Robot Skills Adaptation of Learned Models with Task Constraints

Daniel Hernández García

Director:
Dr. Carlos Balaguer Bernaldo de Quirós
Dr. Concepción Alicia Monje Micharet

Escuela Politécnica Superior
de la Universidad Carlos III de Madrid

Robot Skills Adaptation of Learned Models with Task Constraints

Robot Skills Adaptation of Learned Models with Task Constraints

Daniel Hernández García

Director:
Dr. Carlos Balaguer Bernaldo de Quirós
Dr. Concepción Alicia Monje Micharet

2014

Escuela Politécnica Superior
de la Universidad Carlos III de Madrid

Universidad Carlos III de Madrid

Publication Data:
Daniel Hernández García
Robot Skills Adaptation of Learned Models with Task Constraints.
Universidad Carlos III de Madrid

For a smile, like yours.
Everything, always
for a smile.

ABSTRACT

A major goal in robotics research is to develop human-like robotic systems capable of interacting and collaborating with humans. The ultimate goal is for a robotic platform capable of performing, autonomously, in the unstructured scenario of human's natural environment. Humanoid robots must carry out any number of tasks which their human operators could reasonably expect from them during the normal development of a typical working day. Working alongside humans means dealing with continuously changing environments and a huge variety of tasks, thus the robots should have the ability to continuously learn new skills and adapt their existing skills to new contexts. Therefore, humanoid robots need to display intelligent behaviour. Key attributes required to consider the behaviour of an agent as intelligent are the abilities to learn and acquire knowledge based on its experience, the capacity to understand or comprehend current relevant features in the environment, the capacity for reasoning and, also the ability to adapt.

A framework for humanoid robots needs to provide a minimum degree of intelligence, that is, the ability to sense the environment, learn and, adapt its actions to perform successfully under a given set of circumstances. Humanoids must be provided with systems that allow them to continuously learn new skills, represent their skill's knowledge, and adapt their existing skills to new contexts, as well as robustly reproducing new behaviours in a dynamic environment in order to cope with working in continuously changing environments and performing a huge variety of tasks.

In our context a skill is defined as a motor trajectory motion learned by the agent, an acquired ability for the execution of a task. A robot skill is a complex action movement, reproducible when appropriate, and generalize to different contexts. Learning systems are required to acquire skills and develop task knowledge of how to act. Algorithms for learning and extracting important features of task actions are fundamental in order to build intelligent behaviours. The *Imitation Learning* approach formulates user-friendly methods by which a human user can teach a robot how to accomplish a given task, and generalize the demonstrated movements across a set of demonstrations. To learn the skills motion, a time independent model of the motion dynamics is estimated through a set of first order non-linear multivariate dynamical systems. We employ *SEDS* algorithm to learn a global dynamical estimate of the motion, through a set of first order non-linear multivariate dynamical systems in a statistical approach, as movement primitives.

Despite the *Imitation Learning* approache's clear advantages, it would still be impractical for the human operator to teach the robot the skills for every needed task

and for every foreseen situation, since the number of demonstrations the human must provide to the robot to generate a new model of a skill could turn it into a tiresome and time-consuming process; furthermore, it wouldn't be possible to cover every required task and every situation. Therefore, it is necessary to extend the classical *Imitation Learning* approach to learning a skill model in a way that allows the adaptation of a robot previously learned motion skills to new unseen contexts. The models of a skill are adapted to generate a new task by a merger, transition, combination or update operation over the given robot skill models.

To reproduce a task adapted for an unseen context the robot must be given knowledge of the state of the environment and the constraints of the task. Using both, the already learned model of a skill, and the extracted constraints information of the current task, the model of the skill can be adapted to reproduce the task. The robotic systems must be able to store and later retrieve and use their knowledge of learned skills. The aim would be to have a knowledge base of the robot available skills for reproduction. The knowledge base needs to hold all necessary information for reproduction of the skills in the environment. Knowledge of the task would be distributed among the representation of objects, actions and events of the task and the state of the world.

This work is centred on the major idea of future robotic systems, more specifically humanoid robots, that are capable of interacting with humans in their homes, workplaces, and communities, providing support in several areas, and collaborating with humans in the same unstructured working environments. The aspiration is to have humanoid robots acting as robot companions and co-workers sharing the same space, tools, and activities.

Our focus is on topics concerning the learning, representation, generation and adaptation, and reproduction of robot skills. In this work a framework is proposed for the learning, generation and adaptation of robot skill models for complying with task constraints. The proposed framework is meant to allow: an operator to teach and demonstrate to the robot the motion of a task skill it must reproduce; to build a knowledge base of the learned skills, allowing for their storage, classification and retrieval; to adapt and generate learned models of a skill, to new contexts, for compliance with the current task constraints.

RESUMEN

Uno de los objetivos principales en la investigación en robótica es el desarrollo de sistemas robóticos humanoides capaces de interactuar y colaborar con humanos. La meta final es desarrollar una plataforma robótica capaz de trabajar, de forma autónoma, en el entorno no estructurado del día a día de los humanos. Los robots humanoides deben realizar un sinfín de tareas, que su operador humano pueda requerir, durante el desarrollo normal de un día de trabajo. Trabajar junto a los humanos significa hacer frente a cambios continuos en el entorno y una gran variedad de trabajos, por lo tanto los robots deben tener la capacidad para aprender nuevas habilidades constantemente y para adaptar las habilidades ya aprendidas a nuevos contextos. Por ellos, los robots humanoides necesitan presentar un comportamiento inteligente. Atributos clave para considerar el comportamiento de un agente como inteligente son la capacidad para aprender y adquirir conocimientos basado en la experiencia, la capacidad para entender y comprender características relevantes presentes en el entorno, la capacidad para razonar, y también la capacidad para adaptarse.

Un sistema para robots humanoides debe proporcionar un mínimo nivel de inteligencia, i.e., la capacidad de percibir el entorno, aprender, y adaptar sus acciones para desempeñarse exitosamente bajo un conjunto de circunstancias. Robots humanoides, para poder afrontar los desafíos de trabajar en entornos cambiantes realizando una gran variedad de tareas, deben estar provistos de sistemas que les permitan aprender nuevas habilidades, representar el conocimiento de sus habilidades, y adaptar sus habilidades a nuevos contextos, así como también reproducir robustamente nuevos comportamientos en un entorno dinámico.

Una habilidad en nuestro contexto se define como una trayectoria motora aprendida por el agente, una capacidad adquirida para la ejecución de una tarea. Una habilidad robot es un movimiento de acción complejo reproducible cuando sea necesario, y generalizada para diferentes contextos. Sistemas de aprendizaje son necesarios para adquirir habilidades y generar el conocimiento de la tarea sobre cómo actuar. Algoritmos de aprendizaje, y de extracción de características importantes de una tarea son fundamentales con el fin de construir comportamientos inteligentes. El *Aprendizaje por Imitación* formula métodos mediantes los cuales un usuario humano puede enseñar a un robot como ejecutar una tarea, y generalizar los movimientos a partir de demonstraciones. Para aprender los movimientos de una habilidad un modelo independiente del tiempo de la dinámica del movimiento se estima mediante un conjunto de sistemas dinámicos de primer orden multi-variable. El algoritmo *SEDS* se emplea para aprender una estimación dinámica global del movimiento, en un enfoque

estadístico, como una primitiva de movimiento.

A pesar de las claras ventajas del *Aprendizaje por Imitación*, resulta de todas formas poco práctico para un operador humano enseñar al robot las habilidades requeridas para cualquier tarea y para toda situación previsible, ya que por el número de demonstraciones que el humano debe dar al robot para generar un nuevo modelo se convertiría en un proceso costosos y tedioso. Por lo tanto es necesario extender el modelo clásico de *Aprendizaje por Imitación* para aprender un modelo de la habilidad de forma tal que permita la adaptación de los modelos previamente aprendidos por el robot a nuevos contextos. Los modelos de una habilidad son adaptados para generar una nueva tarea por una operación de fusión, de transición, combinación o actualización sobre los modelos de habilidad robot dados.

Para reproducir una tarea adaptada a un nuevo contexto el robot necesita tener conocimiento sobre el estado del entorno y las restricciones de la tarea. Utilizando tanto el modelo de una habilidad ya aprendido como las restricciones de la tarea extraídas del entorno, el modelo de una habilidad robot puede ser adaptado para realizar la tarea. Es necesario que el sistema robótico permita guardar, y luego recuperar, y usar el conocimiento de las habilidades aprendidas. El objetivo sería tener una base de conocimientos de las habilidades del robot disponibles para la reproducción. La base de conocimientos debe contener toda la información necesaria para la reproducción de las habilidades en el entorno. El conocimiento de la tarea se distribuye entre la representación de los objetos, acciones y eventos de la tarea y el estado del entorno.

Este trabajo se centra alrededor de la idea global de los sistemas robóticos del futuro, en particular de los robots humanoides, que deben ser capaces de interactuar con los humanos en sus hogares, lugares de trabajo y comunidades, prestando apoyo en varias áreas y colaborando con los seres humanos en su entorno de trabajo. La aspiración es tener robots humanoides actuando como compañeros de trabajo que compartiendo el mismo espacio, herramientas y actividades que los humanos.

Nuestra atención se centra en temas relacionados con el aprendizaje, la representación, la generación y la adaptación, y la reproducción de habilidades robot. En este trabajo se propone un sistema para el aprendizaje, la generación y adaptación de modelos de habilidad del robot para cumplir con las restricciones de la tarea. El sistema propuesto permite: enseñar y demostrar al robot el movimiento de una habilidad que debe reproducirse; construir una base de conocimientos de las habilidades aprendidas lo que permite su almacenamiento, clasificación y recuperación; generar y adaptar modelos de habilidades ya aprendidos a un nuevo contexto para el cumplimiento de las restricciones de la tarea actual.

ACKNOWLEDGEMENT

First of all I'm thankful to my advisers for their support, their suggestions, and their patience during my research and completion of this thesis. I want to acknowledge Dr. Carlos Balaguer, not only for his work as director of this thesis, but for granting me the opportunity to be a part of his research group and to work in such fascinating topics as the one's dealt on this work. I want to acknowledge Dr. Concepción Monje for her guidances and valuable advices and for her corrections and help during my research.

A very special thanks to Prof. Aude Billard for receiving me at the *LASA* laboratory of the *EPFL* for a research visit, where I studied the learning techniques that support this work. I also extend my thanks to the rest of the *LASA* laboratory who make me feel welcome at their lab.

I also want to thank all my partners working at the *RoboticsLab*. To Paolo and Miguel who closely worked with me with the *HOAP-3* robot. Santi, Juan, Tamara, Juanmi, Jose and Alberto with the humanoids group. Sonia, Edu, Angela and Fernando which invaluable support makes all of our work possible. I want to extend my thanks to everyone in the Departamento Ingeniería de Sistemas y Automática where I have worked in the completion of this thesis, and special acknowledgements to Martin, Javi, Abdullah, Victor and Raúl for their friendship and to Silvia for everything.

I want to acknowledge everyone at the Universidad Carlos III de Madrid were I have spent 6 wonderful years completing my Masters and Doctoral studies, specially to the Escuela de Doctorado, Miriam Sanchez, María Belén García and María Escudero for their help.

I must always thank Prof. Gerardo Fernández who introduced me to the world of robotics. And to my dear friends from across the pond: Sylvia, Joshue, Idania, Luis, Juan and Mariana.

Finally, I want to thank all my family which I love and who has always supported me and cared about me.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

Work on this thesis focuses on the development and implementation of techniques that allow humanoid robots the ability to continuously learn new skills and adapt theirs existing skills to new contexts. For this goal it is proposed to follow a framework that allows, i) for an operator to teach and demonstrate to the robot the motions of a skill it must reproduce; ii) to build a knowledge base representation of the learned skills allowing for its storage, classification and retrieval; iii) generate and adapt learned models of a skill, to new contexts, for compliance with the current task constraints.

Long before the work of Karel Capek gave us the word ***Robot***, which first appeared in "R.U.R. (Rossum's Universal Robots)" in 1920, the idea of automated machinery, capable of performing a variety of functions and tasks, and of working and serving humans, has been a part of the collective imagination of mankind. Various examples of attempts to build such automatons can be found, from the earlier endeavours of the ancient Greeks and Arab civilizations, to the work of influential thinkers like Leonardo da Vinci's robot, c. 1495, etc. The current vision of robotics in society stems from television, films and science fiction; however, technological advances throughout the 20th century have allowed for the development of robotic solutions, in industrial and manufacturing applications, to become a reality. It is the author's vision, that in perhaps a not too distant future, there will be a world in which humanoid robots and humans will work and interact side by side, sharing the same space, tools, and activities.

This chapter lays out the motives and goals for our research and presents the background of the topic as a basis for the remainder of the document. Section 1.1 presents the issues and motivations that inspired the work on this thesis. Section 1.2 presents the aim and objectives pursued in this work. Section 1.3 presents the contributions of this thesis. In Section 1.4 the outline for the remainder of this work is described.

1.1 Motivations

Since the 1980s, robots have been progressively introduced in the industry for the automation of manufacturing processes performing precise and repetitive tasks, handling delicate or dangerous substances, lifting heavy objects, etc. Robotic systems have enjoyed wide applications in several areas such as the automotive, chemical, electronics and food industries. As technological developments in robotics science have advanced, the range of robotic applications has expanded from its initial dominant industrial settings into more day to day aspects of the human world. The

next generations of robots will need to be able to interact with humans at homes, in the workplace, and in the community, providing support in several areas, such as, services, entertainment, education, healthcare, manufacturing, and assistance [Siciliano and Khatib, 2008].

One major goal in robotics research is to develop human-like robotic systems capable of interacting and collaborating with humans in the same unstructured working environments. Humanoid robots are particularly suitable for these duties because they are able to interact with the environment using the same tools designed for humans, and can collaborate with humans in several ways [Ambrose et al., 2000], [Monje et al., 2008]. Also, it is believed [MacDorman and Cowley, 2006], that the most human-like of robots will be best equipped for reciprocal relationships with human beings. Since humanoid robots are designed to resemble a human shape and to possess human capabilities, they would be ideally suited for performing tasks and to safely share the same space and activities with people without the need to adapt the environments and with a higher level of acceptance and a more intuitive way for interaction between human operators and the robotic agents. We envision a world where humanoid robots and humans would work, collaborate and interact together, sharing the same space, tools, and activities.

From the first full-scaled humanoid robot, WABOT-1 developed by Waseda University [Sugano and Kato, 1987], and the series of robotic prototypes from Honda, E-series 1986-1993, P-series 1993-1997 [Hirai et al., 1998], steady progress can be seen in the development of humanoid robots. Recent years have seen an increase in research of humanoid robots such as the WABIAN-2 from the University of Waseda [Ogura et al., 2006], ASIMO of Honda [Sakagami et al., 2002], the HRP-2 from the National Institute of Advanced Industrial Science and Technology of Japan (AIST) [Kaneko et al., 2004a], the development of the iCub robot [Tsagarakis et al., 2007], for research into human cognition and artificial intelligence at the Italian Institute of Technology, the Robonaut project at NASA's JSC [Ambrose et al., 2000], Robonaut 2 was moving aboard the International Space Station on October 2011, the first humanoid robot in space [Diftler et al., 2011], Boston Dynamics PETMAN anthropomorphic robot which can move dynamically like a real person [Raibert, 2010], the HOAP robot series of Fujitsu [Riezenman, 2002] or the RH series of humanoid robots [Arbulú et al., 2009], [Martinez et al., 2012], designed at the Universidad Carlos III de Madrid.

Figure 1.1, presents some of the most relevant developments in humanoid robotics research. The field of humanoid robots has presented important advances over the years. Yet many challenges still remain before robots can be fully integrated as part of everyday human activities, especially when thinking about humanoid robots, which must naturally be expected to deal with a wide range of movements and tasks; the inherent complexities associated with the need to operate in the real world must also be taken into consideration. In order to overcome some of these challenges, humanoid robots must be provided with the capabilities to interact autonomously and intelligently with humans and the environment. They must also be able to learn and adapt their behaviour to achieve goals and react to changes in a complex and evolving range of different situations.

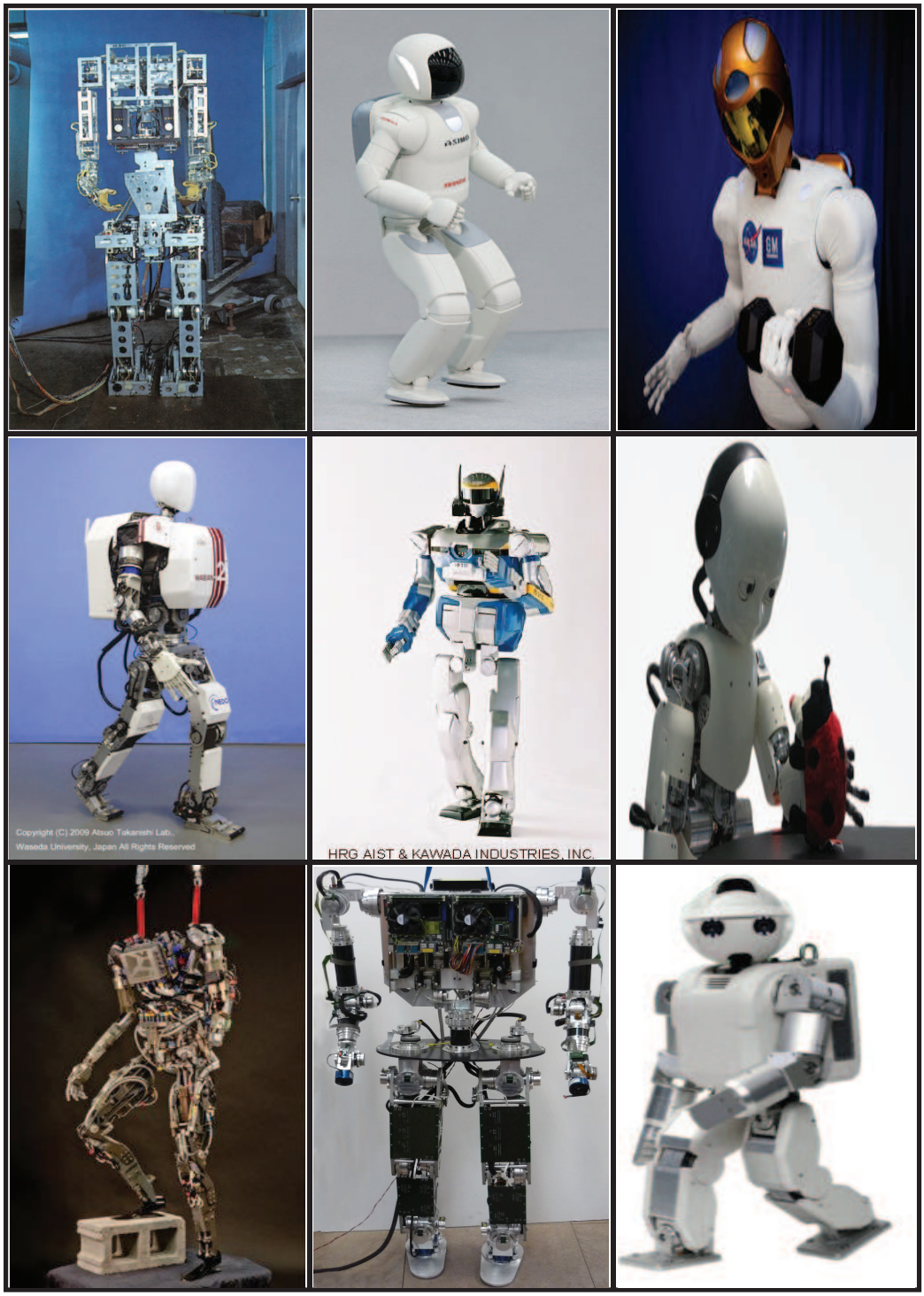


Fig. 1.1: *Historical developments in the field of humanoid robotics.*
Top row (left to right): WABOT-1 of Waseda University (1973), ASIMO of Honda (2000), NASA Robonaut-2 (2010).
Middle row (left to right): WABIAN-2 Waseda University (2006), HRP-2 from AIST (2002), iCub from IIT (2004).
Bottom row (left to right): PETMAN of Boston Dynamics (2010), TEO from Universidad Carlos III de Madrid (2012), HOAP-3 of fujitsu (2005).

A humanoid robot needs to be provided with intelligence, that is, the ability to sense the environment, make decisions and take actions, to recognize objects and events, represent knowledge, reason and plan for the future and to act successfully under a large variety of circumstances. Key attributes required to consider the behaviour of an agent as intelligent are the abilities to learn and acquire knowledge based on its experience, the capacity to understand or comprehend current relevant features in the environment, the capacity for reasoning, and also the ability to adapt. In order to have humanoid robots acting fluently in the world, interacting with different objects and people, they must be able to adapt their motor control to dynamic changes in their interaction with the world. Robot systems must be continuously self-adapting [Brooks, 1996]. An intelligent agent is one that is flexible to changing environments and changing goals, learns from experience, and makes appropriate choices given perceptual limitations and finite computation [Poole et al., 1998]. Intelligence requires an interconnecting system that enables the various system elements to interact and communicate with each other, integrating perception, reason, learning and behaviour generation [Albus, 1991]. [Langley et al., 2009] identified, from a robotic systems point of view, the different functions of cognition as perception, learning, motor control, reasoning, problem solving, goal orientation, knowledge representation and communication. Control architectures for intelligent humanoid robots need to consider these systems. A framework for humanoid robots needs to provide a minimum degree of intelligent behaviour, that is, the ability to sense the environment, learn, and adapt its actions to perform successfully under a given set of circumstances.

Learning systems are required to acquire skills and develop task knowledge of how to act. Algorithms for learning, and extracting important features of task actions, and exhibiting altered behaviour because of what has been learned, are fundamental in order to build intelligent behaviours. For humanoid robots to work with humans in unstructured environments, the robot must be able to perform dynamically changing tasks that require great adaptations to react to new constraints. The programming of specialized controllers for every single task and situation that could be encountered would not be a practical approach. To develop the capacities expected from future humanoid robots, flexible and generic control methods that can adapt to various tasks and robot's constraints are necessary. *Robot Programming by Demonstration (RPbD)* [Billard et al., 2008], also known as *Imitation Learning* or *Learning from Demonstrations (LfD)* [Argall et al., 2009], has appeared as one way to respond to this growing need for intuitive control methods.

The *Imitation Learning* approaches focus on the development of algorithms that are generic in their representation of the skills and in the way they are generated. Implementing *LfD* methods offers the possibility of making learning faster, in contrast to tedious reinforcement learning methods or trial-and-error learning. *LfD* formulates user-friendly methods by which a human user can teach to a robot how to accomplish a given task, simply by demonstrating this task [Gribovskaya et al., 2010], and generalizing the demonstrated movements across a set of demonstrations. *LfD* focuses on three important issues: efficient motor learning; the connection between action and perception; and modular motor control in the form of movement primitives [Schaal, 1999]. To reproduce a skill in a new situation, the robot cannot simply

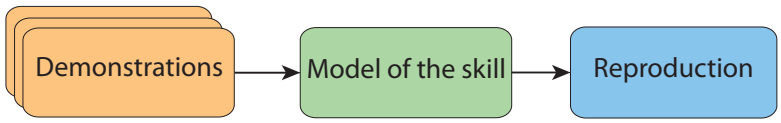


Fig. 1.2: *Generalization of a skill in Robot PbD by extracting the statistical model across multiple observations. Adapted from [Billard et al., 2008]*

copy an observed behaviour; it must have the capability to generalize. In *LfD*, one common approach for generalizing a skill consist in creating a model of the skill based on several demonstrations, performed in slightly different conditions [Calinon, 2009]. The goal is on exploiting the variability inherent to the various demonstrations and to extract the essential components of the task. Figure 1.2 illustrate this process.

Observing multiple demonstrations can help to generalize a skill by extracting the task requisites. Current approaches to generalizing skills can be broadly divided between two trends. Firstly, symbolic level representation, described by the sequential or hierarchical organization of a discrete set of primitives that are predetermined or extracted with predefined rules. Secondly, trajectory level representation, described by temporally continuous signals representing different configuration properties changing over time [Calinon, 2009]. One trend of research followed in this work investigates how statistical learning techniques deal with the high variability inherent to the demonstrations [Calinon et al., 2007], using *Gaussian Mixture Models (GMM)* to encode a set of trajectories, and *Gaussian Mixture Regressions (GMR)* to retrieve a smooth generalized version of these trajectories and associated variabilities, allowing learning non-linear dynamics of the motions as movement primitives [Gribovskaya et al., 2010].

The *Learning from Demonstration (LfD)* approaches offer natural, fast and implicit means of teaching a robot new skills. But even then, the number of demonstrations the human must provide the robot with, in order to generate a new model of a skill could turn it into a tiresome and time-consuming process; and it would also become impractical for the human operator to teach the robot every necessary task and every foreseeable situation. Hence, it will be important to enrich this approach with the capacity to generate new skill models. Also, though *LfD* offers the capability to generalize a learned model, this generalization is somewhat limited to changes in initial conditions or to relatively small perturbations during the execution. Therefore, it is necessary to extend the classical *LfD* approach of learning a skill model in a way that allows the adaptation of a robot previously learned motion skills to new unseen contexts. Some very important questions need solving in this field: Is there a basic set of primitives? How can new primitives be learned, and old primitives be combined to form higher level movement primitives? How can sequencing and recognition of sequences of movement primitives be accomplished? [Schaal, 1999].

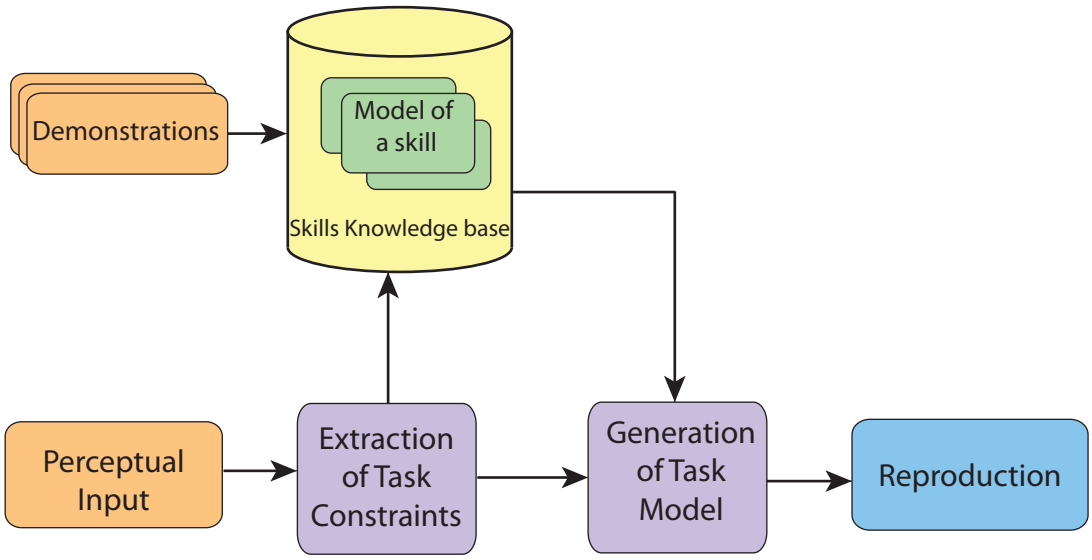


Fig. 1.3: *Propose framework for the generation and adaptation of learned models of a skill for complying with task constraints. A knowledge base of skills models, learned through demonstration, is built. From the perception of the world state the constraints of a requested task are extracted. A skill model is retrieve from the knowledge base a new adapted task model is generated for reproduction using the current task constraints and the models of a skill in the knowledge base.*

1.2 Aim of this Thesis

For robots, working alongside humans means dealing with continuously changing environments and a huge variety of tasks which they are expected to perform. Thus humanoid robots should have the ability to continuously learn new skills and adapt the existing skills to new contexts. As stated in the previous section, for future humanoid robots the ultimate goal is for a robotic platform capable of performing, autonomously, in the unstructured scenario of humans natural environment, be this by itself or sharing the workspace with a human. Humanoid robots must realize any number of task which could be reasonably expected from them by their human operators during the normal development of a typical working day.

It is necessary for humanoid robots to display a sufficient level of intelligent behaviour; this must include the capacity to perceive and understand, to choose wisely, and to act successfully under a large variety of circumstances [Albus, 1991]. Humanoid robots, in order to cope with working in continuously changing environments and performing a huge variability of tasks, must be provided with systems that allow them to continuously learn new skills and adapt their existing skills to new contexts, as well as to robustly reproduce new behaviours in a dynamic environment.

To advance in the achievement of this vision, though still a long way from the ultimate goals of a perfect humanoid, we propose to follow a framework that allows:

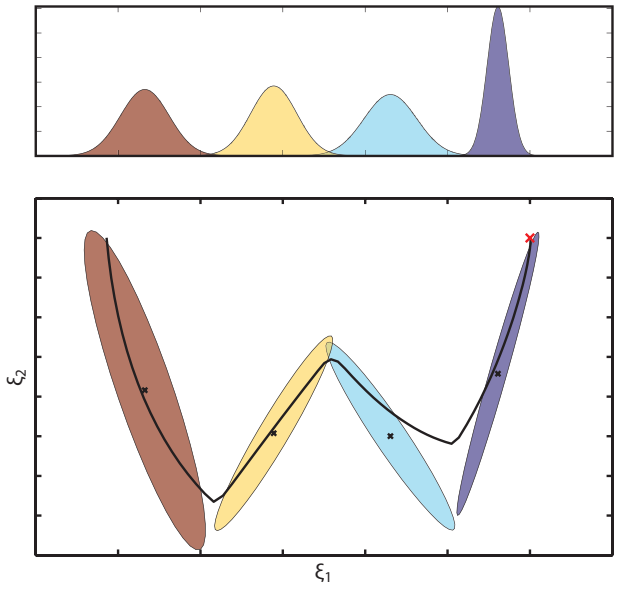


Fig. 1.4: *Robot skill modelled with a mixture of Gaussian functions. The robot skill trajectory in the bottom figure is modelled by 4 Gaussian functions drawn with the 3-sigma ellipses with their centres at their respective means μ, the magnitude and direction of the ellipses are given by the eigenvectors and eigenvalues of the covariance matrix Σ modelled in the learning process to follow the shape of the skill trajectory. The top figure shows the corresponding Gaussian distributions of the 4 Gaussian components in the robot skill model. The figure is shown in a two dimensional plane for arbitrary state variables ξ_1, and ξ_2.*

- An operator to teach, and demonstrate, to the robot the motion of a task skill it must reproduce.

- To build a knowledge base of the learned skill models allowing for their storage, classification and retrieval.

- To adapt and generate learned models of a skill, to new context, for compliance with the current task constraints.

Our propose framework is illustrated in Figure 1.3.

A **Skill** in our context is defined as a motor trajectory motion learned by the agent, an acquired ability for the execution of a task. A robot skill is a complex action movement reproducible when appropriate, and generalized to different contexts.

To learn the skills motions, a time independent model of the motion dynamics is estimated through a set of first order non-linear multivariate dynamical systems. [Ijspeert et al., 2002] propose an approach to *Imitation Learning*, and on-line trajectory modification, by representing movement plans based on a set of non-linear differential equations with well-defined attractor dynamics. We follow a framework

presented on [Gribovskaya and Billard, 2009], that allows learning of non-linear dynamics of motion in manipulation tasks and generating dynamical laws for control of position and orientation, and employed [Khansari-Zadeh and Billard, 2011] algorithm to learn global dynamical estimate of the motions through a set of first order non-linear multivariate dynamical systems in a statistical approach.

We build a model estimate of our robot skill, $\bar{\mathcal{M}}$, from a set $\mathcal{D}$ of N-dimensional demonstrated data points, $\{\xi_i, \dot{\xi}_i\}_{i=0}^{\mathcal{D}}$, where ξ is a state variable describing the state of the robot system. The motion is governed by a first order autonomous ordinary differential equation, $\dot{\xi} = f(\xi, \theta)$. Following a statistical approach an estimate $\hat{f}$ is defined through a *Gaussian Mixture Model (GMM)*. The robot skill is modelled by the parameters θ of $\hat{f}$ determined by $\hat{f} = \{\mathcal{N}^1(\xi; \theta^1), \cdots, \mathcal{N}^{\mathbf{K}}(\xi; \theta^{\mathbf{K}})\}$, where $\theta^i = \{\pi, \mu, \Sigma\}$ of the $\mathcal{N}^i$ Gaussian define the prior, mean and covariance matrix, parameters of the i Gaussian component, and $\mathbf{K}$ is the total number of Gaussian functions required to estimate the motions dynamics. After training, to recover the expected output variable $\hat{\xi}$ a *Gaussian Mixture Regression (GMR)* process is used [Gribovskaya et al., 2010]. Figure 1.4 illustrates the encoding the dynamics of a motion with a mixture of Gaussian functions.

In order for the robot to be able to perform various different actions a repository of the available skills is necessary. The aim is to populate a knowledge base of the robot learned skills for reproduction. The knowledge base needs to hold all necessary information for reproduction of the skills. The tasks the robot is requested to carry out are considered to be of the form ⟨ *robot pick blue ball* ⟩, ⟨ *robot place cup on plate* ⟩, and so on. in which a **Task** is described requesting an operation upon an object for the execution of a goal oriented skill action. Complex sets of behaviours can be built by a planned sequencing of tasks.

One intuitive way in which to represent elements in the knowledge base is over two principal directions of objects and actions. However, objects and actions alone do not provide sufficient and complete information for a robot situated in its environment to be capable of performing its task adequately. For instance, for a single behaviour there could be more than one available pairing of ⟨ *object, skill model* ⟩, leading to ambiguities. At least one more direction for representations would seem necessary, such as a description of the state of the environment. To resolve this problems it is suggested to consider two more representational directives, one for the task goal, and one for the configuration of the current state of the world, mainly objects position and relationships between themselves, the robot and a human operator.

In this way, a **Task** could be represented by the phrase ***"Do an Action (A), To an Object (O), For achieving Goal (G), When State of the World is (W)"***. Therefore, the tuple formed by ⟨ *Do = Action(A), To = Object(O), For = Goal(G), When = World State(W)* ⟩ holds all necessary information for the robot reproduction of a task. The framework in Figure 1.3 would allow the robot to extract the knowledge about objects, goals, and the current state of its working environment from the received perceptual input. The robotic system would be able to retrieve an appropriate **Skill** from the knowledge base by finding the answer to the phrase ***"Do Action (A) ... "*** for its current constraints when being presented with the triple ⟨*Object, Goal, World State*⟩.

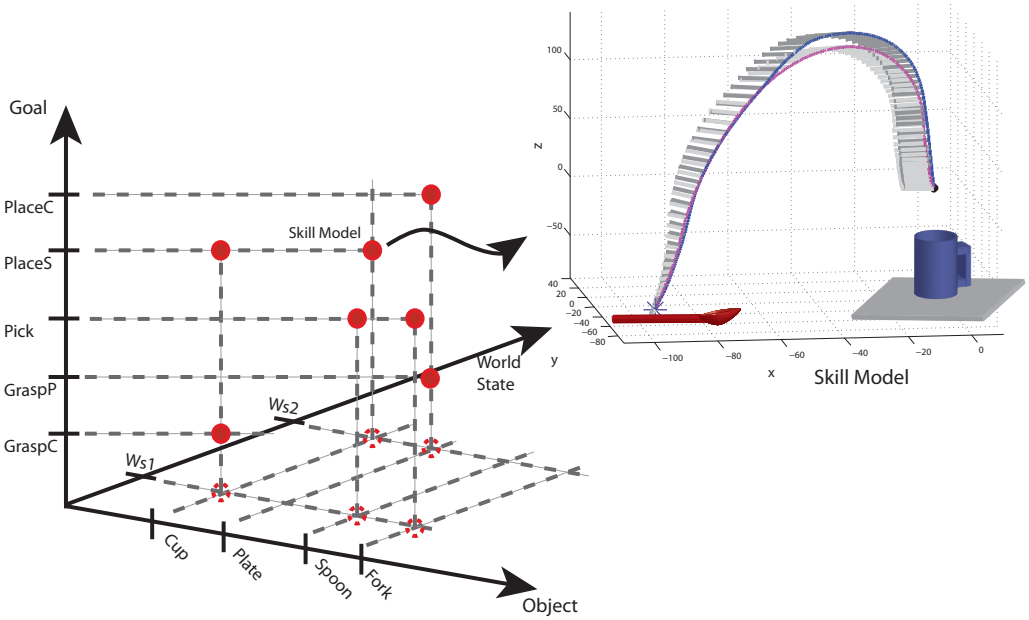


Fig. 1.5: *Representation of the skills in the knowledge base. Example of a robot skill model, represented by a Cartesian trajectory, being selected from the current constraints of the task given by the object, goal and world state.*

As an example let us consider a simple case in which a humanoid robot is requested to place a spoon inside a cup, and place the cup on top of a saucer plate, as if it would be serving a cup of tea or coffee. Therefore to complete this request the robot would be required to perform several tasks, such as grasping the cup and placing the cup on top of the saucer plate. Each of these tasks has an *Object* $\langle Spoon,\ Cup,\ Plate \rangle$, a *Goal* $\langle Grasp,\ Pick,\ Place \rangle$, and *World State* in which the task must be performed. The robot knowledge base would have different models of skills allowing it to perform different actions which may permit the robot to fulfil various tasks in different situations. To successfully complete the given tasks a fitting *Action* must be executed by the robot retrieving from its knowledge an appropriate model of a skill for the constraints given by the **Tasks** $\langle Object,\ Goal,\ World\ State \rangle$.

Imagine for instance the execution of the $\langle$ *robot place spoon inside cup* $\rangle$ tasks. To perform the task it is assumed that the spoon object has already been picked by the robot and is in one of its hands, so the target object for the task is the cup $\langle Object\ :\ cup \rangle$. The goal of the task is to attain a state in which the spoon has been placed inside the cup, so $\langle Goal\ :\ place\ spoon \rangle$. To complete the task goal the robot could have learned and stored different skill models in its knowledge base, each appropriate to successfully executing the task in different states. Therefore the state of the world must be evaluated next, let's assume it could be one of two states; $\langle Ws1 \rangle$ in which the cup is on the table inside the robot's arms workspace, and a $\langle Ws2 \rangle$ in which the cup is grasped in the robot's other hand. And that for the current execution of the

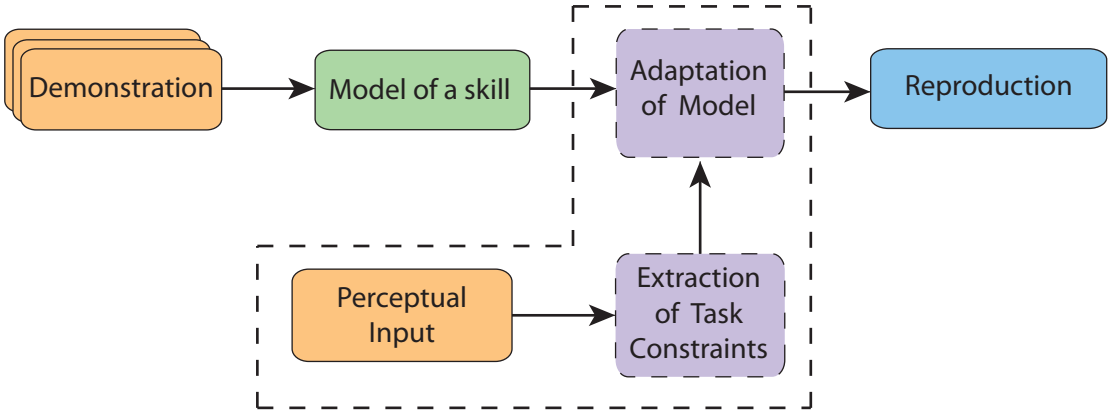


Fig. 1.6: *Adaptation of a learn skill to new context by extracting the task constraints with a new observation. The Robot Programming by Demonstration approach for generalization of a skill is extended to allow adapting previously learn skills models.*

task the robot finds its environment to be $\langle World\ State :\ Ws2 \rangle$. In order to select the right skill model, i.e., the *Action* to perform, the knowledge base is inspected to retrieve the skill model corresponding to the triple $\langle cup,\ place\ spoon,\ Ws2 \rangle$. Figure 1.5 shows the representation of skills in the knowledge base. The tasks described in this example are very simple and their reproduction only requires a single skill. However, more complex tasks could require linking two or more skills together.

Having already stored in the knowledge base a set of robot skill models, learned by different demonstrations of the skills to form a basic set of motion primitives, to reproduce a task the robot is provided with knowledge of the environment and the task constraints extracted from its perceptual input in the knowledge base. Using both, the already learned model of a skill, and the extracted constraints information of the current task, the model of the skill is adapted to reproduce the task. Figure 1.6, illustrates the process of adapting a learned skill in an unseen context.

The robot would receive from the different modules of perception and interaction the required appropriate commands ordering the reproduction of a skill and would extract the constraints of the task and its environmental configuration to instantiated the appropriate knowledge structures in the knowledge base of the robot's skills. With this information taken from the knowledge base, together with the models of the skills corresponding to the requested task, the module for the generation of task models is called to adapt the robot skills accordingly and generate the task models for the robot reproduction of the task.

A desirable application for the learned skill models would be in building libraries of so called movement primitives that can be readily available for later reuse by the robot, when a situation requires it. A **basic or primitive skill** can be understood as sequences of motor commands executed in order to accomplish a certain movement action. To generate complex human like motions from a learned set of primitive skills, methods for operating and manipulating the primitives must be developed. The robot

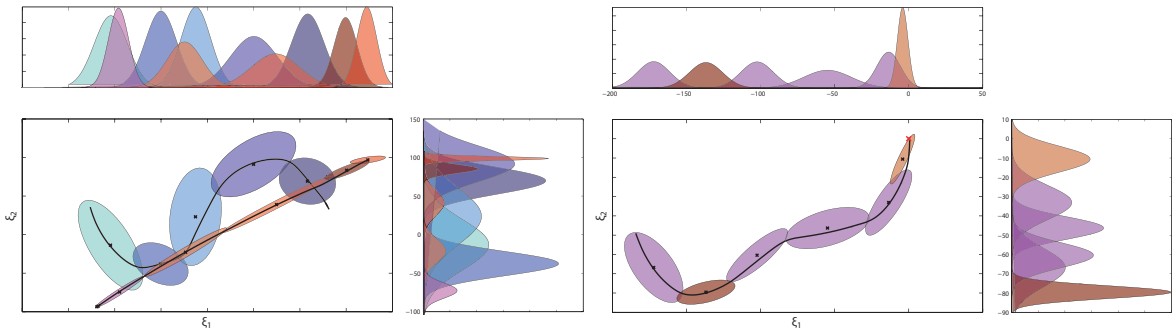


Fig. 1.7: *Generation of a new model of a skill by combining two previously learned skill models. Two or more basic models of a skill can be combined (left) to generate a new complex model of a skill (right) for complying with the constraints of the current task. The figure is shown in a two dimensional plane for arbitrary state variables ξ_1, and ξ_2. The top and side panels shows the corresponding Gaussian distributions of the robot skill models for the ξ_1 (top) and ξ_2 (side) state variables.*

skills must be adaptable to conditions of its operating environment, they must also be updatable when given new information. Additionally, new skills must be generate by merging two or more simpler skills into a new skill or by combining models to generate new models. Also, working with a set of basic or primitives skills must give the ability to create sequences and transitions between robot skill models to generate complex behaviours.

Joining multiple models can provide improvements in performance, different methods can be found for the combination of models in the field of machine learning and pattern recognition [Bishop, 2006]. Methods include using the average predictions made by different learned models, selecting one model out of several, to make the prediction as a function of the input variables. In this way, different models become responsible for making predictions in different regions. Also, probabilistic methods known as mixtures of experts, the models can be viewed as mixture distributions conditioned by the input variables, the idea behind this is that different components of the learned models of skills can model the new skill distribution in different regions of input space, and will also look to determine the functions that decide which components are dominant in which region. Figure 1.7 illustrates the process.

1.3 Contributions

The work that lead to this thesis centred around the major idea of future robotic systems, more specifically humanoid robots, that are capable of interacting with humans in their homes, workplaces, and communities, providing support in several areas, and collaborating with humans in the same unstructured working environments. The aspiration is to have humanoid robots acting as robot companions and co-workers sharing the same space, tools, and activities.

Work on this thesis focuses on topics concerning the learning, representation, generation and adaptation, and reproduction of robot skills. The main contributions presented in this thesis are:

1. The proposition of a framework for the learning, generation and adaptation of skill models to comply with task constraints, Figure 1.3. The goal of the developed framework is to provide a minimum degree of intelligence for humanoid robots to allow them to work and collaborate with humans. The framework provides humanoid robots with systems allowing them to continuously learn new skills, represent skill's knowledge, and adapt existing skills to new contexts, as well as to robustly reproduce new behaviours in a dynamical environment.

2. The application and evaluation of different *Learning from Demonstration* algorithms and modalities. Through the work on this thesis, a number of *Imitation Learning* techniques have been studied and implemented in teaching and learning with the robot, the different sets of skills employed in the rest of the framework. Methodologies used for the reproduction of the learned motion dynamics of the robot skills were reviewed, comparing the performance of the methods presented through this work.

3. The development and implementation of a knowledge base. The knowledge base represents knowledge of objects, actions, world state and task goals. Our representations includes information about objects and actions, the world and situations, events and goals, for effective situated performance. A method for the representation of the knowledge of the skills and task constraints needed for reproduction has been developed.

4. The development and implementation of modalities that allows for the adaptation and generation of new skill models. The development of humanoid robotic systems requires that models of the skill can be operated upon to generate new behaviours of increasing levels of complexity. Different modes were developed and implemented that allow for the adaptation and generation of new skill models based on the already learned models of skills.

5. The evaluation of different scenarios to test the performance of the various modules implemented in our framework and to provide separate validation for the operation of the system. The proposed framework was demonstrated with a humanoid robot *HOAP-3*.

1.4 Outline

This document is divided into 5 chapters, in addition to this introductory one, and a final chapter of discussion and conclusion bringing the total to 7 chapters. The chapters, and the topics they address, are organized as follows:

Chapter 2 In this chapter a *state of the art* review is presented on the challenges of intelligent humanoid robots and on different proposals for robot architectures, in planner based, behaviour based and cognitive execution architectures. At the end, a general description of the framework proposed in this thesis is also given.

Chapter 3 This chapter discusses the learning by demonstration framework use in this thesis. A review of the *LfD* algorithms employed for learning the skills models, as well as the *state of the art* on the field is given.

Chapter 4 This chapter discusses the representation of skills knowledge. The method developed for the representation, storage, classification and retrieval of skills knowledge from the knowledge base is described throughout every section of this chapter. The structure and the organization of the knowledge base is developed.

Chapter 5 This chapter discusses the process for the adaptation and generation of skills models. The process for adapting a learned robot skill to the task constraints is described. The algorithms developed for generating and adapting a skill are detailed throughout the chapter.

Chapter 6 This chapter discusses the reproduction of skills in the proposed framework. A detailed description of the implementation process of the framework is given. The experimental study and validation of the framework showing the adaptation of learned models of a skill, to comply with a current task constraint, is presented.

Chapter 7 The last chapter, discusses the contributions of this thesis and describes current and future work.

2. INTELLIGENT ARCHITECTURES FOR HUMANOID ROBOTS

2.1 Outline of the Chapter

This Chapter presents a review of developments, open issues and challenges in the field of humanoid robotics. It also focuses on different proposals for intelligent agent architectures for robotic systems. As outlined in the introduction, one major goal in robotics research is the development of human-like robot systems capable of interacting and collaborating with humans in the same unstructured working environments. These humanoid robot systems would need to interact autonomously and intelligently with humans and the environment, they must also be able to learn and adapt their behaviour to achieve goals and to react to changes in a complex and evolving range of different situations. Intelligent humanoid robotic systems need to present suitable motor skills; the capacity to sense and perceive their environment; the natural means for human-robot interaction and a high level of autonomy and intelligent behaviour. This presents many challenges that need to be overcome. A review of different approaches and architecture proposals aimed at tackling this issues and developing intelligent robotic systems is presented. From deliberative planning architectures to behaviour-based and hybrid approaches and cognitive architectures. Finally, a general description of a framework for a cognitive model for the generation and adaptation of learned models of robot skills, which can be used to comply with task constraints presented in this thesis is also given. The organization of this chapter is as follows:

- Section 2.2 discuss the challenges of developing intelligent humanoid robot systems. Future human-like robot systems need to perform dynamically changing tasks and be able to operate in the real world. Several issues emerge for motor control, perception, interaction and intelligent behaviour.

- Section 2.3 presents a review of approaches to robot planner-based architectures. Deliberative or hierarchical planning architectures follow the Sense-Plan-Act cycle from classical AI approaches. Intelligence resides on a central planner, with world models and system goals that produce appropriate plans of action for robot reproduction.

- Section 2.4 presents a review of approaches to robot behaviour-based architectures. Behaviour-based architectures present direct coupling between percep-

tion and action with no need for internal models. Intelligence emerges as a result of an embodied agent interaction with the environment.

- Section 2.5 presents a review of approaches to robot hybrid deliberative/reactive architectures. Hybrid architectures advocate the use of the advantageous aspects of both the behaviour-based and the planner-based approaches combining them to produce a new architecture that can deal with more complex scenarios.

- Section 2.6 presents a review of approaches to robot cognitive architectures. Many attempts at developing architectures to provide cognitive process are presented.

- Section 2.7 presents the proposed framework followed in the rest of this work to allow the generation and adaptation of learned models of a skill for complying with current task constraints.

2.2 Challenges in Humanoid Robot Development

The idea of automata moving machines in general, and human-like in particular, which are capable of performing a variety of functions and tasks, and of working and serving humans, have been a part of the collective imagination of mankind for centuries.

The drive to developed human-like robots is supported by three basic ideas: 1) Since humanoid robots are designed to resemble a human shape and to possess human capabilities, they would be ideally suited to performing tasks and to safely share the same space and activities with people without the need to adapt the environments. Designing general purpose humanoid robots would make them more flexible in handling a wide range of chore; and less expensive and efficient than developing specialized robots for every task. 2) A humanoid robot would allow for more natural means of interaction, sharing a similar embodiment would give for humans an easier way to teach a robot and to understand its movements and intentions. Also, the natural human tendency of anthropomorphizing objects would be beneficial in creating human-humanoid robot partnerships. 3) It is expected that human-like robots would be more friendly and acceptable for reciprocal relationships with human beings. A robot system with a human-like shape and behaviour would be more acceptable to regular citizens and non-robotic experts as a household servant and companion or as a co-worker and partner to perform everyday tasks.

Historical Developments in Humanoid Robotics

Many projects and research laboratories have put their efforts into designing, building and testing humanoid robotic systems. Over the years great progress has been made in this field, and currently humanoid robots that can walk, climb stairs, carry objects, perform complex activities like dance routines and interactions with

people are available. These advances are very encouraging and forecast the major progress to come. Robotic researchers envision a world, sooner rather than later, where humanoid robots and humans would work, collaborate and interact together sharing the same space, tools, and activities.

Since the first full-scale humanoid robot, WABOT-1, developed by Waseda University [Sugano and Kato, 1987], great advances have been made in humanoid robotics research, especially during the last two decades. Detailing the advances and research efforts in the field during this time would be too extensive. Table 2.1 summarizes the major historical developments in humanoid robotics research. Currently there are robots that walk, run or climb stairs; robots that can handle and manipulate objects, or carry heavy loads; robots that interact and play games with people and children and robots for entertainment that have taken part in shows and demonstrations, dancing or performing complex choreographies. However, all these robots exist in the scope of research departments of universities or technology companies, there are no commercially available humanoid robots for general public use as of today. Despite all the advances, the ultimate goal of an intelligent and autonomous humanoid robot companion is still far from reach.

Date	**Name (Research Center)**	**Development**
1921	R.U.R. (Rossum's Universal Robots)	Karel Capek introduced the word "robot" in his play R.U.R.
1961	Unimate (Unimation)	The first digitally operated and programmable industrial robot, the Unimate created by George Devol, is installed on a General Motors assembly line.
1973	WABOT-1 (Waseda University)	WABOT-1, the first humanoid robot, consisted of a limb-control system, a vision system and a conversation system, it was able to walk.
1984	WABOT-2 (Waseda University)	WABOT-2 was created as a "specialist robot", a musician humanoid robot able to communicate with a person.
1986	E-series (HONDA)	Honda research and development project was initiated with the E-series of walking biped robots, E0-E6 from 1986 to 1993.
1990s	Cog (MIT)	Cog was an upper-torso humanoid robot build as a general purpose flexible and dexterous autonomous robot with the scientific goal of understanding human cognition.
1993	P-series (HONDA)	Development of the P-series of manlike models with upper limbs and body, P1-P3 from 1993 to 1997.
1995	WABIAN (Waseda University)	WABIAN humanoid robot was developed, is a robot with a complete human configuration that is capable of walking on two legs, and it is capable of carrying things.
1998	HERMES (Bundeswehr University Munich)	Service robot HERMES presented for the first time at Hannover Fair, an experimental robot of anthropomorphic size and shape.
2000	ASIMO (HONDA)	HONDA introduced the first version of ASIMO, it can run, walk on uneven slopes and surfaces, turn smoothly, climb stairs, and reach for and grasp objects.

Tab. 2.1: Historical developments efforts in the field of humanoid robotics

Continued from previous page.

Date	Name (Research Center)	Development
2000	ARMAR (KIT)	The Karlsruhe Institute of Technology built the humanoid robot ARMAR, with a mobile wheel-driven platform, two anthropomorphic redundant arms, two simple gripper and a head.
2001	QRIO (Sony)	Sony unveiled the Sony Dream Robot, later named QRIO, a new line of humanoid robots for entertainment robots.
2001	HOAP-1 (Fujitsu)	Fujitsu produce its first commercial humanoid robot named HOAP-1.
2001	KHR-0 (KAIST)	Korea Advanced Institute of Science and Technology began developing humanoid robots, starting with KHR-0 which has 2 legs without the upper body.
2001	Leroy (UC3M)	Universidad Carlos III de Madrid began efforts researching humanoid robots with the development of the 7 DOF bipedal robot Leroy.
2002	HRP-2 (AIST)	Developed under the HRP project. Biped walking robot HRP-2 is 154 cm in height with a mass of 58 kg, including batteries.
2002	ARMAR-II (KIT)	The second version of the ARMAR series, the anthropomorphic body of the robot was placed on a mobile platform, it was able to bend forward, backward and sidewards.
2002	Robonaut (NASA)	Developed by NASA and DARPA, with a human form and scale, Robonaut was design to use many astronaut tools and work in the same tight corridors as astronauts.
2002	RH-0 (UC3M)	Developed at Universidad Carlos III de Madrid, RH-0 was a full-size humanoid robot, with 21 DOF.
2003	HOAP-2 (Fujitsu)	HOAP robots were designed for broad range applications for Research and Development of robot technologies.
2004	KHR-2 (KAIST)	KHR-2 was built as a complete humanoid with 41 DOF and featured improved sensoring with the addition of CCD cameras, inertial sensors, and tilt sensors.
2004	iCUB (IIT)	Italian Institute of Technology began developing the iCub humanoid robot, its aim replicating the physical and cognitive abilities of a 3 year old baby.
2005	HRP-3 (AIST)	The humanoid robot HRP-3 was presented as the succession of humanoid HRP-2, it presented improving capabilities of manipulation and handling.
2005	KHR-3 HUBO (KAIST)	Continued KASIT KHR series, HUBO design aimed to have as many DOF as possible, long working time, compact appearances, low development costs, minimum maintenance.
2005	HOAP-3 (Fujitsu)	Continued the HOAP series, HOAP-3 added movable axis for the head and hands, CCD cameras, a microphone, a speaker and LEDs to show expression.
2005	RH-1 (UC3M)	Continued UC3M RH-1 series, RH-1 humanoid robot have 21 DOF, 150 cm height, 50 kg weight, main objectives were stability control and gait generation.

Tab. 2.1: Historical developments efforts in the field of humanoid robotics

Continued from previous page.

Date	Name (Research Center)	Development
2006	NAO (Aldebaran Robotics)	Aldebaran robotics presented is first humanoid robot NAO, is a small biped robot, fully articulated, easily programmable and low cost.
2006	LOLA (TUM)	Development of LOLA at Technical University of Munich. LOLA is 180 cm and 55 kg, build for fast, human-like, autonomous walking.
2006	ARMAR-III (KIT)	ARMAR-III was presented to the public at CEBIT in spring 2006 in Hannover, ARMAR-III hopes to closely mimic the sensory-motor capabilities of humans.
2006	Justin (DLR)	Justin developed at DLR, the two-arm system Justin is a powerful upper body humanoid robot that is able to lift weights up to 20 kg.
2008	RH-2 TEO (UC3M)	Development started for the humanoid robot RH-2, renamed TEO, It has 26 DOF, a wider workspace and higher manipulability in the different configurations.
2009	HRP-4C (AIST)	AIST presented HRP-4C, it has the appearance and shape of a human being and can walk and move like one, and interacts with humans using speech recognition.
2009	PETMAN (Boston Dynamics)	Boston Dynamics began developing PETMAN, the first anthropomorphic robot that moves dynamically like a real person.
2010	Robonaut2 (NASA)	NASA developed the second generation Robonaut, upgrades included increased force sensing, greater range of motion, higher bandwidth, and improved dexterity.
2011	ASIMO (HONDA)	Honda unveiled its second generation ASIMO Robot. The new ASIMO is the first version of the robot with semi-autonomous capabilities.
2011	Robonaut 2 (NASA)	Robonaut 2 is the first humanoid robot sent into space, arriving at the International Space Station in early 2011.
2012	COMAN (IIT)	IIT released the Compliant huManoid robot, CoMan, designed for robust dynamic walking and balancing in rough terrain.
2013	ATLAS (Boston Dynamics)	Boston Dynamics presented ATLAS for the DARPA Robotics Challenge.

Tab. 2.1: Historical developments in the field of humanoid robotics

Important challenges remain to be solved or addressed. Functional humanoid robots would need to execute a wide range of movements, with high efficiency in terms of energy and performance, and in a natural human-like manner. They would also need to process information from multiple sensors into a reliable representation of the world in order to understand and react to their environment. Humanoid robots would need to provide means for a meaningful interaction with their human partners; they must be engaging and responsive. And they must present intelligent, natural, predictable and reasonable behaviours. Much work remains to be done in order to improve the capabilities of humanoid robots for locomotion, perception, interaction, cognitive behaviour and competence at performing tasks.

2.2.1 Motion Control

A major issue for robotics in general, and more so when dealing with humanoid robots, is motion control. Unlike industrial robots, which are limited to a well known set of movements, and which in general are stationary or need many displacements across established rails inside a room, the appeal and interest in developing humanoid robots is that they are general and flexible in the range of tasks they can perform, as are humans, and that they would be able to move around the whole environment as it is, instead of needing the environment to be adapted in order to allow them to navigate it. Furthermore, since humanoid robots are thought to work, collaborate and interact in proximity to people, unlike industrial robots that perform on their own as part of automated production lines, humanoid robots movements must be compliant and safe for human-robot interaction. In addition to being safe, physically, humanoid robots must act in human-like form and their movements must seem natural and predictable in order to facilitate their acceptance and the comfort of their human companions. All these special needs and demands present great challenges in the development and implementation of complex control systems, as well as the building and designing of humanoid robots in terms of materials, power supplies, actuators, motors, sensors, etc.

Building humanoid robots requires complex mechanical designs in order to reproduce and mimic the features of human motions. A typical human being possesses several joints DOF. A human leg, considered with rigid toes, would have 3 DOF in the hip, 1 DOF in the knee, and a 2 DOF ankle, in total, each leg has 6 DOF of angular motion [Herman, 2007]. For each arm, considering all fingers rigid, the shoulder has 3 DOF, the elbow is a hinge of 1 DOF, the wrist 2 DOF, and a additional 1 DOF, a pivot motion of the radius rolling on the ulna, for a total of 7 DOF in the arm [Herman, 2007]. With 6 DOF for every leg and 7 DOF in each arm, in addition to 3 DOF for the head and waist, a typical human person would have in excess of 30 DOF. This does not take into consideration the DOF in the human hand, which has 4 DOF for each finger plus 5 DOF for the thumb and its over 20 DOF in total. Typical humanoid robots have in excess of 20 DOF to over 40 DOF [HONDA, 2012], [Kaneko et al., 2002], [Kim et al., 2005], [Asfour et al., 2006], [Vernon et al., 2007], [Martinez et al., 2012]. Most of these humanoid robots do not have fully articulated hands. The Shadow hand, one of the most advanced, offers 24 DOF, position sensing on every joint and pressure sensing on every muscle [Company, 2012].

Key designing decisions in humanoid robots are which materials and actuators to employ, as this would determine the weights and loads of the robots, and limit some capacities of the robot, such as speed and strength, maximum carrying payloads, and complexity of the low-level control. Most humanoid robots have employed DC motors, either brushed or brushless, but some examples can be found that implements hydraulic actuators, as Sarcos, or pneumatic actuators, as Lucy [Behnke, 2008]. DC servo motors with harmonic drive and reduction gear systems are employed for the ASIMO [Hirai et al., 1998], HRP [Akachi et al., 2005], KHR [Park et al., 2004], ARMAR [Albers et al., 2006] and TEO [Monje et al., 2011] humanoid robots. Future technological advances will allow the development of smaller, more powerful,

more efficient and less expensive actuators. Since the role of humanoid robots is one closer to humans than that of industrial robots, precision, force and speed are not as important; however while the requirement for safety and compliance take precedence. Recent approaches aim at developing humanoid robots safe for human-robot interaction exploring the use of controllable stiffness actuators like artificial muscles [Sugisaka, 2009] or series elastic actuator, used by Robonaut2 [Diftler et al., 2011]. Such compliant actuators will significantly contribute to the safe operation of robots in the close vicinity of humans [Behnke, 2008].

Another issue, that remains to be solved, is the development of adequate power supplies. Powering a humanoid robot requires big and heavy battery packages or other power supplies. Currently battery powered humanoid robots can provide no more than 30 minutes of autonomy [Hirai et al., 1998], [Martinez et al., 2012]. Research into better and more efficient technologies for power supplies is fundamental [Monje et al., 2011]. For functional humanoid robots the life and energy capacities of their batteries, or any future power supply, must be greatly improved, in terms of duration, efficiency, weight and space, heat dissipation, recharging, etc.

Humanoid robots' movements need to be done in the most natural and human-like way possible. Primary for full-body humanoid robots is the ability for biped locomotion. Motions like walking, running, going up or down stairs, which seems intuitive and simple for humans, are very complex and difficult to imitate in humanoid robots, and though great advances have been made it is a problem that is not yet fully solved due to the complexity of the non-linear dynamics that must be resolved. Most humanoid robot approaches to biped walking are based on the theory of Zero Moment Point (ZMP) [Vukobratovic and Borovac, 2004]. ZMP defines the point on the ground about which the sum of the moments of all the active forces equals zero. The bipedal robot is dynamically stable if it can guarantee that the ZMP would fall within the support polygon of all the contact points between the feet and the ground during the locomotion. Prominent humanoid robots, relying on ZMP-based control, include Honda ASIMO, which is capable of running at a pace of 6km/h. However, its gait with bent knees does not look human-like and it requires the ground to be flat and stable for walking [Behnke, 2008]. A different strategy consists of the simplification of the complex dynamics of the robot by limiting the model of the robot to a simplified form. [Kajita et al., 2001a] introduced a 3D linear inverted pendulum to model the robot dynamics of the center of mass. The other well know model for the dynamic of a biped robot is the cart-tabled model [Kajita et al., 2003]. Other approaches follow biological inspired controls and rely on central pattern generators involving non-linear oscillators [Tsuchiya et al., 2003], [Righetti and Ijspeert, 2006]. Another approach is to utilize the passive dynamics of the robot to take advantage of the swinging limb momentum for greater efficiency. It has been proved that planar walking down a slope is possible without actuators and control. These machines are able to walk on level ground. However, they cannot stand still not they can start or stop walking and are not able to change speed or direction [Behnke, 2008]. The development of balance control algorithms is fundamental for humanoid robots if they are to be as functional as humans, moving over different types of terrains and, slopes and, avoiding obstacles, etc.

The ability to handle and manipulate tools is also essential for humanoid robots to achieve their full potential and exploit their adaptability. Dexterous manipulation would not only require capable hands, but also hand-arm coordination and the coordination of two hands and the vision system [Behnke, 2008]. As mentioned above, the human hand possesses a high number of DOF, is very flexible and strong and with a high level of sensibility which makes replicating its functionalities a very challenging research goal. Researchers are working on various dexterous tasks ranging from juggling and catching balls, to performing telesurgery or pouring coffee and chopping vegetables [Pradesh, 2006]. Robonaut2 [Diftler et al., 2011], ARMAR-III [Asfour et al., 2006], and Justin [Ott et al., 2006], are among the most advanced humanoid robots in manipulation, though none of them has legs, while the performance of these robots is impressive, it stills presents limitations, like, for example, their not being able to grasp and manipulate unknown objects [Behnke, 2008].

Humanoid robots need to incorporate control systems that can deal with a broad repertoire of motions, variable speeds and constraints, and most importantly, uncertainty in the real-world environment in a fast, reactive manner [Peters et al., 2003]. To allow humanoids to move in complex environments, planning and control must focus on self-collision detection, path planning, obstacle avoidance and reaction to perturbations. Some approaches have relied on teleoperation control of the humanoid robots. A teleoperation system for controlling a humanoid robot can present advantages; the teleoperated humanoid robot can be more versatile in dealing with various tasks and environments. However the characteristics of humanoid robots present more difficulties for controlling the whole body motion of the humanoid robot from teleoperated commands. Challenges arise from the control of the many DOF of humanoid robots, satisfying severe balance constraints and the geometrical and dynamical differences between humanoid robots and humans [Hasunuma et al., 2006]. Teleoperation systems for controlling humanoid robots can be employed for various interesting scenarios, acting as proxies for humans in hazardous or dangerous tasks, the teleoperation of humanoid robots for space operations could be an important application [Pierro et al., 2009]. [Glassmire et al., 2004] presents NASA efforts at developing teleoperation systems for the Robonaut astronaut humanoid robot. In [Neo et al., 2007] a teleoperation system for whole-body motion generation, using joysticks to control a humanoid robot performing a variable set of tasks is introduced. [Stilman et al., 2008] presents the manipulation of objects with varying loads with a teleoperated system. In [Evrard et al., 2009] a teleoperation control is used for intercontinental, multimodal, wide-range telecooperation. As useful as teleoperation control can be for certain humanoid robot missions, in order to benefit from the full potential of humanoid robots. control architectures cannot rely on teleoperation since humanoid robots are expected to perform their tasks in an autonomous way. Efforts in teleopration control look for ways of providing robots with increasing levels of autonomy, going from the fully teleoperated robots towards shared control collaborative robots [Pierro et al., 2012b], [Stilman et al., 2008].

Planning control of motions and tasks is the focal point of humanoid robots control architectures. The control of humanoid robots is most often distributed in a hierarchical manner and comprises of several layers from lower joint motor control

to higher modules for path planning, collision, obstacle avoidance and stabilization control. The main task for the motion control is the generation of stable biped locomotion gaits or full-body trajectories for the humanoid in its environment. Appropriate joint motor control is essential for humanoid robots; the joint motor control problem in humanoid robots is more complex because of the high number of DOF and the possible disturbances from high vibration and external forces that can occur during robot locomotion [Kaynov et al., 2007]. Due to the complex dynamics of the mechanical structure involved in the performance of bipedal translational motions of the humanoid robots, even if each joint follows a correct and well controlled motion pattern this does not guarantee stable biped locomotions; the implementation of additional controls for stabilization are needed. A stabilization controller is proposed in [Kaynov et al., 2009] for joint-position control stabilization with a general, practical and open strategy.

At higher levels of humanoid robot motion control researchers focus on the planning of safe motions, collision and obstacle avoidance. [Harada et al., 2007a] presents a real-time gait planning of humanoid robot for force-controlled manipulation. In [Yoshida et al., 2008] a planning framework is presented for generating 3-D collision-free motions that take complex robot dynamics into account. An iterative algorithm is introduced in [Lengagne et al., 2011] for the replanning of safe motions, ensuring safety, balance and integrity of humanoid robots over the duration of the motions. A collision avoidance methods is described in [Ohashi et al., 2007]. [Guan et al., 2006] addresses the problem of humanoid robots stepping over obstacles, focussing on the planning and the feasibility analysis of motions. [Stasse et al., 2009] presents strategies for dynamically walking over large obstacles. This is only a small list since almost every work on humanoid robot control offers modules for tackling these issues.

The majority of these approaches generates the humanoid robot planned trajectories offline, thus making it impossible or challenging to respond to unforeseen events. Since replanning of new motions is a computationally heavy and time consuming process, it is therefore necessary to have control algorithms that are capable of reacting to perturbations and online adaptation. Machine learning techniques are seen as the best alternatives to offer fast, safe, adaptable control for humanoid robots. [Schaal et al., 2000] offers several locally weighted learning algorithms that have been tested successfully in real-time learning of complex robot tasks. [Atkeson et al., 2000] explores easier ways of programming behaviours in a humanoid robot employing learning from demonstration algorithms. Learning from demonstration has appeared as one way to respond to the need for intuitive control methods [Calinon et al., 2007] presents a demonstration framework for generally extracting the features of a task and generalizing the skills in a different context. [Tani et al., 2008] presents a humanoid robot learning to manipulate objects with a recurrent neural network that has a hierarchical structure. [Hwang et al., 2006] focuses on determination of optimal configuration posture for a pushing task of the humanoid robot employing simple genetic algorithm. [Kamio and Iba, 2005] has proposed an integrated technique of genetic programming (GP) and reinforcement learning (RL) to enable a real robot to adapt its actions to a real environment. Reinforcement learning offers a general framework to offer robotics true autonomy and versatility. However, applying RL to

humanoid robots systems, with its high dimensionality, remains an unsolved problem. [Peters et al., 2003] discusses different approaches of reinforcement learning for their applicability in humanoid robotics. [Stulp et al., 2010] presents a probabilistic reinforcement learning approach, derived from the framework of stochastic optimal control and path integrals, demonstrated to be able to efficiently learn humanoid motor skills which require full-body motion.

2.2.2 Sensory Perception

Sensory perception is one prominent topic of research for robotics, and one of major importance for humanoid robots. It is quite clear that, just like humans, humanoid robots need to perceive their own state and their environment for them to perform successfully. One hurdle in sensory perception would be the integration of the large set of multiple sensor modalities and the processing of this information into a reliable input to the rest of the control architecture. There is a large range of sensors that could be implemented in humanoid robots to measure many kinds of environmental variables, yet visual and auditory perception remain the most important modalities for sensory perception, together with the necessary proprioception for self-estimation. Providing humanoid robots with tactile sensors seems like a natural approach given the importance that the sense of touch has for humans. There have also been attempts to give robots the sense of smell.

For proprioception, most robot motors are equipped with encoders, relative or absolute, to measure their own joint positions; others could employ force sensors, or potentiometers. Most humanoid robots are also equipped with some type of inertial sensors to estimate the robot attitude; either accelerometers, gyroscopes, magnetometers or combinations of all three. Force-torque sensors at the wrist and ankles are also used in many humanoid robots for sensing ground reaction forces or forces at grasping and manipulating objects with the hands.

For humans, the sense of vision is the most important and versatile of all, used to quickly perceive the environment and generate functional representation of work. Providing robots with vision capabilities by means of computer vision is therefore one of the great challenges in robotic research. Great advances have been made over the years, yet computer vision is still not close to replicating the capacities and abilities of the human eye. In general, humanoid robots are equipped with two cameras in their heads, to simulate human eyes, and provide the robots with stereo vision. These cameras are used as active vision systems, allowing the robots to focus their attention towards relevant objects in their environment. Most humanoid robots are equipped with on-board computers for image interpretation. Interpreting real-world image sequences is not a solved problem, and many humanoid vision systems only work well in a simplified environment [Behnke, 2008]. Recent developments on RGB-D cameras could greatly increase humanoid robot capacities for depth perception and 3D interpretation of their world.

Providing humanoid robots with auditory perception is an important research objective, particularly for human-robot interaction where they would be expected to understand the human natural language. Auditory perception is provided by a

microphone or an array of microphones. In addition to facilitate hearing, an array of microphones can provide the capability of also identifying the source of the sound, this however can increase the difficulty of interpreting the audio signal. One major problem is the separation of the sound source of interest from other sound sources and noise. Turning the microphones towards the source of interest and beam forming in microphone arrays are means of active hearing [Behnke, 2008]. Though many speech recognition systems exist, very few are openly available. CMU Sphinx [Huggins-Daines et al., 2006], is one of the leading open source toolkits available for speech recognition. Speech recognition systems performance has been continuously getting better, even if substantial word error rates remains.

Research efforts are also made in providing humanoid robots with a sense of touch. An idea is to cover the robot with a force-sensitive skin; these robot skins are composed of a large number of spatially distributed tactile elements organized in patches, which are surface compliant structures covering large parts of a robot body [Baglini et al., 2010]. Some attempts can be found in this area. The iCub robot, for instance, is being fitted with a capacitive skin system in the fingertips and palms that enables measurement of contact [Consortium, 2012]. The sense of smell is also important for humans. A robot working collaboratively with a human that can't alert the presence of a smell relevant to the task would lack an important functionality,[Coradeschi et al., 2006]. For Ishida, the ability to recognize smells will bring robots closer to humans and provide new ways of directing navigation of autonomous robots [Ishida et al., 2005]. The first research on an artificial sensing system able to discriminate different odours was published in 1982. Since then, researchers have done extensive research on developing electronic noses [Coradeschi et al., 2006]. An electronic nose consists of an array of chemical sensors with partial specificity and a pattern-recognition system. The major problem in the development of artificial olfaction is that no sensor as versatile as odour receptor cells exists.

A key aspect for robot perception is the processing, filtering and representation of the information gathered by the multi-sensory system into manageable structures for the robot interpretation of its state and that of the environment. To provide robots with scene understanding and proper situation awareness the robots would need to build adequate representations of the environment base on the signals received form the various sensors. This is not a trivial task, and much work in this area remains to be done.

2.2.3 Human-Robot Interaction

Humanoids robots are one of the main topics in service robots investigation. Humanoid robots have many features that make them a very suitable partner in collaborative working environments. Therefore, a major focus of research is in the interaction between robots and humans, as this presents one of the main tasks which has to be achieved if we want a world where humans and robots can work together. One important motivation is the idea that the efficient techniques which evolved in our culture for human-human communication can work also for intuitive human-machine communication, since they are designed to have a similar or identical embodiment. This

includes multiple modalities like speech, facial expressions, gaze and body language [Behnke, 2008].

Much work in this area is focused on coding or training mechanisms that allow robots to pick up visual cues such as gestures and facial expressions that guide interaction. One important example of a robot built for studying interaction and socializing with humans is the robot Kismet. Kismet is designed to perceive a variety of natural social cues from visual and auditory channels, and to deliver social signals to the human caregiver through gaze direction, facial expression, body posture, and vocalizations [Breazeal, 2001]. Movable eyes, head, and chest communicate where the robot focuses its attention. When the robot looks at the interaction partner, the partner feels addressed.

Robots present different models of interaction, from direct control or teleoperation of the robot, to robots with an autonomous and independent behaviour and ambient intelligence. The optimal ideal for the human-robot interaction is for the human operator to accept and recognize the robot system, just as one more partner in a working team composed of multiple human and robotic agents. A human-robot team can present many advantages. Robots can be used in order to cover human limitations or to assist them in numerous tasks. Human-Robot Collaboration is an important topic of research in this area. Since robots are expected to live with us and share our environment, studying the possible means of collaboration is of major interest. One example of a humanoid robot working in collaboration with humans is NASA JSC's Robonaut [Johnston and Rabe, 2006]. Another important platform in the field of the human-robot collaboration is the HRP-2 robot from Kawada industries [Kaneko et al., 2004b]. This robot is able to manipulate objects under the orders of a human [Neo et al., 2008] and also to assemble a panel by cooperating with a human [Harada et al., 2007b]. Robots can also be of great assistance for a human worker at a construction scenario, taking most of the workload in a transportation or an assembly task and performing more risky activities. A robot partner can also perform precise or sensitive tasks in an industrial or factory scenario.

Humanoid robots that allow the users to perform tasks in the real world by switching between continuous teleoperation and autonomous operation have been proposed by Yokoi in [Yokoi et al., 2008]. In order for collaboration to be meaningful, it is important for the human operator to see the robot as not just a tool but as a colleague in a team [Siino et al., 2008]. In the work of [Fong et al., 2002] a model for collaboration is proposed in which, instead of a supervisor dictating to a subordinate, the human and the robot engage in dialogue to exchange ideas, to ask questions, and to resolve differences. In [Pierro et al., 2012b] a shared control concept is proposed, the collaboration focuses on a human-robot interaction were the human is not just a supervisor directing the robot, but a partner in which the robot can look for assistance. By sharing control according to each one best capabilities the advantages of a human-humanoid partnership can be exploited.

Another issue to take into account for the design and development of humanoid robots is the phenomenon known as the "uncanny valley". In 1970 Professor Mori introduced the term "uncanny valley" to explain the hypothesized eerie response a person would have at encountering a robot trying to resemble a human shape but

failing to replicate a lifelike appearance [Mori et al., 2012]. The "uncanny valley" describes the relation between human likeness of a machine and affinity towards it, the relationship behaves as a monotonically increasing function until a point in which, a failure in correspondence between the human-like appearances and its artificial performance becomes unsettling and produces a steep drop in the affinity creating the local minimum named the "uncanny valley" [Gee et al., 2005]. When motion is considered, the effect of unsettling eeriness is heightened. Even though the extent of Professor Mori's hypothesis has not been fully validated, the concept of the "uncanny valley" is generally accepted and applied in areas like computer-graphics, animation, films and robotics. For humanoid robots research, where it is expected for humans and robots to generate close relationships and interact and collaborate together, it is very important to consider the level of acceptance the humanoid robots would have by the general population. Humanoid robots design must take into consideration the "uncanny valley", and try its best to prevent it or overcome it. This requires design guidelines for both the appearance and the motion performance of robots. Discussion on the "uncanny valley" often focuses mainly on the appearance dimensionality, forgetting the problem of replicating human-like motion and aiming to achieve a lesser similarity in physical appearance. Even Professor Mori recommends taking the first peak as the goal, aiming at a moderate human likeness with a considerable sense of affinity [Mori et al., 2012]. This, however, omits an important part of the problem as both aspects are relevant to humanoid robotics and necessary for human-robot interaction. Consistency between appearance and motion play a large part in acceptance when they cannot be reviewed independently [Gee et al., 2005]. The continuous developments in robotics should move forward both dimensions, appearance and motion performance, retaining the acceptance. It is, therefore, necessary that developments in humanoid robots go hand in hand with appearance and performance to generate better human-robot interactions. Humanoid robots must not only simulate our embodiment and try to mimic our physical appearance they must also replicate humans motions and try to resemble our behaviour.

Human-robot interaction is an open and very active field, involving several disciplines and a large set of topics. Important progress has been made in this field, and several working robotic systems can be found allowing for multimodal human-robot interaction [Stiefelhagen et al., 2007], [Gorostiza et al., 2006], providing robots with speech recognition [Gomez et al., 2012b], object attention localization and identification [Haasch et al., 2005], gesture recognition interfaces [Bertsch and Hafner, 2009], [Stiefelhagen et al., 2004], face detection [Bueno et al., 2012], teaching and learning interactions [Schmidt-Rohr et al., 2010], [Kronander and Billard, 2012], natural dialogue processing [Alonso-Martin and Salichs, 2011], user interfaces [Chen et al., 2007], etc. Still, the most relevant topics in human-robot interaction can be considered unresolved. Major work on human-robot interaction focuses on assistive and health care robots, lifelike robots, remote robots, robot companions, long term interaction, multi-modal interaction, awareness and monitoring, robot-team learning and collaboration, software architectures for HRI, user studies and experiments on interaction, collaboration and acceptance.

2.2.4 Intelligent Behaviour

In addition to robust and efficient motor controls, for allowing humanoid robots to generate smooth, natural human-like motions, comprehensive multi-sensory perceptual systems and appropriate strategies for meaningful and engaging human-robot interactions, humanoid robots need to present behaviours with a minimum level of autonomy and intelligence. Development of intelligent systems is a long term goal in the fields of robotics research, artificial intelligence and cognitive science. To truly exploit humanoid robots full potential it would be necessary to provide them with an intelligence that is similar to that of humans. This presents an even greater challenge than endowing humanoids with the ability to replicate human-like motions or simulate human interactions. Particularly since the process of human intelligence is one that is not fully understood, in which many competing ideas can be found and where no generally accepted theory of intelligence exists that satisfies every group.

The study of intelligence is a relevant topic of research in many fields, such as psychology, philosophy, neurobiology, education, cognitive science, and artificial intelligence, each one with its own views on what constitutes intelligent agents and intelligent behaviours. Despite all this debate, which has encompassed many years and a wide field of research, no one single standard definition of intelligence has emerged. However, from the many definitions that have been proposed, it is not hard to find some strong similarities and a common ground between them on which behaviours are to be expected from an agent in order to be considered intelligent.

Reviewing the various definitions, as the basis of intelligence the abilities to learn and acquire knowledge, to make judgements and decisions based on reason, to effectively adapt to the environment, to succeed in solving problems and achieving goals can all be found. Intelligence is defined in [American-Heritage, 2006] as the ability to acquire, understand and use knowledge. Seeing it from the view point of psychology [Gardner, 1993], intelligence is the ability to solve problems, or to create products, that are valued within one or more cultural settings. Also [Anastasi, 1992] intelligence is a composite of several functions, a combination of abilities required for survival and advancement within a particular culture. In [Albus, 1991] intelligence is defined as the ability of a system to act appropriately in an uncertain environment, where appropriate action is understood as that which increases the chances of success for the behavioural goal and subgoals. In a more computational intelligence frame, for [Poole et al., 1998] an intelligent agent is one that is flexible to changing environments and changing goals, learns from experience, and makes appropriate choices given perceptual limitations and finite computation. For [Legg and Hutter, 2006] intelligence measures an agent ability to achieve goals in a wide range of environments.

A survey of definitions of intelligence collected in [Legg and Hutter, 2007], leads them to construe intelligence as a property of agents in their interaction with the environment, that are related to the agent ability to succeed in respect to some goal, depending on the agent capacity to adapt to different objectives and environments. As a summary from the various views of intelligence it is possible now for an identification of the key attributes required for considering the behaviour of an agent as intelligent. An intelligent agent can be thought of as one that features the abilities to

learn and acquire knowledge based on its experience, the capacity to understand or comprehend current relevant features in the environment, to exhibit situation awareness, the capacity for reasoning, to compute or deduce the course of actions to follow, the forming of conclusions and value judgements. Also, the ability to adapt, be it of itself, its objectives or its environment, according to every situation and objective. Finally, a fundamental ability to succeed, i.e., to survive, in the wider possible range of environments, to efficiently accomplish one's goals. In order to consider that an agent has displayed intelligent behaviour, it must be required that it presented a successful performance, that is, it has achieved its goal and objectives effectively, regardless of any form of unplanned disturbance that could have been encountered in the environment. An environment, that could be arbitrarily complex in nature and that could be dynamically changing and unpredictable, from which the agent does not necessarily have any prior knowledge. Finally, the agent behaviour must be replicable over time and across different situations.

For humanoid robots to become intelligent agents, and present intelligent behaviours it is necessary to have replicable models of intelligence. [Sternberg, 2000] discusses some relevant, contemporary, models of human intelligence. In the triarchic theory of intelligence there are three interacting factors of intelligence: an internal aspect, consisting of information processing skills guiding intelligent behaviour; an external aspect, the practical ability to adapt a particular environment to match one own skills; and an experimental factor, involving the ability to capitalize on experiences in processing novel or unfamiliar information [Sternberg, 2000]. The theory of multiple intelligences of Gardner focuses on domains of intelligence. There are eight fairly independent, equally important types of intelligence, which are based on abilities valued within different cultures. The intelligences described are, visual-spatial, verbal-linguistic, bodily-kinaesthetic, logical-mathematical, interpersonal, musical, intrapersonal and naturalistic intelligence. The models reviewed above present contrasting differences, however, one common aspect between them is that they all value adaptability of cognitive processing as an important aspect of intelligence.

[Albus, 1991] has proposed a model that integrates knowledge from research in both natural and artificial systems. The model consists of a hierarchical system architecture. Different levels of intelligence in the hierarchy can be achieved, depending on the computational power of the system and the sophistication of its processing algorithms for various functionalities, such as, world modelling, behaviour generation, value judgement, and global communication, and the information and values the system has stored in its memory. A minimal level of intelligent requires at least the ability to sense the environment, make decisions and take actions. Higher levels of intelligence may include the ability to recognize objects and events, to represent knowledge in a world model and to reason about and plan for the future. More elevated forms of intelligence provide the capacity to perceive and understand, to choose wisely, and to act successfully under a large variety of circumstances [Albus, 1991]. The current humanoid robots may only be around the minimum and mid-levels of intelligence. As developments of systems, architectures and algorithms continue to advance the intelligent capabilities of humanoid robots will increase. Humanoid robots need to reach a functional level of intelligence that allows them to function properly

interacting with humans and the environment, even if perhaps the ultimate levels of intelligence could turn out to be out of reach, and creating robots that replicate the total scope of human intelligence may prove impossible. Humanoid robots need to achieve a sufficiently high level in the hierarchy in order to be considered by their human partners as intelligent, namely, a sensing, acting system that perceives, learns, plans, and succeeds in achieving its goals in the world. This is a major challenge in humanoid robot research.

As a minimal requirement, an intelligent robot system or agent needs to be thought of as perceiving its environment through sensors and acting upon that environment through actuators [Russell and Norvig, 2010]. Sensors and actuators represent the inputs and outputs from intelligent systems. Its ability to rationalize and make decisions in the middle of the perception-action determines its level of intelligence. To achieve a higher level it is needed to integrate perception, reason, knowledge, emotion, and behaviour. The model in [Albus, 1991], identifies four elemental systems of intelligence: sensory processing, world modelling, behaviour generation, and value judgement. Similarly, from the field of cognitive science and intelligent agents, the importance of the different functions of cognition were identified in a robotic system point of view as perception, learning, motor control, reasoning, problem solving, goal orientation, knowledge representation and communication [Langley et al., 2009]. The phenomena of intelligence, however, require more than a set of disconnected elements. Intelligence requires an interconnecting system architecture that enables the various system elements to interact and communicate with each other in intimate and sophisticated ways [Albus, 1991].

Figure 2.1 illustrates a model of an architecture for an intelligent agent based on the general principles stated above. For an intelligent agent, with the needs of a humanoid robot, it is necessary to have systems for perception, action, interaction, reasoning, world knowledge and learning. The perception, interaction and action systems are the outward components of the architecture, in charge of dealing with, and affecting the environment. Perception systems process sensor information to acquire and maintain internal models of the world. World knowledge systems store and maintain memory data gathered and processed from the reasoning and learning systems. Learning systems must learn appropriate behaviours from the perception and the interaction data, and also store them in memory. The reasoning system interacts with the action system so as to pursue behavioural goals, it also may interact with the perception, world knowledge and learning system to reason about the environment and the task, the space, time, geometry, etc., and to formulate or select action plans.

Perception establishes and maintains correspondence between the internal model and the external real world. Sensory processing is the mechanism for perception. The sensory input data form multiple ranges of sensors are processed and integrated into a consistent unified perception of the state of the world. Sensory processing algorithms compute distance, shape, orientation, surface characteristics, physical and dynamical attributes of objects and regions of space.

Action is a process of the systems actuators that move, exert forces, move manipulators and, handle tools. It represents the means by which the agent produces an effect on the world, interacting and altering its environment in order to achieve its

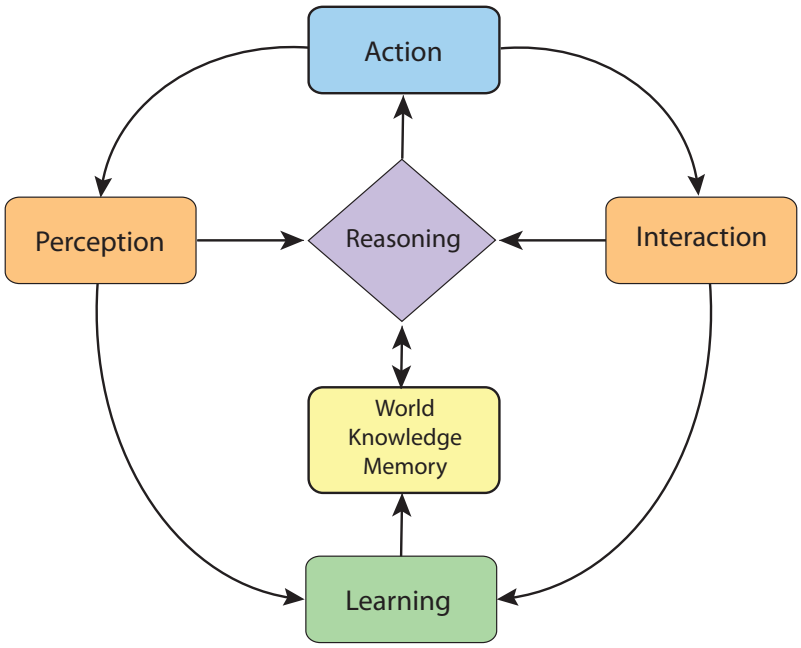


Fig. 2.1: *Model for an architecture of intelligent agents. For an intelligent agent, with the needs of a humanoid robot, it is necessary to have systems for perception, action, interaction, reasoning, world knowledge and learning.*

goals. Task and goal behaviours are decomposed into actions.

Reasoning systems evaluate the observed state of the world and the predicted results of hypothesized plans. They provide the criteria for making intelligent decisions. Reasoning systems compute costs, risks, and benefits both of plans and actions, the desirability, attractiveness, and uncertainty of objects and events. Reasoning systems select goals, and plans and executes tasks. Tasks are recursively decomposed into subtasks, and subtasks are sequenced so as to achieve goals. Logic has been a favoured tool of artificial intelligence theorist, practical intelligent systems have tended to use techniques such as rules, cases and neural networks. However there is a growing use of probabilistic reasoning in intelligent systems [Thagard, 2005].

World modelling estimates of the state of the world. The world model thus can provide answers to requests for information about the present, past, and probable future states of the world. It includes databases of knowledge about the world, and database management systems that store and retrieve information. The world model is the representation of the external world, it provides the reasoning system information necessary to make decisions. It maintains world knowledge, keeping it current and consistent.

Learning is required to acquire and develop task knowledge. Learning systems work the mechanisms for storing knowledge about the external world and for acquiring skills and knowledge of how to act; the algorithms for learning and extracting important features of task actions in order to build intelligent behaviours. The learning system consolidates short-term memory into long-term memory, and exhibiting

altered behaviour because of what was remembered.

For humanoid robot systems to present the intelligent behaviours that would be expected from them, from the ability to sense the environment, make decisions and take actions, to recognize objects and events, represent knowledge, reason and plan for the future, and act successfully under a large variety of circumstances, the intelligent robot architectures developed to control humanoid robots must implement, all or a subset of, these systems.

2.3 Robot Planner-Based Architectures

As stated throughout this chapter, for the development of functional humanoid robots, that work together with humans, helping them in achieving everyday tasks, robotic agents need to become intelligent, they need to be endowed with control mechanism that enables them to produce intelligent behaviours. That is, they must be capable of performing successfully complex tasks in a dynamic environment. Autonomous robotic systems need to be able to perform a wide range of functions, in order to work in complex evolving environments, seeking for the successful accomplishment of their goals. Robots would need to present many different skills, and implement several competing behaviours. Among the desirable abilities that autonomous robots should present is the ability to perceive and understand, the ability to act and interact, the ability to learn, the ability to reason and acquire knowledge, the ability to plan actions and goals, and make decisions, the ability to adapt to tasks and/or environmental changes, etc. All these functionalities present many challenges that the control system architectures of autonomous robots need to address.

Intelligent agents, at their most basic definition, can be thought of as something that perceives and acts in an environment [Russell and Norvig, 2010], one in which actions are well thought of, logically inferred, and reasoned from the information, gathered and processed, from the environment. In this simplified construe for an intelligent agent, it is easy to identify three basic building blocks for a control systems architecture: a perception module, that senses the external world; a reasoning module, that processes the collected information from the environment and reasons about the plans of actions to accomplish goals; and an action module that translates the planned commands into physical actions in the world. Other modules could be thought of such as a learning module, or a memory module, a knowledge module, an adaptation module, etc. However when considering the basic definition of agents, as systems that sense and act in an environment, in pursuit of their own objectives and goals, in order to build intelligent robot systems, efforts could well be first concentrated on the fundamental modules for perception, reason, and action. Front this point of view, the classical approach from AI emerged, focusing on decomposing the control systems for autonomous robots into the three functional elements forming the sense-plan-act cycle. The sensing system's function is to translate raw sensor input into a world model. The planning system's work is to take the goals and the world model and generate plans that achieve these goals. The execution system's job is to generate the actions prescribed by the plan [Gat, 1997].

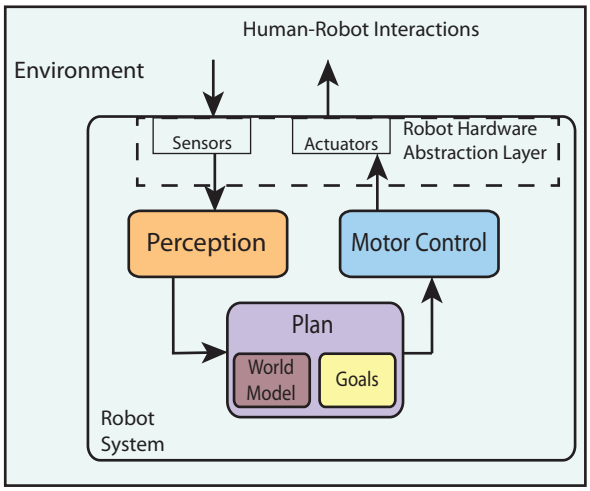


Fig. 2.2: *Hierarchical planning or deliberative architectures follows the Sense-Plan-Act cycle from classical AI approaches. These architectures are form by three components. Information flows unidirectionally from sensors to perception, to planning, to motor control, to actuators. The system intelligences resides on the planner, with world model and system goals, produces appropriate plans of action for the robot.*

Classical approaches from the field of AI focus their effort on building intelligent systems on the symbolic representation of physical world entities, which could be combined, computed or operated upon. And in the belief that intelligent agents could be formulated as information processing systems, taking a representation of the world as input and outputting appropriate sets of actions. The development of planning or deliberative strategies that generate the sequences of tasks to accomplish robot goals is the central aspect of the classical AI control architectures. Figure 2.2 presents the general planning or deliberative architectures. The architectures are formed by three components, from the sense-plan-act hierarchical problem solving paradigm: a perception component for sensing the environment; a plan component, with world model and system goals, for producing a plan of action; a motor control component for translating the planned actions to proper motor commands. The control scheme of information flows unidirectionally and linearly from sensors to perception, to the world model, to planning, to motor control, to actuators.

Hierarchical planning or deliberative architectures use a high level structured approach, relying on a traditional top-down strategy centred on planning for decomposing the robot goal tasks, having an explicit symbolic model of the world, and in which decisions are made via logical reasoning [Wooldridge and Jennings, 1995]. Work on hierarchical planner-based or deliberative architectures has focused on the planning of long-term actions for achieving a set of basic goals. The intelligence of the system architecture, is said to live, in the planner or the programmer, not the execution mechanism [Gat, 1997]. As represented in Figure 2.2, the robot architecture follows a strict sequence of distinct stages during execution: first the robot senses the world,

then plans the next move and acts accordingly. In the sense stage of perception, the robot would acquire information about its environment from its available sensors. The world model data structure is created using this information. The world model are symbolic descriptions, comprising priori information of the environment, with the information collected from the robot sensors and any other cognitive knowledge that the specified task could need to assist the robot [Murphy, 2000]. The planning stage takes a symbolic description of both the world and goal states, it then attempts to find a sequence of actions that will achieve the goal [Wooldridge and Jennings, 1995], several different automated planning algorithms could be employed. When the final goal comprised of complex situations and world states, the planner breaks the goal into sub goals and account for each one in turn to achieve the final goal. The act stage represents the execution of actuator commands that are generated according to the sequence orders from the planning stage. After the robot's actuators finish off the planned task, the cycle begins again and continues until the goal is reached.

The most representative methodology that has been built, based on the planner-based paradigm, was STRIPS [Fikes and Nilsson, 1971]. The STRIPS method takes a symbolic description of both the initial state and a desired goal state, and a set of action conditions and operations, which characterise the pre and post-conditions that are associated with the various actions. Constructing the world model was imperative and the action to be chosen at a certain point was selected from a descriptive table called the difference table. For the planning stage of each cycle a difference-evaluator would measure the difference between the goal state and the current state, enabling the planner to choose the best corresponding commands from the difference table, that would minimize the difference, and pass them on to the actuators. The STRIPS method was used for the robot Shakey by the Stanford Research Institute [Nilsson, 1984]. The STRIPS planning algorithm was very simple, and proved to be ineffective on problems of even moderate complexity. Hierarchical and non-linear planning were proposed in efforts to raise the efficiency of the planner, but remained somewhat weak while working in a system with time constraints [Nilsson, 2007].

In spite of these difficulties, various attempts to construct an agent planner component can be found: the Integrated Planning, Execution and Monitoring (IPEM) system is based on a sophisticated non-linear planner [Ambros-Ingerson and Steel, 1988]. The AUTODRIVE system has planning agents operating in a highly dynamic environment [Wood, 1993]. The PHOENIX system includes planner-based agents that operate in the domain of simulated forest fire management [Cohen et al., 1989]. The Belief-Desired-Intention model has also been of relevance for deliberative planner architectures, the model call for a rational agent must allow for means-end reasoning, for the weighing of competing alternatives, and for interactions between these two forms of reasoning. One example is the Intelligent Resource-bounded Machine Architecture (IRMA) [Bratman et al., 1988]. It presents a high-level specification of the practical-reasoning component of an architecture for a resource-bounded rational agent. This architecture has four key symbolic data structures: a plan library, and explicit representations of beliefs, desires, and intentions. Another examples is GRATE* [Jennings, 1993], a layered architecture in which the mental attitudes of beliefs, desires, intentions and joint intentions, guide the behaviour of an agent.

Numerous examples of hierarchical control systems can be found [Arkin, 1989b] addresses the task of navigational path-planning, which provides the robot with a path guaranteed to be free of collisions with any modelled obstacles. [Albus, 1997], [Meystel, 1988], promote the idea of top-down, hierarchical controllers, each executing a sense-plan-act feedback loop. The NASREM architecture [Albus et al., 1987], is a strict hierarchical framework for task decomposition, perception and world modelling. [Meystel, 1986] proposed a theory for a nested hierarchical controller (NHC), enhancing the planner by decomposing it into three distinct components, namely, the mission planner, navigator and the pilot. NHC looks to give a more receptive response to changes in the environment by having the sensors continuously updating the world model even while the actuators were carrying out the commands.

The symbolic approaches to intelligent agents, embodied by the planner-based or deliberative architectures presented numerous shortcomings. Among the biggest issues that hinder the hierarchical planner-based paradigm with time were the transduction problem, translating the real world into an accurate, adequate symbolic description, the close world assumption, and the representation or frame problem, of how to symbolically represent information about complex real-world entities in time for the results to be useful [Wooldridge and Jennings, 1995]. The required assumption for the close world model, that the robot obtains all the information from the environment that it needs, presents significant problems since the planner cannot keep track of all the changes in the environment in a continuous manner. The frame problem refers to the inability to represent all the world information that was needed by the robot in a computationally viable method. Consequently, addressing uncertainty in the event of a bigger problem was too tedious and was not worth the effort [De Silva and Ekanayake, 2008].

Planning and world modelling turned out to be very difficult problems, and open-loop plan execution was clearly inadequate in the face of environmental uncertainty and unpredictability [Gat, 1997]. Uncertainty in sensing and action, and changes in the environment, can require frequent replanning, the cost of which may be prohibitive for complex systems [Mataric, 1997]. The planner-based or deliberative architecture has presented its strengths and its weakness: they can handle complex tasks by breaking them into more manageable sub tasks, specifying the current and future activities and constraints [Simmons, 1994]. They allow for explicitly formulating task and goals of the system and estimating the quality of the agent's performance [Mataric, 1997]. And they can produce optimal, domain-independent solutions. However, they generally fail to address uncertainty, and are therefore unfit to operate in changing environments, since they are unable to re-plan their actions quickly enough. Planner-based approaches have high computational costs, making their performance poor when there is a need for frequent replanning.

Researches in the 80s began to feel unsatisfied with the poor results obtained from planning-based architectures and started to look for other alternative techniques. The problems of the planner-based or deliberative architectures led to questioning the viability of the whole paradigm, and to the development of what are generally known as reactive architectures [Wooldridge and Jennings, 1995]. Many researchers begin a shift of viewpoints away from the traditional AI symbolic representation,

abandoning the requirement for a central world model, and the idea that intelligence is a computational process that takes an input and produces an output [Brooks, 1996].

2.4 Robot Behaviour-Based Architectures

Earlier attempts to develop intelligent agents, following the classic AI approaches for symbolic reasoning, failed to produce adequate levels of intelligent behaviours for robots. Although the deliberative thinking approach proved successful for certain tasks, for planning operations, by real autonomous agents in complex dynamic environments, the obtained results have been poor [Maes, 1991b]. Therefore, many researchers saw the need for developing different types of architectures and mechanism for replicating intelligence. Attention turned away from the symbolic and AI and attempts to model behaviour through explicit representations and abstract reasoning. Instead, the ideas that real intelligence is situated in the world, and that intelligence behaviours can only emerge as a result of an embodied agent interaction with the environment, gained preference.

This novel AI approach was based on the hypothesis that to build intelligent systems it is necessary to have their representations grounded in the physical world [Brooks, 1990]. Instead of focusing on the design of systems capable of intelligent thinking, the emphasis changed to creating agents that could act intelligently. Researchers took inspiration from biological and ethological advances, studying animal behaviour and coordination. Approaches centred on the reflexive behaviours of animals as stimulus-response mappings, responses to a particular sensory input are directly wired with an action response which is carried out without any higher cognitive involvement [De Silva and Ekanayake, 2008]. The behaviour-based or reactive paradigm is founded on the building of behaviours, direct couplings of sensory inputs to a pattern of actions that in turn carries out a specific task [Murphy, 2000].

Central to the definition of a reactive architecture is that it does not include any kind of central symbolic world model, and does not use complex symbolic reasoning [Wooldridge and Jennings, 1995]. Decisions are based on real-time information from sensors, and the global system behaviour emerges from the interactions of local behaviours with the environment. Behaviour-based or reactive architectures implementations are founded on the constant-time run-time direct encodings of the appropriate actions for each input state, these mappings rely on a direct coupling between sensing and action, and fast feedback from the environment [Mataric, 1997]. This allows reactive autonomous agents to respond faster, and in a somewhat more natural manner, and for achieving real-time performances. Reactive systems maintain no internal models and perform no search. A generally simple functional mapping between stimuli and appropriate responses is employed, usually in the form of a look-up, this being on a table, a set of action rules, a simple circuit, a vector field, or a connectionist network [Mataric, 1997].

Figure 2.3 shows a generic representation for a reactive behaviour-based architecture. The behaviour-based paradigm presents a direct coupling between perception and action. A collection of preprogrammed condition-action pairs is embedded into

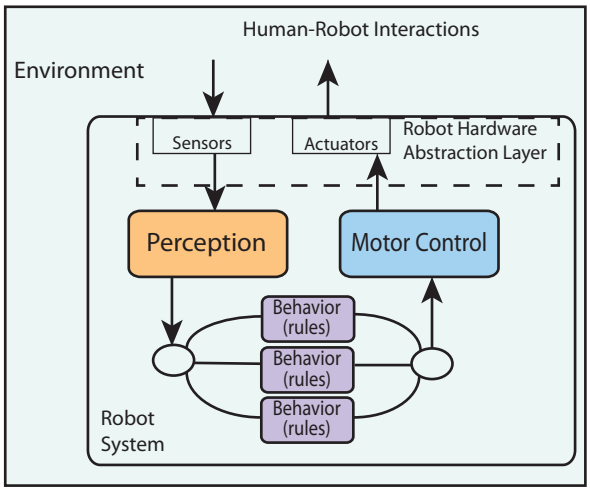


Fig. 2.3: *The robot behaviour-based architectures presents a direct coupling between the perception and action modules. A collection of preprogrammed behaviours, typically consisting of a collection of rules, is embedded into the agent control strategy. Behaviour-based systems maintain no internal models and perform no state space searchs. The reactive behaviour-based autonomous agents can respond faster and achieve real-time performances.*

the agent control strategy. The behaviours in reactive or behaviour-based architectures typically consist of a collection of rules, taking inputs from sensors or other behaviours in the system, and sending outputs to the effectors, or other behaviours [Nicolescu and Mataric, 2002]. The system intelligent behaviours emerge from the bottom-up, instead of the top-down approach of the planner-based models.

[Brooks, 1986] introduced the subsumption architecture as an instance of a behaviour-based approach to building robots that operate in the real world. The subsumption architecture is the best known effort from the behaviour-based paradigm for agent intelligence: it enables a tight connection of perception to action, embedding robots concretely in the world. The subsumption architecture presents a hierarchy of task-accomplishing behaviours, built on layering progressively more complex task-specific competencies, each one connected to its own sensory inputs. The behaviours are decomposed in a vertical arrangement, on top of each other, based on task achieving behaviours in their order of sophistication. Thus, the most basic "survival" behaviours, such as avoiding objects are at the lowest layer while more complex 'cognitive' behaviours, such as reasoning about the behaviour of objects, are at the higher levels [Brooks, 1986]. The higher layers would have the ability to replace or subsume the behaviours of the lower layers. Each layer of behaviour competences receives its own sensorial input system, and is supposed to execute independently, with each behaviour being unaware of what is happening in the other layers. Since each layer executes a dedicated behaviour, this avoids the need for it to know the complete scenario it is trying to solve, which in turn simplifies each layer's computational needs and allows it to abandon the need for internal models of representation. The subsumption

architecture does not come without its shortcomings: including the competencies of lower layers into higher levels leads to a waste of resources; the subsumption architecture precludes the layers from passing information between themselves, and fails to take into account any advantage a planning module can introduce to the system [De Silva and Ekanayake, 2008].

[Chapman and Agre, 1987] also began to explore alternatives to the AI planning paradigm proposing that an efficient agent architecture could be based on the idea of 'running arguments'. The idea is that as most decisions are routine, tasks, once learned, can be accomplished in a routine way, they can be encoded into a low-level structure, which only needs periodic updating. The approach was illustrated by the PENGI system [Agre and Chapman, 1987]. PENGI is a simulated computer game, with the central character controlled using a scheme such as that outlined above.

[Maes, 1991a] developed an agent architecture in which an agent is defined as a set of competence modules loosely resembling the behaviours of the subsumption architecture. Each module is specified in terms of pre and post-conditions and an activation level. The higher the activation level of a module, the higher the probability that this module will influence the agents behaviour. Once specified, a set of competence modules is compiled into a spreading activation network, in which the modules pre- and post-conditions define the ways they are linked to one another. Similarities between the agent network architecture and neural network architectures exist. Perhaps the key difference is in the difficulty of saying what the meaning of a node in the net is. In a neural net it only has a meaning in the context of the net itself. Since the competence modules are defined in declarative terms, it is very much easier to say what their meaning is [Wooldridge and Jennings, 1995].

[Mataric, 1992] implemented an architecture that integrates a map representation into a reactive, subsumption-based mobile robot. It presented a fully integrated reactive system removing the distinction between the control program and the map. Programmed with a collection of simple, incrementally designed behaviours, the robot performs collision-free navigation, dynamic landmark detection, map construction and maintenance, and path planning. [Nicolescu and Mataric, 2002] presents an approach for implementing hierarchical task representations concepts into behaviour-based systems. It describes a Hierarchical Abstract Behaviour Architecture that allows for the representation and execution of complex, sequential, hierarchically structured tasks within a behaviour-based framework. The architecture introduces the notion of abstract behaviours and enables the re-usability of behaviours across different tasks. [Nicolescu and Mataric, 2003] uses a behaviour-based approach as an underlying control architecture in which time-extended actions that achieve or maintain a particular goal are grouped for representing robot skills behaviours. The behaviours are built from two components: one related to perception (Abstract behaviour), the other to actions (Primitive behaviour). This architecture provides a simple and natural way of representing robot tasks in the form of behaviour networks [Nicolescu and Mataric, 2002]. The architecture is used to endow the robots with the ability to convey their intentions by acting upon their environment and to learning complex tasks from observing a demonstration by a teacher [Nicolescu and Mataric, 2003].

[Lenser et al., 2001] describes a highly modular hierarchical behaviour-based con-

trol system for robots. The architecture is designed to present features for easy addition and removal of behaviours, easy to program hierarchical structure, ability to execute non-conflicting behaviours in parallel, a unique reward based combinator to arbitrate amongst competing behaviours such as to maximize reward. [Balch and Arkin, 1998] presents and evaluates reactive behaviours implementing formations in multirobot teams. The formation behaviours are integrated with other navigational behaviours to enable a robotic team to reach navigational goals, avoid hazards and simultaneously remain in formation. Another approach in the reactive paradigm is the methodology known as the Potential Fields Methodology (PFM). In PFMs each behaviour is represented as a vector, thus this methodology is inherently regarded to be confined to the navigational robots. Behaviours are combined in vector summation to produce the emergent behaviour. These behaviours are assumed to exert on the robot in the form of force fields, the robot is assumed to be a particle entering into the force field and the behaviour of the robot is the path it takes as a result of the multiple potential fields [De Silva and Ekanayake, 2008]. Many other navigational systems using reactive control have been developed. These include Paytons reflexive behaviours [Payton, 1986], Kadonoffs arbitration strategies [Moravec et al., 1986], Arkins motor schemas [Arkin, 1989a].

The behaviour-based or reactive architectures lead to a significant advance in the development of autonomous robots, although not everything was positive. The behaviour-based approaches presented greatly improved performances in robot navigation and obstacle avoidance. Reactive architectures showed great flexibility and adaptability, and were ideally suited to performing in dynamic and unpredictable environments. Also, these approaches were robust, simple and computationally tractable. However, they also have some drawbacks: behaviours-based or reactive architectures do not include explicitly the achievement of a goal in their behaviour description; plans and goals are to emerge from the robot interaction with the environment; the approaches only include 'local' information, collected from the environment; they present a short-term view, with no long-term planning capabilities, and offered limited applicability. One of the most important characteristics of the behaviour-based paradigm is their abandonment of the abstract symbolic representation, this presented their advantages but also limits the possibility to employ them at higher levels task. The purely reactive approaches achieved great efficiency at run-time, but their limited representational power results in a lack of run-time flexibility [Mataric, 1997].

Another shortcoming of the reactive behaviour-based paradigm is in the complexity of the interaction dynamics between the behaviours and the environment, and between the behaviours themselves. This hampers the debugging and understanding of the robots emerging behaviour, it also hinders the development and implementation of a large number of behaviours. Also, the behaviour-based architecture prevents the automatic reusability of behaviours across different tasks and thus, the automatic generation of behaviours. Even though the behaviours themselves are usually reused and accumulated into behaviour libraries, the behaviour-based systems are to be manually programmed, involving the customized redesign of behaviours in accordance with the specifics of any new task [Nicolescu and Mataric, 2002].

The reactive behaviour-based paradigm emerged as a response to the problems

presented in the planner-based architectures. The reactive approaches offered a solution for the rigidities encountered within the hierarchical paradigm, and their limitations in performing in dynamic environments. The behaviour-based paradigm proved to be more than satisfactory in robots executing simple tasks, and performs remarkably well and fast in collision free navigation tasks, and in working within the environment. However, a need for planning and higher representations emerged in order to deal with more complex tasks. Many roboticists turned to new ways of combining the planning process with the reactive behaviour of robots and new breed of architectures under the name Hybrid Deliberative/Reactive paradigm was born [De Silva and Ekanayake, 2008].

2.5 Robot Hybrid Architectures

For some time researchers trying to develop intelligent robotic agents explored their implementations in two competing paradigms, the deliberative planner-based architecture, centred on classical AI approaches in the symbolic representation of the world and the deliberative planning of robots' actions, and the reactive behaviour-based architecture, that focused on alternative approaches generating appropriate behaviours to react to real-time robot interactions with their environments. Both the reactive and deliberative based architectures had their advantages and presented early satisfactory results. Nevertheless, each approach displayed various shortcomings.

The deliberative planner-based approaches, dominant through the first decades of AI, tried to build intelligent agents by means of symbolic reasoning and representations of the world that were capable of generating deliberative plans of actions, after reasoning in relation their goals in the world. However, these approaches proved unsuccessful in dealing with dynamic changing environments, where the computational speed for planning was slower than the environment rate of change. Two major problems hindered the progress of the deliberative planner-based architectures. First, as mentioned, the world may change during computation of the planning phase in a way that invalidates the resulting plan. Second, unexpected outcomes or errors during the execution of the planned steps can cause the subsequent steps in the plan to be executed in an inappropriate context [Gat, 1997].

The reactive behaviour-based approaches appear as a reaction to the failures of classical AI approaches. An attempt was made at building intelligent agents that could perform in real-time, situated in the real world. The idea of symbolic reasoning and of maintaining a world model was abandoned in favour of a direct coupling between the sensing and the action, extracting information directly from the world, as its best model [Brooks, 1990]. Though the approach achieved dramatic early success, its limitations and drawbacks were quickly apparent. Behaviour-based approaches offered limited applicability, often confined to low-level tasks. One significant problem was the lack of modularity: upper layers interfere with the lower layers' functionalities so that they cannot be designed independently. Also, the complexity of the interaction dynamics between the behaviours and the environment, and between the behaviours themselves, in cases where a large number of behaviours are implemented,

makes the understanding of the robot's emerging behaviour quite difficult to predict and design, therefore hindering their development and implementation. Also, by eliminating internal state representations, the reactive approach avoided the problem of maintaining that state, but ran headlong into the problem of extracting reliable information about the world through sensors [Gat, 1997].

Intelligent robot agents, in order to be successfully employed, working alongside human partners, need to address three main challenges: adapt quickly to changes in the environment; understand high level human commands; be engaging for people [Stoytchev and Arkin, 2001]. Traditionally, the first challenge has been adequately addressed by the behaviour-based reactive controllers. The second challenge can well be addressed by using a deliberative planner-based approach. The hybrid deliberative/reactive architectures naturally emerged as attempts to bridge these two approaches and use the strengths of each other in reducing their respective shortcomings. The hybrid deliberative/reactive paradigm advocates for the use of the advantageous aspects of both the behaviour-based and the planner-based approaches, combining them to produce a new architecture that can deal with more complex scenarios. In practice, this means the integration of the planning aspect of the hierarchical deliberative paradigm with the rapid execution capabilities of the reactive paradigm [De Silva and Ekanayake, 2008].

The hybrid architectures idea was to attempt a compromise between the purely reactive and deliberative approaches and integrate both of them as subsystems of the architecture. Generally, the reactive system, capable of performing behaviours at faster speeds, is given precedence over the deliberative system. The hybrid deliberative/reactive architectures usually adopt a reactive system at the low-level control, where modules are closer to sensors and actuators, and a planner-based approach at the high-level, for higher decision making [Mataric, 1997]. Therefore, the motion control loops are closed at the lower levels producing different behaviours. At the same time, decisions based on internal models and plans can be reached, modifying lower behaviours variables. The reactive behaviour system makes short term decisions in local areas, and the deliberative planning system makes mid and long term decisions in global areas. This type of structure leads naturally to the idea of a layered architecture. The architecture is arranged into a hierarchy of control subsystems, with the lower levels closer to the physical world, sensors and actuators, and in which the higher levels deal with information at increasing levels of abstraction.

In general, hybrid deliberative/reactive architectures usually divide the control system into a layered structure. This architecture structure to control intelligent robots needs the integration of three separate components: a reactive feedback mechanism for controlling low level primitive activities; a deliberative planning system for decision-making computations; and a sequencing system that controls the interactions between the other two components. This three layered structure, or similar configurations, can be found in the majority of hybrid architecture approaches, such as the ATLANTIS architecture [Gat, 1992], the SSS architecture [Connell, 1992], and the 3T architecture [Bonasso et al., 1995].

Figure 2.4 represents a general hybrid architecture divided into three functional layers: a behaviour control layer, for reactive feedback control of the robot low-level

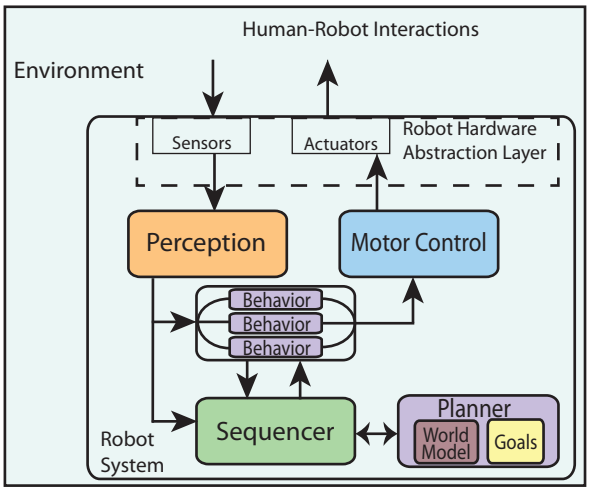


Fig. 2.4: *Hybrid deliberative/reactive architectures usually divide the control system into layered structures with three main components: a behaviour control layer, for reactive feedback low level control; a sequence execution layer, for controlling the execution of behaviours in a planned sequence; and a planning layer for time-consuming deliberative computations, planning high level goals and maintaining the world model.*

behaviours; a sequence execution layer, whose tasks are the activation and inhibition of the other layers and the control of the execution of behaviours in a sequence order to carry out their task; and a planning layer for performing time-consuming deliberative computations, planning high level goals and maintaining the world model. These components run as separate asynchronous computational processes. Usually, algorithms in the three-layer architectures are organized according to the role of their internal state representation. Sensor-based algorithms, that contain no state representation, inhabit the control behaviour layer component. Algorithms that maintain memory of the past inhabit the sequencer layer. Algorithms that make predictions about the future inhabit the planner deliberator layer [Gat, 1997].

In the ATLANTIS architecture [Gat, 1992], these layers are called the controller, the sequencer, and the deliberator. For the 3T architecture [Bonasso, 1991], the components are called the skill layer, the sequencing layer, and the planning layer, respectively. The behaviour control layer is responsible for the control of primitive activities, that is, simple reactive sensorimotor processes. Usually it contains libraries of primitive behaviours or skills, the activation of which is determined by an external input to the control layer, the sequencer or certain sensory inputs. The algorithms that go into the behaviour control layer need to follow some important constraints [Gat, 1997]. The computing cycles must be of constant-bounded time and space complexity, small enough to afford stable closed loop control for the desired behaviour. The algorithms should detect failure to perform the adequate functions, allowing higher components of the system to take corrective actions for failure recovery. Internal states in the controller should have limited time life, and should not introduce

discontinuities. It is the responsibility of the sequencer to manage transitions between regimes of continuous operation.

The sequence execution layer is responsible for controlling the sequences of primitive behaviours and deliberative computations. The job of the sequencer is selecting which primitive behaviour the behaviour control layer should be activated at a given time, and to supply parameters for the behaviours. By controlling the activation and deactivation of behaviours at appropriate moments the robot can be made to perform useful tasks. The sequence execution layer initiates and terminates primitive behaviours by activating and deactivating sets of modules in the behaviour control layer. In addition, it can send parameters to the behaviour control layer, and monitor the progress of the active behaviours [Gat, 1992]. The control of sequences is required to handle several difficult situations. The sequencer must be able to deal effectively with unexpected failures. Also, if behaviours must be interrupted, then the sequencer must ensure that the interrupted activity is properly terminated, and the system must ensure that two activities which interfere with each other are not enabled simultaneously.

The planning layer is responsible for the performance of time-consuming computational tasks such as decision making, planning generation and maintaining world models. The planning layer performs under the control of the sequencer which initiates and terminates its processes. The planning layer often runs as a concurrently computational process in one or more separate control threads. Several behaviour transitions could occur between the time a deliberative algorithm is invoked and the time it produces a result, with no restrictions on the computational structure except the sequencer's ability to initiate and terminate its functions. The planning layer interaction with the rest of the system usually follows one of three broad methods [De Silva and Ekanayake, 2008]. The planning layer provides lower layers directly with the information on which to act. It can produce plans for the sequencer to execute, or it can respond to specific queries from the sequencer. The planning layer works prior to or jointly with the behaviour layer, updating the robots behavioural parameters or changing the world state. Coupled planning and behavioural layers occur concurrently making plans and reactive execution.

Many examples of implementations of hybrid architectures can be found. The Autonomous Robot Architecture, AuRA [Arkin and Mackenzie, 1994], is one of the earliest approaches attempting the integration of hierarchical planning and reactive behaviours mechanisms. In AuRa two major planning and execution components are present: a behaviour reactive component, schema controller, coupled with a hierarchical planning component system that is formed by a mission planner, at the highest level of the architecture, concerned with establishing high level goals, a spatial reasoner, that construct sequences of paths using stored knowledge, and a plan sequencer, that translates each path generated by the spatial reasoner into a set of motor behaviours for execution [Arkin and Balch, 1997].

Under the hybrid paradigm the most popular hybrid deliberative/reactive architectures are the three-layered architectures. [Gat, 1992] introduced the ATLANTIS architecture as an early example, it was first implemented on robot Robby in 1990. ATLANTIS is a heterogeneous asynchronous architecture for controlling mobile robots

based on the activity model of action. It has three layers, namely the controller, the sequencer, and the deliberator. The controller is a reactive control of primitive activities with no decision-making computations. The sequencer is a special-purpose system which controls initiation and termination of the primitive activities, and the time-consuming deliberative computations, performed in the deliberator, like planning and world modelling [Gat, 1992].

Another architecture, similar in structure to ATLANTIS is SSS, Servo-Subsumption-Symbolic, which combines a servo-control layer, a "subsumption" layer, and a symbolic layer. Unlike ATLANTIS in the SSS architecture the middle layer is based on the subsumption architecture [Brooks, 1986], and the symbolic layer is inside the control loop. The 3T architecture [Bonasso et al., 1995], separates the general robot intelligence problem into three interacting tiers or layers. First, a skill layer where a dynamically reprogrammable set of behaviour reactive skills is coordinated by the skill manager. A sequencing layer, that activates and deactivates the sets of skills to accomplish specific tasks, this use the Reactive Action Packages (RAPs) system. And the planning layer with deliberative planning capabilities that reason about the goals, resources and time constraints.

[Ferguson, 1991] developed the TOURINGMACHINES hybrid agent architecture. It consists of components for perception and action in direct interaction with the environment, and three independent control layers concurrently executing process under a control framework. The reactive layer, implemented in the style of the subsumption architecture [Brooks, 1986], as a set of situation-action rules, generates courses of action in response to quick changing events. The planning layer constructs plans and selects actions to execute in order to achieve the agents goals. The modelling layer contains symbolic representations of the cognitive state of other entities corresponding to the environment. The three layers are embedded in a control framework that mediates between the layers, and deals with conflicting action proposals from the different layers.

INTERRAP [Müller and Pischel, 1994], is a layered architecture, with each successive layer representing a higher level of abstraction. The INTERRAP architecture further subdivides these layers into two vertical ones: the first containing layers of knowledge bases and, the other containing control components. The lower-layer is a world interface control component that manages the interface between the agent and its environment, and thus , deals with acting, communicating, and perception as an abstraction layer for the rest of the structure. The next layer is the behaviour-based component that implements and controls the basic reactive capability of the agent. Above the behaviour-based component is the plan-based component layer which contains a planner that is able to generate single-agent plans in response to requests from the behaviour-based layer. The knowledge component at this layer contains a set of plans, including a plan library. The highest layer for the INTERRAP architecture is the cooperation layer, which is able to generate joint plans that satisfy the goals of a number of agents. These plans are generated in response to requests from the plan-based component. The knowledge component at this layer contains a social plan library, from which the cooperation layer can select plans for elaboration.

The Task Control Architecture (TCA) [Simmons, 1994], provides an integrated

set of control constructs for implementing deliberative and reactive robot behaviours. The control constructs mean to facilitate the development of modular and evolutionary systems, they are used to integrate and coordinate planning, perception, and execution, and to incrementally improve the efficiency and robustness of the robot systems. The TCA control constructs include support for distributed inter-process communication, task decomposition, management and allocation of resources, exception handling and execution monitoring. A TCA robot system consists of a number of robot-specific modules, and a central control module, which is common to all systems that use TCA. The modules communicate by passing coarse-grained messages to the central control, which then routes messages to the appropriate modules that would handle them. In this structured control approach, the deliberative components handle normal situations and the reactive behaviours, which are explicitly constrained as to when and how they are activated, handle exceptional situations. The TCA architecture has been used in over a half-dozen robot systems, including a six-legged robot that autonomously walks over rugged terrain [Simmons, 1994].

The Action-Deliberative (AD) architecture [Malfaz et al., 2011], was designed trying to avoid rigidity in the planning-sequencing-acting paradigm that can be found in the three layer architectures. It is composed of only two levels: one for deliberative activities and a second one for automatic activities. The sequencing processes are distributed between the deliberative and automatic levels, providing more flexibility to the hybrid architecture. The AD architecture has been further enhanced by also adding a biologically inspired decision making system [Malfaz et al., 2011].

The hybrid deliberative/reactive architectures present some advantages over both purely deliberative and purely reactive architectures, mostly in shortening their respective drawbacks. Hybrid deliberative/reactive architectures combine the rapid real-time responses and ability to adapt to quickly changing environments provided by behaviour-based systems with the higher level reasoning, planning and decision making capabilities of planner-based approaches, enabling them to perform in a better wider range of tasks, coupling the strengths of both paradigms, providing more successfully acting intelligent agents. However, these types of architectures are not devoid of problems and critics. Hybrid deliberative/reactive architectures tend mostly to be very specific, application dependent, and lacking general design guiding methodologies. A potential difficulty with hybrid architectures is that while their structures are well-motivated from a design point of view, it is not clear that they are motivated by any deep theory [Wooldridge and Jennings, 1995]. The lack of good theoretical models for agent architectures prevents the true understanding of the mechanism from which the systems works, difficulting the generalization and reproduction of their results for varying domains. However, psychological and neurophysiological evidence can be found for the co-existence of two distinct planning systems in humans [Norman et al., 1980], supporting this approach as a potentially effective methodology for robotic systems.

2.6 Robot Cognitive Architectures

When developing robot systems with human like embodiments and functional capacities and behaviours that are similar to that of humans, such as those needed for the humanoid robots that have been discussed in the above sections, it becomes clear that different mechanisms are necessary to replicate the complex level of skills and operations presented by humans than those employed to simulate simpler behaviours. In order to deal with the richer set of intricate abilities that are expected from humanoid agents, the intelligent architectures need to provide new structures and models. The deliberative planning and behaviour-based approaches on their own seem to be insufficient to deal with the inherent complexities related to representation and modelling of reasoning in the human mind. In [Brooks, 1996], a needed shift in viewpoint is discussed for cases when the focus of research goes to building humanoid robots, designed to present a full human level intelligence, that must be capable of operating and interacting in the world in much the same way a human agent would. Here, approaches are led to different architectural decompositions from those considered from both the traditional AI planning approaches and the behaviour-based approaches, largely implemented for mobile robots. These decompositions are motivated by fundamentally different concerns at many different levels of analysis, requiring to deal with a number of important issues, such as, bodily form, motivation, coherence, self adaptation, inspiration from the brain, etc.

In dealing with these concerns, which arise when thinking about building robots with human level intelligence and functionality, the agents' architecture structural paradigm shifts from the production and emergence of intelligent behaviours as a system output towards a viewpoint whose main pursuit is in the development of intelligence thinking at the system internal processing. These approaches are centred on the mechanism that allows for the generation of thought and the interior workings of cognition. This calls for an organization of intelligence in terms of cognitive models. Dealing with these issues, and the organization and interaction of cognitive components, is one important aspect for the development of cognitive architectures and cognitive robotics.

The deliberative planning approaches, while applicable for state-space search and scheduling systems, proved to be unfit to operate in changing environments which would be required of humanoid robots. The reactive and behaviour-based approaches presented great performances in robot navigation and obstacle avoidance, and in dynamic and unpredictable environments, yet their true applicability is limited to low level behaviours and they would not be suited to dealing with the complexities of behaviours present in humanoid robots. The hybrid approaches have attempted to combine the strengths of deliberative and reactive approaches and can be readily employed as the system architecture for several robotic platforms. However, they ignore issues of perception, learning, world model, and different mechanisms that would be necessary to replicate the complex level of skills and operations presented by humans and lack of good theoretical models. Research in cognitive architectures constitute a solid basis for building intelligent systemdecompositionss centred on the configuration and interaction of cognitive modules dealing with the various mechanism

and abilities that constitute the various process of human intelligence.

The study of the mind, intelligence, and the working processes of intelligent thought are the competencies of cognitive science. Research in cognitive science stands at the intersection of various fields, embracing philosophy, psychology, artificial intelligence, neuroscience, linguistics, and anthropology. A central point for the development of cognitive theories lies in studying the nature of knowledge. The most agreed view by cognitive scientists is that knowledge in the mind consists of mental representations, and that intelligent behaviour and thought are the resultant products of manipulating, reasoning and operating upon these internal representations. Much debate in the field is focused on the class and nature of these knowledge representations, on the representational mechanisms for acquisition, organization, and utilization of knowledge, and on whether the internal representations are even needed at all or whether or not another paradigm is required.

The central task of a knowledge representation is capturing the complexity of the real world [Davis et al., 1993]. Representations thus perform as functional abstractions of the perceived environment, encoding an agents' knowledge of its world, objects, actions, events, etc., into manageable internal structures. An agent system, having useful representations, can therefore operate on them by abstracting itself beyond the world. The knowledge representation constitutes an important property for the design of a cognitive agent architecture, along with the organization and use of the represented knowledge, and the mechanism supported for the acquisition and revision of the knowledge in the representation [Langley et al., 2009].

The dominant analogy in cognitive sciences has been to compare the mind, and the brain, to computers, where thinking can be understood as computational procedures. The metaphor assumes that the mind has mental representations analogous to data structures in a computer program, and computational procedures similar to programmed algorithms [Thagard, 2005]. The computational hypothesis has been the most expanded and dominant theoretical and experimental theory of mind developed so far. Other theories have also arisen to challenge the major premises of the computational-representational understanding of mind (CRUM) thesis as the most suitable one for cognition. Connectionist models have proposed novel ideas expanding theoretical frame of cognitive science about representation and computation that uses neurons and their connections. The connectionist analogy is that mental phenomena can be described by interconnected networks of simple and often uniform units, where neuron patterns and network connections can be compared to data structures, and neuron firing and spread activation is analogous for algorithms [Thagard, 2005]. More recent approaches in cognitive science have taken a growing interest in dynamical systems. The dynamical systems metaphor promotes thinking about the underlying forces, vector fields, from which observed patterns of behaviours emerge [Schöner, 2008]. In this view, the brain is thought of as a dynamic physical system and the processes in the mind can be described by differences and differential equations. The driving idea motivating the dynamical systems approach is that cognitive processes, contrary to the computational hypothesis of discrete representational operations, must unfold continuously and simultaneously in real time. Therefore, a cognitive system would not be a sequential manipulation of discrete static representa-

tional structures, but rather, a structure of mutually and simultaneously influencing change [van Gelder and Port, 1995].

The traditional commitment of cognitive sciences to a computational-representational view of the mind, where intelligence is a problem of symbol manipulation, has faced increasing challenges and scepticism over the years, in which, the very central notion of internal representation has been questioned. This challenges have been explicitly stated by [van Gelder, 1995], and are also present in works by [Thelen and Smith, 2007], [Wheeler et al., 1994], [Haselager et al., 2003], etc. The representational approach, according to this hypothesis, is viewed as incapable of producing timely suitable cognitive responses, and as detrimental and counterproductive for developing intelligent physical agents. The critical distinction is not between representational and non-representational solutions but among an action-neutral form of internal representation, requiring disembodied symbolic computational processing, and action-oriented forms, in which a behavioural response is embedded into the representation itself [Clark, 2004]. A necessary emphasis is placed on the close link of cognition with the sensory and motor processes and the environments in which these are immersed. Models of cognition must be embodied processes that capture the unfolding of cognition in time and the associated sensory and motor surfaces embedded in the environment in which cognitive phenomena takes place [Schöner, 2008]. The embodied cognition view maintains that there is more to cognition than just mental representations. Humans' problem solving ability involves "intensive cooperation" between internal representation, computations and interactions with the environment. The claim is not an outright rejection of the legitimacy of representations, however in order to be valid, for embedded cognition, the representations are to be limited, physically grounded to the environment and oriented toward the specific needs of the given agent [Anderson, 2003]. Development of cognitive robotics will relied on off-lined modelling and operation on internal representations, and emulation mechanism for environmentally coupled responses [Clark and Grush, 1999].

The ideas of knowledge representation and reasoning are central for high level cognitive robotic control [Levesque and Lakemeyer, 2008]. The development of robot systems endowed with a human like embodiment, functional capacities and behaviours, capable of replicating the complex level of skills and operations presented by humans, would require complex control architectures, which allowed them to display cognitive abilities. Cognitive architectures specifies the underlying infrastructure for an intelligent system. The representational formalisms by which an agent would encode its knowledge are a central aspect of a cognitive architecture [Langley et al., 2009]. As a definition, let's take the one provided by [Albus and Barbera, 2005]: a cognitive architecture is an organizational structure, of knowledge representations and functional structures, set for enabling the modelling of cognitive phenomena. A cognitive architecture would attempt to provide the basic primitive computational resources needed for developing intelligent systems. Among their basic properties are those related to memory, representation, processing, organization, performance, interaction, reasoning, and learning. Research on cognitive architectures is a very important topic since it supports a central goal of artificial intelligence, cognitive science, and robotics, the creation and understanding of agents built for supporting the same capabilities as

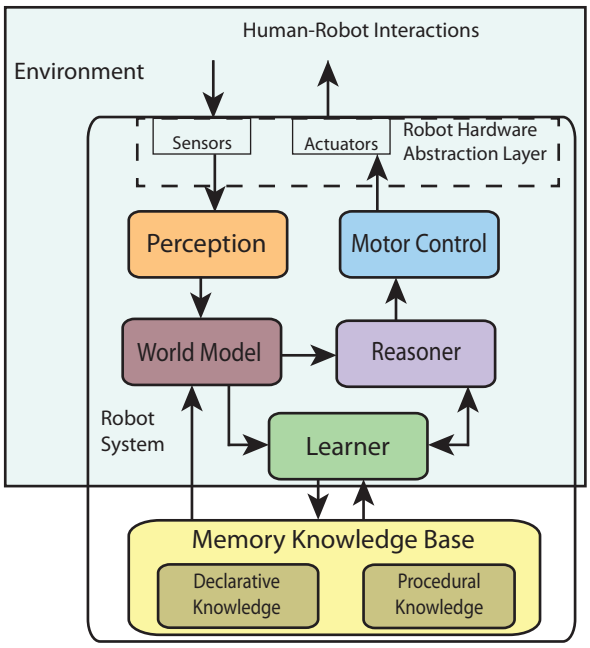


Fig. 2.5: *A Robot Cognitive Architecture must support several capabilities. Perception and motor abilities must be present for a cognitive robot agent acting in an environment. The capacity for modelling and characterizing the environment is central to the performance of all other cognitive functions. Memory storage is also fundamental for a cognitive robot agent. The ability for learning, adapting and improving the agent skills is vital for the performance of a cognitive agent over time. Reasoning and decision making abilities guide agent action, choosing, from the perceived, stored, and learned knowledge, appropriate set of skills and proper behaviours to execute.*

humans [Langley et al., 2009].

The Cognitive architecture function is to provide a comprehensive initial framework for the modelling and understanding of cognitive phenomena, in a variety of task domains, [Sun, 2009]. The architecture design must specify overall structures, essential divisions of modules and their interrelationships, basic representations, essential algorithms and a variety of other aspects. The various attempts at developing cognitive architectures can differ in the assumptions they make, and the design decisions they take about how to manage these aspects. A cognitive architecture can support several capabilities, and can differ variedly in their set of abilities. Perception and recognition, decision making, memory, and learning are the most central abilities an architecture must support to cover the range of human-level intelligence. Other relevant abilities are those of problem solving and planning, prediction, reasoning, communication and action execution [Langley et al., 2009].

Figure 2.5 represents a general model for a robot cognitive architecture. Here various interlinked models are present for supporting perception, motor control, world

modelling, memory, reasoning, and learning abilities. A cognitive robot agent, embedded in an environment requires modules for perception and motor control of its actions with the world. A comprehensive model of the agents environment, allowing for the agent situatedness and understanding of the current state of the world is a necessary prerequisite for practically every other cognitive faculty the agent could display. A memory module for knowledge storage is of central importance for a cognitive, adaptive, intelligent agent. Stored knowledge could be of declarative or procedural nature. Modules for reasoning or decision making guide all motor activity, based on the perceive, store, and learned knowledge of the agent. The module for reasoning would choose appropriate sets of actions and proper behaviours to execute. Supporting modules for learning, and improving, the agents skill set, is a vital part of a cognitive architecture in order to guarantee an agents' success over time.

An intelligent agent exists inside an external environment that it must sense, perceive, and interpret. Multiple sensor modalities could be implemented by the agent. Perception involves integrating results from the different modalities into a model of the environment which could be used by other cognitive processes. The perception must go beyond perceiving isolated objects or events to interpret the broader environmental situation, and compose a large model of the current environment.

Intelligent agents require the ability to recognize situations as instances of known or familiar patterns, and categorize such objects, situations, or events to known concepts. To support recognition and categorization, a cognitive architecture must provide some way to represent patterns and situations in memory [Langley et al., 2009].

Decision making abilities to select from alternatives is an important ability required for an intelligent agent. A cognitive architecture, in order to support decision making, must possess a way to represent alternative choices or actions, and also offer a process of selection between these alternatives.

A cognitive architecture requires mechanisms that draw inferences using its knowledge structures. Reasoning lets an agent augment its knowledge state, drawing conclusions from beliefs and assumptions that the agent already holds. The cognitive agent can engage in various forms of reasoning such as, deductive reasoning, inductive reasoning, adductive inference, as well as the architectures afforded by it. Cognitive architectures are essentially models of human reasoning [Russell and Norvig, 2010].

Storing, and retrieving, an agent cognitive process in memory is an important ability that crosses all other cognitive capacities of the agent. In order to 'remember' an agent cognitive activity, the architecture must encode and store the cognitive structures that are generated during the agent's activity. Memory must store and index this knowledge, and be able to retrieve it when needed. Cognitive architectures most often distinguish between a short-term memory, holding information relevant to current environment models, and long-term memory storing knowledge capture by the agent over periods of its actions.

Cognitive architectures must incorporate some way from which to learn, and improve, their cognitive capacities. Learning involves processing, and generalizing, memory cognitive structures to improve the capabilities of the agent, beyond specific beliefs and events. The data on which learning operates may come from all sources supported by the architecture, including observation of another agent, problem-solving

behaviour, perception and categorization, prediction, reasoning, skills and execution policies. A cognitive architecture should also be able to learn from instruction and experience.

Problem solving and planning abilities are necessary in order to generate plans and achieve an agents goals in several situations. Intelligent agents operating in dynamic environments must often modify existing plans in response to unanticipated changes [Langley et al., 2009]. To support these abilities, the cognitive architecture must be capable of representing the planned actions, as ordered set of activities, and expected results. It should also be able to generate plans and solution from components available from its memory, or learning.

Cognitive agents can benefit from the ability to predict future situations. This requires the architecture to provide mechanisms capable of predicting future situations using present knowledge structures. Prediction requires a model of the environment and of the effects an action has on it [Langley et al., 2009].

Communication is another important ability for cognitive architectures to support since a cognitive agent interacts with other agents and the transfer of knowledge from one agent to another is a possible occurrence. Cognitive architectures should support mechanisms for transforming knowledge into the form and medium through which it will be communicated [Langley et al., 2009]. Agents can communicate about perceptions and actions, plans, inferences, decisions made, predictions and anomalies, etc. Building cognitive architectures facilitates the interaction between humans and intelligent systems because of similarities in cognitive abilities [Sun, 2009]. Increasing the cognitive capacities of a robotic system is an important task in order to achieve a meaningful and natural interaction and collaboration in a human-robot team.

The cognitive architectures must allow for the execution of skills and actions in the environment. The architecture must be able to represent and store motor skills that enable the agents activity. Cognitive architectures should present the flexibility to support a behavioural range, as can humans, from autonomous open-loop behaviours, to reactive closed-loop behaviours.

In the field of Artificial Intelligence and Cognitive Systems there are various works on the development of cognitive architectures to model cognitive processes and functionalities of humans. Among the better known architectures there is Soar [Laird et al., 1987], ACT-R [Anderson et al., 2004], PRODIGY [Veloso et al., 1995], EPIC [Kieras and Meyer, 1997], ICARUS [Langley and Cummings, 2004], CLARION [Sun et al., 2001], etc.

The Soar (State Operator And Result) [Laird et al., 1987], cognitive architecture has been under continuous development since the early 1980s. The architecture is based on the theoretical framework of knowledge-based systems seen as an approximation to physical symbol systems [Duch et al., 2008]. Soar stores its knowledge in the form of production rules, which are in turn organized in terms of operators that act in the problem space. The basic deliberative acts of the system are performed by the operators, with knowledge used to dynamically determine their selection and application [Langley et al., 2009]. In Soar, tasks are formulated as goal achieving attempts. The primary learning mechanism in Soar is *chunking*. Chunking occurs when one or more results are produced in a subgoal. The chunk actions are based on

the result, and theirs conditions are based on the relevant aspects of the goal above the subgoal. Soar has multiple learning mechanisms: chunking and reinforcement learning acquire procedural knowledge, whereas episodic and semantic learning acquire their own corresponding types of declarative knowledge [Langley et al., 2009]. Researchers have used Soar architecture to develop a variety of sophisticated agents that have demonstrated several high-level cognitive functions [Duch et al., 2008].

ACT-R (Adaptive Control of Thought-Rational) [Anderson et al., 2004], architecture is primarily concerned with modelling human behaviour. The aim is to build systems that perform the whole space of humans cognitive tasks and describe mechanisms' underlying perception, thinking and action [Duch et al., 2008]. The ACT-R architecture is organized into a set of modules, including sensory modules for visual processing, motor modules for action, an intentional module for goals, and a declarative module for long-term declarative knowledge. Each module processes different types of information and has its own associated buffer to hold *chunks* of declarative structures, taken together these buffers comprise the architecture short-term memory [Langley et al., 2009]. ACT-R employs a top-down learning approach to adapt to the structure of the environment [Duch et al., 2008]. Productions or chunks are matched to perceptions and facts, mediated by activation levels of objects. There execution is made to affect the environment or alter declarative memory. The architecture operates by matching productions on perceptions and facts, mediated by the real-value activation levels of objects, and executing them to affect the environment or alter declarative memory. Learning in ACT-R involves creating new facts and productions, as well as updating base activations and utilities associated with these structures. The ACT-R architecture has been applied in intelligent tutoring systems, psychological studies, including aspects of memory, attention, reasoning, problem solving, etc., and to control mobile robots that interact with humans [Langley et al., 2009].

ICARUS [Langley and Cummings, 2004], defines an integrated cognitive architecture for physical agents where two distinct forms of knowledge are stored. Concepts, containing knowledge of general classes of objects and relationships, and skills specifying knowledge about ways of doing things. The architecture includes a number of modules: a perceptual system, a planning system, an execution system, and several memory systems [Duch et al., 2008]. The ICARUS interpreter operates on a recognize-act cycle. Conceptual memory directs bottom-up, percept-driven inference with the process continuing until ICARUS infers all deductively implied beliefs. Skill memory controls top-down, goal-driven selection of actions, starting from a top-level goal: it finds a path downward through the skill hierarchy when a path terminates in a primitive skill with executable actions; the architecture applies these actions to affect the environment [Langley et al., 2009]. ICARUS is able to learn new concepts incrementally, in an efficient way, by constructing feature trees that the system can comprehend [Duch et al., 2008]. ICARUS architecture has been used to develop agents for a number of domains involving a combination of inference, execution, problem solving, and learning. Ongoing work aims to link ICARUS to physical robots that carry out joint activities with humans [Langley et al., 2009].

PRODIGY [Veloso et al., 1995], incorporates two kinds of knowledge structures, domain rules, which encode the conditions under which actions have certain effects

and control rules, which specify the conditions under which the architecture should select, reject, or prefer a given operator. PRODIGY performs searches through a problem space to achieve one or more goals, relying on means-ends analysis, selecting an operator that reduces differences between the current state and the goal. If control knowledge is absent, the architecture makes a choice at random and pursues a depth-first means-ends search with backtracking [Langley et al., 2009]. Research in PRODIGY framework has focuses mainly on problem solving and planning issues. However, PRODIGY has also formed the basis for a mobile robot with interleaved planning and execution and accepted asynchronous requests from users [Langley et al., 2009].

CLARION (Connectionist Learning with Adaptive Rule Induction ON-line), is an integrative architecture [Sun et al., 2001], it consists of four distinct subsystems: action-centered subsystem (ACS), non-action-centered subsystem (NCS), motivational subsystem (MS), and metacognitive subsystem (MCS). Each of these interacting subsystems consists of two levels of representation. CLARION architecture incorporates a distinction between explicit (symbolic) and implicit (sub-symbolic) processes and captures the interactions between the two [Duch et al., 2008]. In general, for each subsystem, the top level encodes explicit knowledge and the bottom level encodes implicit knowledge [Sun, 2009]. The role of the ACS module is to control and regulate the agent actions, whether they are external physical movements or for internal mental operations. The role of the NCS module is to maintain the general system knowledge, either implicit or explicit. The role of the MS module is to provide underlying motivations for perception, action, and cognition. The role of MCS module is to monitor, direct and alter the operations of the other three modules. CLARION cognitive architecture has seen applications to multi-agent social simulations [Sun, 2009].

EPIC, (Executive Process Interactive Control) [Kieras and Meyer, 1997], aims at capturing human perceptual, cognitive and motor activities through several interconnected processors working in parallel, and to build models of human-computer interaction for practical purposes [Duch et al., 2008]. The architecture encodes long-term knowledge as production rules, and a set of perceptual (visual, auditory, tactile) and motor processors. Research on EPIC has included a strong emphasis on achieving quantitative fits to human behavior, especially in tasks that involve interacting with complex devices [Langley et al., 2009].

Polyscheme [Cassimatis et al., 2004], cognitive architecture integrates multiple methods for representations, reasoning, and problem solving [Duch et al., 2008]. Each representation has a specialist associated module, modelling a different aspect of the world, it supports forward inference, subgoaling, and other basic operations, which are matched against the shared dynamic memory with elements grounded in perception and action [Langley et al., 2009]. The architecture could be used for abstract reasoning and also for common sense physical reasoning in robots. The PolyScheme architecture makes a stronger semantic commitment than most other architectures: it encodes all structures within a basic set of relations of time, space, events, identity, causality, and belief [Langley et al., 2009]. Polyscheme architectures has been used to model infant reasoning, including object identity, events, causality, spatial relations

[Duch et al., 2008].

IBCA (Integrated Biologically-based Cognitive Architecture), is a biologically inspired cognitive architecture [O'Reilly et al., 1998], it imitates automatic and distributed notions of information processing in the brain. The architecture contemplates three modules inspired by the role of three regions in the brain, posterior cortex (PC), frontal cortex (FC), and hippocampus (HC) [Duch et al., 2008]. The PC module focuses on sensory-motor as well as multi-modal, hierarchical processing, assuming overlapping, distributed localist organizations. In the FC module, working memory units are isolated from one another, contributing combinatorially, in a non-overlapping, recurrent localist organization. The HC module utilizes a sparse, conjunctive globalist organization, in which units contribute interactively to a given representation. In the IBCA framework, the underlying regularities of the world and sensory-motor activities, are captured by employing slow integrative learning, in the PC and FC modules, that blends many individual experiences. The HC module adds a fast learning retaining and discriminating over he individual experiences. Cooperation between HC and FC/PC reflects the complementary learning paradigms in the brain [Duch et al., 2008].

RCS (Real-time Control System) [Albus, 1997], is a cognitive architecture, originally designed for the sensory-interactive goal-directed control of laboratory manipulators. It has evolved over three decades into real-time control architecture for intelligent machine tools, factory automation systems, and intelligent autonomous vehicles [Albus and Barbera, 2005]. The RCS architecture consists of a multi-layered hierarchy of computational modules, operating in parallel, containing elements of sensory processing (SP), examining the current state, world modelling (WM), predicting future states, value judgment (VJ), selecting among alternatives, behaviour generation (BG), carrying out tasks, and a knowledge database (KD). The Knowledge representation is heterogeneous, including frames, rules, images, and maps [Langley et al., 2009]. At the lower levels, goal-seeking reactive behaviours are generated. At higher levels, decision making, planning, and deliberative behaviour takes place [Albus and Barbera, 2005]. The higher level modules influence, in a top down manner, the lower level modules, which in turn pass information back up.

Other approaches to cognitive architectures or frameworks includes, PRS (Procedural Reasoning System) [Ingrand et al., 1992], a well know agent architecture, based on the belief-desire-intention paradigm. PRS includes a plan library, of partially-elaborated plans called knowledge areas, as well as explicit symbolic representations of beliefs, desires, and intentions [Wooldridge and Jennings, 1995]. The framework stores the hierarchical procedures, effects, and ordered steps that invoke sub procedures. Among dynamic structures includes, agent belief about the environment, desired goals to achieve, and planned intentions of the agent. At each control cycle, PRS architecture decides on whether to continue executing its current intention or to select a new intention to pursue [Langley et al., 2009]. PRS has been evaluated in a simulation of maintenance procedures, as well as other domains [Wooldridge and Jennings, 1995]. SULTAN (Simultaneous User Learning and TAsk executioN) [Balaguer et al., 2011], offers a framework for an intelligent service robotic system that can be capable of physical and cognitive collaboration. The SULTAN

concept sets the problem in a user-task-object domain, aimed at solving the challenge of how an agent can robustly perform a set of tasks for different users in different environments. In SULTAN the learning process is based on hierarchical Bayesian networks build on the base of the Bayesian approach to cognitive system [Balaguer et al., 2011]. A model of the user is maintained by SULTAN learning module, and the representation of the physical interaction tasks is concurrently refined keeping explicit account of user learning. The framework allows the augmentation of personal capabilities, its main focus is the creation of a human+robot binomial in which physical and cognitive collaboration is achieved as a whole with potential applications for assistive robotics [Balaguer et al., 2011]. ISAC cognitive architecture [Kawamura et al., 2008], developed for the humanoid robot ISAC, is a multi-agents architecture, based on the IMA [Pack et al., 1997]. The ISAC cognitive architecture provides three control loops for cognitive control of robots: Reactive, Routine and Deliberative. It relies on the parallel operation of several cognitive agents, such as a Perceptual Agent, an Action Agent, a Self Agent, a Central Executive Agent, a Goal Agent. Also three memory components are implemented in the architecture, including: Working Memory System (WMS), Short Term Sensory Memory (STM), Long Term Memory (LTM) [Tan, 2012]. Work on ISAC focused on human-robot interaction and development of cognitive control for humanoid robots [Pack et al., 1997].

Efforts in cognitive architectures have produced important advances in cognition, reasoning and conceptual aspects of human thinking. [Levesque and Lakemeyer, 2008] offers an overview of the challenges and efforts taken in the subject of cognitive robotics. A comprehensive review of various different cognitive architectures, issues and challenges, can be found in [Langley et al., 2009], many of which have seen practical use in real-world problems. To date, contributions to the development of cognitive architectures for humanoid robots have been rather sparse. However, attempts to provide cognitive processes and functionalities for a humanoid robot can be found in the works of [Brooks et al., 1999], [Burghart et al., 2005], [Zoliner et al., 2005a], [Galindo et al., 2005], [Jung et al., 2007], [Lemaignan et al., 2010], [Choi et al., 2009], [Kim et al., 2010], and [Tan, 2012], among others.

Further research into cognitive architectures, frameworks and cognitive models is important to improve the control and design of the intelligent robotic agents. The most obvious arena for improvement concerns the introduction of new capabilities, and additional research on the structures and processes that support such capabilities [Langley et al., 2009], which bear the wide range of human skills and cognitive abilities. The architectures must address the issue of the agents' physical limited resources. Frameworks are needed that can encode knowledge in a variety of formalisms, and use them with greater flexibility and more effectively to support intelligent behaviours [Langley et al., 2009]. Cognitive architectures need to confront the roles of the interaction with the environment, agents' internal drives, emotions, etc. There is also the need for experimental methods for the thoughtful evaluation of cognitive architectures [Langley et al., 2009]. The development of cognitive architectures support the central goal of artificial intelligence, cognitive science and robotics: and of building artificial systems that are as capable as human beings. The reviewed cognitive architectures constitute a solid basis for building intelligent systems, since they

Architecture	Deliberative	Reactive	Hybrid	Cognitive
Design Paradigm	Sense-Plan-Act cycle	Hierarchy of coupled sense-act behaviours	Low-level reactive layers and high-level deliberative layers	Interconnected structure of functional cognitive modules
Strengths	-Planning of long term actions. -Break complex task into subtasks. -Can produce optimal, domain-independent solutions.	-Improved navigation and obstacle avoidance. -Great efficiency at run-time. -Robust, simple and computationally tractable.	-Combine real-time response and adaptability of reactive systems with planning and decision making of deliberative approaches.	-Solid basis for building intelligent systems. -Interconnected models of cognitive abilities support range of skills and actions.
Challenges	-Fail to address uncertainty. -High computational cost. -Poor performance when frequent replanning.	-No long-term planning. -Limited applicability. -Difficult to debug and understand emerging behaviour.	-Very application dependant. -Lack general design, methodologies. -Difficult to generalize in varying domains.	-Introduce new capabilities. -Address agent physical limited resources. -Experimental methods to evaluate architectures.
Implementations	STRIPS, IRMA, etc.	Subsumption	ATLANTIS, SSS, 3T, AuRA, etc.	Soar, ICARUS, ACT-R,EPIC, etc
Applicability for Humanoid Robots	Unfit to operate in changing environments.	Offers limited applicability confined to low level tasks.	Couple strengths of deliberative/reactive paradigms. Lack of good theoretical models.	Support goal for intelligent artificial systems. Further research is important.

Tab. 2.2: *Comparison of Intelligent Architectures with their strengths, challenges and possibilities for application in the field of humanoid robotics.*

are based on well-motivated and properly grounded cognitive research [Sun, 2009]. The desired cognitive agents must display capacities for environmentally coupled embedded action: and at the same time, they must think or reason abstractly about the world in a de-coupled manner, as argued by the theories of embodied situated cognition.

Table 2.2 summarizes the most relevant aspects and shortcomings of the intelligent architecture approaches for developing humanoid robots that have been reviewed throughout this chapter. Comparing their their design, strengths, challenges, implementations and possibilities for application in the field of humanoid robotics.

2.7 Framework for Learning and Adaptation of Skills to Task Constraints

From everything that has been stated throughout this chapter, it becomes clear that the envisioned humanoid robots of the future, capable of working autonomously and serving humans, are required to have advanced motor control skills, comprehensive perceptual systems, and suitable intelligence, with an intelligent agent being understood as in [Poole et al., 1998], as one that is flexible to changing environments and changing goals, and one that learns from experience and makes appropriate choices, given perceptual limitations and finite computation. The previous sections presented a review of different approaches for developing a robot's functional architecture that would endow it with the capabilities for performing intelligent behaviours in the environment. Clearly this is a very challenging topic in which completely satisfactory solutions have not yet been reached. Although great efforts and advances have been made over the years, obtaining important contributions through the field.

When thinking about what could constitute a general typical task for a humanoid robot operating in a domestic environment together with other human agents, let us consider a kitchen setting and a cooperative task of setting a table for supper. The robot would be required to pick up, place and hand different objects into different places at different times, not necessarily following a particular order or sequence established beforehand, and in a world being changed not only by its actions, but also by other agents working in the same space. In this scenario, deliberative planning approaches would be unsuitable, since they are inappropriate to operate in dynamic changing environments. A reactive and behaviour-based approach would be limited in its applicability, and concentrated only on low level reactive behaviours. Hybrid approaches could be employed when designed to resolve the challenges of one particular task but a different mechanism would be necessary when the focus is on humanoid robots presenting human level intelligence and in replicating the complex level of skills and operations presented by humans. The agent's architecture paradigm must concentrate on the development of intelligent thinking at the system internal processing, centred on an organization of intelligence in terms of the configuration and interaction of cognitive modules. Research in cognitive architectures constitutes a solid basis for building intelligent systems, but even though some attempts on the field have been made for providing cognitive processes for humanoid robots, there are no

fully developed cognitive architectures capable of endowing robots with the necessary functional intelligence readily available.

[Albus, 1991] proposed a multi-layered hierarchical system architecture, where different levels of intelligence in the hierarchy can be achieved, depending on the computational power of the system and the sophistication of its processing algorithms. A minimal level of intelligence requires at least the ability to sense the environment, make decisions and take actions. Higher levels of intelligence may include the ability to recognize objects and events, to represent knowledge in a world model, and to reason about and plan for the future. More elevated forms of intelligence provide the capacity to perceive and understand, to choose wisely, and to act successfully under a large variety of circumstances.

The current humanoid robots may only be around the minimum and mid-levels of intelligence. Even if perhaps the ultimate levels of intelligence could turn out to be out of reach, and creating robots that replicate the total scope of human intelligence may prove impossible, it is necessary for future humanoid robots to achieve a sufficiently high level in the hierarchy. A cognitive framework for humanoid robots needs to provide a minimum degree of intelligent behaviour; this is the ability to sense the environment, learn, and adapt its actions to perform successfully under a set of circumstances.

The reference model architecture [Albus and Barbera, 2005], [Albus, 1991], identifies five elemental systems contained in each layer, such as, sensory processing, world modelling, behaviour generation, value judgement and knowledge, interconnected in a way that enables the various system elements to interact and communicate with each other in intimate and sophisticated ways. Research efforts must focus on building the necessary modules of cognition that would form the layers in this hierarchy and allow for assembling the levels of intelligence.

Humanoid robot agents to be successfully used for working alongside human partners would need to address important challenges such as high level understanding, engaging interactions and quick adaptations to environmental dynamical changes [Stoytchev and Arkin, 2001]. The ability to self-adapt and learn from experience is a major concern. In order to have humanoid robots acting fluently in the world, interacting with different objects and people, they must be able to learn and adapt their motor control to dynamic changes in their interaction with the world, that is, robot systems must be continuously self-adapting [Brooks, 1996].

It becomes apparent that humanoid robots must be provided with systems that allow them to continuously learn new skills and adapt their existing skills to new contexts, as well as to robustly reproduce new behaviours in a dynamical environment in order to cope with working in continuously changing environments and performing an unlimited variability of tasks.

Motivated by the design of multi-layered reference model architectures, in the spirit of [Albus, 1991], and influenced by the ideas of the *Dynamical System* approach to embodied cognition, as promoted by the works of [van Gelder and Port, 1995], [Clark and Grush, 1999], [Clark, 2004], [Beer, 2000], and in the *Learning from Demonstration* approaches for encoding complex motions as *Dynamical Systems*, first introduced by [Ijspeert et al., 2001], [Ijspeert et al., 2002], representing movement plans

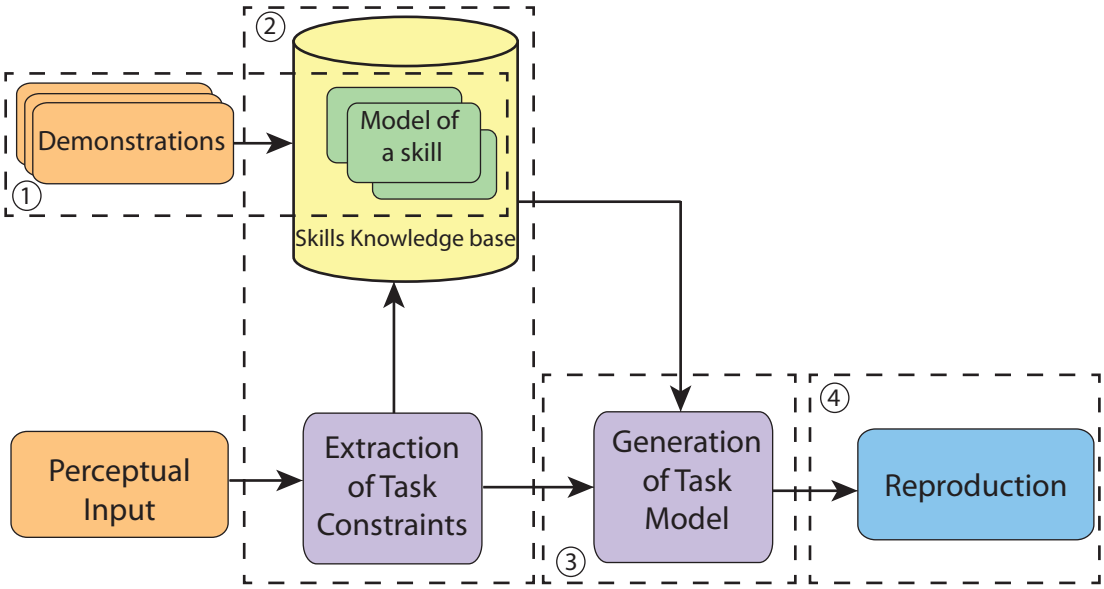


Fig. 2.6: *Proposed framework of a cognitive model for the learning and adaptation of robot skills to task constraints. A knowledge base (2) is built with the models of the robot skills learned through demonstrations (1). The constraints of a requested task are extracted from the perception of the world state. With the current task constraints and the models of a skill retrieved from the knowledge base an adapted task model (3) is generated for reproduction (4).*

as mixtures of non-linear differential equations with well-defined attractor dynamics, in this work a framework is proposed for a cognitive module for the generation and adaptation of learned models of robot skills for complying with task constraints.

We follow a view which claims that models of cognition must be embodied processes capturing the unfolding of cognition in time, mindful of the associated sensory and motor surfaces embedded in the environment in which cognitive phenomena takes place [Schöner, 2008]. And that systems' internal representations may be modelled not as simple inner states but as dynamical patterns of just about any conceivable kind [Clark, 2004]. Here, thought can be described by variables governed by a set of non-linear differential equations and an agent behaviour can be generated from the complex dynamical evolution of stable states and their instabilities in a non-linear dynamical system [Schöner, 2008].

For the rest of this work, and throughout the following chapters, a framework for the generation and adaptation of learned skills to task constraints is presented, developed, implemented and validated. Figure 2.6 illustrates our proposed framework. The main purpose of the framework is to provide the humanoid robot with a basic level of intelligence, namely, the ability to sense the environment, learn and adapt its actions to perform successfully under a set of circumstances. In the developed framework a knowledge base of skills is built with the models of the skills learned

through demonstrations. During execution the constraints of a requested task are extracted from the perceptual system from the working environment and the models of an appropriate skill are retrieved from the skills knowledge base. With all available information a new adapted task model is generated for reproduction.

The framework provides humanoid robots with systems that allow them to continuously learn new skills, represent their skills' knowledge, and adapt their existing skills to new contexts, as well as to robustly reproduce new behaviours in a dynamical environment. The proposed framework is formed by 4 fundamental modules:

1. Module for the learning of robot skills.

2. Module for the management and representation of robot skill knowledge.

3. Module for the generation and adaptation of robot skill models.

4. Module for the reproduction of robot skills.

The robot skill learning module collects the learning processes and algorithms used for learning and encoding the models of the skills. The robot skill knowledge module controls the developed knowledge base. The robot skill generation and adaptation module governs the process by which the learned model of a skill can be operated to reproduce a new task. The robot skill reproduction module produces the adequate control signals to the robot for the reproduction of those skills. Additionally, a perception and interaction module is in charge of processing the outside information of the robot's working environment to use in the other modules. The following chapters will describe in more detail the modules for learning the robot skills models, representing the robot skills knowledge, generating and adapting robot skills, and the reproduction of the robot skills.

The ultimate goal for a humanoid robot would require them to present full level cognitive and intelligent architectures, yet current developments are not yet even near close to these capacities. The cognitive architecture archetype could, eventually, very well be the most suitable approach for building the humanoid robots' intelligence capabilities. However, a majority of current cognitive approaches focus more on solving intelligence as an abstract reasoning process and do not take into account the physically embedded aspects of cognition and the particular challenges humanoid robotics represents. Furthermore, fully developed cognitive architectures with the capabilities for endowing robots with the needed functional intelligence are not readily available. Therefore we begin our approach by trying to attain a basic functional level of intelligence allowing a robot the ability to sense the environment, learn, and adapt its actions to perform successfully under a set of circumstances. The framework developed in this work was proposed as a cognitive model intended to provide the robot with an essential cognitive ability for learning and adaptation of skills. Our framework can be thought of as one module level in the hierarchy of a more complex architecture, or as a first stepping stone upon which to incrementally build more complex cognitive processes.

2.8 Summary of the Chapter

Throughout this chapter a review of the developments and challenges in humanoid robotics research has been presented along with different proposals for intelligent agent architectures for robotic systems. Table 2.1 summarizes the major historical developments in humanoid robotics research. Section 2.2 discussed the issues emerging for humanoid robot developments and for motor control, perception, interaction and intelligent behaviour. Much work remains to be done in order to improve the capabilities of humanoid robots for locomotion, perception, interaction, cognitive behaviour and competence at performing tasks. Humanoid robots must present intelligent, natural, predictable and reasonable behaviours. Different approaches were reviewed in planner based, behaviour based, hybrid, and cognitive architectures for intelligent robots. Section 2.3 presents a review of approaches to robot planner-based architectures. They follow the Sense-Plan-Act cycle, intelligence resides on a central planner that produces appropriate plans of action for the robot reproduction. Section 2.4 presents a review of approaches to robot behaviour-based architectures. They present direct coupling between perception and action. Intelligence emerges as a result of an embodied agent interaction with the environment. Section 2.5 presents a review of approaches to robot hybrid deliberative/reactive architectures. They attempt to use the advantageous aspects of both the behaviour-based and the planner-based approaches. Section 2.6 presents a review of approaches to robot cognitive architectures. Planning approaches are unfit to operate in changing environments, as would be required of humanoid robots. Behaviour-based approaches are limited in their applicability to low-level behaviours and they would not be suitable to deal with the complexities of behaviours present in humanoid robots. Hybrid approaches combine the strengths of deliberative and reactive approaches and can be readily employed as the system architecture for several robotic platforms, but they tend to be very specific and application dependent; also, the lack of good theoretical models makes generalization and reproduction of their results difficult for varying domains. Research in cognitive architectures constitute a solid basis for building intelligent systems, but even though some attempts in the field have been made for providing cognitive process for humanoid robots, there are no fully developed, cognitive architectures capable of endowing robots with the needed functional intelligence readily available. Cognitive approaches are centred on the mechanism that allows for the generation of thought and the interior workings of cognition. This calls for an organization of intelligence in terms of cognitive models. Table 2.2 summarizes the most relevant aspects and shortcomings of the intelligent architecture approaches for developing humanoid robots that have been reviewed throughout this chapter. Section 2.7 presents the proposed framework, followed in the rest of this work, of a cognitive model for learning and adaptation of skills to task constraints. Our approach attempts to attain a basic functional level of intelligence, allowing a robot the ability to sense the environment and learn and adapt its actions. The framework provides humanoid robots with systems that allow them to continuously learn new skills, represent their skills' knowledge and adapt their existing skills to new contexts, as well as to robustly reproduce new behaviours in a dynamical environment.

3. LEARNING ROBOT SKILLS MODELS FROM DEMONSTRATIONS.

3.1 Outline of the Chapter

This chapter presents the methodology followed in this work for learning models of robot skills. Robots working alongside humans means there will be continuously changing environments and a huge variability of tasks that the robot is expected to perform: thus, the robot should have the ability to continuously learn new skills and adapt the existing skills to new contexts. An important part of the framework proposed in the previous chapter, and developed through this work, is the learning of robot skills. Figure 3.1 shows the framework proposed throughout this work for the learning and adaptation of robot skills to comply with task constraints, highlighting the module for learning the robot skills discussed in this chapter. *Learning from Demonstration (LfD)*, also known as *Robot Programming by Demonstration (RPbD)* or *Imitation Learning*, has appeared as a major trend for developing intuitive control methods. This chapter presents important concepts in *LfD* and the most relevant developments in demonstration learning approaches. It also describes the learning process and algorithms used for learning and encoding the models of the skills. Finally, the results of the teaching and learning process for various different robot skills are presented. The organization of this chapter is as follows:

- Section 3.2, presents the basic notions, and a review of the field, of *Learning from Demonstration (LfD)*.
- Section 3.3, presents a review of methodologies for teaching and building the demonstration datasets for learning. This include kinaesthetic teaching, visual demonstrations, motion capturing systems to record demonstrations and, generating robot trajectories with virtual reality or simulated environments.
- Section 3.4, presents the framework employed through this work to learn robot skill motions from demonstrations. The approach is based on learning time independent models of the motion dynamics estimated through a set of first order non-linear multivariate dynamical systems.
- Section 3.5, presents a review of the methodologies used for the encoding of the models of the motion dynamics for learning robot skills.
- Section 3.6, presents a review of the methodologies used for the reproduction of the learned motion dynamics of robot skills.

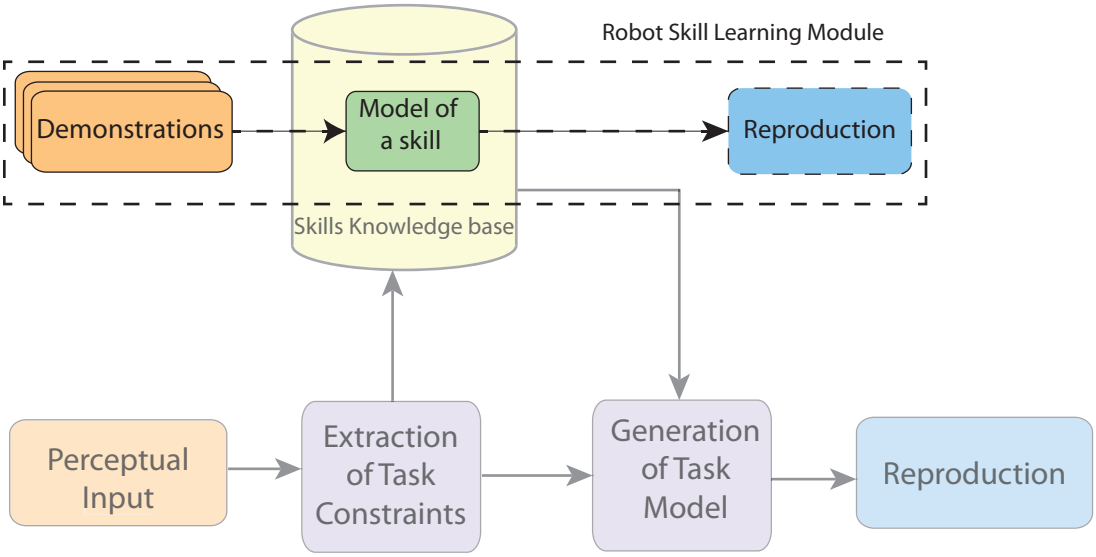


Fig. 3.1: *Learning models of robot skills module, highlighted over the proposed cognitive framework for learning and adaptation of robot skills in compliance with task constraints. To learn a robot skill, models of the motion dynamics are built from various human demonstrations of the skill. Robot replicates a skill by reproducing the model of the demonstrated skill motion.*

- Section 3.7, discusses approaches for using the learned robot skills as basic primitives of movement.

3.2 Learning from Demonstration

Previously in section 2.2, the challenges of developing humanoid robots were discussed. When trying to develop the next generation humanoid robots, with the capabilities to collaborate and interact together with humans and, sharing the same space, tools, and activities with them, there are many important issues that arise and which motivate the field's research directions. Finding suitable solutions to the challenges in system design, appropriate materials, power supply, processing capacities, motor control and sensory perception is a fundamental goal. However, even if it would be possible to have access to an ideal robotic system with every desirable property, the successful operation of a humanoid robot would not be possible without developing proper control mechanisms. The control algorithms traditionally available are not nearly versatile, robust or flexible enough to achieve the level of complexity of the biological systems which are to be emulated. Missing are the abilities to deal with large movement repertoires, variable speeds, constraints and uncertainty in the real-world environment in a fast, reactive manner [Peters et al., 2003]. Most current robotic systems can only solve tasks after the task has been carefully analysed and added to the robot program by a human [Schaal, 1999]. This requires an impressive

amount of work, research and time, and it is very inefficient when it is needed to develop a broad set of behaviours. The classical robotics approach relies heavily on teleoperation or fixed "pre-canned" behavior based control with very little autonomous ability to react to the environment [Peters et al., 2003]. There are many approaches that rely on the teleoperation control of humanoid robots [Hasunuma et al., 2006], [Pierro et al., 2009], [Glassmire et al., 2004], [Neo et al., 2007], [Stilman et al., 2008], [Evrard et al., 2009]. A teleoperation system for the control of a humanoid robot can present advantages, like versatility, provided by the human operator when dealing with various tasks and environments. Yet several challenges arise in humanoid robot teleoperation, from the control of the many DOF of humanoid robots, satisfying both severe balance constraints and the geometrical and dynamical differences between humanoid robots and humans, in addition to the regular issues presented in teleoperated systems [Chen et al., 2007], such as, limited FOV, degraded perception, time delay, user interface, operator cognitive load, etc. As useful as teleoperation control can be for certain humanoid robot missions, to benefit from the full potential of humanoid robots control architectures cannot rely only on teleoperation since humanoid robots are also expected to perform their tasks in an autonomous way. In order to overcome the need for teleoperation and manual "hard-coding" of every behaviour, a learning approach is required [Schaal, 1999].

Robot learning covers a large field, encompassing learning to perceive, control, to plan and, make decisions, etc. [Schaal and Atkeson, 2010]. Machine learning algorithms have been extensively developed in the last couple of decades. Machine learning techniques present wide application at several levels of robot planning and control [Münch et al., 1994], offering solutions in computer vision, object recognition, grasp planning, robot motion, pattern recognition, language processing, etc. Robotic systems, of the characteristics of the humanoid robots we want to develop, need to be able to learn, and adapt to uncertainty and unforeseen changes in their dynamic environments. Focus on this work will center on topics of learning control, in particular of robot learning of motion trajectories and skills. Learning control refers to the process of acquiring a control strategy, at the core of this is the problem of learning a mapping between world states and actions. This mapping, or policy, enables a robot to select an action based upon its current world state [Argall et al., 2009]. The goal for a robot learner is to generalize from its experience [Bishop, 2006], to find appropriate control policies to accomplish a given movement task. The traditional approaches to robot control of modelling dynamics and deriving mathematically-based policies is most often a challenging task and heavily dependent upon the accuracy of the world model. As a result, machine learning techniques have been applied to policy development [Argall et al., 2009]. Robot learning can be classified, from the viewpoint of machine learning, as supervised learning, reinforcement learning, learning modularizations or learning feature representations that subserve learning [Schaal and Atkeson, 2010].

Robot Programming by Demonstration (RPbD) [Billard et al., 2008], appeared as a promising route to automate the tedious manual programming of robots and as a way to reduce the costs involved in the development and maintenance of robots in a factory. Moving from purely preprogrammed robots towards flexible interfaces for training robot tasks follows a three-fold motivation. *RPbD* or *LfD* is a powerful

mechanism for reducing the complexity of search spaces for learning. It offers an implicit and natural means of interacting and teaching a machine. It also helps to understand the coupling mechanism of perception and action [Billard et al., 2008].

Acquiring efficient motor learning, exploring the connection between action and perception and modular development of motor control in the form of movement primitives, are three issues at the core of *Imitation Learning* [Schaal, 1999]. The *Imitation Learning* approaches focus on the development of algorithms that are generic in their representation of the skills and in the way they are generated. Implementing *LfD* methods offers the possibility of making learning faster, in contrast to tedious reinforcement learning methods or trial-and-error learning. *LfD* formulates user-friendly methods by which a human user can teach a robot how to accomplish a given task, simply by demonstrating this task [Gribovskaya et al., 2010], and generalizing the demonstrated movements across a set of demonstrations. Due to the intuitive nature of the demonstrations, *LfD* algorithms have the potential of making robots accessible for everyday users, not requiring extensive programming experience but rather the ability to provide demonstrations of the chosen behaviours [Argall et al., 2009].

A most important question here is what is it that should be learned? The major goal of learning control is acquiring a task-dependent control policy π that maps a continuous-valued state vector $\mathbf{x}$ of a controlled system and its environment, to a continuous-valued control vector $\mathbf{u}$. The motor control learning is thus centred on finding the generally non-linear function π that is adequate for a desired behaviour [Schaal and Atkeson, 2010].

As mentioned above, the machine learning approaches for policy development can be mainly divided between unsupervised learning, supervised learning and reinforcement learning. Unsupervised learning refers to the problem of finding hidden structures in data. No reward or error signal exists to evaluate a potential solution since the examples given to the learner are unlabelled.

Reinforcement learning in robotics offers one of the most general frameworks towards true autonomy and versatility [Peters et al., 2003]. The reinforcement learning approach should enable humanoid robots to autonomously learn motor skills from interaction with the environment, and given only a relatively unspecific feedback on the quality of completing the task. However, in practice, applying reinforcement learning to humanoid robots poses several challenges [Stulp et al., 2010]. The state and action spaces are continuous, the learning problems are high-dimensional thanks to the large number of DOF in humanoid robots. Exploration in high-dimensional spaces is costly and time consuming, and it is difficult to acquire an accurate model of the robot and its interaction with the environment. The greedy policy improvement algorithms are likely to fail to scale to the high dimensional systems as their large changes in the policy during learning makes stable algorithms, so far, infeasible. The policy gradient methods are promising techniques in terms of scaling to high dimensional continuous control systems, and have been applied in humanoid robotics for both walking and fine manipulation [Peters et al., 2003].

In Supervised Learning the agent is presented with labelled training data and learns an approximation to the function that generated such data. *LfD* can be seen as a subset of Supervised Learning [Argall et al., 2009]. In the scope of *LfD*, the

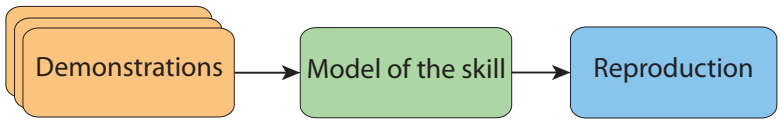


Fig. 3.2: *Generalization of a skill by extracting the statistical model across multiple observations. Adapted from [Billard et al., 2008]*

training dataset is composed of example executions of the task by a demonstration teacher. As *LfD* it is understood the general category of algorithms in which a policy is derived based on demonstrated data. A probabilistic transition function defines the mapping between the world states S and actions A, $S \times A \times S \rightarrow [0, 1]$. The learner has only access to observed states Z, since state S is not fully observable, through the mapping $U : S \rightarrow Z$. A policy selects from the set A of actions, containing low-level motions to high-level behaviours, based on observations of the world state [Argall et al., 2009].

To reproduce a skill in a new situation, the robot cannot simply copy an observed behaviour; it must have the capability to generalize [Calinon, 2009]. A common approach for generalizing a skill consists of creating a model of the skill based on several demonstrations, performed in slightly different conditions. The goal is to exploit the variability inherent to the various demonstrations and to extract the essential components of the task. Figure 3.2 illustrate this process.

LfD covers methods by which a robot learns new skills through human guidance. Common to all these approaches is the presence of a teacher, providing examples for the execution of a desired behaviour, and a learner, provided with a set of these demonstrations and deriving a policy from such examples capable of reproducing the demonstrated behaviour. Distinctions among *LfD* methods can be made based on their choice of demonstration approach: the choice of demonstrator or demonstration technique, the choice of state action representation, either discrete or continuous representation and the selection of an algorithm for generating the policy. The determination of these decisions can greatly be influenced by factors such as the general domain, task complexity and robot capabilities, and developers preference. Within *LfD*, the learning problem is thus segmented into two phases: gathering the examples and deriving a policy from such examples [Argall et al., 2009]. In *LfD* a popular method employs a probabilistic framework gathering information from cross-situational observations of a skill with information extracted from different social cues observed during the interaction [Calinon and Billard, 2008]. A key concept at the bottom of these approaches is that of determining a metric of imitation performance. First it must be determined the metric, weights of the function for the reproduction of each of the components of the skill. Then it is possible to find an optimal controller for imitation by minimizing the metric. Relevant problems to address in these approaches are, the problem of extracting the relevant features of a given task, the problem of evaluating

how the task should be reproduced and the problem of finding optimum controllers to generalize the acquired knowledge of various contexts [Calinon et al., 2007].

The demonstrated behaviours can be used to learn the appropriate control policy directly by supervised learning. In these methods, called "task-level imitation" [Schaal, 1999], a serious constraint is imposed on the need for the state and action of the teacher to be observable and identifiable. Therefore, a coordinate frame based on variables that can be perceived would be needed to define a movement primitive. Prior knowledge of how a task-level command can be converted into an actuator-level command is required. For this purpose, motor control needs to be modular, assuming at least separate processes for movement planning and execution. A second approach to learning novel behaviours is based on building policies out of the demonstrated trajectories. This process results in data about the movement of the manipulated object in Cartesian coordinates, as well as the movement of the actuator in terms of joint angle coordinates. Knowledge of the task goal is manually provided in the form of an optimization criterion. Based on this knowledge, the robot's performance improves by trial and error learning until the task is accomplished [Schaal, 1999]. A third method employs model-based learning from the demonstrated behaviours to learn a novel primitive; the dynamics of the task are approximated in the form of a predictive forward model. Given knowledge of the task goal, the task-level policy of the movement primitive can be computed with reinforcement learning procedures based on the learned model [Schaal, 1999].

Research within *LfD* has seen the development of three core approaches to policy derivation from demonstration data: a mapping function approach consisting of learning an approximation to the state-action mapping; A system model approach based on learning a model of the world dynamics and deriving a policy from this information. An alternately is planning approaches where a sequence of actions can be produced by a planner after learning a model of action pre and post-conditions [Argall et al., 2009]. The mapping function approach to policy learning calculates a function that approximates the state to action mapping, $f() : Z \rightarrow A$, for the demonstrated behaviour. These types of algorithms aim to reproduce the underlying teacher policy and to generalize over the set of available training examples. The goal is to acquire valid solutions for similar states that may not have been encountered during demonstration [Argall et al., 2009]. The system model approach to *LfD* policy learning derives a policy $\pi : Z \rightarrow A$ using a state transition model, $T(s'|s, a)$, of the world. The transition function, $T(s'|s, a)$, is generally determined from the demonstration data and any additional autonomous exploration the robot may do [Argall et al., 2009]. In the planning framework, the policy is represented as a sequence of actions that lead from the initial state to the final goal state. Actions are often defined in terms of the state that must be established before the action can be performed, pre-conditions, and the state resulting from the actions' execution, post-conditions. Demonstration-based algorithms differ in how the rules associating pre and post-conditions with actions are learned, and whether additional information is provided by the teacher [Argall et al., 2009].

The most important issues in the field of *Imitation Learning* are categorized under the broad spectrum of four major questions, namely, the set of generic questions *what*

to imitate, *how to imitate*, *when to imitate* and *who to imitate* [Billard et al., 2008]. A fifth central question on research on *Imitation Learning* relates to *how to evaluate* a successful imitation attempt [Alissandrakis et al., 2002b]. Intense research has been made into solving these questions, focused mainly on technical approaches to answering the *what to imitate* and *how to imitate* questions. *What to imitate* is related to the general problem of 'what to learn of a skill'. There are several aspects of a behaviour that could be imitated. An agent must be able to extract the relevant features of a given task from the 'cues' and constraints that define the 'skill' to imitate. An agent is required to build the structure of the knowledge transferred, choosing between two different kinds of imitation, copying the organizational structure of the behaviour versus copying the surface form of the behaviour [Alissandrakis et al., 2002b]. *How to imitate* considers the problem of 'how to encode a skill', the problem of evaluating how the task should be reproduced and the problem of finding the optimum controller with which to generalize the acquired knowledge [Calinon et al., 2007]. The learning algorithm must provide means from which to learn the encoding of relevant knowledge of the 'skill' to build models appropriated for reproduction. Agents must employ the appropriate mechanisms to learn and reproduce necessary imitating actions [Alissandrakis et al., 2002b]. The *when to imitate* and *who to imitate* questions are strongly related to the social interaction between the imitator and the imitated, these questions have been less explored. *When to imitate* question refers to the problem of 'when it is fit to reproduce a skill'. Agents need to learn to recognize from social and environmental 'cues' when a learned imitation 'skill' is to be used. The imitating agents have to segment the demonstrator behaviour and have to decide on a suitable time and place for imitation, based on the appropriateness of previous or current observed behaviour in their current context [Alissandrakis et al., 2002b]. *Who to imitate* covers the problem of 'observing from whom to learn a skill'. An agent must recognize from social 'cues' and interaction imitation demonstrations provided from other agents, and evaluate their usefulness as an appropriate behaviours to imitate. The agent must choose its demonstrator in order to engage in imitation, produce an imitated behaviour that is beneficial, and at the same time, not to imitate agents whose tasks and needs are not relevant to the imitator [Alissandrakis et al., 2002b]. The question of *how to evaluate an imitation attempt* refers to the need to find proper measures to evaluate behavioural matching [Alissandrakis et al., 2002b]. Determining a metric of imitation performance is very important. It must be determine the metric, weights of the function for the reproduction of each of the components of the 'skill' [Calinon and Billard, 2008]. The above questions and their solutions aim at being generic in the sense of making no assumptions about the type of skills that may be transmitted [Billard et al., 2008].

Once all the features and relevant knowledge of a given task have been extracted from a set of suitable teacher's demonstrations, the most fundamental issues becomes how such information should be converted into actions; this concerns the *How to imitate* question above. For this purpose the concept of movement primitives, also called movement schemas, or units of actions, is proclaimed. Movement primitives are sequences of action that accomplish a complete goal-directed behaviour [Schaal, 1999]. A movement primitive can have different forms of representation. Two major trends

can be identified for the generalization of these task representations: a trajectory level and a symbolic level representation. A task at a trajectory level is described by temporally continuous signals representing different configuration properties changing over time. A task at a symbolic level is described by the sequential or hierarchical organization of a discrete set of primitives that are pre-determined or extracted with pre-defined rules [Calinon, 2009].

RPbD or *LfD* contribute to major advances in robot learning, and advance the development of robust controllers for service, personal, and humanoid robots. *LfD* is an intuitive communication medium for human teachers and enables the development of control algorithms to non-robotics experts. For a complete review on the field see [Billard et al., 2008]. It also, offers solutions to certain weaknesses in traditional approaches and complements traditional policy learning techniques. *LfD* has been successfully applied to many robotic applications. The field has seen very active research, as exemplified in the works of [Kober and Peters, 2010], [Schaal et al., 2007], [Ijspeert et al., 2009], [Gribovskaya et al., 2010], among others.

Research into *RPbD* or *LfD* has seen important developments in many areas, yet some issues have received limited attention, these issues include, sufficient availability of state features to adequately describe the task and to allow its learning, the encoding of temporary information and event memory into the demonstration sequence; also recovery operations in the event of encountering failures in policy derivation or execution, the ability to continuously learn from its experience, the application of *LfD* in multi-robot settings, the development of standardized evaluation metrics, etc. [Argall et al., 2009]. Certain outstanding questions remain to be addressed, such as, how can appropriate movement representation be developed in an automated fashion? How can new primitives be learned, and old primitives be combined to form higher level movement primitives? How can sequencing and recognition of sequences of movement primitives be accomplished? Are the mechanisms for movement generation also directly employed for movement recognition? How can the demonstrated movement intentions be recognized? And how can they be converted to the imitator's goal? [Schaal, 1999].

3.3 Providing Demonstrations of a Skill

A *LfD* framework has many favourable features, as stated in the previous section; one such very attractive feature for the development of a demonstration approach is that of an intuitive medium for communication from humans who already use demonstration to teach other humans [Argall et al., 2009]. Providing demonstrations to a humanoid robot agent offers a familiar and instinctive way for non-expert users to communicate and program the robot behaviours. The *Imitation Learning* approach allows for a well-known mechanism, regularly employed for teaching and learning the performance of tasks among the general public, to be easily used to naturally interact with a robot. *LfD* provides an implicit means to facilitate learning for humanoid robots. Demonstrations also have the practical feature of focusing the dataset to areas of the state-space actually encountered during task execution

[Argall et al., 2009]. The *LfD* or *RPbD* paradigm to learning control has at its core the goal of enabling robots to perform new task autonomously, focused on building appropriate robot control policies derived from observations of a human demonstration performance. Within *LfD*, the learning problem is thus segmented into two phases: gathering the demonstration examples and deriving a policy from such examples [Argall et al., 2009]. In this section, various techniques for executing and recording demonstrations are discussed.

The first approaches to *Imitation Learning*, adopted for manipulator robotics, chose to rely on symbolic reasoning [Billard et al., 2008]. Due to reduced computational power demonstrations consisted of manually pushing the robot through a movement sequence [Schaal et al., 2003], divided into subgoals and into appropriate primitive actions, commonly chosen to be simple point-to-point movements. The demonstrated tasks were segmented into sequence of state-action-state transitions, and from them 'if-then' rules were extracted, describing the states and actions according to symbolic relationships [Billard et al., 2008]. The field moved gradually from copying movements to generalizing over sets of demonstrations. [Münch et al., 1994] suggested using machine learning to recognize Elementary Operators, defining discrete sets of basic motor skills, learning tasks by generalizing over a sequence of discrete actions. However, this was only one part of the problem and learning continuous trajectories to control actuators were also required [Billard et al., 2008]. As machine learning, robotic and sensor systems have experienced advances in their respective fields. *Imitation Learning* has been influenced by non-symbolic learning tools, including, artificial neural networks, radial-basis function networks, fuzzy logic, statistical learning, etc. [Schaal et al., 2003]. More recent trends take inspirations on processes of animal imitation, taking into account evidence of neural-mechanism for visuo-motor imitation in primates, and developmental stages of imitation capacities in children [Billard et al., 2008]. In essence current works follow mostly a conceptual approach, very similar to that of early approaches, as recent progress has mainly affected only the interfaces to support teaching. New elements include the use of computer vision, data gloves, laser range finder, kinaesthetic teaching, marker-based observation systems, etc. [Schaal et al., 2003].

An *LfD* dataset is composed from the state-action pairs recorded during teacher execution of demonstrated behaviours. A majority of work on *LfD* makes use of humans demonstrations, while some techniques explore the use of robotic teachers, hand-written control policies and simulated patterns [Argall et al., 2009]. An important matter for a demonstration approach to be successful is that states and actions provided by the learning dataset be usable by the robot, by constraining the demonstrations modality the robot can understand and providing sufficient examples to achieve desired generality [Billard et al., 2008]. Demonstrations are defined as recorded trajectories in the teacher's state space, with identifiable start and end points and proceeding through a finite number of steps. For a well formed set of demonstrations the teacher must convene to the learner all necessary information of the task space to fully generalize the demonstrated knowledge of a task. The definition outlined above aimed at being general and makes no assumptions about the type of trajectories or task that are demonstrated, what and how the variables are

recorded, what platform is use during execution, and what types of representations are employed. The choice of demonstrator and the demonstration technique are two key decisions when gathering teacher demonstrations [Argall et al., 2009]. Choosing a demonstrator is additionally decomposed into the controller of the demonstrator and the executer of the demonstration. Choosing a demonstration technique further refers to strategies for providing the data to the learner and the selection of algorithms for deriving a policy.

The major focus in *RPbD* or *LfD* works is in the selection and development of algorithms for policy derivation. Yet studying the learning process, the flowing of information from teacher to robot learner, is also important. In choosing demonstration technique strategies for the learning process and for providing data for the learning mechanism are selected. Options include batch learning, self-improvement and interactive approaches. For batch learning, the demonstrations are sampled beforehand, either because collecting the data is difficult or processing it is too time consuming and it is more practical to collect the data all at once. In batch learning the policy is learned only after all data has been gathered [Argall et al., 2009]. Teachers' demonstrations must cover the behaviour sufficiently to ensure adequate generalization. For self-improvement learning, demonstrated data is also collected at the onset of the learning approach; it is separated from batch learning in that self-improvement involves generating new samples from the learning of the original demonstrations which in turn are used to drive the improvement of the policy itself. For interactive learning approaches, the learning process is also iterative, learning must be quicker and demonstration easier to acquire compared to batch learning approaches. In interactive learning, the policy can be updated incrementally as learning data is made available [Argall et al., 2009]. Interactive learning allows teachers to provide additional demonstration to target observed errors in the robot's reproduction.

Another important choice is selecting the information to record from the demonstration examples. Recorded sensory information must be parsed into knowledge about objects and their spacial location in a coordinate system whether internal or external. Some information should become available on the posture of the teacher and/or positions of objects, if any are involved, while moving [Schaal et al., 2003]. Afterwards, this information needs to be converted into action. Common approaches create model of the skill based on sets of demonstrations performed in slightly different conditions generalizing overt the inherent variability to extract the essential components of the skill [Billard et al., 2008].

The knowledge of the task, extracted from demonstrations of the states and actions in the teacher's dataset, must be relevant and usable to the learner for a successful *Imitation Learning* approach. In an ideal set-up, states and actions of the teacher execution would map directly to the learner's embodiment. However, in practice, a direct mapping is generally not possible, as it is most likely to find that the learner and the teacher will differ in their sensing and mechanical systems and capacities [Argall et al., 2009]. In nature, even two humans or animals of the same species in spite of their morphological similarities would still present dissimilar mappings as their height, weight, muscle build, stamina and so on, would differ between them. For humanoid robots learning from a human teacher, even though an attempt is made at

replicating its functionalities, a direct mapping would not be possible as they don't act in the environment in the same manner. Even when dealing with a robot teacher and learner, including robots of identical types, dissimilar mappings are likely to occur due to differences in their respective sensory-motor characteristics. The challenges which arise from these differences are broadly referred to as the Correspondence Problem [Nehaniv et al., 1998].

In order to match reproduction of an observed behaviour in a copying, imitation or mimicry approach, it is important that a suitable correspondence is established. To achieve a behavioural match a correspondence must explicitly or implicitly be present [Nehaniv et al., 1998]. Different correspondences could be required depending on the type of task to imitate, whether it is desired to match individual actions or global goals, or how teacher and learner sensory-motor characteristics differ from each other.

The Correspondence Problem is circumscribed to determining partial correspondence between states and events for the imitator and those of the model to imitate, and to search for an appropriate relational morphism ensuring a sufficient degree of correspondence between them for imitation to be possible [Nehaniv and Dautenhahn, 2001]. As outlined above, dealing with issues of correspondence is important since exact copying of behaviours, even when there is similar embodiment, is almost never possible [Nehaniv et al., 1998]. Solving these discrepancies in sensory-motor capabilities of agents is a problem related to the *how to imitate* question. It is important to note that correspondence need not be a one-to-one mapping, it can take many forms; also successful imitation does not necessarily involve a fixed mapping correspondence, a partial mapping could also be an useful correspondence [Nehaniv and Dautenhahn, 2001]. Simple one-to-one correspondence cannot exist between the joints of two agents with a different number of DOF, as often would be the case among robots. A robot may, however, still imitate a human successfully, e.g. in a waving task, without requiring that it has the same number and type of joints as the human whose behaviour it emulates using a particular correspondence [Nehaniv et al., 1998]. Humans and humanoid robots, although interacting in the same environment and using the same objects, would still perceive and act in the world in very different ways due to their difference in structure, form, DOF, sensors, and abilities. Correspondence, nevertheless, can still be found regarding to two different dimensions, a perceptual equivalence, dealing with the differences in which the agents can perceive the world, and a physical equivalence dealing with the differences in which the agents can perform the task in the world.

According to [Nehaniv and Dautenhahn, 2001], for a behaviour to be called imitation, a correspondence of perception, both exteroceptive and proprioceptive, must exist between model and follower. While correspondence in form, structure, dynamics of actions and behavioural repertoire are also very important aspects, a corresponding perception of a shared context between the model and imitator is a fundamental requirement for imitation. This shared context can be fixed, whether designed by nature or artificially engineered, accidental, opportunistic, or actively established [Nehaniv et al., 1998]. With an insufficient perceptual correspondence, an agent could still be able to follow or mimic another agent's behaviour; however, the agent cannot perform the behaviour alone; unless it has perceptions correlating to those of the

model, no true imitation of that particular behaviour can take place.

It is possible to distinguish with respect to how the events of an imitator 'correspond' to the distinct types of imitation of the imitatee: action-level imitation, where the imitator is set to carry out actions exactly as in the imitated system; where the purpose of the behaviour lies in action-matching rather than focusing on a particular resulting state [Nehaniv and Dautenhahn, 2001]; program-level imitation, where the imitator carries out an identical program conceived as a structure of hierarchical subroutines; it entails acquiring a program of action that makes use of a solution to the correspondence problem. Program-level imitation focuses attention on components of the behavioral program rather than the structural [Nehaniv and Dautenhahn, 2001]; effect-level imitation, where the imitator's concern is obtaining results similar to those of the imitatee, rather than with matching specific actions; for effect-level imitation, trying to imitate, relies on discovering affordances to attain effects corresponding to those attained by the agent being imitated [Nehaniv and Dautenhahn, 2001].

No generic solution exists to solve correspondence problems so task specific equivalences are formulated for each case. To find proper mappings between like individuals of the same kind or species can be natural and direct. Determining the mappings between dissimilar bodies is a problem dependent on the observer point of view [Nehaniv and Dautenhahn, 2001]. The judgement of the degree of success or failure of an imitative behaviour is observer-dependent; the observer, either the demonstrator, the learner, or a possible third party, has a central role at judging whether or not an exhibited behaviour matches that of a model [Nehaniv et al., 1998]. The subjective notion of observer-attributed goals must be transformed to a well-defined notion of metrics. By choosing the metrics, one is choosing which states of the demonstrator are deemed to match those of the imitator and how closely they match [Nehaniv and Dautenhahn, 2001]. A metric provides a quantifiable method to measure the error of an attempted imitation, which the imitator uses to evaluate its own success [Alissandrakis et al., 2002a]. Degrees of success can be formalized by metrics in states and actions and measures of correspondence with respect to achieving some result.

As a formal definition [Nehaniv et al., 1998], a correspondence between two autonomous agents is a relation of states $\Phi \subseteq X \times Y$ and sequence of actions $\Psi \subseteq \Sigma^* \times \Delta^*$ satisfying:

$$\forall x \in X \, and \, y \in Y, \, if (x, y) \in \Phi \, and \, (\sigma, \delta) \in \Psi, \, then \, (x, y, \sigma, \delta) \in \Phi$$

where the state of the systems are represented as X, and Y, and the actions-events are represented as Σ, and Δ. A correspondence or mapping to model (Y, Δ) from imitator (X, Σ) is a relational homomorphism: $\phi : (X, \Sigma) \rightarrow (Y, \Delta)$. A sequence of action-events for system (X, Σ) given by $w \in \Sigma^*$ is said to match successfully a sequence $z \in \Delta^*$ in another system (Y, Δ) if w achieves the same effects as z [Nehaniv and Dautenhahn, 2001]. The solutions to correspondence problems result from successful attempts at imitation [Nehaniv et al., 1998].

Different methods could be classified in accordance to the variables employed by learning which are assumed to be observable, are they kinematic or kinetic, are internal or external coordinates used for the demonstrations, are task goal explicit

or not, etc. [Schaal et al., 2003]. Approaches for gathering demonstration data can be categorized in respect to correspondence of two mappings, an embodiment mapping, which is concerned with how the recorded state-actions on the dataset matches those which the learner would execute; and a record mapping, which relates to how the states-actions of the teacher are recorded within the demonstration dataset [Argall et al., 2009]. When exact states-action experienced by the teacher are directly recorded in the demonstration dataset the identity $I(Z, \Delta)$ constitutes the record mapping, where Z are the observable states, and Δ are the actions. If not a record mapping, $g_E(Z, \Delta) \neq I(Z, \Delta)$, is needed to encode the teacher's demonstrations within the dataset. Analogously, when the states-actions are mapped directly to the learner for execution the embodiment mapping is the identity $I(Z, \Delta)$. In any other way an embodiment, $g_R(Z, \Delta) \neq I(Z, \Delta)$, exists to map the learner's execution of the recorded demonstrations. This mapping does not change the contents of the demonstrated dataset, only the reference frame which represents it [Argall et al., 2009]. Further categorization of *LfD* approaches for data acquisition can be made according to whether record and embodiment mappings are present. The presence of more mappings increases the difficulty of recognizing and reproducing the teacher's behaviour. Yet, it also reduces teacher constraint and helps improve generality of the demonstration technique [Argall et al., 2009]. Approaches are first split into two categories based on the embodiment mapping and then further distinguished, within these categories, based on the record mapping.

The case when there does not exist an embodiment mapping, that is $g_E(Z, \Delta) \equiv I(Z, \Delta)$, is denominated as *demonstration*. Here, the teacher demonstrations of behaviour are performed directly by the learner platform or a representation thereof, and the embodiment mapping is not an issue. However, a non-direct record mapping can exist, thus dividing approaches for providing demonstration data as *teleoperation* and *shadowing*, [Argall et al., 2009]. For *teleoperation*, the teacher operates the learner platform and the execution is recorded by the learner's own sensors. The record mapping is direct, $g_R(Z, \Delta) \equiv I(Z, \Delta)$. Among all the methods *teleoperation* is the most direct for transferring learning data, however, it is required that the operation of the robot be manageable for the learners which is not always possible, making it a technique not suitable for all systems [Argall et al., 2009]. In *shadowing*, the learner records its execution while trying to mimic or copy the teacher demonstrated behaviours. The record mapping is not direct, $g_R(Z, \Delta) \neq I(Z, \Delta)$. The *shadowing* method requires an additional component to enable the learner to track and shadow the teacher execution [Argall et al., 2009].

The case where the embodiment mapping do exist, that is $g_E(Z, \Delta) \neq I(Z, \Delta)$, is denominated as *imitation*. Here, the teacher demonstrations are performed on a platform that is different from the learner platform, therefore embodiment mapping is an issue to regard. Equally as in the case of *demonstration*, the record mapping can exist or be the identity, thus dividing approaches for providing imitation as *sensors on teacher* and *external observation* [Argall et al., 2009]. In *sensors on teacher*, the platform executions are recorded by sensors directly on itself. The record mapping is direct, $g_R(Z, \Delta) \equiv I(Z, \Delta)$. The *sensors on teacher* method can provide more precise measurements of the example execution, however, applicability of this technique

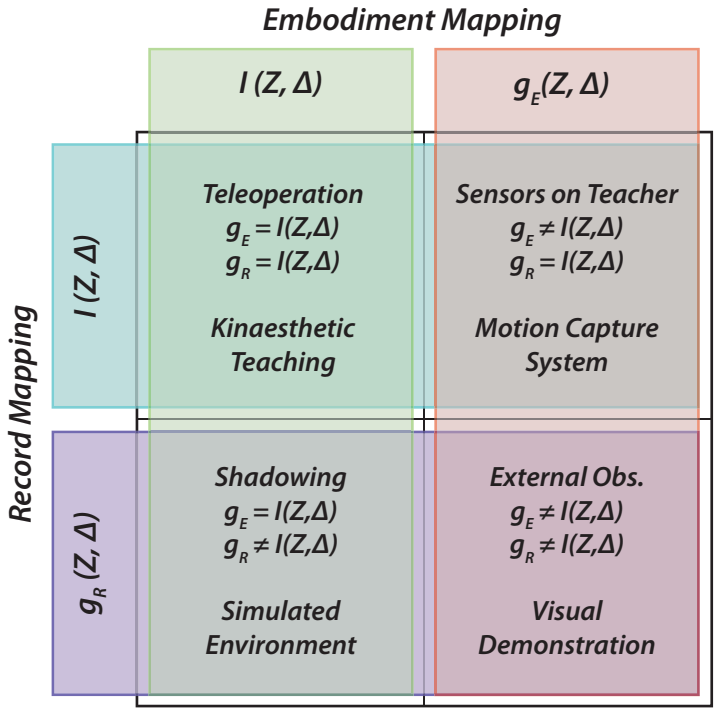


Fig. 3.3: *Intersection of the record and embodiment mappings. Adapted from [Argall et al., 2009]. The demonstration technique can be divide, according to its record and embodiment mappings into four quadrants. (top-left) Teleoperation demonstration when both record and embodiment mappings are direct. (top-right) Sensors on teacher imitation when there exist a non-direct record mapping. (bottom-left) Shadowing demonstration when there exist a non-direct embodiment mapping. (bottom-right) External observation when both record and embodiment mappings are non-direct.*

can be limited by the overhead associated with the need to use specialized sensors [Argall et al., 2009]. For *external observations*, the data from the teacher executions are recorded by sensors externally located to the executing platform, these sensors may or may not be located on the learner's platform. The record mapping is not direct, $g_R(Z, \Delta) \neq I(Z, \Delta)$. The *external observation* method is less reliable as uncertainty increases from having to infer the teacher states-actions from recorded data [Argall et al., 2009].

Many techniques have been employed throughout the field for providing and gathering the demonstration datasets; most popular among them are teleoperation, data gloves, haptic devices, kinaesthetic teaching, motion capture systems, virtual simulations environments, speech interaction and computer vision [Billard et al., 2008]. As outlined above, the role of the interface employed at gathering the demonstrations plays a significant role. In this section, four important techniques for providing demonstrations to a humanoid robot, corresponding to the previous categorization, are reviewed. Figure 3.3 summarizes the approaches' review for gathering and build-

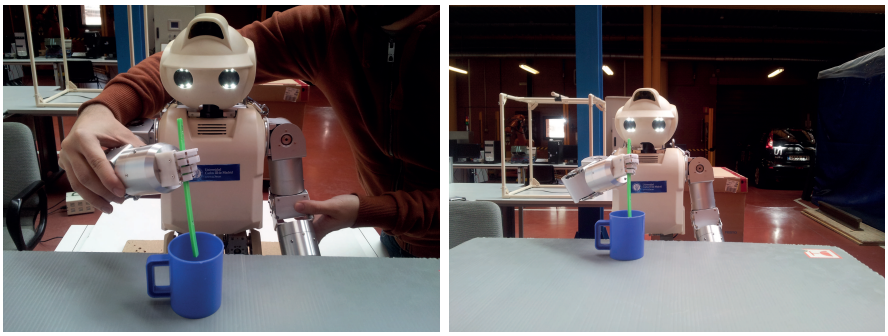


Fig. 3.4: *Kinaesthetic Teaching of a Skill: (left) A human teacher operates the HOAP-3 robot arms through a demonstration of the skill. (right) The HOAP-3 robot reproduces the recorded demonstration of the skill.*

ing the demonstrations datasets, and how they intersect with the considered record and embodiment mapping of correspondence.

Kinaesthetic Teaching

One method for providing demonstrations of the motion to the robot is by means of kinaesthetic teaching. The kinaesthetic teaching process [Calinon, 2009], consists of using the motor encoders of the robot to record information while the teacher moves the robot's arms. To record the demonstrations the robot motors are set in a passive mode, a human demonstrator, standing beside the robot, moves simultaneously the robot's arms as it performs the motions.

In kinaesthetic teaching the robot is operated by the teacher while recording from its own sensors. The record mapping is therefore direct, $g_R(Z, \Delta) \equiv I(Z, \Delta)$. Since the demonstration is performed on the actual robot learner, the embodiment mapping will also be direct, $g_E(Z, \Delta) \neq I(Z, \Delta)$, just like the described category of *teleoperation*, as represented by the top left quadrant of Figure 3.3.

In this work, kinaesthetic teaching was employed to teach several demonstrations to a humanoid robot by operating the robot arms in the performance of different motions. The kinematics of each joint motion were recorded at a rate of $1000Hz$ during the demonstrations and were then re-sampled to a fixed number of points. The robot is provided with motor encoders for every DOF, except for the hands and the head actuators. The process is illustrated in Figure 3.4 for the teaching of a skill with the humanoid robot HOAP-3.

Providing demonstrations to the robot by means of kinaesthetic teaching is advantageous on several fronts. As discussed above for *teleoperation*, it is the most direct method, and since both mappings are the identity there is not correspondence problem. Also, it provides the human teacher with knowledge of the robot platform limitations when performing the demonstrations. However, the manageability of the robot operation is an issue. It would be difficult to provide complex demonstrations requiring the human to move several limbs simultaneously. And it limits the human

Fig. 3.5: *Demonstrations in a Motion Capture Systems: (left) A human teacher performs a demonstration. (right) The generated skeleton of the human recorded demonstration.*

teacher's ability of performing the demonstration naturally, as they would be doing it themselves; specially when the human and the robot have very different embodiments.

Motion Capture Systems

Motion Capture (MoCap) is the term that describes the process of recording the motion of the human or animal body, where the recording process could be in real or delayed time. MoCap involves the mapping of human motion onto the motion of a computer character or Skeleton. This mapping can be direct, such as a human arm motion controlling a character's arm motion, or indirect, such as a human hand and finger patterns controlling a character's skin color or emotional state. There are two main technologies used in motion capture. Inertial Motion Capture technology, were the systems are based on inertial measurement sensors. During the motion, the data captured from the inertial sensors is often transmitted wirelessly to a computer. Optical systems, were optical sensors and one or more cameras are used to estimate the 3D position and orientation of the human body segments during the motion. For the optical systems, markers are generally attached to the human body, which are located on the joint or the body part needed to be captured. The number and the type of these markers and the number of the cameras used in the system depend on the complexity of the motion to be captured [Dyer et al., 2013].

In teaching demonstration to the robot recorded by a motion capture system, the recording sensors are located directly on the teacher executing the task. This means therefore, that there is no record mapping, $g_R(Z, \Delta) \equiv I(Z, \Delta)$. Imitation, however, is directly performed by the teacher and not the robot, therefore the embodiment mapping is not direct, $g_E(Z, \Delta) \neq I(Z, \Delta)$, just as it is for the described category of *sensors on teacher*, as represented by the top right quadrant of Figure 3.3.

Currently there are several different options of equipment that can be used for MoCap systems, although they remain a little pricey. Alternatively, many libraries

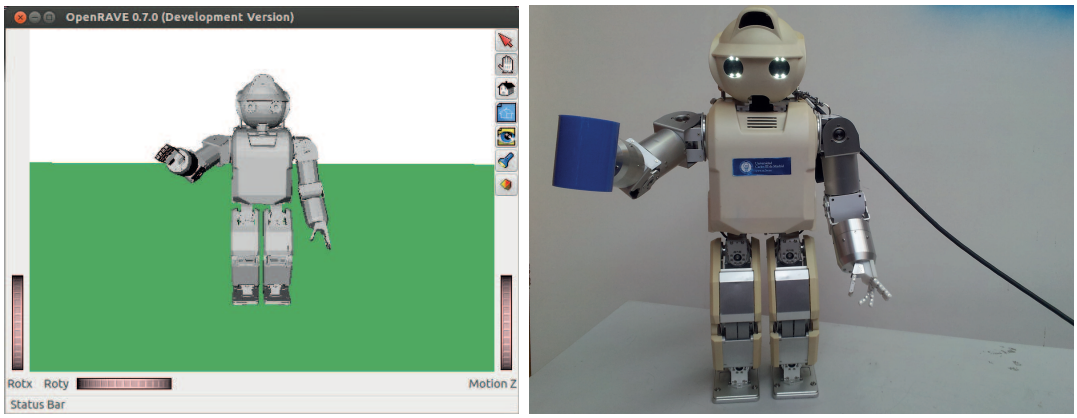


Fig. 3.6: *Teaching Demonstrations in Simulated Environment: (left) A human teacher provides demonstration to the virtual HOAP-3 robot. (right) The HOAP-3 robot reproduces the recorded demonstration of the skill.*

and databases of available capture motions, such as, the *CMU Graphics Lab Motion Capture Database* do exist. Future work for this thesis would require to also use these libraries to build a vast repertoire of demonstrations to learn robot skills. Figure 3.5 illustrates the process of motion capture from one of the motions in the CMU database.

Providing demonstrations to the robot by means of a MoCap system can be advantageous, as discussed above for *sensors on teacher*, as it can provide more precise measurements of the example execution. Also, the human teacher can perform the demonstrations naturally. However, this technique requires the use of specialized sensors, and sometimes the conditioning of a dedicated room just for these systems, making the system complex and expensive and was thus not used in this work.

Simulated Environment

Another method for providing demonstrations of the motion to the robot is by employing simulated environments. For instance, the demonstrations of the skill can be provided by a human teacher by means of a virtual simulator interface. A human teacher can provide demonstrations to a simulated robot in a virtual environment, either by a joystick, mouse or any other appropriated input device.

In teaching the robot under a simulated environment the demonstrations are recorded by the simulator's virtual interface, meaning that a record mapping exists, and $g_R(Z, \Delta) \neq I(Z, \Delta)$. The demonstrations, however, are performed on a simulated robot learner, the embodiment mapping will therefore be direct, $g_E(Z, \Delta) \equiv I(Z, \Delta)$, just like the described category of *shadowing*, as represented by the bottom left quadrant of Figure 3.3.

In this work the Open Robotics Automation Virtual Environment, (OpenRAVE) [Diankov and Kuffner, 2008], was used to develop a simulated environment to control a humanoid robot HOAP-3. A 3D model of the real HOAP-3 robot is loaded into the

Fig. 3.7: *Visual Demonstrations Teaching of a Skill: (left) A human teacher performs a demonstration. (right) The generated skeleton of the human recorded demonstration.*

OpenRAVE environment. The simulated HOAP-3 is controlled by a human operator which provides the virtual robot with demonstrations of the task. The process is illustrated in Figure 3.6 for the teaching of a skill to a humanoid robot HOAP-3 simulated in OpenRAVE.

Providing demonstrations to the robot with a simulated environment can be advantageous in that it can allow the human teacher better control over the learner and the demonstration environment. Also, it can be safer to interact with a simulated version of the robot and not with the real robot platform. However, as discussed above for *shadowing*, it requires an additional component to enable the learner to track and shadow the teacher execution. Also, the record mapping is not direct and the correspondence problem must be dealt with.

Visual Demonstration

A robot platform provided with its own set of cameras and vision sensors can also record the teacher demonstrations by itself. The human teacher would simple perform the demonstrations of the motion in front of the robot vision system. Computer vision algorithms can be employed to build a system capable of tracking human motions, as it performs the demonstrations, with the robot cameras.

In teaching the robot by visual demonstrations of the skill, the imitation relies on data recorded by sensors located externally to the executing platform, meaning that a record mapping exists, and $g_R(Z, \Delta) \neq I(Z, \Delta)$. The demonstrations, however, are performed by the human teacher and the embodiment mapping would not be identical, $g_E(Z, \Delta) \neq I(Z, \Delta)$, just like the described category of *external observation*, as represented by the bottom right quadrant of Figure 3.3.

In this work, visual demonstrations are provided to the robot learner using the stereo robot cameras equipped within the robot or via a Microsoft Kinect sensor [Microsoft, 2013], and the appropriate computer vision software modules implemented to track accordingly the motions of the human teacher performing the desired skills. The process is illustrated in Figure 3.7 for the teaching of the skill with visual demon-

strations employing the Microsoft Kinect sensor.

Providing demonstrations to the robot by visual demonstrations is advantageous in that it is a simpler and cheaper method than a MoCap system. Also, demonstrating the skills visually to a robot recording from its own sensors provides a more natural and intuitive way for a human teacher to interact with a robot as it could be able to perform as it would typically do if interacting with another human partner. However, as discussed above for *external observation*, it is less reliable as uncertainty increases from having to infer the teacher states-actions from recorded data. And it forces us to deal with both record and embodiment correspondence problems.

Each of these interfaces presents positive and negative aspects, for instance, in the context of teaching humanoid robots, imitation approaches relying on motion capture systems or visual demonstrations are particularly suitable to human teachers since they would be able to provide the demonstrations naturally, performing as it would in regular situations of human interaction, while a teleoperation or kinaesthetic approach would prove more difficult to the teacher, since controlling all the DOF of the learner could be a complex task. However, a kinaesthetic approach also provides its own advantages by allowing the demonstrator more direct control of the learner platform reproduction. Exploring the way these interfaces could be employed together to exploit complementary information and short-circuiting its respective disadvantages, would be an interesting topic.

Apart from the teaching interface employed, within the context of gathering teacher demonstrations several issues and limitations should also be addressed. Such as the manageability of the large streams of data comprising the dataset, which could be typically at rates from 60 to 1,000 data points per second [Schaal and Atkeson, 2010], which must be used for continuous learning without degradation over time. Availability of the training data, is an important limitation, for most general cases; the teacher would be incapable of providing a demonstration for every possible state of the task, dealing with under demonstrated datasets raises many questions that the learning systems need to address, most common approaches would attempt at generalizing from the existing demonstrations or re-engaging the teacher to provide additional information [Argall et al., 2009]. One major complexity comes from the high dimensionality of the learning data, in particular for more complex robot systems such as humanoid robots. Ideally learning should happen in real time; this requires computationally tractability, efficiently data management, robustness towards shifting input distributions and capacity for discovering relevant features, while automatically excluding irrelevant or redundant inputs, from hundreds or thousands of input dimensions [Schaal and Atkeson, 2010].

The performance of the learner can also be limited by poor quality of the data provided by the demonstrations. A teacher demonstration may be ambiguous, unsuccessful or suboptimal in certain areas of the state space [Argall et al., 2009]. Teacher feedback must, beyond evaluating performance, also provide correction of the executed behaviour. The gathering demonstration process is greatly influenced by the evolution of the robot interaction with the human. Several insights from the field of Human-Robot Interaction (HRI) are explored in order to make the transfer of skill more efficient [Billard et al., 2008]. The role of the teacher is one of the most im-

portant key components of attention for an efficient transfer of skill, where an active participation of the teacher, not only for demonstrating the skill but also to refine the acquired model, allows the learner to adapt the skill for particular body capacities [Calinon and Billard, 2008].

3.4 Learning a Robot Skill

An aim of this work is to learn models of robot skills for humanoid robots; the learned robot skills should latter be used to build a knowledge base of robot skills. To teach and learn the robot skills a *LfD* framework is implemented. The motivations for adopting a *LfD* approach have been outlined in the previous sections; it provides intuitive and user-friendly methods to teach tasks to a robot by demonstrating the skills, and they don't require the user to have expert programming skills. It also, reduces the cost of developing automated planning and manual programming of robot control, and speeds up the learning process, as opposed to reinforcement learning methods, reducing complexity of search spaces, giving prior knowledge of task performance.

The *LfD* approaches focuses on the development of algorithms that are generic in their representation of the skills and in the way they are generated. One common approach creates models of the skill based on sets of demonstrations performed in slightly different conditions, generalizing about the inherent variability to extract the essential components of the skill [Calinon, 2009]. Current approaches to generalizing a skill can be broadly divided into two trends: a symbolic encoding, providing a high-level representation of the skill, in which the demonstrated task is decomposed into a sequence of state-action-state transitions; and trajectory encoding, providing a low-representation for the skill, taking the form of non-linear mapping between sensory and motor information. The most promising approaches are those that encapsulate the dynamics of the movement into the encoding [Billard et al., 2008]. Generalization is important since it is not possible to demonstrate all the motions the robot is expected to perform and the learned motions must be applicable to contexts not seen during training. Working in dynamically changing environments, it is necessary to adjust the desired trajectories appropriately, or to generate new ones by generalizing from previously learned knowledge [Schaal et al., 2007]. Statistical machine learning approaches are a popular mechanism for encoding changing correlations across variables and observed variations from multiple demonstrations of the movement. Generic approaches must allow the robot to automatically extract relevant features of the task and search for a controller to optimize their reproduction.

Employing statistical learning techniques is a popular trend for dealing with the high variability inherent to the demonstrations. Traditional means were based on spline fitting techniques to deal with the uncertainty contained in several motion demonstrations [Ude, 1993], [Aleotti and Caselli, 2006]. Non-linear regression techniques were proposed as a statistical alternative to spline-based representations. A number of authors exploited the robustness of Hidden Markov Models (HMMs) for encoding temporal and spatial variations and modelling various types of motion [Tso and Liu, 1996], [Yang et al., 1997]. Popular approaches used *Gaussian Mix-*

ture Model (GMM) to encode a set of trajectories, and *Gaussian Mixture Regression (GMR)* to retrieve them [Calinon et al., 2007], [Calinon and Billard, 2008]. The work of [Chatzis et al., 2012] proposed an extension of GMR-based learning by demonstration models to incorporate concepts from the field of quantum mechanics. Different approaches [Schaal and Atkeson, 1998], used Receptive Field Weighted Regression (RFWR) to learn piecewise linear models with non-parametric regression techniques. Autonomous dynamical systems have also been proposed as an alternative approach, representing movements as mixtures of non-linear differential equations with well-defined attractor dynamics [Ijspeert et al., 2001].

Efforts in *Imitation Learning* focus on three important issues: efficient learning of motor control; organizing relation of perception and action units; and achieving modular motor control in the form of movement primitives [Schaal, 1999]. Learning motor control requires mapping world states and actions, a given motor movement can generally be formalized as a policy in terms of the expression,

$$\mathbf{u} = \pi(\mathbf{x}, \alpha) \tag{3.1}$$

which maps the state vector, $\mathbf{x}$, to a control vector of the system, $\mathbf{u}$. The vector, α, contains task specific and adjustable parameters shaping the policy. The major goal of learning control being centred around finding a generally non-linear function π, the motor control policy, adequate to reproduce a desired behaviour [Schaal and Atkeson, 2010]. *Imitation Learning* covers the algorithms by which a robot learns a policy based on demonstrated data. As mentioned in previous sections, the learning problem is segmented between gathering the demonstrations, including the choice of a demonstration technique, and deriving a policy from the demonstrations, including the selection of an algorithm for generating this policy. Significant problems to address in these approaches are the problem of extracting the relevant features of a given task, the problem of evaluating how the task should be reproduced and the problem of finding optimum controllers to generalize the acquired knowledge of various contexts [Calinon et al., 2007].

The *Robot Skills Models* learned in this chapter would form a set of basic primitives of action from which a skills knowledge base is built for generating, adapting, and reproducing more complex tasks in the right context. Suitable models of the robot skills must promote the simple learning and representation of desired trajectories. Robot skills ought to enclose all the general knowledge of the task to allow generalization of the skill for reproduction and to form full goal-directed motions and a set of basic units of action. Robot skills should also present certain properties such as autonomous behaviour without explicit time dependency and adaptation of their parameters, flexible learning, basic stability, coupling phenomena of perception and action, compact representation and ease of categorization of movement trajectories, reusable for similar and related tasks, modifiable to new tasks and contexts not seen during demonstrations; robustness against both temporal and spatial disturbances of movement in dynamic environments and allowing learning discrete and rhythmic movements.

Adopting non-linear dynamic systems theory has become an increasingly accepted

practice in several branches of science, with applications to physics, mechanics, chemistry, electromagnetism, biology, engineering, and so on. [Strogatz, 1994]. The field of neural control of movement has long suggested to model movement phenomena with dynamical systems [Kelso, 1995]. Similarly, ideas of dynamical systems theory have been introduced for developmental psychological theories of human development [Thelen and Smith, 2007]. In the field of cognitive sciences, dynamical systems theory has also been proposed as a better model for understanding the process of human cognition [van Gelder, 1995], [Beer, 2000], [Schöner, 2008], as briefly discussed in Chapter 2. In robot control theory many related approaches, such as potential fields, tried to create flexible attractor landscapes according to which any movement system must move [Okada et al.,], [Tsuji et al., Nov]. Encapsulating the dynamics of the movement into a dynamical system encoding is a promising approach to learning movement trajectories [Billard et al., 2008]. A *Dynamical Systems (DS)* approach to skill learning can offer a fast, simple and powerful formulation for representing and generating movement plans, learned from demonstrations. The *DS* framework allows to comply with the attractor dynamics of the desired behaviour, modulating it with a set of non-linear dynamic systems that form an autonomous control policy for motor control. Statistical learning techniques can be used to arbitrarily shape the attractor landscape of the control policy for encoding within the desired trajectory, moving from an initial state to an end state driven by the attractor dynamics. *DS* provide efficient and clean means for encoding a skill and fulfilling most of the desirable properties stated above. *DS* are intrinsically robust and can adapt their trajectories instantly in the face of spatio-temporal perturbations [Khansari-Zadeh and Billard, 2010a]. The *DS* do not explicitly depend on time indexing and provide closed loop control and are able to model arbitrary non-linear dynamics [Gribovskaya et al., 2010]. The *DS* can also be easily modulated to generate new trajectories that have similar dynamics, performing in areas that where not covered during demonstrations [Khansari-Zadeh and Billard, 2011]. Use of *DS* with statistical approaches permits the development of a representation of movements, encapsulating the relationships between variables and variations of the task into the dynamical systems' parameters [Calinon et al., 2012]. The *DS* approach could also be used to exploit its representational properties for movement generalization, recognition and classification [Pastor et al., 2009]. *DS* can create a rich variety of non-linear dynamic models fitted for point attractor and limit cyclic systems allowing encoding of both discrete and rhythmic movements [Ijspeert et al., 2009].

The dynamic system can be generally expressed as a differential equation,

$$\dot{x} = f(x, \theta), \tag{3.2}$$

this equation is mostly identical to Equation 3.1, except for the left-hand term f denoting a change of state, instead of a motor command π. The *DS* is conceive as a 'kinematic policy' which generates target values, in kinematic varibles, e.g., position, velocity and acceleration [Schaal et al., 2007]; appropriate controllers are needed to subsequently convert them to motor commands. Explicit time dependency is removed from the formulation of the *DS* such that the control policy becomes an autonomous dynamic system; this is advantageous as maintaining timing counters

or signals adds a burdensome level of complexity to control; additionally support for such clocking signals in biological systems is disputed [Schaal et al., 2007]. Autonomous non-linear dynamical systems are a powerful mechanisms to modulate the control policies by learning the model of the skill by building a stable estimate $\hat{f}$ of f based on the set of demonstrations. Ensuring the stability of $\hat{f}$ is a key requirement to provide a useful control policy, since non-linear *DS* are prone to instabilities. Necessary efforts are conducted into guaranteeing global asymptotic stability at the target [Khansari-Zadeh and Billard, 2011].

[Ijspeert et al., 2001] was the first work to emphasize this approach, by designing a motor representation based on dynamical systems in order to encode movements and for later replaying them in various conditions. The approach conceived the motions as movement primitives and named it *Dynamic Movement Primitives (DMP)* [Ijspeert et al., 2003]. The *DMP* can be used as a compact representation of high-dimensional planning policies. The approach starts with a simple dynamical system and transforms it, by means of an autonomous forcing term, into a non-linear system with prescribed attractor dynamics. Non-parametric regression techniques are used to shape the attractor landscapes to the demonstrated trajectories [Ijspeert et al., 2009]. *DMP* can be understood as a two dynamical system with a one-way connection such that one system drives the other one, a canonical system h which drives a transform or output system g for every considered degree of freedom. The *DMP* consists of a system of differential equations given by,

$$\tau \dot{z} = h(z, \theta),$$
$$\tau \dot{x} = g(x, f, \theta),$$

which determine the variables of internal focus x. θ is a place holder for all parameters of the system, like goals, time constants, etc. z denotes the state of the canonical system, and is a substitute for time, and f is a non-linear forcing function [Schaal et al., 2007]. The output of the system are desired positions, velocities and accelerations. A suitable controller is needed to convert them into motor commands. *Locally Weighted Regression (LWR)* was the initial method proposed to learn the system's parameters [Ijspeert et al., 2002]. [Hersch et al., Dec] extended the approach to learning trajectories in multidimensional space, Gaussian models are used to encoded the trajectories modulating the dynamical system. [Calinon et al., 2012] extended the *DMP* model by formulating the estimation of the parameters of the *DS* as a Gaussian mixture regression problem with projection in different coordinate systems. A *DS-GMR* model was proposed opening roads for developments, combining the versatility of dynamical systems and the robustness of statistical approaches.

The original *DMP* approach operated in a single dimension using a pre-defined dynamical system as a motion primitive, where the trajectory of every single DOF was modulated by its own non-linear function and transformation system separately. [Gribovskaya and Billard, 2009] investigated a method whereby the *Gaussian Mixture Models (GMM)* could directly embed the multi-variate dynamics of a motion. Their work presented a generic framework that combined *DS* movement control with *RPbD* in order to teach a robot. The framework requires two systems, a learning system

processing the data from the recorded demonstrations of the task for extracting coordination constraints and encoding the trajectory, and a motor system reproducing the dynamics of the motion while satisfying the constraints learned in the previous system [Gribovskaya and Billard, 2008]. An iterative procedure was employed to learn a statistical estimate of an arbitrary multivariate autonomous dynamical system, through the encoding of the demonstrated data with Gaussian Mixtures. The state of the robotic system ξ is assumed to be governable by an autonomous dynamical system. The motion model is driven by a first order autonomous ordinary differential equation, with a single equilibrium point,

$$\dot{\xi} = f(\xi),$$
$$\dot{\bar{\xi}} = f(\bar{\xi}) = 0$$

the problem consists of constructing an estimate $\hat{f}$ of f from the set of demonstrated trajectories, the *Gaussian Mixture Models (GMM)* are used to define the $\hat{f}$ following a statistical approach [Gribovskaya and Billard, 2009]. The *GMM* define a joint probability distribution function over a training set of demonstrated trajectories as a mixture of a finite set of Gaussian distributions. In order to generate the new trajectories, one can sample from the probability distribution function of the learned *GMM*, this process is named *Gaussian Mixture Models (GMM)*. The proposed framework has three advantages, i) it allowed generalizing the motion to unseen context; ii) provides robustness to spatio-temporal perturbations of the motion; iii) different types of dynamics can be embedded [Gribovskaya et al., 2010]. This framework allowed to learn the non-linear multivariate dynamics for cases in which this correlation between variables is important, unlike other works which generally discard information pertaining to correlation across the joints. Storing the correlations among the joints' variables can be costly; yet it is also advantageous in that the correlations contain information on features characteristic of the motion.

The non-linear *DS* are susceptible to instabilities. An important issue for these approaches is to considerer the stability of the generated control policies. Guaranteeing the estimates $\hat{f}$ results in an asymptotically stable trajectory which is, therefore, a key requirement in order to provide useful control policies. The aforementioned method is not guaranteed to result in a stable estimate of the motion. A learning procedure called *Binary Merging (BM)* was introduced by [Khansari-Zadeh and Billard, 2010b]. It tackles the problem of estimating, from the recorded demonstrations, the unknown non-linear *DS*, while ensuring local stability at the target based on the provided stability conditions. The *BM* approach can build the locally stable estimate $\hat{f}$ by minimizing iteratively the number of Gaussian functions required for achieving both asymptotic stability at the target and high accuracy in estimating the dynamics of motion. The estimated *DS* generates trajectories that accurately follow the motion dynamics based on the metric of accuracy the user defines. However, the method is sensitive to demonstrations and only effective when demonstrations are very similar.

[Khansari-Zadeh and Billard, 2011] proposed a learning method, called *Stable Estimator of Dynamical Systems (SEDS)*, to learn the parameters of the *DS* that ensure all motions to closely follow the demonstration dynamics. The approach follows sim-

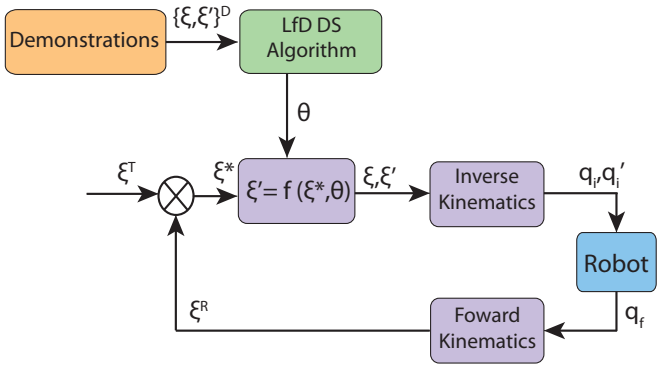


Fig. 3.8: *Control flow of LfD framework. ξ, q correspond respectively to the state variables and robot joint angles describing the motion. The learning block infer the model parameters θ from the set of demonstrations $\{\xi, \dot{\xi}\}^{\mathcal{D}}$.*

ilarly as in [Gribovskaya et al., 2010] and formulates the encoding as a control law that is driven by a first-order autonomous non-linear ODE with Gaussian Mixtures. Their work formulates the problem of computing the estimate $\hat{f}$ and the optimal values of θ by solving an optimization problem. Learning the parameters of the *GMM* proceeds as a constraint optimization problem under strict stability constraints; this ensures that the model satisfies the global asymptotic stability of the *DS* at the target [Khansari-Zadeh and Billard, 2010a]. For the optimization objective function, two different candidates are used. One function based in the *log-likelihood*, as a means of constructing the model. And a function based on the *mean square error (MSE)*, as a means of quantifying the accuracy of estimations that are based on demonstrations. The approach provides a sound ground for the estimation of non-linear *DS* which is not heuristic driven and, therefore, has the potential for much larger sets of applications. Also, by presenting the properties of being time-invariant and globally asymptotically stable at the target, the *DS* estimated with *SEDS* are able to respond immediately and appropriately to perturbations that could be encountered during reproduction of the motion.

In this work, the end-effector trajectories, ξ, in Cartesian space, of a skill motion will be modelled in terms of a dynamic systems approach, as in [Schaal et al., 2007] for an autonomous dynamical system encoding of the motion. The model of our motions is learned by estimating the non-linear function f. The frameworks presented in [Gribovskaya et al., 2010] and [Khansari-Zadeh and Billard, 2011] are followed to learn the motions as multivariate *DS* within a *LfD* statistical approach. A time independent model is estimated through a set of first order non-linear multivariate dynamical systems. Figure 3.8 presents the control flow of the learning framework. ξ^T and ξ^R are the target and real robot state, which could represent position, velocities, forces, etc. The *DS* provides the desired outputs $\xi, \dot{\xi}$. $q_i, \dot{q}_i$ and q_f corresponds to the initial and final positions and velocities of the robot joints respectively. The learning block infers the model parameters θ from the set of demonstrations $\{\xi, \dot{\xi}\}^{\mathcal{D}}$.

3.5 Encoding of a Robot Skill

As stated in the previous section, to learn, and latter to reproduce, the robot skills, a computational model of the motion is built in the framework of dynamic system approaches. The motion dynamics are estimated through a set of first order non-linear dynamical system equations. *DS* approach was proposed as alternative to traditional approaches for motor representation, like spline decomposition and regression techniques. The *DS* framework provides an effective means to encode trajectories through time-independent functions that define the temporal evolution of the motions, by representing movements as mixtures of non-linear differential equations with well-defined attractor dynamics. A *DS* model of the robot skill is built, encoding the relevant information of the demonstrated skill for reproducing the learned dynamics of the motion.

3.5.1 Problem Formalization

First let us assume that the state of the robot system can be unambiguously described using a state variable defined as ξ. And let the recorded demonstrations be the set $\mathcal{D}$ of N-dimensional demonstrate data points $\{\xi^i, \dot{\xi}^i\}_{i=1}^{\mathcal{D}}$, instances of a global motion. Further assume that the motion is governed by a first order autonomous ordinary differential equation (ODE):

$$\dot{\xi}^{\mathcal{D}} = f(\xi^{\mathcal{D}}), \tag{3.3}$$

Here, $\xi^{\mathcal{D}} \in \mathbb{R}^n$, and its time derivative $\dot{\xi}^{\mathcal{D}} \in \mathbb{R}^n$ are vectors that describe the robot motion. From Eq. 3.3 it can be seen that it follows the same form as in Eq. 3.2. To compute the evolution of the motion, giving an initial state $\xi^0 \in \mathbb{R}^n$, it is possible to integrate Eq. 3.2 through time,

$$\xi(t) = \int_0^t f(\xi, \theta)\mathrm{d}t \tag{3.4}$$

the analytical computation of the above integral are usually non-trivial, especially for complex multi-dimensional *DS*.

Let's also consider that a set of parameters, θ, can describe the function $f(\xi)$, as in Eq. 3.2, optimal values for the parameters θ can be obtained employing different statistical approaches. The learning problem is reduce to building a stable estimate $\hat{f}$ of f, and determining the parameter θ, based on the set of demonstrations, $\{\xi^i, \dot{\xi}^i\}_{i=0}^{\mathcal{D}}$. The function f, $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$, is considered to be a non-linear continuous and continuously differentiable function with a single equilibrium point. Without loss of generality, the attractor, $\bar{\xi}$, can be transferred to the origin, $\bar{\xi} = 0$, so that $f(\bar{\xi}) = f(0) = 0$ and by extension $\hat{f}(\bar{\xi}) = \hat{f}(0) = 0$.

$$\begin{aligned} \dot{\xi} &= f(\xi), \\ \dot{\bar{\xi}} &= f(\bar{\xi}) = 0 \end{aligned} \tag{3.5}$$

The motion of the system is uniquely determined by its state ξ. Choosing the appropriate state variables has an important impact on the dynamics to be learned. Here, motions are to be represented in kinematic coordinates, the desired outputs are position, velocities and accelerations, which could be in in joint space or task space. It is assumed that there are appropriate controllers that convert kinematic variables into motor commands.

Non-linear Regression Techniques

Regression is a problem in statistical analysis for estimating the relationships among variables. The non-linear regression techniques focus on building a continuous mapping function $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$, the function f is a non-linear combination of the model parameters, building f is based on determining the set of parameters θ during training based on the set, $\mathcal{D}$, of training data points, $\{\xi_I^i, \xi_O^i\}_{i=1}^{\mathcal{D}}$, with $\xi_I^i \in \mathbb{R}^n$ and $\xi_O^i \in \mathbb{R}^m$ corresponding to the input and output variables respectively. The value ξ_O can be predicted from the input ξ_I with the estimate of f,

$$\hat{\xi_O} = f(\xi_I^*, \theta) \tag{3.6}$$

notice the similarities of this statement with the previous formalization of the *DS* learning problem.

There are numerous regression techniques to build the estimate of f, the statistical methods can be broadly divided into parametric and non-parametric approaches. The non-parametric methods are advantageous in that they make little assumptions about the form of the underlying distribution, they are also well suited to accurately perform data fitting in low-dimensional spaces. However, they suffer from the curse of dimensionality. The parametric methods are better suited to model multivariate datasets, and deal with problems of regression on multi-dimensional data. However, to choose the underlying parameters effectively they rely on heuristical methods [Hastie et al., 2009]. Existing approaches to statistical estimating of f mostly relied on either *Gaussian Process Regression (GPR)* [Rasmussen and Williams, 2006], [Schneider and Ertel, 2010], *Gaussian Mixture Regression (GMR)* [Hersch et al., 2008], [Calinon et al., 2010], or *Locally Weighted Regression*, *Locally Weighted Projection Regression (LWPR)* [Vijayakumar and Schaal,], [Grollman and Jenkins, 2008].

Gaussian Process

Gaussian Process Regression (GPR) provides an estimate of the function f by assuming it as a Gaussian process, in which any set of samples has a joint Gaussian distribution. A set of training data points with uni-dimensional function values $\xi_I = {\xi_I^i}_{i=1}^{\mathcal{D}}$, and $\xi_O = {\xi_O^i}_{i=1}^{\mathcal{D}}$, representing respectively the input and output variables. By conditioning the multivariate Gaussian distribution on the training data, for any point ξ_I^*, the *GPR* is obtained,

$$f(\xi_I^*) \mid \xi_I, \xi_O \sim \mathcal{N}(\mu(\xi_I^*), \Sigma(\xi_I^*)) \tag{3.7}$$

where the estimate $\mu(\xi_I^*)$ and the variance $\Sigma(\xi_I^*)$ are given by

$$\mu(\xi_I^*) = K(\xi_I^*, \xi_I)(K(\xi_I, \xi_I) + \sigma_n I)^{-1})\xi_O$$
$$\Sigma(\xi_I^*) = K(\xi_I^*, \xi_I^*) - K(\xi_I^*, \xi_I)(K(\xi_I, \xi_I))^{-1}K(\xi_I, \xi_I^*)$$

with K, symmetric matrices representing the evaluation of the *GP* covariance function across the specified variables.

The formulation of *GPR* is only applicable to multi-input single-output datasets, for datasets with multiple outputs it is necessary to train a separate *GPR* model for every output dimension. The *GPR* method builds an accurate estimate of non-linear functions, however, it is ill-suited for applications requiring fast computation. the computational costs of *GPR* scale cubically with the number of training examples.

Gaussian Mixture

Gaussian Mixture Regression (GMR) is a non-linear regression technique which operates on the joint probability $\mathcal{P}(\xi_I; \xi_O)$. The joint probability is formed by superposition of linear Gaussian functions,

$$\mathcal{P}(\xi_I; \xi_O) = \sum \pi \mathcal{N}(\xi_I; \xi_O \mid \mu, \Sigma) \tag{3.8}$$

where π, μ and Σ are respectively the prior, mean and covariance matrix of the Gaussian function $\mathcal{N}$.

Given the joint distribution $\mathcal{P}(\xi_I; \xi_O)$ and input point ξ_I^*, the *GMR* process follows the output from the posterior mean estimate of the conditional distribution,

$$\hat{\xi_O} = f(\xi_I^*; \theta) = \mathbb{E}[\mathcal{P}(\hat{\xi_O} \mid \xi_I^*; \theta)] \tag{3.9}$$

with $\theta = [\pi, \mu, \Sigma]$ the parameters of the Gaussian functions.

A more expansive description of the *GMR* process would be given later in this section. The *GMR* method provides an alternative to modelling non-linear trajectories. It usually requires fewer parameters in comparison to other methods, yet it is less accurate. One critical concern with *GMR* based approaches is that they require heuristic methods to determine an optimal number of Gaussian kernels, also, the final results are sensitive to initialization.

Locally Weighted Projection

Locally Weighted Projection Regression (LWPR) is an incremental regression technique which provides an estimate of f in terms of the output from a set of local regions, defined with a Gaussian function,

$$w(\xi) = e^{-(\xi-\mu)^\top W(\xi-\mu)} \tag{3.10}$$

where μ are the centres and W is a positive semi-definite distance metric, determining the influence of the region. The output prediction is computed as the non-linear weighted sum of the output of all regions,

$$\hat{\xi_O} = f(\xi_I^*) = \frac{1}{\sum w(\xi_I^*)} \sum w(\xi_I^*) r(\xi_I^*) \tag{3.11}$$

The *LWPR* method offers in comparison cost-efficiency for non-linear function approximation. *LWPR* describes the system through a finite combination of Gaussian functions. The parameters are estimated in one-shot learning through linear regression. However, the approach is very sensitive to the choice of parameters at initialization and relies on manual tuning to achieve high accuracy.

Regardless of the advantages or weakness of these approaches, they cannot be used as is to estimate the *DS* of Eq. 3.5 since they do not take into account the stability of the dynamical system they model [Khansari-Zadeh and Billard, 2010b].

Dynamic Motor Primitives

The *Dynamic Motor Primitives (DMP)* method [Ijspeert et al., 2009], was proposed to learn the attractor dynamics of the motion and to deal with the instability issues. The *DMP* can be used to generate one dimensional movements with a basic point attractor system instantiated by the second order dynamics as,

$$\begin{aligned} \tau\dot{z} &= \alpha_z(\beta_z(g-y)-z)+f \\ \tau\dot{y} &= z \end{aligned} \tag{3.12}$$

with g the goal state, α_z, β_z time constants, τ a temporal scaling factor, and $y, \dot{y}$ correspond to a desired position and velocity.

For appropriate parameter setting and with $f = 0$ Eq. 3.12 form a globally stable linear dynamic system with g as an unique attractor [Schaal et al., 2007]. The function f is a non-linear function which can be learned to allow the generation of arbitrary complex trajectories. The non-linear function f can be defined in the form of,

$$f(x, g, y_0) = \frac{\sum_{i=1}^{N} \psi_i w_i x}{\sum_{i=1}^{N} \psi_i}(g - y_0) \tag{3.13}$$

where $\psi_i = \exp\left(-h_i(x-c_i)^2\right)$ are Gaussian basis function with center c_i and with h_i, and w_i are learnable adjustable weight that shapes the trajectory. The function f does not directly depend on time, but on a phase variable, x,

$$\tau\dot{x} = -\alpha_x x \tag{3.14}$$

with α_x a pre-defined constant. The *DMP* can be understood as two dynamical system with a one-way connection such that one system drives the other, with the canonical system in Eq. 3.14 driving the output system in Eq. 3.12.

For learning the parameters, a non-parametric regression technique from locally weighted learning can be used to generate the function approximator [Ijspeert et al., 2009]. This method allows us to determine automatically the necessary number of basis functions N, their centres c_i, and widths h_i. For every basis function ψ_i, which defines a small region in input space x, any point that falls into this region is used to perform a linear regression, which can be formalized as weighted regression. The method creates a piecewise linear approximation of f, in which each linear function piece belongs

to one of the basis functions. Other function approximators can also be used, like radial basis function networks, mixture models, Gaussian Process regression, etc., for example [Calinon et al., 2012].

The *DMP* approach however presents two drawbacks; the phase variable employed to modulate the dynamics makes the system time dependent and sensitive to temporal perturbations. Also, a *DS* is learned separately for each dimension, and a heuristic is needed to synchronize for modelling multi-dimensional systems, this neglects the combined effect of all the dimensions in the motion.

3.5.2 Multivariate Gaussian Mixtures

To learn the multi-variate dynamics of a motion trajectory, here, an approach from [Gribovskaya and Billard, 2009] has been followed, as outlined in section 3.4. In their work an iterative procedure was employed to learn a statistical estimate of an arbitrary multivariate autonomous dynamical system, *Gaussian Mixture Models (GMM)* are used to directly embed the multi-variate dynamics of a motion through the encoding of the demonstrated data.

The state of the robotic system ξ is assumed to be governable by an autonomous dynamical system, with a single equilibrium point, as per Eq. 3.5. And the set of N-dimensional demonstrated data points be represented as $\{\xi^i, \dot{\xi}^i\}_{i=1}^{\mathcal{D}}$, as described in the problem formalization in 3.5.1. A probabilistic framework is employed to build an estimate $\hat{f}$, of the non-linear state transition map f, based on the set of demonstrations. The dynamics of the motion are learned thus, by modelling the estimate $\hat{f}$ via a finite mixture of Gaussian functions, f is defined as a non-linear combination of a finite set of Gaussian kernels using the *GMM* [Gribovskaya et al., 2010].

Gaussian Mixture Models

Employing mixture models is a popular approach for the statistical modelling of a wide variety of random phenomena. Mixture distributions provide a convenient framework to model unknown distributional shapes, for density approximation of continuous or binary data [Mclachlan and Peel, 2000]. A mixture model of $\mathbf{K}$ components is defined by a probability density function,

$$p(\xi) = \sum_{k=1}^{\mathbf{K}} p(k)p(\xi \mid k) \tag{3.15}$$

where ξ is a data point, $p(k)$ is the prior probability and $p(\xi \mid k)$ is the conditional probability.

Given our set of demonstrated data points, $\{\xi^i, \dot{\xi}^i\}_{i=1}^{\mathcal{D}}$, each recorded point in the trajectories is associated with a probability density function. The *GMM* define a joint probability distribution $p(\xi^i, \dot{\xi}^i)$ of the training set of demonstrated trajectories as a mixture of the $\mathbf{K}$ Gaussian multivariate distributions $\mathcal{N}^k$, with π^k, μ^k, and Σ^k, respectively the prior, mean and covariance matrix, parameters of the Gaussian component k.

The parameters in Eq. 3.15 become,

$$\begin{aligned} p(k) &= \pi^k \\ p(\xi \mid k) &= \mathcal{N}(\xi; \mu^k, \Sigma^k) \end{aligned} \tag{3.16}$$

The joint probability distribution, $p(\xi, \dot{\xi})$, for the *GMM* is given by,

$$\begin{aligned} p(\xi, \dot{\xi}; \theta) &= \frac{1}{K} \sum_{k=1}^{\mathbf{K}} \pi^k \mathcal{N}^k(\xi, \dot{\xi}; \mu^k, \Sigma^k) \\ \text{with} \quad \mu^k = \{\mu^k_\xi; \mu^k_{\dot{\xi}}\} \quad &\text{and} \quad \Sigma^k = \begin{bmatrix} \Sigma^k_\xi & \Sigma^k_{\xi\dot{\xi}} \\ \Sigma^k_{\dot{\xi}\xi} & \Sigma^k_{\dot{\xi}} \end{bmatrix} \end{aligned} \tag{3.17}$$

where the probability density function of each Gaussian, $\mathcal{N}^k(\xi^i, \dot{\xi}^i; \mu^k, \Sigma^k)$, in the model is then given by:

$$\mathcal{N}^k(\xi, \dot{\xi}; \mu^k, \Sigma^k) = \frac{1}{\sqrt{(2\pi)^{2n}|\Sigma^k|}} e^{\frac{-1}{2}(([\xi,\dot{\xi}]-\mu^k)^T(\Sigma^k)^{-1}([\xi,\dot{\xi}]-\mu^k))} \quad \forall k \in 1 \ldots \mathbf{K} \tag{3.18}$$

The mixture of Gaussian functions would estimate the non-linear function f, thus the unknown parameters of f, θ, becomes the prior, π^k, the mean, μ^k, and the covariance matrix, Σ^k, of the $\mathbf{K}$ Gaussian functions, such that $\theta^k = (\pi^k, \mu^k, \Sigma^k)$, defined as in Eq. 3.17.

The mixture modelling method builds a coarse representation of the data density through a fixed number of mixture components. By considering an adequate number of Gaussian functions, and adjusting their means and covariances matrix parameters, almost any continuous density can be approximate to arbitrary accuracy. Finding the optimal number of components is not trivial and various methods can be found, such as, the *Bayesian Information Criterion (BIC)* [Schwarz, 1978], or the *Deviance Information Criterion (DIC)* [Spiegelhalter et al., 2002]. The parameters $\theta = (\pi, \mu, \Sigma)$ of function f, governed the form of the Gaussian mixture distribution. To learn the parameters a *Maximum Likelihood Estimation* of the mixture parameters is performed. *EM* proceeds by maximizing the likelihood that the complete model represents the training data well.

$$\mathcal{L}(\xi, \Theta) = \sum_{n=1}^{\mathbf{N}} \ln(p(\xi^n \mid \Theta)) \tag{3.19}$$

First, the model is initialized using the k-means clustering algorithm starting from a uniform mesh and it is then refined iteratively through *Expectation-Maximization (EM)* [Dempster et al., 1977], to find the maximum likelihood function of equation 3.17, from Eq. 3.19 as,

$$\mathcal{L}p(\xi, \dot{\xi}) = \sum_{n=1}^{\mathbf{N}} \ln \left\{ \sum_{k=1}^{\mathbf{K}} \pi^k \mathcal{N}((\xi^n, \dot{\xi}^n) \mid \mu^k, \Sigma^k) \right\} \tag{3.20}$$

The parameters $\theta^k = (\pi^k, \mu^k, \Sigma^k)$ of the *GMM* are then estimated iteratively until convergence, trough alternating between an expectation (E) step and a maximization (M) step. The *E-step* creates a function for the expectations of the log-likelihood, using current estimate for the parameters. The *M-step* computes the parameters, maximizing the expected log-likelihood of the *E-step*, these estimates of the parameter are used for determining the next *E-step*. The iterations stop when the increase of the log-likelihood becomes smaller than a threshold, $\frac{\mathcal{L}^{t+1}}{\mathcal{L}^t} < threshold$, with the log-likelihood, $\mathcal{L}$, defined as in Eq. 3.19 and 3.20.

E-step:

$$
\begin{aligned}
p^{k,n}_{(t+1)} &= \frac{\pi^k_{(t)} \mathcal{N}((\xi^n, \dot{\xi}^n) \mid \mu^k_{(t)}, \Sigma^k_{(t)})}{\sum_{k=1}^{\mathbf{K}} \pi^k_{(t)} \mathcal{N}((\xi^n, \dot{\xi}^n) \mid \mu^k_{(t)}, \Sigma^k_{(t)})} \\
E^k_{(t+1)} &= \sum_{n=1}^{\mathbf{N}} p^{k,n}_{(t+1)}
\end{aligned}
\tag{3.21}
$$

M-step:

$$
\begin{aligned}
\pi^k_{(t+1)} &= \frac{E^k_{(t+1)}}{\mathbf{N}} \\
\mu^k_{(t+1)} &= \frac{\sum_{n=1}^{\mathbf{N}} p^{k,n}_{(t+1)} (\xi^n, \dot{\xi}^n)}{E^k_{(t+1)}} \\
\Sigma^k_{(t+1)} &= \frac{\sum_{n=1}^{\mathbf{N}} p^{k,n}_{(t+1)} ((\xi^n, \dot{\xi}^n) - \mu^k_{(t+1)})((\xi^n, \dot{\xi}^n) - \mu^k_{(t+1)})^\top}{E^k_{(t+1)}}
\end{aligned}
$$

A more in deep theoretical analysis of the *Gaussian Mixture Models (GMM)* can be found on [Dasgupta and Schulman, 2000], [Calinon, 2009]. Figure 3.9 illustrates the learning process and encoding of a training data set into a model of mixtures of Gaussian functions. First, several demonstrations of a trajectory are recorded to build the $\mathcal{D}$ dataset. A model of the trajectories is built encoding the given demonstrations with $\mathbf{K}$ Gaussian distributions, defined by the μ and Σ parameters. To generate a new trajectory from the *GMM*, one then can sample from the probability distribution function $p(\xi, \dot{\xi})$, this process is called *Gaussian Mixture Regression (GMR)*.

Gaussian Mixture Regression

Gaussian Mixture Regression (GMR) is used for retrieving a generalized trajectory made up of a set of trajectories used to train the model, where the generalized trajectory is not part of the dataset but instead encapsulates all of its essential features [Calinon, 2009].

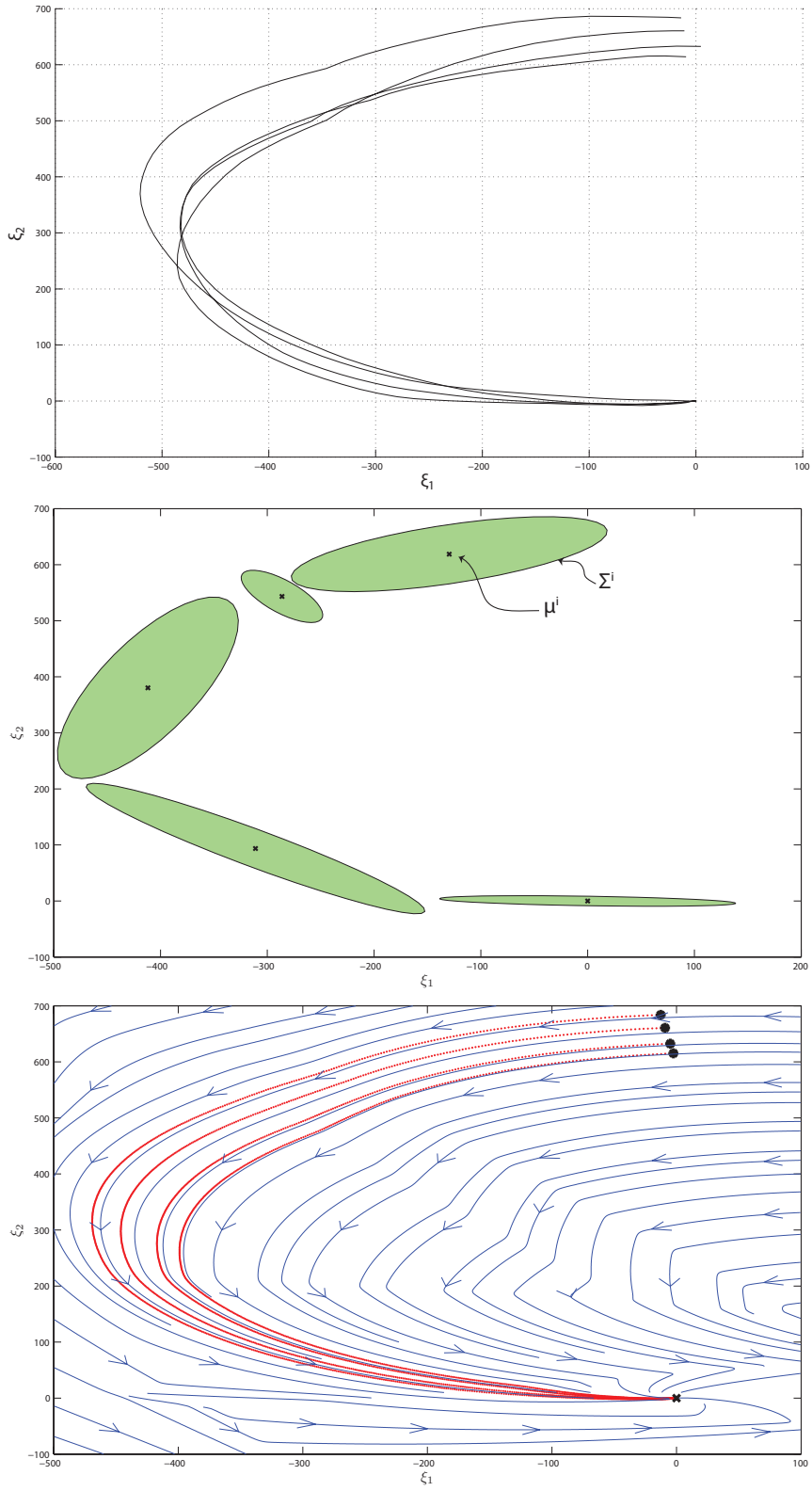


Fig. 3.9: *Illustration of the learning process with GMM-GMR. (top) Recorded training data of the demonstrated trajectories. (center) The learned GMM model represented by ellipses centre at μ^i, magnitude and direction of the ellipses are given by the eigenvectors and eigenvalues of Σ^i. (bottom) Reproduction of several trajectories trough GMR.*

The *GMM* computes a joint probability density function for the input and the output so that the probability of the output conditioned on the input are a Mixture of Gaussian. So it is possible after training, to recover the expected output variable $\hat{\dot{\xi}}$, given the observed input in ξ.

Given the joint probability distribution, $p(\xi, \dot{\xi})$, from Eq. 3.17 and a input query point ξ^*, the *GMR* process takes the conditional mean estimate of $p(\dot{\xi} \mid \xi^*)$, the estimate of our function $\hat{\dot{\xi}} = \hat{f}(\xi^*)$ can be expressed by,

$$\hat{\dot{\xi}} = \sum_{k=1}^{K} h^k(\xi^*)(\Sigma^k_{\dot{\xi}\xi}(\Sigma^k_{\xi})^{-1}(\xi^* - \mu^k_{\xi}) + \mu^k_{\dot{\xi}}) \tag{3.22}$$

$$\text{where,} \quad h^k(\xi) = \frac{p(\xi; \mu^k_{\xi}, \Sigma^k_{\xi})}{\sum_{k=1}^{K} p(\xi; \mu^k_{\xi}, \Sigma^k_{\xi})}$$

$$\text{with} \quad h^k(\xi) > 0 \quad \text{and} \quad \sum_{k=1}^{K} h^k(\xi) = 1$$

A review of theoretical considerations of the *GMR* can be found in [Sung, 2004], [Cohn et al., 1996]. The *GMM* encoding of the demonstrations and *GMR* reproduction of the learned motions process is illustrated in Figure 3.9. The model of the trajectories are learned from several demonstrations and then encoded as a mixture of Gaussian distributions. To reproduce the trajectories one sample from the probability distribution of the *GMM* trough the *Gaussian Mixture Regression* process. The *GMR* approximates the dynamical systems through a non-linear weighted sum of local linear models. The process for encoding the dynamics of a motion through *Gaussian Mixture Models*, and *Gaussian Mixture Regression*, is illustrated in Figure 3.10.

The notation of Eq. 3.22 can be simplified through a change of variable where,

$$\begin{cases} \mathbf{A}^k = \Sigma^k_{\dot{\xi}\xi}(\Sigma^k_{\xi})^{-1} \\ \mathbf{b}^k = \mu^k_{\dot{\xi}} - \mathbf{A}^k \mu^k_{\xi} \\ h^k(\xi) = \frac{p(\xi; \mu^k_{\xi}, \Sigma^k_{\xi})}{\sum_{k=1}^{K} p(\xi; \mu^k_{\xi}, \Sigma^k_{\xi})} \end{cases} \tag{3.23}$$

Substituting Eq. 3.23 into Eq. 3.22 produces an expression of the *GMR* as a non-linear sum of linear dynamical systems,

$$\dot{\xi} = \hat{f}(\xi) = \sum_{k=1}^{K} h^k(\xi)(\mathbf{A}^k \xi + \mathbf{b}^k) \tag{3.24}$$

Rewriting Eq. 3.22 in this way is useful to study the influence of each Gaussian and the stability of the estimate $\hat{f}$. Stability of the system is governed by the *GMR* parameters, the matrices $\mathbf{A}^k$, $\mathbf{b}^k$ and weighting term h^k, which are learned during training. Figure 3.10 represents the influence of the *GMR* parameters in the final

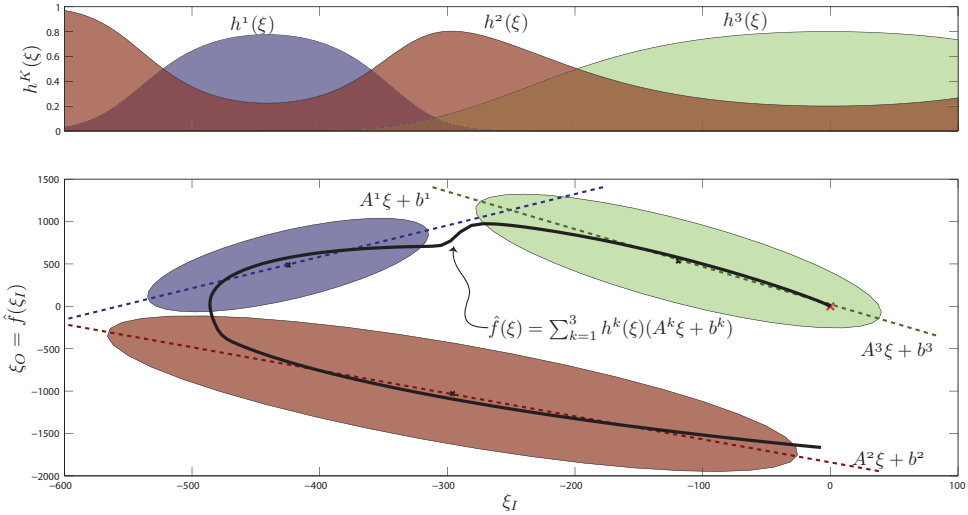


Fig. 3.10: *Illustration of the GMR inference process for reproducing learned trajectories. (top) The non-linear weights $h^k(\xi)$, as defined by Eq. 3.23, give a relative measure of the importance of each Gaussian contribution to the estimate $\hat{f}$ at point ξ. (bottom) The estimate $\hat{f}$ is expressed as a non-linear sum of DS, as per Eq. 3.24. The linear dynamics of every $\mathbf{A}^k\xi + \mathbf{b}^k$ correspond to a line equation with slope $\mathbf{A}^k$ that runs through the centre μ^k. Given an observed input ξ_I, the value of ξ_O is estimated from $\hat{f}$.*

reproduction. Each linear dynamics corresponds to a line that passes through the centres μ^k with slope $\mathbf{A}^k$. The non-linear weighting term, h^k, in Eq. 3.24 gives a measure of the relative influence of each Gaussian locally. Due to the influence of the non-linear weighting term, h^k, the estimate function $\hat{f}(\xi)$ is also non-linear and presents enough flexibility as to model a wide variety of motions. However, it cannot be guaranteed that the system will be asymptotically stable, and the resulting non-linear model $f(\xi)$ can contain spurious attractors or limit cycles even for simple 2D models [Khansari-Zadeh and Billard, 2011].

[Gribovskaya et al., 2010] proposes a modification of the *GMM* procedure to build the mixture resulting in an estimate, locally stable around the target (*GMM-DS*). It is assumed that in the neighbourhood of the origin, the system is governed solely by the last $\mathbf{K}$ Gaussian. In order to guarantee the convergence to the target additional synthetic data is generated within a small neighbourhood around the origin. In addition, the center of the last Gaussian is set at the target and it is not updated during training. The system would be asymptotically stable by ensuring that the eigenvalues of $\mathbf{A}^\mathbf{K}$, from Eq. 3.24, are all strictly negative. The stability is estimated locally within a subregion $\mathbb{C}$, inside the robot's workspace. The function is approximated in $\mathbb{C}$, referred to as the region of applicability of the learned dynamics, such that,

$$\begin{aligned} \hat{f} &: \mathbb{C} \rightarrow \mathbb{C} \\ \hat{f}(\xi) &\cong f(\xi), \forall \xi \in \mathbb{C} \end{aligned} \tag{3.25}$$

Algorithm: Multivariate Dynamic Systems *GMM* [Gribovskaya et al., 2010]

Input: Demonstrations dataset $\{\xi^i, \dot{\xi}^i\}_{i=1}^{\mathcal{D}}$.

1. Initialize stability subregion $\mathbb{C}$.
2. Compute synthetic data at the target attractor $\bar{\xi}$, to guarantee convergence.
3. Choose initial number of Gaussian components $\mathbf{K}$.
4. **LOOP** Until stability verification is satisfied.
5. Initialize the *GMM* parameters with k-means clustering.
6. Train the joint probability distribution $p(\xi^i, \dot{\xi}^i) \sim \mathcal{N}(\xi; \theta)$ with Expectation Maximization.
7. Verify local stability at the origin.
8. **IF** not asymptotically stable at the origin.
9. **THEN** increase the number of Gaussian components.
10. **END**.
11. **END**

Output: $\hat{f} = \{\mathcal{N}^1(\xi; \theta^1), \cdots, \mathcal{N}^{\mathbf{K}}(\xi; \theta^{\mathbf{K}})\}$

Tab. 3.1: *Encoding Multivariate Dynamics with GMM-GMR.*

Initialization of $\mathbb{C}$ is data-driven and its size is defined by the amplitude of the training dataset. After training, initial guesses regarding $\mathbb{C}$ are re-estimated, following a numerical procedure, to empirically verify that $\mathbb{C}$ is a region of attraction and that all the trajectories converge toward the origin; it does not include any other attractors. This approach presented the drawback that it cannot ensured to find even a locally stable estimate and it gave no explicit constraint on the form of the Gaussian functions to ensure stability [Khansari-Zadeh and Billard, 2011].

Table 3.1 summarizes the procedure to model the motion dynamics through Multivariate Gaussian Mixtures, employing *GMM* and *GMR* as proposed in the work by [Gribovskaya et al., 2010].

3.5.3 Binary Merging

[Khansari-Zadeh and Billard, 2010b] proposed a method, as outlined in section 3.4, to tackle the problem of estimating the non-linear *DS* while ensuring local stability at the target. Their work provides a set of stability conditions that can be used to ensure local asymptotic stability of f when it is formulated with a mixture of Gaussian functions.

Ensuring that the estimate $\hat{f}$ of the non-linear dynamical system results in trajectories that asymptotically converge on the attractor, is a key requirement. Here, $\hat{f}$ is a stable estimate of $f \in \mathbb{R}^n$ if it has a single attractor $\bar{\xi} : f(\bar{\xi}) = 0$ and every

trajectory generated by f asymptotically converges to $\bar{\xi}$,

$$\lim_{t\to\infty} \hat{f}(\xi^t) = \bar{\xi} \qquad \forall \xi^t \in \mathbb{R}^n \tag{3.26}$$

[Khansari-Zadeh and Billard, 2010b] defined a region $\mathbb{D} \subset \mathbb{R}^n$ which covers entirely the part of the state space spanned by the demonstrations, including the origin, where the motion can be accurately estimated with $\hat{f}$,

$$\begin{aligned} \mathbb{D} &= \{ \quad \xi \in \mathbb{R}^n \quad : \quad p(\xi) \geq \delta^k \quad \} \\ \delta^k &= \alpha \min(p(\xi^i)) \quad : \quad k = 1 \dots \mathbf{K}, i = 1 \dots \mathcal{D} \end{aligned} \tag{3.27}$$

where $p(\xi)$ is the probability of ξ as estimated from Eq. 3.17, and $\alpha : 0 < \alpha \leq 1$ is a constant. The definition of δ ensures that all data points are included in $\mathbb{D}$. The region $\mathbb{D}$ is then partitioned into $\mathbf{K}$ pairwise, disjointed continuous subregions, Ω^k, via the hyperplanes Φ^k,

$$\Phi^k : (\xi - \mu_\xi^k)\top \cdot v^k = 0 \tag{3.28}$$

with v^k being the eigenvector pointing towards the direction of motion. Φ^k is the hyperplane through μ_ξ^k and normal to v^k. Each subregion, Ω^k, is a part of $\mathbb{D}$ that is defined by,

$$\Omega^k = \hat{\Omega}^k \bigcap \mathbb{D} \quad \forall k \in 1 \dots \mathbf{K} \tag{3.29}$$

For each subregion $\Omega^k \subset \mathbb{D}$, $k = 2 \dots \mathbf{K}$, the estimate given by Eq. 3.24 is truncated so that the dynamics are driven solely by the two dominant Gaussian functions $\mathcal{N}^k$ and $\mathcal{N}^{k-1}$. The estimate for points in partition Ω^1 are set by construction to only be influenced by the dominant Gaussian $\mathcal{N}^1$. Thus becoming,

$$\dot{\xi} = f(\xi) = \begin{cases} \mathbf{A}^1\xi + \mathbf{b}^1 & \forall \xi \in \Omega^1 \\ h^{k-1}(\xi)(\mathbf{A}^{k-1}\xi + \mathbf{b}^{k-1}) + & \\ \qquad h^k(\xi)(\mathbf{A}^k\xi + \mathbf{b}^k) & \forall \xi \in \Omega^k, k \in 2 \dots \mathbf{K} \end{cases} \tag{3.30}$$

The origin, the attractor, of Eq. 3.30 is asymptotically stable if the parameter of f, μ^k and Σ^k, are constructed such that,

$$\begin{cases} \mu_\xi^{\mathbf{K}} = \bar{\xi} = 0 \\ \mu_{\dot{\xi}}^{\mathbf{K}} = -\dfrac{p(0 \mid \mathbf{K}-1)}{p(0 \mid \mathbf{K})}(\mu_{\dot{\xi}}^{\mathbf{K-1}} - \Sigma_{\dot{\xi}\xi}^{\mathbf{K}-1}(\Sigma_\xi^{\mathbf{K}-1})^{-1}\mu_\xi^{\mathbf{K}-1}) \\ \Sigma_{\dot{\xi}\xi}^{\mathbf{K}}(\Sigma_\xi^{\mathbf{K}})^{-1} + (\Sigma_\xi^{\mathbf{K}})^{-1}(\Sigma_{\dot{\xi}\xi}^{\mathbf{K}})^\top \prec 0 \end{cases} \tag{3.31a}$$

$$\begin{cases} (\xi - \mu_\xi^1)^\top(\Sigma_\xi^1)^{-1}\dot{\xi} < 0 \quad \forall \xi \in \Omega^1 \\ (\xi - \mu_\xi^{k-1})^\top(\Sigma_\xi^{k-1})^{-1}\dot{\xi} > (\xi - \mu_\xi^k)^\top(\Sigma_\xi^k)^{-1}\dot{\xi} \begin{cases} \forall \xi \in \Omega^k \\ \forall \xi \neq 0 \\ k = 2 \dots \mathbf{K} \end{cases} \end{cases} \tag{3.31b}$$

$$(v^k)^\top \dot{\xi} > 0 \quad \forall \xi \in \Phi^k, \forall k \in 1 \dots \mathbf{K}-1 \tag{3.31c}$$

$$\mathbb{D} \quad \text{is an invariant set} \tag{3.31d}$$

Putting together the conditions in Eq. 3.31 the system becomes locally asymptotically stable at the origin in the region defined by $\mathbb{D}$ [Khansari-Zadeh and Billard, 2010b]. It is necessary for the estimate to be not only stable, according to the stated definition, but also should follow closely the dynamics of the demonstration. This is evaluated through a measure of accuracy $\bar{e}$ with which $\hat{f}$ approximates the demonstration dynamics. This is quantified by measuring the discrepancy between the direction and amplitude of the estimated and observed velocity vectors for all the training points [Khansari-Zadeh and Billard, 2010b].

$$\bar{e} = \frac{1}{\mathcal{D}} \sum_{i=1}^{\mathcal{D}} \left(\frac{1}{\mathcal{T}^i} \sum_{t=0}^{\mathcal{T}^i} r \left(1 - \frac{(\dot{\xi}^{t,i})^\top f(\xi^{t,i})}{\parallel \dot{\xi}^{t,i} \parallel \parallel f(\xi^{t,i}) \parallel \epsilon} \right)^2 + \dots \right. \\ \left. q \left(\frac{(\dot{\xi}^{t,i} - f(\xi^{t,i}))^\top (\dot{\xi}^{t,i} - f(\xi^{t,i}))}{\parallel \dot{\xi}^{t,i} \parallel \parallel \dot{\xi}^{t,i}) \parallel \epsilon} \right) \right)^{0.5} \tag{3.32}$$

where r and q are positive scalars that weight the relative influence of each factor, and ϵ is a very small positive scalar. An estimate of the dynamics is considered accurate if $\bar{e} \leq e_{max}$, with e_{max} a given a maximal acceptable error.

The *Binary Merging (BM)* learning approach proceeds in two steps to build the stable estimate of f. A first step that initializes the model with a maximum number of possible Gaussian functions. And a second step that tries to reduce the number of Gaussian functions to a minimum, satisfying the stability criteria while also keeping the error of the estimates below maximal error e_{max}.

The initialization step, first with a sample alignment the demonstration trajectories are aligned. The time stamps that result from the sample alignment are used to initialize the Gaussian mixture. The parameters, $\theta^k = (\pi^k, \mu^k, \Sigma^k)$, corresponding to each Gaussian function are then computed as,

$$\theta^k = \begin{cases} \pi^k = \dfrac{1}{\mathbf{K}} \\ \mu^k = mean(\Xi^k) \\ \Sigma^k = cov(\Xi^k + \sigma^0 \mathbf{I} \end{cases} \tag{3.33a}$$

$$\theta^{\mathbf{K}} = \begin{cases} \pi^{\mathbf{K}} = \dfrac{1}{\mathbf{K}} \\ \mu^{\mathbf{K}} = \quad \text{computed following Eq. 3.31} \\ \Sigma^{\mathbf{K}} = \sigma^0 \begin{bmatrix} \mathbf{I} & -\mathbf{I} \\ -\mathbf{I} & \mathbf{I} \end{bmatrix} \end{cases} \tag{3.33b}$$

where $\Xi^k = \{\xi^{k,i}, \dot{\xi}^{k,i}\}_{i=1}^{\mathcal{D}}$ denotes a subset of the demonstrations that belong to the k Gaussian function, σ^0 is a small positive scalar to avoid numerical instability, and $\mathbf{I}$ is an identity matrix of the proper size.

The iteration step proceeds as follows, a pair of adjacent Gaussian functions, $\{\mathcal{N}^k, \mathcal{N}^{k+1}\}$ are picked randomly and merged into a single Gaussian by computing

Algorithm: Binary Merging *BM* [Khansari-Zadeh and Billard, 2010b]

Input: Demonstrations dataset $\{\xi^i, \dot{\xi}^i\}_{i=1}^{\mathcal{D}}$.

1. Initialize parameters. (r, q, e_{max}).
2. Transfer the target attractor $\bar{\xi}$ to the origin.
3. Sample align demonstrations to length $\mathcal{T}$.
4. Define time indices $t^k = k, \forall k \in 1 \cdots \mathcal{T}$.
5. Initialize the *GMM*, with the time indices $t^k = k$ for $\mathbf{K} = \mathcal{T}$.
6. **LOOP** while $\mathbf{K} > 1$ and further merging is possible.
7. Backup the previous model *GMM* $\mathcal{N}^{\mathbf{K}}$.
8. Select randomly an index $k \in 1 \cdots \mathbf{K} - 1$.
9. Compute the parameters, θ^k, for a new Gaussian $\mathcal{N}^m = \{\mathcal{N}^k : \mathcal{N}^{k+1}\}$.
10. **IF** Conditions of stability, 3.31, and accuracy, 3.32 are satisfied.
11. **THEN** replace $\mathcal{N}^k$ with $\mathcal{N}^m$, remove $\mathcal{N}^{k+1}$.
    Correct numbering of Gaussian and time indices, $\mathbf{K} = \mathbf{K} - 1$.
12. **ELSE** discard $\mathcal{N}^m$.
13. **END**
14. **END**

Output: $\hat{f} = \{\mathcal{N}^1(\xi; \theta^1), \cdots, \mathcal{N}^{\mathbf{K}}(\xi; \theta^{\mathbf{K}})\}$

Tab. 3.2: *Encoding the estimate of the DS with Binary Merging*

the new μ^k and Σ^k associated to the Gaussian. The stability and accuracy conditions from Eq. 3.31 and Eq. 3.32 are verified for the updated model. If the conditions are satisfied the two selected Gaussian functions are replaced by the merged Gaussian, and the new model is now composed of $\mathbf{K} - 1$ Gaussian functions. The algorithm terminates when there is no possible pair of Gaussian functions that can be merged without violating the maximum accepted error or becoming unstable. Table 3.2 summarizes the procedure to model the motion dynamics through Binary Merging, as proposed by [Khansari-Zadeh and Billard, 2010b].

There are some shortcomings when using *BM*, it has a limited region of applicability, since the stability domain $\mathbb{D}$ usually corresponds to a narrow region around the demonstrations. Additionally, it relies on determining numerically the stability region, which could become computation costly and intractable in higher dimensions [Khansari-Zadeh and Billard, 2011].

3.5.4 Stable Estimator of Dynamical Systems

[Khansari-Zadeh and Billard, 2010a] proposed a learning method, called *Stable Estimator of Dynamical Systems (SEDS)*, to learn the parameters of the *DS* that ensure all motions closely follow the demonstration dynamics and for the global asymptotic stability at the target of the estimate $\hat{f}$ of the non-linear autonomous *DS*. Their work provided a set of stability conditions to ensure the global asymptotic stability of f at the target. However, as opposed to *BM*, the effect of all Gaussian functions are taken into account, without any need to truncate the estimate to solely using the adjacent Gaussian functions.

In order to build a globally asymptotically stable *DS*, it is needed to set the parameters, θ, of the estimate of f, such that, by starting the motion from any point in the state space the energy of the system decreases until it reaches the target. Assuming that the state trajectory evolves according to Eq. 3.24, the non-linear function $\dot{\xi} = \hat{f}(\xi)$ can be made globally asymptotically stable at the target $\bar{\xi} \in \mathbb{R}^n$ by ensuring the following stability conditions,

$$\begin{cases} \mathbf{b}^k = -\mathbf{A}^k \bar{\xi} \\ \mathbf{A}^k + (\mathbf{A}^k)^\top \prec 0 \end{cases} \qquad \forall k = 1 \dots \mathbf{K} \tag{3.34}$$

where $\mathbf{A}^k$ and $\mathbf{b}^k$ are defined according to Eq. 3.23, and $\prec 0$ refers to the negative definiteness of a matrix, details can be found on [Khansari-Zadeh and Billard, 2011]. Conditions from Eq. 3.34 impose the constraint so that the energy dissipation on each Gaussian becomes negative everywhere except at the target, where it becomes zero.

Established sufficient conditions whereby $f(\xi)$ can be globally asymptotically stable at the target remain to determine a procedure for computing the unknown parameters, $\theta^k = (\pi^k, \mu^k, \Sigma^k)$, of Eq. 3.22 satisfying the stability conditions. Learning the parameters of the *GMM* proceeds as a constraint optimization problem, the *SEDS* learning algorithm computes optimal values for θ under strict stability constraints, ensuring that the model satisfy global asymptotic stability of the *DS* at the target [Khansari-Zadeh and Billard, 2010a]. For the optimization objective function two different candidates are used. One function based in the *log-likelihood*, as a means of constructing the model; and a function based on the *mean square error (MSE)*, as a means of quantifying the accuracy of estimations that are based on demonstrations.

The optimization problem is subject to the following constraints,

$$\mathbf{b}^k = -\mathbf{A}^k \bar{\xi} \qquad \forall k = 1 \dots \mathbf{K} \tag{3.35a}$$

$$\mathbf{A}^k + (\mathbf{A}^k)^\top \prec 0 \qquad \forall k = 1 \dots \mathbf{K} \tag{3.35b}$$

$$\Sigma^k \succ 0 \qquad \forall k = 1 \dots \mathbf{K} \tag{3.35c}$$

$$0 \leq \pi^k \leq 1 \qquad \forall k = 1 \dots \mathbf{K} \tag{3.35d}$$

$$\sum_{k=1}^{\mathbf{K}} \pi^k = 1 \tag{3.35e}$$

Algorithm: Stable Estimator of Dynamical Systems *SEDS* [Khansari-Zadeh and Billard, 2011]

Input: Demonstrations dataset $\{\xi^i, \dot{\xi}^i\}_{i=1}^{\mathcal{D}}$.

1. Initialize optimization parameters.
2. Transfer the target attractor $\bar{\xi}$ to the origin.
3. Choose initial number of Gaussian components $\mathbf{K}$.
4. Find initial estimate for the Gaussian parameters $\acute{\theta}^k = (\acute{\pi}^k, \acute{\mu}^k, \acute{\Sigma}^k), k \in 1 \cdots \mathbf{K}$, running Expectation Maximization.
5. Define optimize parameters as $\pi^k = \acute{\pi}^k$ and $\mu_{\xi}^k = \acute{\mu}_{\xi}^k$.
6. Convert the covariance matrix such that it satisfied the stability and optimization constrains as given by Eq. 3.35.
7. Compute $\mu_{\dot{\xi}}^k$ solving the optimization problem constraints given by Eq. 3.35.
8. Solve constraint optimization problem for $J(\theta)$ as given by the objective function of:
   Eq. 3.36 for the Likelihood.
   Eq. 3.37 for the MSE.
9. **IF** optimization constraints check satisfied.
10. **THEM** return.

Output: $\hat{f} = \{\mathcal{N}^1(\xi; \theta^1), \cdots, \mathcal{N}^{\mathbf{K}}(\xi; \theta^{\mathbf{K}})\}$

Tab. 3.3: *Encoding the estimate of the DS with SEDS*

The first two constraints of Eq. 3.35 are the stability conditions from Eq. 3.34. And the last three constraints are imposed by the nature of the *GMM*, from Eq. 3.17, ensuring Σ^k being positive definite matrices, and the π^k prior probabilities being positive scalars, smaller than or equal to one and their sum equal to one.

The *SEDS-Likelihood* method, using the log-likelihood as a means to quantify the accuracy of estimations, computes the optimal values of θ by solving,

$$\min_{\theta} J(\theta) = -\frac{1}{\mathcal{T}} \sum_{i=1}^{\mathcal{D}} \sum_{t=o}^{\mathcal{T}^i} \ln p((\xi^{t,i}; \dot{\xi}^{t,i}) \mid \theta) \tag{3.36}$$

where $p((\xi^{t,i}; \dot{\xi}^{t,i}) \mid \theta)$ is given by Eq. 3.17 and $\mathcal{T} = \sum_{i=1}^{\mathcal{D}} \mathbf{T}^i$ are the total number of points in the demonstration dataset. For selecting an optimal number of Gaussian functions $\mathbf{K}$ for this method, *Bayesian Information Criterion (BIC)* was used to determine a trade-off between the optimization of the model's likelihood and the total number of parameters needed to encode the data,

$$BIC = \mathcal{T} J(\theta) + \frac{n_p}{2} \ln(\mathcal{T})$$

in which $J(\theta)$ is the normalized log-likelihood of the model in Eq. 3.36 and n_p is the total number of free parameters.

In *SEDS-MSE* method, which uses the *mean square error* as a means to quantify the accuracy of estimations, the optimal values of θ are computed by solving,

$$\min_{\theta} J(\theta) = -\frac{1}{2\mathcal{T}} \sum_{i=1}^{\mathcal{D}} \sum_{t=o}^{\mathcal{T}^i} (f(\xi^{t,n}) - \dot{\xi}^{t,n})^\top (f(\xi^{t,n}) - \dot{\xi}^{t,n}) \tag{3.37}$$

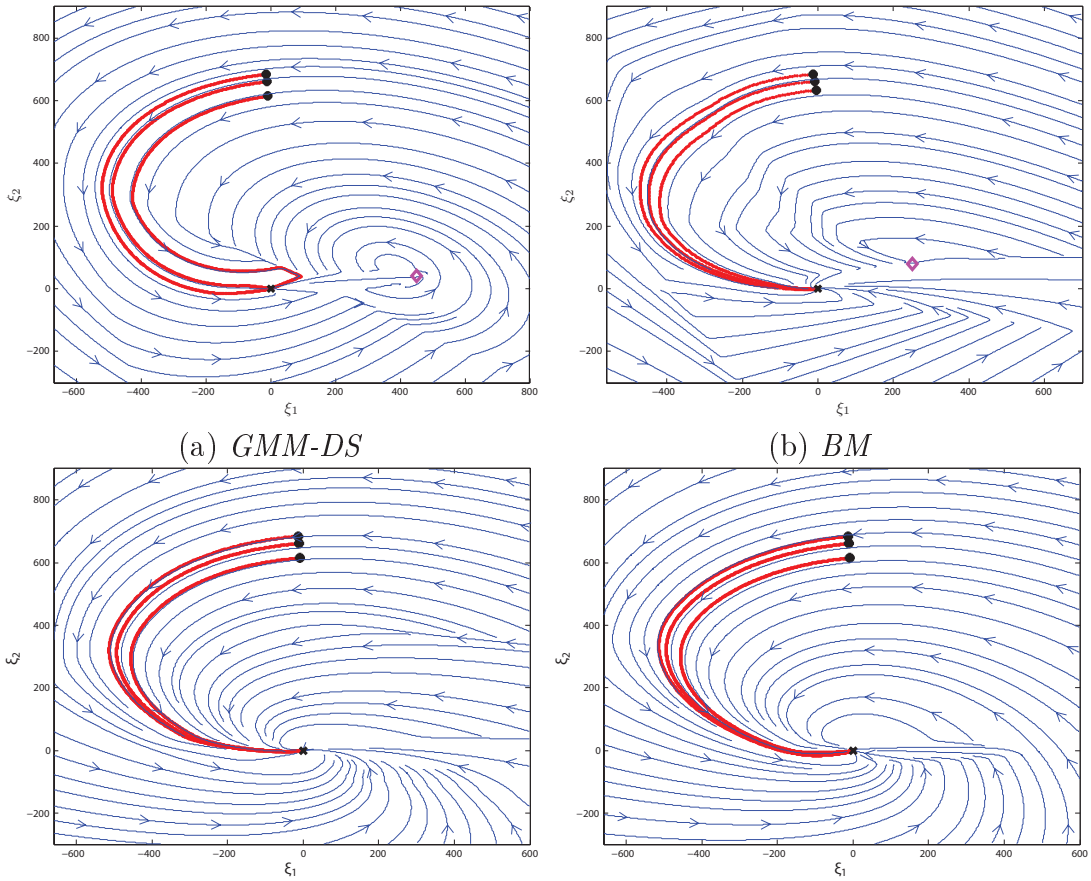
where $f(\xi^{t,n})$ is calculated directly from Eq. 3.22 and $\mathcal{T}$ is, as above, the total number of points in the demonstration dataset. In order to obtain an optimal number of Gaussian function $\mathbf{K}$ for this method, the demonstrations are split into training and test datasets. The optimal number of Gaussian functions corresponds to the minimum value of K that provides an accurate estimate on both datasets [Khansari-Zadeh and Billard, 2011].

Table 3.3 summarizes the procedure to model the motion dynamics through Stable Estimator of Dynamical Systems, as proposed by [Khansari-Zadeh and Billard, 2011].

The resulting models from optimizing with both *SEDS-Likelihood* and *SEDS-MSE* methods benefit from the inherent characteristics of autonomous *DS*. However, each objective function has its own advantages and disadvantages. Employing the *SEDS-log-likelihood* can be advantageous in that it is more accurate and smoother than *SEDS-MSE*. Furthermore, the *SEDS-MSE* cost function is slightly more time consuming since it requires computing *GMR* at each iteration. However, the *SEDS-MSE* objective function requires fewer parameters than the *SEDS-log-likelihood*, which may make the algorithm faster in higher dimensions or when a higher number of components are used [Khansari-Zadeh and Billard, 2011].

From the *GMM-GMR* approach the learning parameters requirements for estimation would be $\mathbf{K}(1 + 3n + 2n^2)$, for π, μ, and Σ of size 1, 2n, and n(2n + 1) respectively, with n the dimensionality of the demonstrations dataset. However, for *SEDS-Likelihood* the total number of parameters can be reduced since Eq. 3.35 provides explicit formulation to compute $\mu_{\dot{\xi}}$ from the other parameters. The number of free parameters to construct the model with *SEDS-Likelihood* is $\mathbf{K}(1 + 2n(n + 1))$. For *SEDS-MSE* the term $\Sigma_{\dot{\xi}}$ is not used, the total number of parameters *SEDS-MSE* encoding reduces to $\mathbf{K}(1 + \frac{3}{2}n(n + 1))$. For both approaches, the number of parameters grows linearly with the number of Gaussian functions and quadratically with the dimension. In comparison, the number of parameters in *SEDS* would be smaller than those needed for the other methods.

Figure 3.11 presents examples of the learned *DS* from a demonstrated trajectory with the methods presented in this section: *GMM-DS*, *BM*, *SEDS-MSE* and *SEDS-Likelihood*. As mentioned before, the non-linear *DS* are susceptible to instabilities. Guaranteeing the estimates $\hat{f}$ results in an asymptotically stable trajectory and is a key requirement to provide a useful control policy. The *GMM-GMR* approach as defined by Eq. 3.24 cannot guarantee the system's asymptotically stability. The method presented by [Gribovskaya and Billard, 2009], Table 3.1, look for an estimate of $\hat{f}$ that is locally stable around the target, but without guaranteeing that such a model would be found or considering the accuracy of its reproduction. As can be seen from 3.11(a) the learned model presents a spurious attractor and some of the trajectories



(a) *GMM-DS* (b) *BM*

(c) *SEDS-Likelihood* (d) *SEDS-MSE*

Fig. 3.11: *Examples of the learned DS. The letter C pattern demonstrations from Fig 3.9 are modelled with the GMM-DS(a), BM(b), SEDS-Likelihood(c) and SEDS-MSE(d) methods. Reproductions are drawn as red lines. The target attractor is drawn as a black* $\mathbf{x}$ *at (0,0). The existence of spurious attractors is drawn as magenta* $\diamond$*. The streamlines of the learned dynamics are drawn in blue.*

are not accurate enough. The *BM* approach [Khansari-Zadeh and Billard, 2010b], Table 3.2, ensures local stability around a defined region $\mathbb{D}$. Here, as can be seen from 3.11(b), spurious attractors can still exist outside of $\mathbb{D}$, which also has a limited region of applicability. The *SEDS* approach [Khansari-Zadeh and Billard, 2011], Table 3.3, provides strict stability constraints ensuring that the model satisfies global asymptotic stability of the *DS* at the target. As can be seen from 3.11(c)-(d), for *SEDS-MSE* and *SEDS-Likelihood* respectively, reproductions of the learned *DS* are guaranteed to be globally asymptotically stable.

3.6 Reproduction of Learned Robot Skills

As stated before in the problem formalization in the previous section, the system motion can be unambiguously determined by the state variable ξ when governed by the estimate $\hat{f}$ of the motion dynamics. Choosing this variable is therefore crucial for the trajectory of the reproduction of the learned robot skill. The learning algorithms described in section 3.5 aim at being a generic framework and make no assumption on the variable that is used for training. Here, the choice is made to represent the motions in kinematic coordinates, the Cartesian space, with the assumption that appropriate controllers are available to convert the kinematic variables to motor commands. Adopting a kinematic formulation is quite suitable for motion control, since the kinematic variables generalize over a large part of the workspace, and planning in kinematic space is often more convenient for motor control. Also, kinematic plans can theoretically be clearly superimposed to form more complex behaviours [Schaal et al., 2007].

First, it is desirable to validate the performance of the methods presented in section 3.5. For this a set of 2-D sample motions is collected from the validation data provided by the authors of the original formulation of these methods in their respective source codes. A total of 8 motions were chosen to compare the performance of the methods, 4 from [Khansari-Zadeh and Billard, 2011], 1 from [Khansari-Zadeh and Billard, 2010b], 2 from [Calinon, 2009] and finally 1 hand drawn motion recorded with *MLDemos visualization tool for machine learning* [Basilio, 2013]. All reproductions are generated in simulation to avoid adding the robot controller errors. The methods' performance are evaluated over two error measurements. An accuracy error measurement, $\bar{e}$ from 3.32, which measures the error in the estimation of $\dot{\xi}$ magnitude and direction. And a "swept area error" measurement,

$$\mathcal{E} = \frac{1}{\mathcal{D}} \sum_{i=1}^{\mathcal{D}} \sum_{t=0}^{\mathcal{T}^i} \mathcal{A}(\xi^i(t), \xi^i(t+1), \xi^{t,i}, \xi^{t+1,i}) \tag{3.38}$$

$\mathcal{A}$ correspond to the area of the tetragon generated by the points $(\xi^i(t), \xi^i(t{+}1), \xi^{t,i}, \xi^{t+1,i})$, were ξ^t, ξ^{t+1} are given by the demonstration datapoints at t and $t{+}1$, and $\xi(t), \xi(t{+}1)$, computed by $\xi(t) = \dot{\xi}(t) * dt$, are an estimate of the demonstrated trajectories starting from the same initial points. Eq. 3.38 measures the cumulative error over the reproduction of trajectories.

Figure 3.12 and Tables 3.4, 3.5 summarize the results of validating the methods with 8 sample 2-D motions. The *GMM-DS* method, tries to satisfy local stability conditions, however, it does not ensure the possibility of finding a stable *DS*. The *BM* method, generated the most accurate estimates among the methods, producing generally better results than both *SEDS* versions. However, the *BM* method is also the more computationally costly and the one which requires the highest number of parameters among the methods, *BM* and *GMM-DS* have the same number of parameters $\mathbf{K}(1 + 3n + 2n^2)$ yet the value for number of Gaussian $\mathbf{K}$ was consistently higher for *BM* since this method began at a max number of Gaussian and merged down from there. Finally, the *BM* method only ensures local stability of the *DS*.

	Accuracy Error $\bar{e}$		Swept Area Error $\mathcal{E}$		# of	Ensure
Method	Mean $\bar{e}$	Range of $\bar{e}$	Mean $\mathcal{E}$	Range of $\mathcal{E}$	Parameters	Stability
GMM-DS	0.75348	0.226-1.660	1930.3	199-8766	101(60-165)	No(Local)
BM	0.50394	0.217-1.118	1582.5	213-7062	165(90-300)	Yes(Local)
SEDS-Likelihood	0.77215	0.628-1.198	2241.3	648-10290	95(52-156)	Yes(Global)
SEDS-MSE	0.74683	0.474-1.128	1767.7	449-8223	64(40-120)	Yes(Global)

Tab. 3.4: *Performance comparison of the methods presented in section 3.5 with a set of sample 2-D motions. BM generates the most accurate estimate, and also require the more number of parameters among the methods. The performance of SEDS-MSE and SEDS-Likelihood is similar.*

The performance of both *SEDS* methods, Table 3.3, was comparable, with very similar results particularly for the accuracy error ($\bar{e}$), and slightly better with *SEDS-MSE* for the sweep area error ($\mathcal{E}$). The *SEDS-MSE* method is advantageous in that it requires fewer parameters than *SEDS-Likelihood*. However, *SEDS-MSE* has a more complex cost function, making the algorithm computationally more expensive [Khansari-Zadeh and Billard, 2011]. Both *SEDS* methods outperforms *BM* in that they ensure global asymptotic stability and are capable of better generalizing motions for trajectories that are far from the demonstrations. Figure 3.12 shows results of estimating the 8 sample 2-D motions of Table 3.5 with *SEDS-Likelihood* method.

The motivation for this chapter is to learn and encode the demonstrated motion dynamics in order to build models of the robot skills as needed by the subsequent modules of the proposed framework in Figure 3.1. The *Robot Skills Models* are defined by the estimate of the motion dynamics, $\hat{f}$, as learned by the methods in Section 3.5, described in Tables 3.1, 3.2, 3.3. Therefore a robot skill is modelled by the parameters θ of $\hat{f}$. We will use the notation $\bar{\mathcal{M}}_{RS}$ for a *Robot Skill Model*, determined by $\hat{f} = \{\mathcal{N}^1(\xi;\theta^1), \cdots, \mathcal{N}^{\mathbf{K}}(\xi;\theta^{\mathbf{K}})\}$, such that,

$$\bar{\mathcal{M}}_{RS} = \{\theta^1, \cdots, \theta^{\mathbf{K}}\} \tag{3.39}$$

where $\theta^i = \{\pi, \mu, \Sigma\}$ of the $\mathcal{N}^i$ Gaussian defined by Eq. 3.18, and $\mathbf{K}$ is the total number of Gaussian functions required to estimate the motions dynamics.

For the robotic system reproduction of motions, the robot skills in this work are to be represented in the Cartesian coordinate system, the desired output variables are then positions, velocities and accelerations, in order to control the system in the operational task space. From [Gribovskaya and Billard, 2009], the task space trajectories of the robot's end-effector are selected so that it can be taught to control the position and orientation of the motion. The variables in the training set were chosen as the translation component of a motion of the end-effector, a vector of Cartesian coordinates $x \in \mathbb{R}^3$; and the orientation of the end-effector, a pair of variables $\{s, \phi\}$ representing the axis and the angle of rotation. According to this representation, the orientation of a moving referential $\{x'y'z'\}$ with respect to a fixed

	Accuracy Error $\bar{e}$			Swept Area Error $\mathcal{E}$			# of Parameters		
Motion	*BM*	*SEDS-L*	*SEDS-M*	*BM*	*SEDS-L*	*SEDS-M*	*BM*	*SEDS-L*	*SEDS-M*
Angle[1]	0.332	0.737	0.708	1082	1931	1178	135	120	80
Sine[1]	0.804	1.051	0.945	6101	7700	5627	195	90	60
Khamesh[1]	0.371	0.703	0.688	861	969	1021	90	90	60
Trapezoid[1]	0.353	0.698	0.675	575	984	789	180	75	50
Arc[2]	0.328	0.640	0.639	1037	1244	1112	90	75	50
Waves[4]	0.759	0.875	0.887	723	1179	889	300	180	120
U-Curve[3]	0.404	0.655	0.752	264	1825	1741	150	60	40
5-Curve[3]	0.509	0.747	0.584	2586	2814	1767	180	75	50

Tab. 3.5: *Performance Comparison of Learning Methods on Sample Set of 2-D Motions. ([1]) taken from [Khansari-Zadeh and Billard, 2011], ([2]) taken from [Khansari-Zadeh and Billard, 2010b], ([3]) taken from [Calinon, 2009], ([4]) recorded with MLDemos visualization tool [Basilio, 2013].*

referential $\{xyz\}$ is described by the rotational axis $s \in \mathbb{R}^3$ and the angle $\phi \in [0; 2\pi]$. Internally an inverse kinematics controller is available to convert the end effector's control variables to appropriated joint space motor commands, $\theta, \dot{theta}$.

Therefore, the estimate $\hat{f}$ of the *DS* that it must be learned from the demonstrations is,

$$\dot{x} = \hat{f}_x(x) \qquad \text{with} \quad \xi = x \in \mathbb{R}^3 \tag{3.40}$$

for the dynamics of the end-effector's position (x). And,

$$\dot{o} = \hat{f}_o(s, \phi) \qquad \text{with} \quad \xi = [s, \phi], \quad s \in \mathbb{R}^3, \phi \in [0; 2\pi] \tag{3.41}$$

for the dynamics of the end-effector orientation (o).

Alternatively, the state variable ξ can be made to encode the coupled dynamics of the end-effector's position and orientation as,

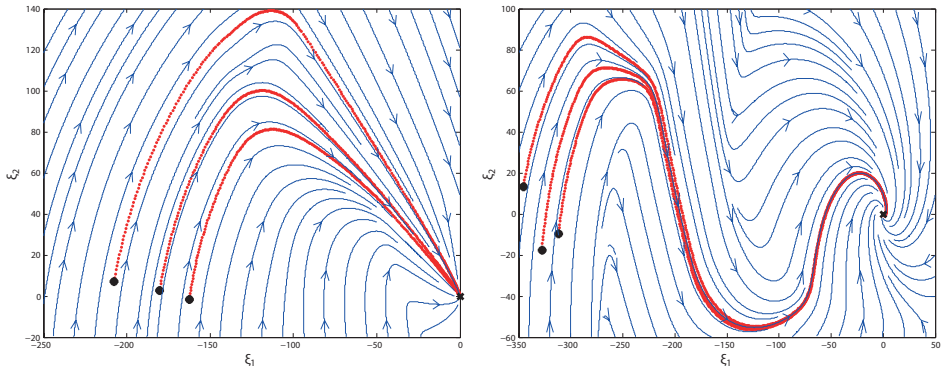
$$\dot{\xi} = \hat{f}_\xi(\xi) \qquad \text{with} \quad \xi = [x, s, \phi], \quad x \in \mathbb{R}^3, s \in \mathbb{R}^3, \phi \in [0; 2\pi] \tag{3.42}$$

Then the estimate $\hat{f}$ of the dynamics can be inferred through the *GMR* process,

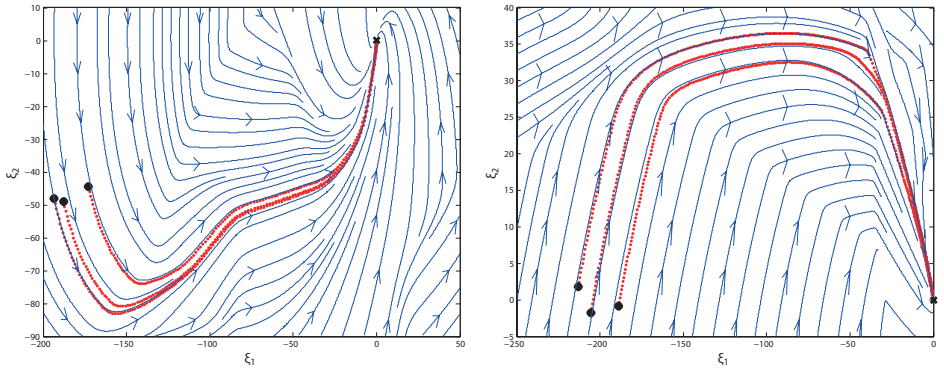
$$\dot{\xi} = \hat{f}(\xi) = \mathbb{E}\left[p(\dot{\xi} \mid \xi)\right] = \sum_{k=1}^{K} h^k(\xi)(\mathbf{A}^k \xi + \mathbf{b}^k) \quad \text{where,}$$

$$x = \hat{f}(x) = \mathbb{E}\left[p(\dot{x} \mid x)\right] = \sum_{k=1}^{K} h^k(x)(\mathbf{A}^k x + \mathbf{b}^k) \tag{3.43a}$$

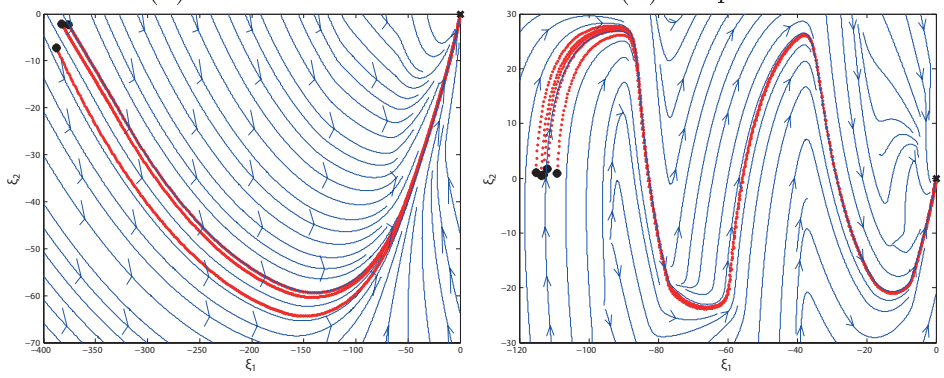
for controlling the position.



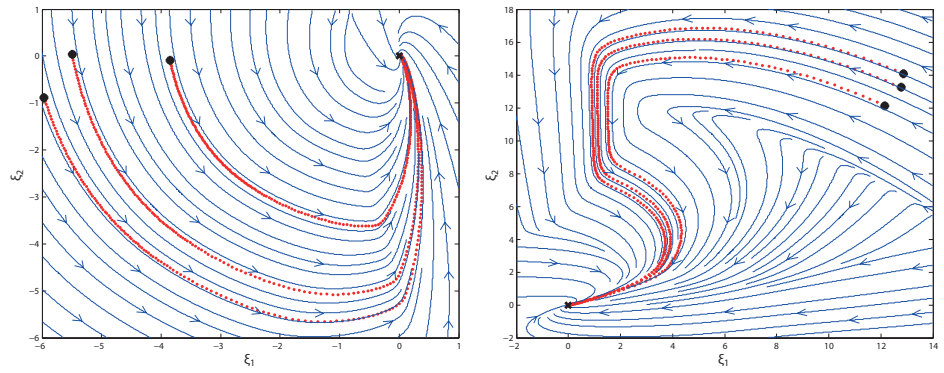
(a) *Angle* (b) *Sine*



(a) *Khamesh* (b) *Trapezoid*



(a) *Arc* (b) *Waves*



(a) *U-Curve* (b) *5-Curve*

Fig. 3.12: *8 2-D motions use to compare the performance of the various methods. The resulting reproductions correspond to the SEDS-Likelihood Models.*

	Accuracy Error $\bar{e}$		Swept Area Error $\mathcal{E}$		# of	Ensure
Method	Mean $\bar{e}$	Range of $\bar{e}$	Mean $\mathcal{E}$	Range of $\mathcal{E}$	Parameters	Stability
BM	1.1079	0.179-2.258	1965	589-4548	112.5(45-210)	Yes(Local)
SEDS-Likelihood	1.8395	0.582-4.920	2024	389-6454	81.25(65-91)	Yes(Global)
SEDS-MSE	1.7191	0.458-4.316	2989	985-8028	62.5(50-70)	Yes(Global)

Tab. 3.6: *Performance comparison of the methods presented in section 3.5 with a set of sample 3-D motions. The estimates generated with BM are more accurate, while also requiring the bigger number of parameters among the methods. Performance of SEDS-Likelihood and SEDS-MSE is very similar, with SEDS-Likelihood outperforming SEDS-MSE in accuracy estimates, and SEDS-MSE doing better with Swept Area Error.*

$$\dot{o} = \hat{f}(o) = \mathbb{E}\left[p(\dot{o} \mid o)\right] = \sum_{k=1}^{K} h^k(o)(\mathbf{A}^k o + \mathbf{b}^k) \quad \text{for controlling the orientation.} \tag{3.43b}$$

$$[\dot{x}, \dot{o}] = \hat{f}(x, o) = \mathbb{E}\left[p([\dot{x}, \dot{o}] \mid [x, o])\right] = \sum_{k=1}^{K} h^k([x, o]))(\mathbf{A}^k [x, o]) + \mathbf{b}^k) \quad \text{for a coupled controller.} \tag{3.43c}$$

as defined by Eq. 3.22 and Eq. 3.24.

Figure 3.13 and Tables 3.6, 3.7 summarize the results of validating the methods with 4 sample 3-D motions. The *GMM-DS* method, was omitted this time from the validations since it does not ensure the stable *DS* it will not be further employed in this work. The *BM* method, again, generated the most accurate estimates, producing generally better results than both *SEDS* versions, although its performance was not always better for all of the task, see Table 3.7, this could be because of bad modelling of the task. Also, just as expected, the *BM* method presented the higher number of parameters among all methods. The performance of both *SEDS-MSE* and *SEDS-Likelihood* was very similar with *SEDS-MSE* performing slightly better for the accuracy error ($\bar{e}$), and *SEDS-Likelihood* outperforming *SEDS-MSE* this time for the sweep area error ($\mathcal{E}$). It must be taken into account that these results are very task dependent, and no other true conclusion can be made between these methods except that they are both sufficiently adequate for their intended purpose in this work. Figure 3.13 shows the results of estimating the 4 sample 3-D motions of Table 3.7 with the *SEDS-Likelihood* method.

The assumption was made from Eq. 3.5, that the motions are modelled with a first order time-invariant ODE. The proposed *DS* are generic enough to represent a wide variety of motions, however, they would fail to define second order dynamics accurately. This problem can be solved by defining the motion in terms of position, velocity and acceleration [Khansari-Zadeh and Billard, 2011]. This means solving

	Accuracy Error $\bar{e}$			Swept Area Error $\mathcal{E}$			# of Parameters		
Motion	*BM*	*SEDS-L*	*SEDS-M*	*BM*	*SEDS-L*	*SEDS-M*	*BM*	*SEDS-L*	*SEDS-M*
iCub-Task[1]	1.956	4.327	3.869	3112	599	1993	105	91	70
Cup-Task[2]	1.522	1.215	1.248	11200	4758	24708	45	65	50
Door-Task[3]	0.263	1.523	0.923	1989	2796	2684	90	78	60
Cap-Task[4]	2.416	1.267	1.383	791	1064	1973	210	91	70

Tab. 3.7: *Performance comparison of learning methods on a sample set of 3-D Motions. ([1]) taken from [Gribovskaya and Billard, 2009], ([2]) taken from [Khansari-Zadeh and Billard, 2010b], ([3]) taken from [Kheddar et al., 2009a], ([4]) recorded with kinaesthetic demonstrations with the robot HOAP-3.*

	Accuracy Error $\bar{e}$		Swept Area Error $\mathcal{E}$		# of	Ensure
Method	Mean $\bar{e}$	Range of $\bar{e}$	Mean $\mathcal{E}$	Range of $\mathcal{E}$	Parameters	Stability
BM	0.50394	0.217-1.118	1582.5	213-7062	165(90-300)	Yes(Local)
SEDS-Likelihood	0.77215	0.628-1.198	2241.3	648-10290	95(52-156)	Yes(Global)
SEDS-MSE	0.74683	0.474-1.128	1767.7	449-8223	64(40-120)	Yes(Global)

Tab. 3.8: *Performance comparison of the methods presented in section 3.5 with a set of sample self-intersecting motions.*

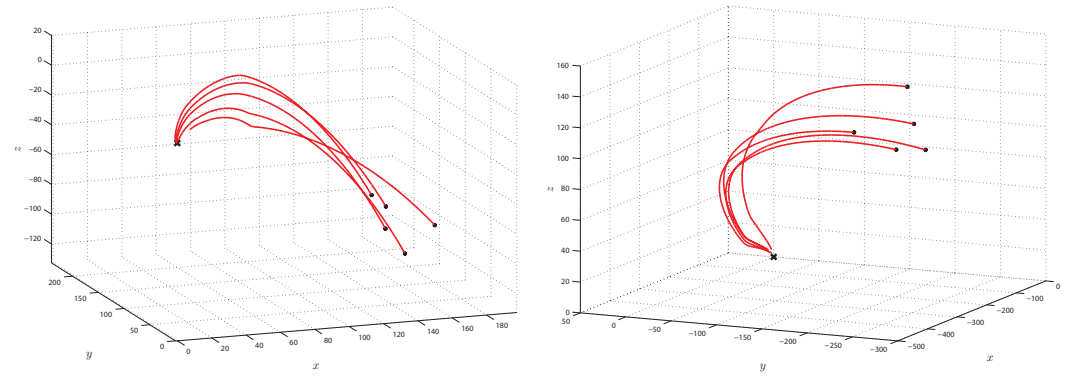
the problem as second order *DS*. When considering the motions in its second order dynamics, that is $\ddot{x} = g(x, \dot{x})$, it would be very advantageous if it could be simplified with a change of variable into a first order ODE,

$$\begin{cases} \dot{x} = v \\ \dot{v} = g(x, v) \end{cases} \Rightarrow [\dot{x}, \dot{v}] = f(x, v) \tag{3.44}$$

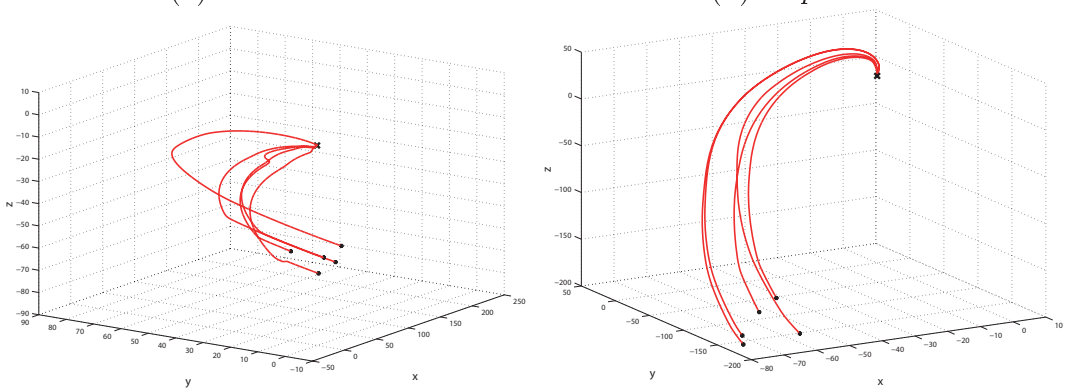
By defining the state variable, ξ, as $\xi = [x, v] \Rightarrow \dot{\xi} = [\dot{x}, \dot{v}]$ the Eq. 3.44 simplifies to $\dot{\xi} = f(\xi)$ as in Eq. 3.24 and can be learned following the methods presented in the previous section.

Figure 3.14 and Tables 3.8, 3.9 summarize the results of validating the methods by learning the second order dynamics of a motion. It can be observed from the trajectories that when encoding the intersecting motion with only the first order dynamics reproduction failed to reflect the full demonstrated motion, Figure 3.14. Encoding the second order dynamics of the motion allows disambiguation of the direction of the motion when reproducing a self-intersecting trajectory.

In order for a robot to reproduce a skill a model, $\mathcal{M}_{RS}$, of the estimate $\hat{f}(\xi)$ of the motion dynamics must have been learned beforehand as per the methods presented in the previous section. Assuming appropriate models exist the first step is to detect a target object of the skill, the attractor of the modelled dynamics, in a "global referen-



(a) *iCub-Task* (b) *Cup-Task*



(a) *Door-Task* (b) *Cap-Task*

Fig. 3.13: *4 3-D motions use to compare the performance of the various methods. The results correspond to the SEDS-Likelihood Models.*

	Accuracy Error $\bar{e}$			Swept Area Error $\mathcal{E}$			# of Parameters		
Motion	*BM*	*SEDS-L*	*SEDS-M*	*BM*	*SEDS-L*	*SEDS-M*	*BM*	*SEDS-L*	*SEDS-M*
Loop[1]	4.348	2.741	2.246	5601	1078	3587	90	65	50
Letter T[2]	3.260	7.211	6.448	3315	1773	2846	155	78	70

Tab. 3.9: *Performance comparison of learning methods on a sample set of self-intersecting motions. ([1]) taken from [Khansari-Zadeh and Billard, 2010b], ([2]) recorded with MLDemos visualization tool [Basilio, 2013].*

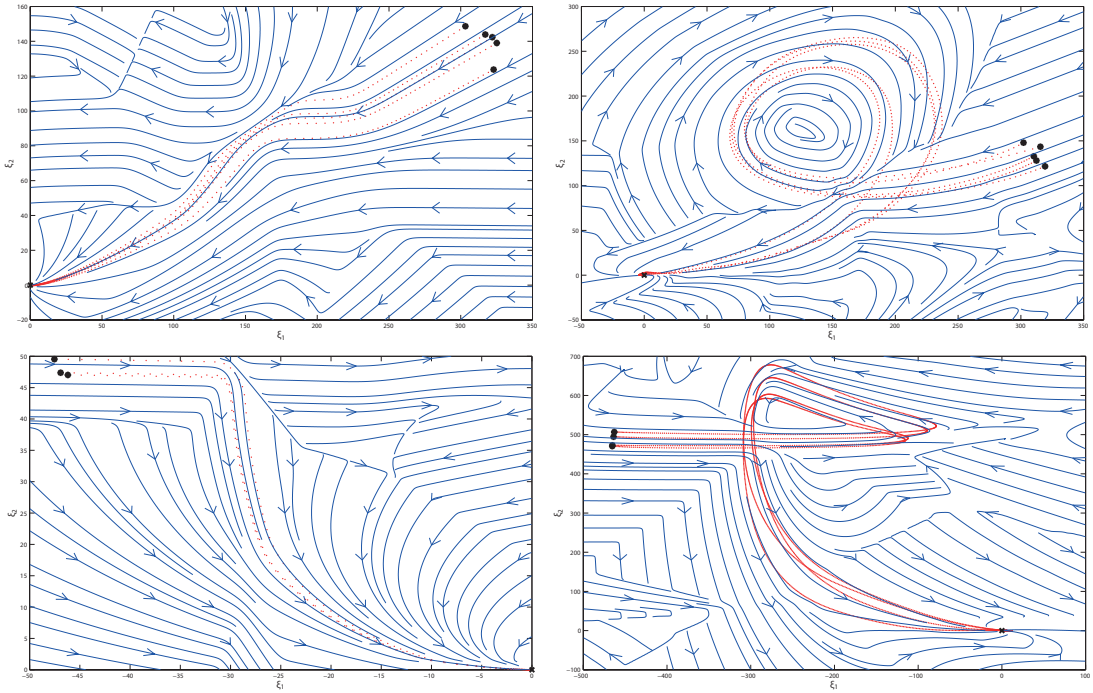


Fig. 3.14: *2 self-intersecting motions DS use to compare the performance of the various methods. Top trajectory is a loop motion taken from [Khansari-Zadeh and Billard, 2010b]. Bottom trajectory is a letter T motion recorded with MLDemos visualization tool [Basilio, 2013]. (left) The result of encoding the first order dynamics of the motion. (right) The result of encoding the second order dynamics of the motion as it is presented in Eq. 3.44.*

Algorithm: On-line reproduction of the learned robot skill

Input: Learned *Robot Skill Model*, $\mathcal{M}_{RS}$, of the estimate $\hat{\xi}$ of the motion dynamics underlying the skill.

1. Detect a target position in the global referential $\{x_t, y_t, z_t\} : \mathbf{x}_G$.
2. Transfer the origin of the task referential frame to the detected target $\{x_t, y_t, z_t\} : \mathbf{x}_T = \{0, 0, 0\}$.
3. Recompute the current state of the end-effector in the target referential $\xi^* = \{x'_e, y'_e, z'_e\} : \mathbf{x}_T$.
4. **LOOP** until the target position is reached. $t = 0$.
5. Infer the velocity at the time step t through *GMR*, Eq. 3.24.
   $\dot{\xi}(t) = \sum_{k=1}^{K} h^k(\xi^*)(\mathbf{A}^k \xi^* + \mathbf{b}^k)$.
6. Compute the end-effector's state for the next time step, $\xi(t+1)$.
   $\xi(t+1) = \xi(t) + \dot{\xi}(t+1) \cdot dt$.
7. Compute the robot motor command for the next step,
   $\xi, \dot{\xi} \mapsto q, \dot{q}$, solving the inverse kinematics problem.
8. Execute the robot command and sense the new q effector position.
9. Update the end-effector state in the target referential
   $q \mapsto \xi : \xi^* = \{x'_e, y'_e, z'_e\} : \mathbf{x}_T$.
10. **END**

Output: $\dot{\xi}(t); \xi(t) = \xi(t-1) + \dot{\xi} * dt$ robot skill trajectory.

Tab. 3.10: *Procedure for on-line reproduction of the learned robot skills.*

tial frame", $\{x_t, y_t, z_t\} : \mathbf{x}_G$, this could be from the robot's own viewpoint referential or any other perception system referential available for the task. The origin for the task reference frame is then attached to the target, $\{x_t, y_t, z_t\} : \mathbf{x}_T = \{0, 0, 0\}$. Therefore, the robot skill motion is controlled with respect to this frame of reference, with a target attractor at the origin as in the formulation of Eq. 3.5. This representation also makes that the parameters of the *DS* invariant to changes in the target position. Subsequently, the current state of the end-effector, $\{x_e, y_e, z_e\} : \mathbf{x}_G$, is recomputed in the referential frame of the target, $\xi^* = \{x'_e, y'_e, z'_e\} : \mathbf{x}_T$. The trajectories of the reproduced motion are governed by the modelled dynamics, progressing from the current state, ξ^*, towards the attractor point of the *DS* located at the origin of the target referential frame, $\mathbf{x}_T = \{0, 0, 0\}$, according to the estimated attractor landscape of the learned *DS*, see Figures 3.12 and 3.13. At every step, the end-effector's next state, ξ^{*+1}, is inferred by sampling from the *GMR*, to obtain $\dot{\xi} : \xi^{*+1} = \xi* + \dot{\xi} \cdot dt$, this is repeated successively until the target is reached. The robot reproduction of the trajectories of the learned motion dynamics can be computed on-line through the *GMR* of the modelled robot skill. The process for on-line reproduction of the learned motion dynamics is summarized in Table 3.10.

3.7 Robot Skills as Basic Primitives of Movement

A desirable application for the learned *Robot Skill Models* is the building of a library of so called movement primitives that can be readily available for later reuse by the robot when a situation required it to.

For motion control in robotics different motor behaviours can be seen as different control policies, representing different actions. It is desired to have methods for representing human movement compactly in terms of a linear superimposition of simpler movements, which are termed primitives. The motor primitives, also called, movement primitives, basic behaviours, units of actions, etc., are sequences of actions that can accomplish a certain movement goal [Schaal, 1999]. Movement primitives are biological structures that organize the underlying mechanism of complete movements [Fod et al., 2000]. An approach based on movement primitives relies on possessing available sequences of motor commands, executed in a certain order, to accomplish a given motor task. The movement primitives can be viewed as a basic set of motor programs that are sufficient for generating entire movement repertoires [Muelling et al., 2013].

A starting point for this approach is in the assumption that complex movement skills are composed from smaller units of action. The established belief is that human activity is decomposed into building blocks of elementary actions. There are many theories which propose human motion being divided into their elementary trajectories [Fod et al., 2000].

Dealing with these issues leads to facing the problems of segmentation and classification of human motion. The work of [Vecchio, 2002] deals with decomposing continuous trajectories of the human body into their components, which are called 'movemes', and aims to build a so called "alphabet of movemes" to represent and

describe human motion similar to the way phonemes are used in speech. The underlying idea for a robotic system to cope with the complexity of replicating human motor skills is for the learned, demonstrated movements to be first segmented into sub-goals from which appropriate primitives can be obtained.

Movement primitives in their most simple form can be thought of as simple as the elementary actions in the symbolic approach to imitation, with simple point-to-point movements employed by industrial robots [Schaal, 1999]. Learning motions of such a low-level of representation failed to scale well to systems with many degrees of freedom. Movement primitives would benefit from coding complete temporal behaviours, that result in state-action representation that are compact and which need to adjust only a few parameters for a specific goal [Schaal, 1999]. Learning the *Robot Skills Models* as in section 3.6 can be a most suitable way of forming basic primitives of movement, encoding within the model the motion dynamics of a demonstrated skill.

Learning such basic units of action has long been thought useful for generating libraries of motor skills. A robotic system equipped with a library of movement primitives with a sufficient number of skills can be thought of possessing an adequate repertoire of actions to deal with a vast range of situations. Also, it is generally regarded that complex motions can be dealt with by building a library of movement primitives [Pastor et al., 2009], providing basic components from which multiple desired robot tasks can be performed by combination and superposition of the primitives.

From leading views of motor control in neurobiology it is generally regarded that humans do employ basic motor primitives as an underlying mechanism of biological motor control. Evidence exists from human and animal experiments supporting the belief that sets of motor primitives are used to build a basis for voluntary motor control [Konczak, 2005]. It is well accepted in these approaches that for coping with the complexity of motor skills learning for robots, it is necessary to rely on the insight that humans decompose motor skills into smaller subtasks. There are many theories about motor primitives which suggest that they are a viable means for encoding humanoid movement. Primitives are fundamental building blocks of motor control determining an effective basis set of primitives is therefore a difficult problem [Fod et al., 2000].

Motor controller components of the movement primitives may be manually derived or learned. It is important that the representations used for extracting units of actions also relate to the movement generation [Meier et al., 2011]. The primitives must be characterized in parametric form to allow generalization and their applicability to different scenarios. Adequate representations are needed for the movement primitives, in order to build a library of skills.

The work of [Ijspeert et al., 2003] was the first to suggest the idea of using *DS* as motor primitives. Their approach employed the *DMP* to learn and encode the dynamics of demonstrated motions. The control policies could be used to represent basic movements that form a library of motions. Defining the primitives in terms of causal dynamical systems allows then to be parametrized by a small set of dynamical parameters and an input driving the overall dynamics [Vecchio, 2002].

Various examples can be found to represent movement primitives such as that repertoires of motions that can be built from learned motion tasks. [Ude et al., 2007] presents a framework for synthesizing goal-directed actions from a library of example

movements; different methods can be utilized for the construction of these movement libraries. [Zoliner et al., 2005b] deals with the integration of learned manipulation tasks into a knowledge base as well as enabling the system to reason and reorganize the gathered knowledge in terms of re-usability, scalability and explainability of learned skills and tasks. In [Pastor et al., 2009] a collection of dynamic movement primitives is used to build a library of movements by labelling each recorded movement according to task and context. [Vecchio, 2002] proposed understanding human motion by decomposing it into a sequence of elementary building blocks that belong to a known alphabet of dynamical systems, which can be composed to represent and describe human motions and shown dynamical characteristics which are sufficient to distinguish between them. [Muelling et al., 2013] created a movement library from *Imitation Learning*; movements stored in the library can be selected and generalized using a mixture of primitives algorithm. [Fod et al., 2000] presented a method used to derive a set of perceptual-motor primitives directly from movement data. The primitives can be used as a lower-dimensional space for representing movement.

A difficult problem remains in these approaches in the segmenting of complex movements and classification of the movement primitives. [Meier et al., 2011] approach aimed at movement segmentation with simultaneous movement recognition, assuming that a library of movement primitives already existed, and reduced the segmentation problem to online movement recognition.

The ability to imitate is based on a mapping mechanism which can automatically classify all observed movements onto their set of perceptual-motor primitives [Fod et al., 2000]. Building systems that can detect and recognize human action are an important goal. Thus segmentation and classification become key interrelated processes of movement interpretation. The segmentation problem can be divided into three sub problems, first determining the number of segments, then estimating the start and end time of each segment and recognizing which primitive from the library is executed in each segment [Meier et al., 2011]. Understanding motor behaviour becomes a process of classifying the observed movements into the known collection of movement primitives [Fod et al., 2000].

Distinguishing between general classes of motor skills is useful. [Vecchio, 2002] selects between "reach" and "drawing" motions. The work of [Schaal and Atkeson, 2010] makes a classification along "regulator" tasks which keep the system over a point of operation. "Tracking" task control systems to follow a given desired trajectory. "One-shot" tasks defined by achieving a particular goal. And "periodic" movement tasks. A complex movement would be composed of sequencing and superimposing of these simpler motor skills.

[Ijspeert et al., 2002] showed that trajectories with similar velocity profiles fit similar encoding parameters and proposed to use the learned control policies to classify movements, computing the correlation between them. The goal is to build a base of robot skills learned from the demonstrations and to select and generalize among these skills to adapt to new situations. A robot skill could be categorized according to its velocities and acceleration profiles and the correlation between its variables into several categories, such as reaching movements, striking or hitting movements, tracing or drawing movements and coordinated and uncoordinated movements.

3.8 Summary of the Chapter

Throughout this chapter a review of the field of *Learning from Demonstration (LfD)* has been presented along with the process and methods used for learning and encoding the models of the robot skills. In section 3.2 basic notions of *LfD* were presented. Section 3.3 reviewed methodologies for gathering demonstrations and the correspondence problem. Various techniques for teaching and building the demonstration datasets were presented, such as kinaesthetic teaching, visual demonstrations, motion capturing systems for recording demonstrations and the generation of robot trajectories with virtual reality or simulated environments. The framework employed through this work to learn robot skill motions from demonstrations was introduced in section 3.4. The approach is based on learning time independent models of the motion dynamics, estimated through a set of first order non-linear multivariate dynamical systems. Section 3.5 presented the formalization of the learning problem, a review of various regression techniques was presented. Also, three algorithms to learn the dynamics of demonstrated motions were introduced. A first approach to learning multivariate Gaussian was developed. This original formulation could not guarantee the learning of a stable estimate of the dynamics. The *BM* method was presented next, this method could produce a model of *DS* with local asymptotic stability at the target. Finally the *SEDS* method was reviewed with two objectives functions, *SEDS-likelihood* and *SEDS-MSE*. The *SEDS* formulation to learn the underlying dynamics of a motion can guarantee that the estimate of the dynamics is globally asymptotically stable at the target. Section 3.6 reviewed the methodologies used for the reproduction of the learned motions dynamics of the robot skills. Comparing the performance of the methods presented in section 3.5. Validation was performed by learning the estimates of 8 2-D motions and 4 3-D motions. The performance of the methods was compared across the demonstrated motions, the results are presented over Tables 3.4 to 3.9. Section 3.7 discussed the existing approaches for building libraries of basic movement primitives with the learned robot skills. A library of robot skills can be built with the learned models of motion dynamics in order to build an appropriate repertoire of movements for a robot to perform in several situations. In this work a module was successfully implemented allowing the robot to learn skills from demonstrations, we employed three different modalities to teach a robot the skill motions, recording the robot's trajectories as manipulated by a teacher with kinaesthetic teaching, employing vision system to track the teacher demonstrations, and recording the robot trajectories in an OpenRAVE simulated environment. After studying, comparing, and implementing various algorithms and techniques a *Dynamical System* approach, based on learning time independent models of the motion dynamics through a set of first order non-linear multivariate dynamical systems, was choose in this thesis to learn the robot skills employing the *SEDS-likelihood* method for the remaining experiments presented in this work.

4. REPRESENTATION OF ROBOT SKILLS KNOWLEDGE

4.1 Outline of the Chapter

This Chapter describes the development of a knowledge base for the storing and retrieval of the learned models of the skills. For a robotic system to perform different skills and tasks in a changing and unstructured scenario, it is important to develop a framework in which to organize the acquired knowledge in a manner that allows its retrieval in order to use it to deal with the current context constraints. An important aspect of the framework developed in this work is the existence of a knowledge base of the learned robot skills. Figure 4.1 shows the framework proposed through this work for the adaptation of learned skills to task constraints, highlighting the knowledge base for robot skills model discussed in this chapter. To introduce the contents of this chapter, first a review of the basic notions and concepts of knowledge representation and reasoning is given; a review of different approaches with a similar aim of building repertoires of basic motor skills is given next; the representational structure and organization of the knowledge base is presented; the representations used in the knowledge base for the storing of the robot skills and the process for searching the knowledge base are also described. The organization of this chapter is as follows:

- Section 4.2, presents an introduction to the topic of knowledge. The basic notions and concepts in the field of knowledge representation and reasoning are reviewed.
- Section 4.3, presents a review of similar approaches aimed at building repertoires of basic units of action, also known as movement primitives, which can represent a basic set of elementary robot motor skills.
- Section 4.4, presents approaches for the representation of the object's knowledge in a robot skill's knowledge base.
- Section 4.5, presents approaches for the representation of the action's knowledge in a robot skill's knowledge base.
- Section 4.6, presents approaches for the representation of the event's knowledge in a robot skill's knowledge base.
- Section 4.7, presents the developed representational structure of the robot skill's knowledge base.

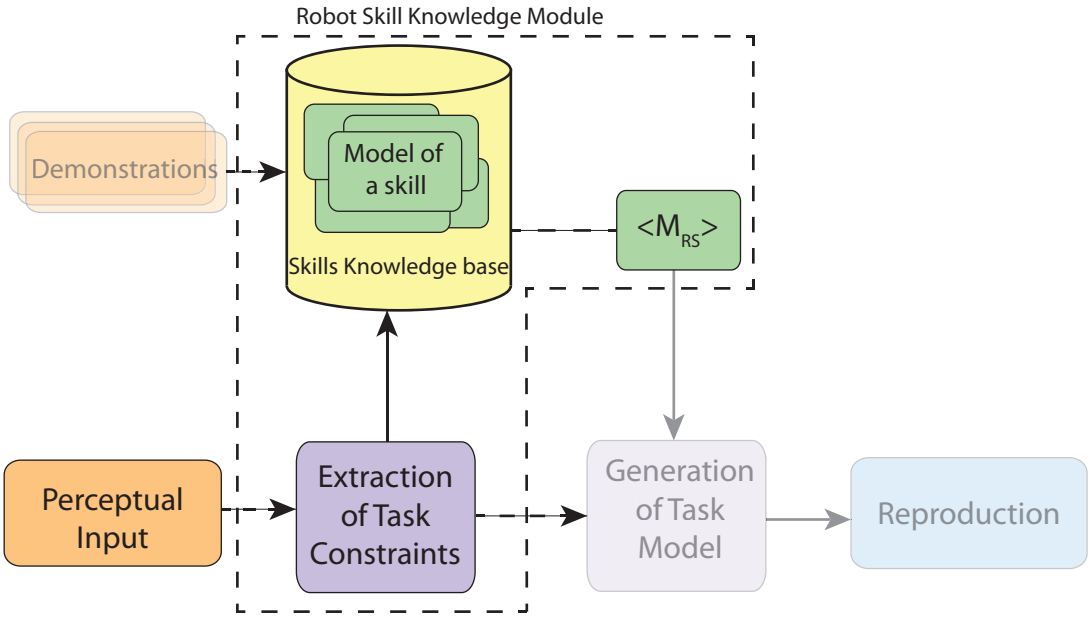


Fig. 4.1: *Module for representation of learned models of skills in a knowledge base, highlighted over the proposed cognitive framework for learning and adaptation of robot skills in compliance with task constraints. A knowledge base of the learned Robot Skills Models is built for their storage, classification and retrieval.*

4.2 Knowledge Representation and Reasoning

An important challenge for robotics, and particularly for robots acting in unstructured dynamic environments, which is a requirement for humanoid robots, is in dealing with internal representation and understanding the world. Cognitive science approaches aim at understanding, also with the hope of replicating, the processes of human intelligence, and the workings of the mind, with an emphasis on, the mental representations and mental operations involved in the development of thought and intelligent behaviour. A central point for the development of cognitive theories lies in studying the nature of knowledge; understanding the mechanism by which knowledge is acquired, stored, represented and operated upon in a way that generates intelligent thinking and behaviour. The field of philosophy has historically tried to explain the roots and essences of knowledge. One position, rationalism, supported by the philosophers Descartes, Spinoza, and Leibniz, among others, believed that knowledge can be gained solely by employing thinking and reasoning skills about things. In contrast, empiricism, defended by Locke, Hume, among others, believes that knowledge is acquired primarily from sensory experience [Russell, 2012]. Transcendental idealism, founded by Kant, and continued by Schopenhauer, and others, tried to reconcile the differences between rationalism and empiricism views, arguing that human knowledge, and our understanding of the external world depend on not merely our experience, but in both sensed experience and a priori concepts, innate to the mind

[Deleuze, 1985]. Experimental psychology theories of behaviourism, such as those of J.B. Watson, denied the mind, and suggested restricting themselves to examining the relationship between perceived stimulus and observed behavioural responses [Baum, 2003]. The constructivism view in philosophy, defined as a term by Jean Piaget, whose ideas could be traced back to Giambattista Vico, and others, states that concepts are mental constructs proposed in order to explain sensory experience, and that all knowledge is a compilation of human-made constructions, created through a series of individual constructs [Rockmore, 2005].

The most agreed view by cognitive scientists is that knowledge in the mind consists of mental representations, and that intelligent behaviour and thought are the resultant products of manipulating, reasoning and operating upon these internal representations. People, states the view of cognitive sciences, have mental procedures operating on mental representations that produce thought and action [Thagard, 2005]. Much of the debate in the field is centred upon the class and nature of these knowledge representations, on the representational mechanisms for acquisition, organization, and utilization of knowledge, and on whether the internal representations are even needed at all or another if paradigm is required. All through history, philosophers, psychologists, and other scientists, have formulated a variety of metaphors for the mind; for example, comparing it to a blank slate, 'tabula rasa', starting empty, without any mental content or knowledge built into it, and on which impressions are made recording knowledge from experience and perception. Other comparisons involve the analogy to a hydraulic device, with various forces operating on it, governing the energy flows which control behaviours; and to the operations of a telephone switchboard, with an intercommunicating network of cells, involving complex switching of information, responding to sensations, perceptions, thoughts, etc., [Thagard, 2005].

Currently, the dominant analogy in cognitive sciences has been comparing the mind and the brain to computers, where thinking can be understood as computational procedures. This metaphor assumes that the mind has mental representations analogous to data structures in a computer program, and computational procedures similar to programmed algorithms [Thagard, 2005]. Other theories have also arisen to challenge the major premise of the computational-representational understanding of mind (CRUM) thesis as the most suitable one for cognition. Connectionist models proposed novel ideas expanding theoretical frames of cognitive science about representation and computation that use neurons and their connections. The connectionist analogy is that mental phenomena can be described by interconnected networks of simple and often uniform units. Where neuron patterns and network connections can be compared to data structures, and neuron firing and spread activation is analogous for algorithms [Thagard, 2005]. Recent approaches in cognitive science have taken a growing interest in dynamical systems. The dynamical systems metaphor promotes thinking about the underlying forces, vector fields, from which observed patterns of behaviours emerge [Schöner, 2008]. In this view, the brain is thought of as a dynamic physical system and the processes in the mind can be described by difference and differential equations. The driving idea motivating the dynamical systems approach is that cognitive processes, contrary to the computational hypothesis of discrete representational operations, must unfold continuously and simultaneously in real time.

Therefore, a cognitive system would not be a sequential manipulation of discrete static representational structures, but rather, a structure of mutually and simultaneously influencing change [van Gelder and Port, 1995]. The agents' behaviour, in its full complexity, can finally be generated from the complex dynamical evolution of stable states and their instabilities in an interlinked non-linear dynamical system [Schöner, 2008]. The systems internal representations may be modelled, thereafter, not as simple inner states but as dynamical patterns of just about any conceivable kind [Clark, 2004].

Traditional commitment of cognitive sciences to a computational-representational view of the mind, that is a view of intelligence as a problem of symbol manipulation, faced increasing challenges and scepticism. These challenges have been explicitly stated in works of [van Gelder, 1995], [Thelen and Smith, 2007], [Wheeler et al., 1994], [Haselager et al., 2003], etc. Theirs is a challenge to the isolationist conception of the mind, and offer a rather radical rejection of representations. Their thesis is based on the idea that the symbolic computational-representational views of cognition are mistaken, and that cognitive agents do not require use of internal representation to act upon the world. The anti-representationalists claim is that computations of static symbolic internal representations form an inadequate analogy to explain the continuous dynamically complex patterns of behaviour that cognitive agents display; moreover, that internal representational mechanisms are not readily employed in nature in biological cognition. Many researchers have looked for approaches trying to completely disregard the use of representations and internal models as a whole. These efforts are best summarized by the behaviourist radical mantra of "the world is its own best model" [Brooks, 1990].

The view of the proponents of this hypothesis is that the representational approach is incapable of producing timely, suitable cognitive responses. The ontological commitment incurred by ascribing to a knowledge representation [Davis et al., 1993], can be seen as detrimental and counterproductive for developing intelligent physical agents. [Clark, 1997], addresses these challenges, and argues in favour of affording complementary approaches for adaptive success, instead of thinking in terms of competing perspectives. Here it is established, as a minimal common ground between representationalist and anti-representationalist, that complex persisting inner states are at the heart of cognitive phenomena, and that is not necessarily required a revision of the notion of internal representations, but rather a revision of the ideas on the kinds of inner states and processes which can possibly serve as vehicles of such representations. The critical distinction is not between representational and non-representational solutions but among an action-neutral form of internal representation, requiring disembodied symbolic computational processing, and action-oriented forms, in which a behavioural response is embedded into the representation itself [Clark, 2004]. The call is to beware of approaches relying only in intelligence on the head, and narrow representational contents, and rather, to take a harder look at temporally extended process that span brain, body and world. The major contribution of these challenges is for a general broadening of cognitive science from its historically narrow focus on disembodied, language-like reasoning towards approaches of embodied, embedded, situated, action and cognition [Beer, 2000].

The theories of embodied cognition underlined that cognition is constrained by the kind of body we possess, and emphasized the importance of action grounding, and the role played by bodily states [Borghi and Cimatti, 2010]. The focus in an embodied, embedded, approach, is in examining possibilities for action provided by the body and the environment. A necessary emphasis is placed on the close link of cognition with the sensory and motor processes and the environments in which these are immersed. Models of cognition must be embodied processes that capture the unfolding of cognition in time, and the associated sensory and motor surfaces embedded in the environment in which cognitive phenomena takes place [Schöner, 2008]. Therefore, an agent's potential for cognition is bounded to the motor capabilities of its body, dependent upon its physical characteristics and abilities, and its situatedness and possibilities of interaction with the environment. The claim is not an outright rejection of the legitimacy of representations, however in order to be valid, for embedded cognition, the representations are to be limited, physically grounded to the environment and oriented toward the specific needs of the given agent [Anderson, 2003]. It is clear that, despite the many challenges, some form of reasoning and representation of knowledge mechanism must be featured in a cognitive agent to produce the intelligent and adaptable behaviour that are desired.

The computational-representational, connectionist and dynamical systems theories of cognition mentioned earlier, beyond their differences in formalism and the technologies employed, differ markedly in their theoretical vocabulary and explanation on cognitive phenomena [Beer, 2000]. However, while there is a substantial difference between the presented accounts of cognition, this does not render the approaches incompatible; they can be complementary [Bechtel, 1998].

The computational theory is based on the existence of mental representations, and the presence in the mind of "algorithmic" processes that operate upon the representations; behaviours are produced by applying processes to the representations [Thagard, 2005]. The explanatory focus of the symbolic computational model is solely on the structure and content of the representations and the nature and efficiency of the algorithms [Beer, 2000]. Various kinds of representation can be considered, such as rules, concepts, analogies, frames, images, etc.

Connectionist theory is expressed as layered networks and simple, neuron-like, nodes and links. The connectionist approach employs a more implicit style of representation, replacing the symbolic nature of computational approach with numerical vectors and operations of vectors completion and transformation [Clark, 1997]. Here, representations involve simple processing units connected to each other and processes spread activations between the units via their connections, which produces the behaviour [Thagard, 2005]. In a connectionist theory the focus of explanation is on the network architecture, the learning algorithm, and the intermediate distributed representations that are developed [Beer, 2000].

The dynamical systems theory conveys a very different format than other cognitive theories. A dynamical model is expressed as a set of differential or difference equations, describing system state changes over time. This focus on system evolution and change over time is an important contribution of the dynamical system approach [Bechtel, 1998]. Dynamical system parameters, attractors, trajectories, bi-

furcations, etc., can be regarded with a representational status, storing knowledge which can influence behaviour [van Gelder and Port, 1995]. Here, thought can be described by variables governed by a set of non-linear difference equations, these equations state space and the nature of the systems dynamics can explain stable patterns of behaviours, phase transitions, or the appearance of unpredictable behaviours [Thagard, 2005]. The explanatory focus of the dynamical systems theory is thus on the space of possible trajectories and the internal and external forces that act over the trajectory unfolding over time, and not on the nature of the mechanisms that instantiate the dynamics [Beer, 2000].

Decisions about how to act are made, for a wide range of activities, based on what is known about the world. Intelligent behaviour is thus, clearly conditioned by knowledge [Brachman and Levesque, 2004]. The field of knowledge representation and reasoning is a part of artificial intelligence concerned with the mechanism of how an agent can use what it knows to decide what to do. Knowledge representation deals with how knowledge can be represented and manipulated in an automated way. The goal of knowledge representation and reasoning is the study of how knowledge can be, simultaneously, represented as comprehensively as possible and reasoned with as effectively as possible [Brachman and Levesque, 2004]. The most important issues related to an agent's needs in order to behave intelligently and to the computational mechanism which may allow for knowledge to be readily available to an agent as required. In knowledge representation and reasoning one's focus is on the symbolic structures for representing knowledge and the computational process for reasoning with those structures that must be created. In dealing with the topic of knowledge when building intelligent systems, the problems of representation and reasoning must always be taken together. It is not sufficient to state what needs to be known, in whatever formal representational language, and it is not sufficient either to develop reasoning procedures, which are effective for various tasks. There is a necessary trade-off between these two concerns; and it necessary to take into account the needs that reasoning with knowledge structures has on the form of languages used to represent knowledge. It is the interplay between representation and reasoning which makes the field relevant [Brachman and Levesque, 2004].

To understand the concepts of knowledge representation [Davis et al., 1993] proposes to review its meaning in terms of the five fundamental roles it plays. These roles provide a framework useful for characterizing a wide variety of representations and knowledge representation technologies; that is, basic representation tools such as logic, rules, frames, semantic nets, etc., which are used to build knowledge representations. For representations the fundamental task is capturing the complexity of the natural world. It must form an ontological commitment and provide a theory of intelligent reasoning. Representation and reasoning are inextricably intertwined. A knowledge representation is also a medium for pragmatically efficient computation and of human expression [Davis et al., 1993]. Those five roles help to characterize the spirit of the representations and representation technologies that are developed.

All representations function as surrogates for abstract notions, such as, actions, processes, beliefs, causality, categories, etc., allowing for a description of them to be available so they can be reasoned with. However, every representation would

ultimately be an imperfect approximation to reality, attending to some things and ignoring others, since a complete description of the world would not be possible or even practical or desirable. By choosing a representation and representation technology a set of decisions are made about what and how to see the world. The stances a representation takes on these issues and its rationale for those stances are indicators of what the representation says about how to view and reason about the world [Davis et al., 1993].

The representation technologies, logic, rules, frames, etc., embody a viewpoint of the kinds of things that are important in the world. For example, logic involves viewing the world in terms of individual entities and the relationships between them. Rules view the world in terms of attribute-object-value triples and the rules of plausible inference that connect them. Frames view in terms of prototypical objects. Thus, the commitment to a particular view of the world begins with the selection of a representation technology. The selection has a significant impact on the perception of the world and the task being modelled. Thus, existing representation technologies would supply its set of guesses about what to attend to and what to ignore in the world. Choosing among any of them means more than the selection of a representation, in it a conception of the nature of intelligent reasoning is also being made [Davis et al., 1993]. The selected representation would have inevitable consequences on how one sees and reasons about the world, so it must be selected consciously and carefully, trying to find one that is appropriate for the task. While the selection of tools and techniques are important, however, the field of knowledge representation is also much richer than that. It must be the central preoccupation of the field to understand and describe the richness of the world [Davis et al., 1993].

The fundamental commitment for representations is as tools for describing the natural world; their main role being working as a stand-in for real entities, substituting them for direct world interaction. The representations convey the gathered knowledge content, and function as stand-ins for the things that exist in the real world. Representations thus perform as functional abstractions of the perceived environment, encoding an agent's knowledge of its world, objects, actions and events into manageable internal structures; allowing for it to work, and reason, over the representations instead of acting directly upon the world. Since reasoning is an internal process, while the things it needs to reason about exist externally, this functional abstraction is important. The representations are structures standing in for something else outside the system, by virtue of relations such as similarity, casual history, and connections with other representations [Thagard, 2005]. An agent system, having useful representations, can therefore operate on them, abstracting itself beyond the world. A representation is a relationship between the two domains, an inner self and an external world, where the first is meant to take the place of the second [Brachman and Levesque, 2004]. This notion of representations, as proxies of the world and bridging interaction with the environment, is a vehicle of human thought. Performing operations with the representation is a substitute for operating with real things, that is, a substitute for direct interaction with the world. The role of representations as surrogates for the world leads to two important questions of correspondence and fidelity. There must be some form of correspondence between

its surrogates and its intended referents in the real world. Second is the problem of how close a surrogate can be to the real thing. A perfect fidelity would be, both in practice as in principle, impossible to obtain [Davis et al., 1993]. The imperfect surrogates also leads inevitably to having incorrect inferences. Independent of the reasoning and representation technologies employed, every sufficiently broad attempt at reasoning about the world will eventually reach incorrect conclusions. Therefore, the importance of the selection of a good representation is in minimizing the error for the specific task [Davis et al., 1993].

As already mentioned above, choosing a representation involves making a set of decisions about how to see the world, and making a set of ontological commitments about what part of the world to focus on. This is useful because the judicious selections of commitments provides the opportunity to focus attention on aspects of the world believed to be relevant [Davis et al., 1993]. The natural world offers an overwhelming complexity, the commitments incurred by the representational stance offers necessary guidance in deciding the parts of the world to attend to and the ones to ignore. By determining what and how to see the world, the representations allow one to cope with what could be otherwise untenable complexity and detail. The commitment that is made by choosing from different ontologies can produce sharply different views of the task at hand. An ontology can be written down in a wide variety of languages and tools. The commitment to a particular view of the world, thus, starts in the selection of a representation technology and accumulates from there as choices are made about how to see the world in these terms [Davis et al., 1993].

To use a representation, computations must be made with it. Reasoning in purely mechanist terms can be seen as a computational process. Questions about the computational efficiency are central to the notions of representation, but one can also not be overly concerned with them to the point of producing representations that are fast but inadequate for real use [Davis et al., 1993]. Knowledge representations must also be means for communication in which to express things about the world. The representations must fulfil the role of medium for expression and communication. This role matters since one must be able to speak the language, with heroic efforts, in order to use it to communicate with the reasoning system [Davis et al., 1993].

A representation can guide and facilitate reasoning if it has at its heart a theory of what reasoning to do. Representation and reasoning are inextricably and usefully intertwined, in this view, reasoning itself is in part a surrogate for action in the world. A knowledge representation is also a theory of intelligent reasoning. A representation can be examined in three components, first its conception of intelligent inference. The second component of a representation theory of intelligent reasoning is the set of sanctioned inferences. Thirdly, more than an indication of which inferences can legally be made is needed; an indication of which inferences are appropriate is also needed. Where the ontological commitment tells one how to see, the recommended inferences suggest how to reason [Davis et al., 1993]. The concept of reasoning is as disputed as those of representation, knowledge and intelligence, collecting inputs from various fields. Recalled by [Davis et al., 1993], the mathematical logic view is that, reasoning is a variety of formal calculation. The view in psychology sees reasoning as a characteristic human behaviour, symbolized by human problem solving. An approach rooted

in biology takes the view that the key to reasoning is in stimulus-response behaviours emerging from the parallel interconnection of simple processors. Approaches derived from probability theory, add to logic the notion of uncertainty, in which reasoning intelligently means obeying the axioms of probability theory. Reasoning is the formal manipulation of the represented collection of believed propositions in such a way as to construct representations of new propositions [Brachman and Levesque, 2004]. Different conceptions of the nature of intelligent reasoning lead to different goals and definitions of success, and different artifacts being created [Davis et al., 1993].

Knowledge representation hypothesis implies that we would want to build systems for which the intentional stance is grounded by design in symbolic representations. A knowledge base is a collection of symbolic structures representing what it believes and reasons with during the operation of the system. A knowledge base system can be understood at two different levels. At the knowledge level, questions concern the representation language and its semantics. It deals with expressing adequacy of a representation language and characteristics of entailments, including computational complexity. At the symbol level, questions concern the computational architecture and the properties of the data structures and reasoning procedures, including their algorithmic complexity [Brachman and Levesque, 2004]. Broader conception of representations are important, recognizing that a representation embeds a theory of intelligent reasoning, the ability to dissect some of the arguments about formal equivalence of representations, and that the central task of knowledge representation is capturing the complexity of the real world [Davis et al., 1993]. Human problem solving depends on what is important and interesting given the situation. A human expert learns to recognize and to react, they do not think and reason, as a knowledge base system would do, over an explicit representation. [Dreyfus et al., 2000] would describe the difference in terms of "knowing-that" and "knowing-how". "Knowing-that" is a conscious, step-by-step problem solving ability, with context free symbols, which we manipulate using logic and language. "Knowing-how" is the natural way one deals with things, when we just know what to do, and learn to subconsciously, recognize a situation and react. It generally makes a system slow down having to look up facts in a knowledge base and reason with them at runtime in order to decide what actions to take. The ability to make behaviours which depend on explicitly represented knowledge only seems to pay off when it is not possible to specify in advance the ways that knowledge will be used [Brachman and Levesque, 2004].

4.3 Developing a Repertoire of Robot Skills Knowledge

The main goal for humanoid robotics research is to build human like robots that can work alongside humans dealing with continuously changing environments and performing a wide variability of tasks. To achieve a complex behaviour such as this, it would be necessary to have an inclusive and comprehensive repertoire of robot skills. For this purpose the concept of movement primitives, also called movement schemas, basic behaviours, or units of actions, is proclaimed. Movement primitives are sequences of action that accomplish a complete goal-directed behaviour [Schaal, 1999],

as it has been reviewed in Section 3.7.

From the field of neurobiology it is generally regarded that humans employ basic motor primitives as an underlying mechanism of biological motor control. Evidence exists from human and animal experiments supporting the belief that sets of motor primitives are used to build a basis for voluntary motor control [Konczak, 2005]. Neuroscience studies in animals point to two neural structures, spinal fields and mirror neurons, which support the basis for a theory of perceptual-motor primitives [Mataric, 2000]. It is well accepted in these approaches, that for coping with the complexity of motor skills learning for robots, it is needed to rely on the insight that humans decompose motor skills into smaller subtasks. There are many theories about motor primitives which suggest that they are viable means for encoding humanoids' movements.

The movement primitives are sequences of action that accomplish a certain movement goal. The primitives encode groups or classes of stereotypical movements [Mataric, 2000]. Movement primitives in their most simple form can be thought of as simple as the elementary actions in the symbolic approach to imitation, with simple point-to-point movements employed by industrial robots [Schaal, 1999]. To deal with complex motions, a library of movement primitives can be built [Pastor et al., 2009], providing basic components from which multiple desired robot tasks can be performed by combination and superposition of the primitives. A robotic system equipped with a library of movement primitives, with a sufficient number of skills, can be thought of as possessing an adequate repertoire of actions to deal with a vast range of situations. A theory of primitives is a fundamental building block for motor control.

Learning the *Robot Skills Models* as they were presented in Section 3.6 can be a most suitable way to form basic primitives of movement, encoding within the model the motion dynamics of a demonstrated skill. Such collections of primitives are used to build a knowledge base from the learned motions of a task. Various examples can be found on building up a knowledge base from learned motion tasks. [Ude et al., 2007] presents a framework for synthesizing goal-directed actions from a library of example movements; different methods can be utilized for the construction of this movements library. [Zoliner et al., 2005b] deals with the integration of learned manipulation tasks into a knowledge base as well as enabling the system to reason and reorganize the gathered knowledge in terms of the re-usability, scalability and explainability of learned skills and tasks. In [Pastor et al., 2009] a collection of dynamic movement primitives is used to build a library of movements by labelling each recorded movement according to task and context.

The work of [Ijspeert et al., 2003] was first to suggest the idea of using *DS* as motor primitives. Their approach employed the *DMP* to learn and encode the dynamics of demonstrated motions. The control policies could be used to represent basic movements that form a library of motions. Defining the primitives in term of causal dynamical systems allows then to be parametrized by a small set of dynamical parameters and an input driving the overall dynamics. [Schaal et al., 2003], presents a conceptual imitation learning system which alludes to the concept of movement primitives to generate action behaviours. Perceptual elements are transformed into spatial and object information and are mapped onto a set of existing primitives, where a set

of movement primitives compete for a demonstrated behaviour. Motor commands are generated from input of the most appropriate primitive. Learning can adjust both movement primitives and the motor- command generator [Schaal et al., 2003]. Effectiveness of imitation learning with these dynamic system primitives was successfully demonstrated in a humanoid robot that learned a series of movements such as tennis forehand, tennis backhand and drumming sequences from a human teacher [Ijspeert et al., 2003].

[Mataric, 2000], proposed to structure the motor system into a collection of movement primitives, which then serve both to generate a movement repertoire to the humanoid robots, and to provide prediction and classification capabilities for visual perception and interpretation of movement. The movement primitives or behaviours are the unifying mechanisms between visual perception and motor control in their approach. They represent the generic building blocks of motion that can be implemented as parametric motor controllers [Mataric, 2000]. Such a primitive lets a robot reach toward various goals within a multitude of tasks; this allows for a small number of general primitives to represent a large class of different movements, such as reaching various places on and around the body. The general system segments the trajectory over time these segments are, at each point, matched to the expected output of each of the primitives with the observed input and the best match is selected. The output of the classification is a sequence of primitives and their associated parameters. These then go to the motor control system and activate the primitives in turn that reconstruct the observed behaviour [Mataric, 2000].

[Fod et al., 2000] presented a method for representing human movement in terms of a linear superimposition of simpler movement primitives. The primitives can be used as a lower-dimensional space for representing movement. In their model, the perceptual system is biased by the set of motor behaviours the agent can execute. Thus, an agent can automatically classify observed movements into its executable repertoire. In [Jenkins et al., 2000] perceptual-motor primitives formed a biologically-inspired notion for a basis set of perceptual and motor routines. Primitives serve as a vocabulary for classifying and imitating observed human movements, and can be derived from the imitator's motor repertoire. Their notion of a motion vocabulary comprises movement primitives that structure a human's action space for decision making and predict human movement dynamics. Through prediction, such primitives can be used to both generate motor commands for specific actions and perceive humans performing those actions, using a known vocabulary of primitives [Jenkins et al., 2007].

[Vecchio, 2002] developed a study of primitives of human motion, termed "movemes", using tools from dynamical systems and systems identification, decomposing it into a sequence of elementary building blocks that belong to a known alphabet of dynamical systems, which, in turn, can be composed to represent and describe human motions. [Vecchio et al., 2003] address the problem of defining conditions under which collections of signals are well-posed according to a dynamical model class and, thus, can generate the "movemes". Also, developed segmentation and classification algorithms in order to reduce a complex activity into the sequence of "movemes" that have generated it. [Vecchio, 2002] attempted to define primitives in terms of causal dynamical systems, that could be parametrised by a small set of dynamical parameters and by

an input which drives the overall dynamics. An "alphabet of movemes" is built to represent and describe human motion. Their experiments showed that it was possible to distinguish between the "movemes" in drawing tasks.

[Zoliner et al., 2005b] built up a knowledge base of manipulation tasks by extracting relevant knowledge from demonstrations of manipulation problems. Their work dealt with the integration of learned manipulation tasks into a knowledge base, as well as enabling the system to reason and reorganize the gathered knowledge in terms of re-usability, scalability and explainability of learned skills and tasks. The main goal was comparing newly acquired skills or tasks with already existing tasks and knowledge and deciding whether to add a new task representation or to expand the existing representation with an alternative. Gathered knowledge is reorganized and structured on the level of manipulation segments, enabling an execution system to select from multiple alternative operations [Zoliner et al., 2005b].

[Ude et al., 2007] presents a framework for synthesizing goal-directed actions from a library of example movements, different methods can be utilized for the construction of this movements library. The approach used a general representation based on fifth order splines. The proposed approach enables the generation of a wide range of movements that are adapted to the current configuration of the external world without requiring an expert to appropriately modify the underlying differential equations to account for perceptual feedback. In [Forte et al., 2012] trajectories are generalized by applying Gaussian process regression, using the parameters describing a task as query points into the trajectory database.

In [Pastor et al., 2009] a collection of dynamic movement primitives is used to build a library of movements by labelling each recorded movement according to task and context. Their work provides a general approach for learning robotic motor skills from human demonstration. Generalization can be achieved simply by adapting a start and a goal parameter in the equation to the desired position values of a movement. Feasibility of the approach is demonstrated with a pick-and-place operation and a water-serving task and could generalize these tasks to novel situations.

[Meier et al., 2011] approach aimed for movement segmentation with simultaneous movement recognition, assuming that a library of movement primitives already existed, and reduced the segmentation problem to online movement recognition. In [Muelling et al., 2013] the goal was to acquire a library of movement primitives from demonstrations and to select and generalize among these movement primitives to adapt to new situations. The primitives stored in the library are associated with a set of parameters that form an augmented state that describes the situation present during demonstration; these parameters are used as components in a mixture of motor primitives algorithm. To generate a movement, the system selects movement primitives from the library. A parametrized gating network is used in the mixture of primitives algorithm to activate components based on the augmented state and generate a new movement using the activated components.

The motor controller components of the movement primitives could be manually derived or learned. In this work a framework to build the models of the robot skills using *Learning from Demonstration* techniques, as described through Chapter 3, was chosen to learn the basic primitives of action. Traditionally, learning motions at a

low level representation failed to scale well to systems with many degrees of freedom. The learning of movement primitives, therefore, would benefit from coding the complete temporal behaviours that result in state-action representation that are compact and which need to adjust only a few parameters for a specific goal [Schaal, 1999]. Primitives must be characterized in parametric form to allow generalization and their applicability to different scenarios. It is important that the representations used for extracting units of actions also relate to the movement generation [Meier et al., 2011]. The representation of the robot skills must be flexible and compact enough to be able to store, use and retrieve this knowledge in efficient ways and allow the robot have a comprehensive repertoire of skills. Adequate representations are needed for the skill primitives in order to build a repertoire of robot skills.

4.4 Representing Objects in the Robot Skills Knowledge

Before one can start to deal with the issues of building a representation of the world and the commitments it must ascribe to for the representations of knowledge and the process of reasoning to work, a key decision must be made on which aspects of the world one will focus on and which aspects of the world one will choose to ignore and how the knowledge about the world would be structured. For any representational system the question of what is needed to be modelled and what can be ignored or abstracted away is a fundamental issue [Anderson, 2003]. The abstractions are necessary because no system can possible manage a world model that includes the whole of the world. Knowledge of the world, in a cognitive agent, can come from different sources and present different formalism. Knowledge about one's environment can come through perception, knowledge about a current situation may come from planning, reasoning and prediction, knowledge about other agents can come via communication and knowledge of the past come from memory and learning [Langley et al., 2009]. It is an important ability for an intelligent agent to deal with these various forms of knowledge in an effective manner. For instance, an agent must have the ability to recognize situations or events as instances of known patterns, and to assign these objects, situations or events to known concepts or categories [Langley et al., 2009]. Also, the ability to select among alternative actions and make decisions is needed. Therefore, an agent must be able to represent and store knowledge that would enable its activity. Preceding discussions have dealt with the numerous views that claim a rejection of the internal representation paradigm for developing cognitive agents, nevertheless, it is our belief that some form of representations is not only inevitable but also necessary and adequate for robotics. However, the representations must be limited and physically grounded to the environment; good representations must be selective and oriented to a particular use by a particular agent [Anderson, 2003].

One way to see the world, borrowing representational ideas from natural languages, would have us dealing with "objects", such as people, houses, etc., and "relations" among "objects", or "properties", such as red, round, etc., and "functions", such as fatherof, etc. Where almost any assertion can be thought of as referring

to "objects", and "properties" or "functions" [Russell and Norvig, 2010]. Traditional representations in artificial intelligence have focused on the symbolic discrete representation of objects and actions [Geib et al., 2006]. The objects-actions dichotomy is an important abstraction for the performance of robots as embodied cognitive situated agents. A majority of approaches in cognitive architectures focus on skill knowledge of how to generate or execute sequences of actions, while often relegating equally important conceptual knowledge dealing with categories of objects, situations or other concepts [Langley et al., 2009]. Therefore, much of an agent's knowledge must consist of skills, concepts and facts about the world. From what has been considered so far, one is lead to a view of the world consisting of objects, concepts, actions, skills, situations and events. The importance of dealing with objects when developing robotic agents which must perform in the world seems quite evident, since most of a robot's operations in the environment would be bound to the manipulation of an object. It seems clear that the representational attributions must be oriented to dealing with objects in the environment and the actions that can be executed on them. In order to deal with changing dynamic environments' representations must also have the ability to handle different situations or events. Recognizing different events or situations in the environments and the objects and actions that pertain to the current configuration of the environment is a crucial ability that the robotic systems discussed here must be able to possess.

Prior to populating a knowledge base with object classes it is necessary to state out what is called an ontology. An ontology determines the kinds of things that can be said to exist. The word "ontology" means a particular theory of the nature of being or existence [Russell and Norvig, 2010]. In philosophy, as an ontology is understood the study of the nature of being, becoming, existence, or reality, as well as the basic categories of being and their relations. An ontology defines, in the context of computer science, a set of representational primitives with which to model a domain of knowledge [Gruber, 2009]. The ontology defines the concepts, relationships, and other distinctions that are relevant for modelling a domain. By means of an ontology one determines the kinds of objects that will be important to the agent, the properties those objects will be thought to have, and the relationships among them [Brachman and Levesque, 2004]. Committing to an ontology requires choosing a particular view of the world. Once the choices are made one is left with a representational vocabulary specifying the domain, with definitions for typical classes or sets, attributes or properties, and relationships among class members [Gruber, 2009]. Ontologies can't provide complete descriptions of everything, but they leave place-holders where new knowledge for any domain can fit in [Russell and Norvig, 2010]. Ideally, an ontology would try to unify different areas of knowledge, general purpose ontologies should be applicable in more or less any special purpose domain. However, general ontological engineering has so far seen only limited success [Russell and Norvig, 2010]. Agreeing to an ontological representation is a difficult proposition, and usually most applications in artificial intelligence make use of special purpose knowledge engineering, designing their own ontology, tailored for a particular use.

Organizing objects into categories is a vital part of knowledge representation [Russell and Norvig, 2010]. It is essential to circumscribe the basic types of objects

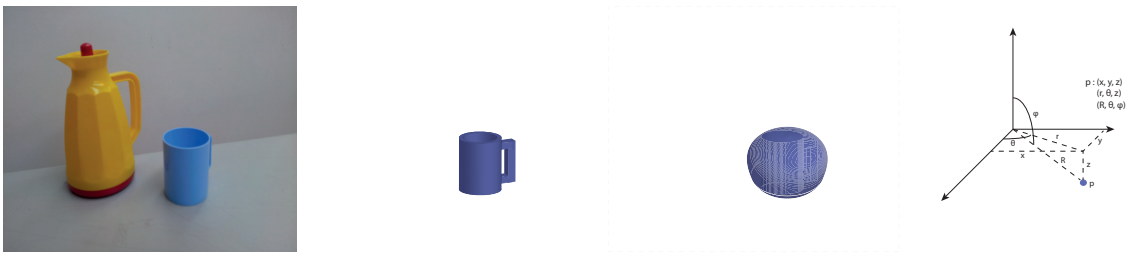


Fig. 4.2: *Representing knowledge of objects. (left to right): The real-world object. 3D model representation of the object. Convex bounding volume representation of the object. Representation of the object in Cartesian, spherical or cylindrical coordinates.*

our knowledge base would have, and to determine the set of attributes that our objects can have. In general, a good ontology should require only a few general rules. Once the types of our objects have been established one can capture the properties of the objects [Brachman and Levesque, 2004]. The ontology provides a set of features that serve to identify objects that can fit typical categories. One infers the presence of certain objects from perceptual input, infers category membership from the perceived properties of the objects, and then uses category information to make predictions about objects [Russell and Norvig, 2010].

A typical problem building a representational approach is that knowledge about an object could be scattered around the knowledge base [Brachman and Levesque, 2004]. The organization of the knowledge of objects in the world towards a manageable structure of objects' knowledge is a critical aspect of the design of a knowledge base. Organizing objects into categories is a vital part of knowledge representation; the approach is to group facts or rules in terms of the kind of objects they pertain to. Categories are the primary building blocks of knowledge representation schemes, the real world can be seen as primitive objects and composite objects built from them [Russell and Norvig, 2010]. Objects naturally fall into categories, but can also be members of multiple categories. The objects can also be made of parts, the relationship among an object's parts is essential to it being considered a member of a category. Building taxonomies is also an important aspect of general commonsense knowledge; the subclasses relations organize categories into taxonomy hierarchies [Russell and Norvig, 2010].

The framework of first-order logic encodes knowledge about the objects as logical expressions, each cast in terms of predicates and arguments, plus statements that relate these expressions in terms of logical operators. A model in first-order logic consists of a set of objects and an interpretation that maps constant symbols to objects, predicate symbols to relations on those objects and function symbols to functions on those objects [Russell and Norvig, 2010]. Production systems on the other hand, provide a more procedural notation, which represents object knowledge as a set of condition-action rules that describe plausible responses to different situations [Langley et al., 2009]. Semantic networks provide graphical aids for

visualizing the knowledge base and algorithms for inferring the properties of an object base on its category membership. Semantic networks allow the capability of representing individual objects, categories of objects, and relations among objects [Russell and Norvig, 2010]. The description logics system provides formal language for constructing and combining category definition, and for deciding subset and superset relationships between categories. The notation of description logics was designed to make it easier to describe definitions and properties of categories [Russell and Norvig, 2010]. [Geib et al., 2006], proposes the pairing of objects and actions in a single interface representation. Object-Action complexes are suggested as a framework for representing actions, objects, and the learning process that construct such representations [Krüger et al., 2009]. [Lemaignan et al., 2010] presents an embeddable knowledge processing framework along with a common-sense ontology designed for robotics.

A system dealing with objects in the real world must deal with various different forms and types of knowledge. Representing the objects' knowledge requires a structured approach. [Minsky, 1975], suggested the idea of using object-oriented groups of procedures, which where called frames. The frame concept offers a representation of an object or category, with attributes and relations to other objects or categories, assembling facts about particular object and event types and arranging the types into a large taxonomic hierarchy analogous to a biological taxonomy [Russell and Norvig, 2010]. Frames focus mainly on the recognition and description of objects and classes. The frame data structure specifies concepts in terms of attributes, called slots, and their values, called fillers. One would have special systems for important objects, but also a variety of frames for generally useful "basic shapes". [Minsky, 1975] pictured a great collection of frame systems stored in permanent memory, when the perception evidence suggests one will fit a frame is evoked to working memory.

The frame knowledge structure can be seen as an instance of an object-oriented representation analogous to the development in an object-oriented programming language. This could allow the frame representation of objects to share many advantages of object-oriented programming systems, like the specification of general classes, logical control, inheritance of methods, encapsulation of abstract procedures, etc. [Brachman and Levesque, 2004]. In general, there are two types of frames, individual frames used to represent single objects, and generic frames, used to represent categories of classes of objects [Brachman and Levesque, 2004]. Through inheritance of properties one can organize and simplify the knowledge base using categories. Much of the reasoning done with frames involves the instantiation of individual frames out of the generic frames. Filling some of the slots with some values and inferring others. The reliance on default values for when a reliable inferring of the slots is not possible is one important aspect of the frame system [Hayes, 1979]. A generic object frame holds all necessary information for the recognition and identification of an object into a category class, and any positional information and constraints relate to it and its situation in the environment. Object instance in the knowledge base would be described by the characteristic attributes of the objects; this could be, for instance, its color, shape, size, id tags, or any other relevant intrinsic information property of the

Object Frame: Example of generic object frame and instances of an object frame	
⟨Object-frame⟩ gObj ⟨Color⟩ none ⟨/Color⟩ ⟨Volume⟩ 0 ⟨/Volume⟩ ⟨Model⟩ none ⟨/Model⟩ ⟨Roles⟩ obstacle ⟨/Roles⟩ ⟨Position⟩ ⟨Cartesian⟩ 0 0 0 ⟨/Cartesian⟩ ⟨/Position⟩	
⟨Object⟩ ObjA ⟨instanceOf⟩ gObj ⟨Color⟩ Blue ⟨/Color⟩ ⟨Volume⟩ none ⟨/Volume⟩ ⟨Model⟩ none ⟨/Model⟩ ⟨Roles⟩ tool ⟨/Roles⟩ ⟨Position⟩ ⟨Cartesian⟩ 120 34 56 ⟨/Cartesian⟩ ⟨/Position⟩	⟨Object⟩ ObjB ⟨instanceOf⟩ gObj ⟨Color⟩ # FFFF00 ⟨/Color⟩ ⟨Volume⟩ none ⟨/Volume⟩ ⟨Model⟩ none ⟨/Model⟩ ⟨Roles⟩ obstacle ⟨/Roles⟩ ⟨Position⟩ ⟨Cartesian⟩ 30 -45 78 ⟨/Cartesian⟩ ⟨Spherical⟩ 95 35 -56 ⟨/Spherical⟩ ⟨/Position⟩

Tab. 4.1: *Object Frame example for a generic object frame and instantiations of particular object frames.*

object that allows for its identification.

Figure 4.2 presents different modes for the representation of an object location knowledge. The leftmost image corresponds to a real-world scene, representing the object as it is. Managing a full model of the world is a very demanding task. The real-world object can be represented by its 3D model; this could be directly computed from sensory input or retrieve from memory given a prior recognition step. Complete 3D models are not always necessary, a simpler convex bounding volume representation capturing the occupied space of the object can suffice, for instance when thinking of the object as an obstacle to avoid in a path. Typical tasks in robotics need only to rely on the knowledge of the object position in either of Cartesian, cylindrical or spherical coordinate frames of reference, making an object representation in terms of its point coordinates a valid one for this objectives. It must be noted that having one or other representation can lead to a very different set of computations and tasks that the robot could be able to perform with an object. Yet, these representations are not exclusive and any combination of these modes could be present in a knowledge base if the system is designed for it. In this work the data structure of Frames is used to store knowledge about the objects in the environment in our knowledge base. Table 4.1 shows an example of the object frame. A generic object frame is described, and two instances derived from the generic frame are also present. An instance of an object frame inherits from the properties and default values of the generic frame, but this does not prevent it having properties and updating its values on its own.

An object frame could also represent instances of two or more generic frames or be composed of other object frames as sub-parts. Important properties of the object frames their name, position and role values for their identification, localization and relationship to the rest of the knowledge base.

4.5 Representing Actions in the Robot Skills Knowledge

As outlined in the previous section, the representational attributions of objects and actions, and perhaps more importantly the interrelation between the objects and action representations, is a fundamental concern when executing tasks in the world. The main role of a humanoid robotic system is to act and achieve tasks and goals operating in complex environments. The robots' actions would generally involve the presence of an object, or several objects, plus the possible interaction with human partners. Deciding on the model for the representation of actions is an essential undertaking for robotics and cognitive science research efforts. Thinking beings ought to be considered as acting beings in which cognition is a situated activity [Anderson, 2003]. It is important to note that actions are not performed in a vacuum, the cognitive process does not occur in isolation, actions are not performed disconnected from their embodied presence and the effects they have on the world [Nehaniv and Dautenhahn, 2001]. These effects would be described in terms of combinations of actions, states and goals.

Robotic systems, executing tasks in unstructured environments, must have functional representations for actions that facilitate the robot performance with objects and their environment. In section 4.2, the challenges to the symbolic internal representations and the stance for an embodied approach to cognition were reviewed. The embodied view of cognition's most pressing concern lies in the interaction between an agent's body with the environment [Haselager et al., 2003], and a distrust of the idea that cognition and knowledge representations are purely symbolic mental processes separated from action in the world. However, despite the various challenges, an outright rejection of internal representations also seems to be an incomplete approach. To produce the intelligent and adaptive behaviours that we desire, a cognitive agent must feature some form of reasoning and representation of knowledge.

Human problem solving abilities involve the cooperation between internal representations, computations and environmental interactions. [Clark, 1997] addressed the challenges to internal representations and argued in favour of adopting complementary approaches rather than thinking in terms of competing perspectives. Representations, in order to be valid for embedded cognition, are to be limited, physically grounded to the environment and oriented towards the specific needs of the given agent [Anderson, 2003]. Therefore, a distinction must not be made between representational and non-representational solutions but among the action-neutral forms of internal representations, requiring for disembodied symbolic computational processing and more action-oriented forms of representation, in which the behavioural response is embedded into the representation itself [Clark, 2004]. Real world cognitive processes occur in very particular environments and are employed for very practical

ends and exploit the interaction and manipulation of external props [Anderson, 2003].

The previous discussion points to the realisation that the ways in which a robotic system can act hinges on what its embodiment and the environment allow. This is tied to the object and situation oriented concept of affordances [Krüger et al., 2009]. When thinking of actions' representations the concept of affordances is essential, as the concept of an ontology was for the discussion of object representations. The representations of objects and actions are related in terms of their affordances. The system's actions are embedded in affordances' representations of objects and action pairs [Şahin et al., 2007]. The concept of affordances refers to the perceived and actual properties of things; particularly to properties that are fundamental to determine how a thing could possibly be used [Norman, 1988]. An affordance is the relationship between a situation, usually including an object of a definite type, and the actions that it allows [Krüger et al., 2009]. The concept relates to the perceived features in an entity, regarding how they can be used to do something. The affordances are proprieties of the objects and of the kinds of interactions they can support. An affordance is the observed availability of things to certain intervention [Anderson, 2003]. Affordances of an object are thought to be directly perceived by the agent, perception is shaped in terms of actions; the world constantly invites action [Anderson, 2003]. However, an affordance is not accurately explained as an element of an object representation, they are also related to the environment and to acting agents. An affordance is also a relationship between the abilities of an agent and the features of an environment; it is equally a reality of the environment and of the actions of an agent and it can be both physical and psychical and, at the same time, neither [Gibson, 1986]. Hence, affordances refer to the actions' possibilities that the object presents in an environment. Yet, not only the tools, but also the rest of the environment can provide affordances in a situation [Nehaniv and Dautenhahn, 2001]. The affordances of the environment are what it offers to the agent, what it provides or furnishes is a relationship with the environment, the object, and the agent. An affordance can point both ways, to the environment situation and to the observer morphology [Gibson, 1986]. Affordances depend not only on the objects and their design but also on their embeddedness to the environment and on the particular bodily structure and configuration of the agent who might use them [Nehaniv and Dautenhahn, 2001]. Finally, an affordance can be defined as an acquired relationship between a behaviour, or action, of an agent and an entity, for instance an object, such that the application of the behaviour on the entity generates a certain effect [Şahin et al., 2007]. [Barck-Holst et al., 2009] presents two approaches to modelling affordance relationships between objects, actions and effects. A first approach uses a voting function to learn which objects afford which types of grasp. The second approach uses an ontological reasoning engine for learning affordances. [Varadarajan and Vincze, 2012] describes AfRob, an extension of an affordance network or robotic applications. AfRob offers modules to enable robots to interact and grasp objects through the generation of grasp affordances.

Now, attention must be turned to the mechanism for action representation. In order to be general actions must be characterized in parametric form [Fod et al., 2000]. When thinking in terms of robotic control, computations for actions are captured as continuous transformations of continuous vectors over time. These vectors may be

used to represent different continuous values, like absolute points in three dimensional spaces, joint angles, force vectors, etc. [Geib et al., 2006]. Efforts in artificial intelligence research has typically focused on modelling high-level conceptual state changes that result from the execution of actions, and not on the low level continuous detail of action execution. The representation in artificial intelligence focuses on discrete symbolic representations of objects and actions, generally employing propositional or first-order logic [Geib et al., 2006]. In order to describe dynamic environments and the effects actions have on the world in a symbolic logic formalism situation calculus can be used. The basic concepts in the situation calculus are situations, actions and fluents [Fangzhen, 2007]. A situation is an instant of the state of the world [Funge, 1999]. Situation are defined as a period of time during which a certain set of properties hold; whereas the actions are the cause of state transitions [Belleghem et al., 1995]. The actions are what make the dynamic world change from one situation to another when performed by agents [Fangzhen, 2007]. The fluents are situation-dependent functions used to describe the effects of actions [Fangzhen, 2007]. Any property of the world that can change over time is known as a fluent [Funge, 1999]. Another formalism is the use of event calculus. The event calculus is a formalism for reasoning about action and change. In event calculus there is one real line of time points, and the events are the occurrence of an action at a certain point in time [Belleghem et al., 1995]. Both the situation and event calculus provide rich frameworks for solving problems in dynamic systems. Situation calculus and event calculus share the property of being initiated and terminated by actions [Belleghem et al., 1995]. In situation calculus the actions are hypothetical and time is tree-like. In the event calculus, there is a single time line on which actual events occur. [Mueller, 2007].

In [Krüger et al., 2009] object action complexes are proposed as a framework for representing actions, object and the process that constructs such representations at all levels. The object action complexes can be used as an interface between the very different representation languages of robot control and artificial intelligent planning [Geib et al., 2006]. They combine the representation strengths of STRIPS planners, the concept of action affordance, and the logic of event calculus [Krüger et al., 2009]. Pairing actions and objects in a single representation interface captures the needs of both high level action representation and low level control [Geib et al., 2006]. The execution of object action complexes is done in a hierarchical system with different level coding actions at different levels of abstraction [Krüger et al., 2009].

As has been discussed throughout this section, the principal aim of a situated agent is to take actions appropriate to its circumstances [Beer, 2000]. Fitting representations are essential for that goal. General approaches from artificial intelligence and logic base reasoning see the world more in terms of discrete time experiences. However, real-world action is a continuous time phenomena. State and action representations are dynamic entities [Krüger et al., 2009]. Cognitive systems are not discrete sequential manipulations of static representational structures, but rather, a structure of mutually and simultaneously influencing change [van Gelder and Port, 1995]. In order to acquire an internal representation of an affordance, an agent must carry out a complex encoding of the sensory stimulus; to reproduce the corresponding action, an agent must decode the encoded representation of the actions into proper signals.

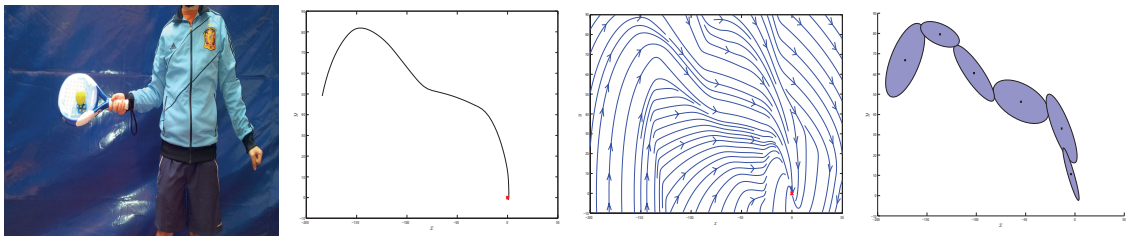


Fig. 4.3: *Representing knowledge of actions. (left to right): Agent action execution in the real-world. Representation of the action as a point to point vector trajectory. Representation of the action as an attractor landscape of skill dynamics. Representation of the action encoding the skill dynamics in a Mixture Gaussian Model.*

The embodied approach of cognition calls for the representations to be encoded in the body and not in the head [Anderson, 2003]. A dynamical system theory approach to cognition provides a way to overcome the separation between mind and the world that was largely prevalent in most work on artificial intelligence [Bechtel, 1998]. A dynamical approach is promising for providing a unified theoretical framework for cognitive science, especially when coupled with a situated embodied perspective on cognition [Beer, 2000]. The working hypothesis of the dynamical approach is that through increasingly sophisticated use of internal states to mediate between perception and action, more cognitive behaviours emerge from the dynamics of situated action [Beer, 2000].

Various proponents of a dynamical system approach to cognition also advocate for a complete rejection of representations, notably the work of [van Gelder, 1995]. Yet, as addressed in section 4.2, the provision of an inner model is not an impediment for real-time success, but actually enhances fluent real time action [Clark and Grush, 1999]. Most of these challenges stem from a mistaken idea that representations are useful as a representation for the system must be contemplated as a representation by the system processing [Clark, 1997]. The dynamic systems theory provides an alternative to the traditional formats of representations considered in cognitive science, yet, despite the differences between the approaches, they need not be incompatible they can be complementary [Bechtel, 1998]. A dynamical relationship of a representation with what it represents does not undercut its status as a representation. Something can stand-in for something else being coupled to it in a dynamical manner, and determining its response by being so coupled, which alters the thing being represented [Bechtel, 1998]. A wide variety of aspects of dynamical models can be regarded as having a representational status, such as states, attractors, trajectories, bifurcations, and parameter settings [van Gelder and Port, 1995]. The dynamical models are not based on the transformations of representational structures, the representation in a dynamical systems theory has radically different formats from others used in cognitive science [Bechtel, 1998]. However, the dynamical systems can store knowledge and have this stored knowledge influence their behaviour [van Gelder and Port, 1995].

Research in cognitive science has explored a wide variety of representational for-

Action Frame: Example of action-affordance frame

⟨Action-Affordance⟩ gAct
⟨ObjectList⟩ gObj1 gObj2 gObj3 ⟨/ObjectList⟩
⟨iniConditions⟩ 0 ⟨/iniConditions⟩
⟨Skill⟩ $\mathcal{M}_{RS}1$ ⟨/Skill⟩

⟨Action⟩ Act1
⟨instanceOf⟩ gAct
⟨iniConditions⟩ ... ⟨/iniConditions⟩
⟨Skill⟩ $\mathcal{M}_{RS}1$ ⟨/Skill⟩

⟨Object⟩ gObj2 ⟨/Object⟩

⟨SkillModel⟩ $\mathcal{M}_{RS}1$
⟨Prior⟩ 0.302 0.295 0.403 ⟨/Prior⟩
⟨Mean⟩
-424.72 173.09 487.24 -747.64
-118.99 4.04 534.15 -72.19
-295.90 538.47 -1030.21 -644.18 ⟨/Mean⟩
⟨Covar⟩
4.04e+3 -5.63e+3 1.33e+4 1.09e+4
-5.64e+3 9.60e+3 -2.65e+4 -1.70e+4
1.33e+4 -2.65e+4 1.02e+5 4.60e+4
1.09e+4 -1.70e+4 4.61e+4 3.65e+4
8.42e+3 -7.55e+2 -3.23e+4 8.17e+3
-7.55e+2 1.34e+2 2.11e+3 -1.17e+3
-3.23e+4 2.11e+3 2.07e+5 -2.96e+4
8.17e+3 -1.17e+3 -2.96e+4 1.27e+4
2.43+4 1.27e+4 -6.27e+4 4.01e+4
1.27e+4 8.81e+3 -3.83e+4 2.05e+4
-6.27e+4 -3.83e+4 2.81e+5 -6.72e+4
4.01e+4 2.05e+4 -6.72e+4 1.01e+5 ⟨/Covar⟩

Tab. 4.2: *Action-Affordance Frame example for generic action frame and instance of a particular action frame and skill model.*

mats; the dynamical system theory introduces new notions, such as the concepts of trajectories and dynamic attractors. One important contribution of dynamical system theory is that it focuses on representations that change as the system evolves [Bechtel, 1998]. A crucial difference between traditional symbolic computational models and dynamical models is that the rules that govern how the system behaves are defined over the entities that have representational status in a computational model, whereas for a dynamical model, the rules are defined over numerical states [van Gelder and Port, 1995]. For a dynamical system theory approach, the processes within the system are not defined over representations [Bechtel, 1998]. Namely, the dynamical systems can be representational without this meaning having the rules that govern their evolution defined over representations [van Gelder and Port, 1995]. The dynamical system theory is revolutionary in adopting a different concept of explanation than the mechanistic conception adopted by most cognitive scientists [Bechtel, 1998].

All through Chapter 3 the framework for teaching and learning the robot skills by *Robot Programming by Demonstration* was presented. The robot skills ought to enclose the knowledge of the task to allow generalization of the skill for reproduction

and to form full goal directed actions. The idea of employing autonomous dynamical systems was proposed as an alternative approach for representing movements as mixtures of non-linear differential equations with well-defined attractor dynamics [Ijspeert et al., 2001]. The dynamic system can be generally expressed as differential equations of the form $\dot{x} = f(x, \theta)$, as per Eq. 3.2 . Autonomous non-linear dynamical systems are a powerful mechanism to modulate the control policies by learning the model of the skill building a stable estimate of f based on a set of demonstrations. The dynamical systems approach to skill learning can offer a fast, simple and powerful formulation for representing and generating movement plans. The dynamical systems framework allows it to comply with the attractor dynamics of a skill, modulating it with a set of non-linear dynamical systems that form an autonomous control policy for motor control. Statistical learning techniques are used to arbitrarily shape the attractor landscape of the control policy for encoding in it the desired trajectory. The end-effector trajectories of a skill action are modelled in terms of a dynamic systems approach, as in [Schaal et al., 2007] for an autonomous dynamical system encoding of the action. The *Robot Skills Models* are learned by estimating the non-linear function f, a time independent model of the action is estimated through a set of first order non-linear multivariate dynamical systems as in the frameworks presented in [Gribovskaya et al., 2010] and [Khansari-Zadeh and Billard, 2011], described in Section 3.5, following the method of Table 3.3. Therefore the robot skills are modelled by the parameters θ of $\hat{f}$. $\bar{\mathcal{M}}_{RS}$ defines a *Robot Skills Model* determined by $\hat{f} = \{\mathcal{N}^1(\xi; \theta^1), \cdots, \mathcal{N}^{\mathbf{K}}(\xi; \theta^{\mathbf{K}})\}$, where $\theta^i = \{\pi, \mu, \Sigma\}$ of the $\mathcal{N}^i$ Gaussian function, defined by Eq. 3.18, are the prior, π^k, the mean, μ^k, and the covariance matrix, Σ^k, of the $\mathbf{K}$ Gaussian and they encode the representation of the skill action in a dynamical system approach.

Figure 4.3 shows different representations for a skill action. The leftmost images display a tennis swing skill execution of the action by a humanoid agent. Common action representations in robotics rely on vector trajectories describing explicitly the positions for the robot control at every point. Dynamical systems theory allows to represent the skill action in terms of their attractor dynamics. A dynamic system representation allows it to focus on the internal and external forces that act over the trajectory unfolding over time. Here, the dynamics of the skill action are encoded in a statistical approach employing the Gaussian mixture models.

Table 4.2 shows an example of the action-affordances frame. Generic action frames have an associated *Robot Skills Model* of the encoded skill action dynamics. As stated by [van Gelder and Port, 1995], the dynamic models representation status are defined over numerical states. In addition to the model of the skill, the action frame links actions with the corresponding objects that afford them. Generic action frames list all available objects for such action, the particular instances of an action frame, created by the system in the environment, presents only one object affordance for the execution of the action. For instance, lets consider a $\langle Pick \rangle$ action. A generic $\langle Pick \rangle$ action-affordance frame would hold a robot skill model encoding the action and a list of objects which afford the action, like spoons, forks, knifes, etc. While particular instances of the action frame, such as $\langle Pick\ Spoon \rangle$ serve as representational tools for the execution of an action upon a specific object found in the environment.

4.6 Representing Events in the Robot Skills Knowledge

Sections 4.4 and 4.5 have discussed mechanism for representing objects and actions respectively. However, focusing only on these two aspects would not be enough to develop the knowledge representation structures needed by the humanoid robotic systems that are the aim of this work. In addition to objects and actions the representational attributes need to take into account the state of the world grounding the representations to the environment, the task at hand and the current situation or present events. As discussed in section 4.2, one of the roles of representations is as stand-ins for external things outside the system. A robotic system would use representations to operate on them and not directly over the world. The system representations should include objects, actions, tasks goals and world event configurations, as in the representations of Figure 1.5. This does not require building up complete models of the agent's body and the environment, the stand-ins are only needed for those aspects that are relevant for guiding behaviour [Bechtel, 1998]. The major goal for humanoid robots, cognitive systems and embodied situated agents is to take the actions which are appropriate to take in the present circumstances of the world. There are many resources in service of this objective, including the physical properties of an agent's body, the structure of its immediate environment and its social context [Beer, 2000].

One of the most important properties of the world is change. Change is having an action move you from a given situation to a new one [Brachman and Levesque, 2004]. Propositional logic representations have limitations, such as tying directly the notion of time. Situation calculus gets around these limitations by replacing the notion of linear time with branching situations [Russell and Norvig, 2010]. Situation calculus takes into account situations and actions in the domain [Brachman and Levesque, 2004]. The situations are complete states of the world at some point in time, and a sequence of actions leading from some initial situation to the given actual situation [Brachman and Levesque, 2004]. Situation calculus was designed to describe a world in which actions and situations are discrete, instantaneous and happening one at a time, making situation calculus limited in its applicability [Russell and Norvig, 2010]. Event calculus was introduced as an alternative formalism which is based on points of time rather than on situations. Event calculus opens the possibilities of talking about time, and time intervals. Events, actions and time could still be represented either in situation calculus or event calculus representations [Russell and Norvig, 2010].

The object action complexes, described in [Geib et al., 2006], define instantiated state transition fragments to be a situated pairing of an object and an action that captures a fragment of the planning domain's state transition function. The fragments are defined as a tuple $\langle s_i, mp_j, Obj_{mp_i}, s_{i+1} \rangle$, comprising the initial sensed state of the world s_i, a motor program instance mp_j, the whole object containing the component the motor program was defined relative to Obj_{mp_i}, and the state that results from executing the motor program s_{i+1}. The instantiated state transition fragments contain all of the information the robot has about the two states of the world [Geib et al., 2006]. Much of the world space in S will be irrelevant for a particular complex since it is not required for the performance of the action and the action will not affect it, the

system should avoid expending resources in observing these non-relevant parts of the world [Krüger et al., 2009]. It is important to reduce the world space to only the features pertinent to the current action. The representation in [Geib et al., 2006] are bounded unto two states of the world, an initial state and a desired end state, and to planning state transition function from one to another. The discussion in Chapter 2 discourages us from this type of planning approach for applications such as humanoid robotics.

For our goal the representations are to be at one time larger, but also more structured and more intimately connected, combining knowledge of action and objects with the situations of the environment, the system tasks and the effects of execution. [Minsky, 1975], suggested the idea of using object-oriented groups of procedures to recognize and deal with new situations. The term *frame* was used for the data structure that represents these situations [Brachman and Levesque, 2004]. Frames were put forward as a set of ideas for the design of a formal language for expressing knowledge [Hayes, 1979]. A Frame is a collection of questions to be asked about a hypothetical situation; it specifies issues to be raised and methods to be used in dealing with them [Minsky, 1975]. To use frames is to make a certain kind of assumption about what entities will be assumed to exist in the world being described [Hayes, 1979].

Frames are essentially bundles of properties. A frame is a data-structure intended for representing a stereotyped situation [Minsky, 1975]. Attached to each frame are several kinds of information. Some of this information is about how to use the frame. Some is about what one can expect to happen next. Some is about what to do if these expectations are not confirmed [Minsky, 1975]. It is made up of slots which can be filled by expressions named fillers which may themselves be other frames. Given a frame representing a concept, we can generate an instance of the concept by filling in the slots. A frame instance denotes an individual, and each slot denotes a relationship which may hold between that individual and some other [Hayes, 1979]. An individual frame could look like *(Name-frame: < slot1 filler1 >, < slot2 filler2 >, ...)*. To help understand the concept consider a generic room frame as representing the general idea of a room with generic slots that can later be filled by individual room frames, such as a *kitchen room*, *living room*, *bedroom* inheriting from the generic room frame and filling them with their own special characteristics. The individual frames are a specialization of the general one, [Brachman and Levesque, 2004]. *(Kitchen-frame: < Is-a: room >, < Role: cooking >, ...)*.

Frame theory adopts a structured approach, assembling facts about particular objects and event types and arranging the types into large taxonomic hierarchies [Russell and Norvig, 2010]. The idea behind the approach is that when one encounters a new situation one would select from memory a structure called a *frame*. This is a remembered framework to be adapted to fit reality by changing details as necessary [Minsky, 1975]. Collections of related frames are linked together into frame-systems. The differences between the frames of a system can represent actions, cause-effect relations, or changes in conceptual viewpoint [Minsky, 1975].

It is useful to think about frames as a network of nodes and relations. The top levels of a frame are fixed, and represent things that are always true about the

supposed situation. The lower levels have many terminal slots that must be filled by specific instances or data [Minsky, 1975]. Collections of frame systems are stored in memory, and one of them is evoked when perceptual evidence makes it plausible that the scene will fit [Minsky, 1975]. When a proper frame is retrieved its slots are filled with available information, its default assignments become instantly available, and the more complex assignment negotiations are completed later as they become available [Brachman and Levesque, 2004]. Certain assignments to the slots terminals are compulsory, others are optional, and others take default assignment values in the absence of better information. The theory states that frames are never to be stored in long-term memory with unassigned terminal values. Instead, frames are stored with weakly-bound default assignments at every terminal, where values can be changed dynamically when more suitable information is deemed to be accessible [Minsky, 1975]. The process of matching a proposed frame suitable to represent the current situation is controlled by the system current goals and by information attached to the frame [Russell and Norvig, 2010].

The representations of events are, thus, largely concentrated on two major frames, one of the system tasks and goals' knowledge, and one representing the current state of the world knowledge. Representations of the task event knowledge consist of the agent's knowledge about what it is doing or trying to achieve. This is the knowledge about its purpose, its commands, its goals, both global and local, its planned actions, and the relationships between them and the state of execution of the task in the world. Task event frames would hold knowledge for the requested execution of a task. Such as the task goal, task actions, including proper instances of required action frames, task start, end and invoking conditions, etc. Task events are instantiated from recognizing matching invoking conditions for the event frame or by directly giving the system high level commands for a task execution corresponding to a particular task event. The representation of world event knowledge consists of the agent's knowledge about the situation of the environment its operating in. That is, knowledge about objects and places and their relationships. The representation of a world event frame would try to maintain an accurate model of the agent's environment, as it is being explored, so the world frame holds knowledge of objects being perceived as well as the most recent assumptions of objects no longer in the current view that are reasonably thought to still lie around. Continuous operation of the robot and its perceptual system provides updates and reinstantiations of the world event. To clarify these points we can revise the example raised in Chapter 1, represented in Figure 1.5. A robot is requested to place a spoon inside a cup on top of a saucer plate on a table. Aside from object and action frames already stored in the knowledge base, two frames are created at the start of the robot operation. A world event frame is created from the robot's perception of its environment, so in this case it would reflect knowledge about detected objects such as the cup, the spoon, the plate or any objects that are present and perceived by the robot, representing, for the robot system, their positions, states, etc. The world event frame representation correlates to the world state dimension from Figure 1.5. The task event frame is constructed from the given task robot command representing the knowledge of the task execution, in this case it describes task goals, state, steps. The task event frame representation correlates to the goal dimension from Figure 1.5.

Fig. 4.4: *Representing event knowledge. (left) Chessboard state during play. A chess master can recognize from this state, world event, the pieces and patterns important to its objective. (top) Board is reduced to relevant knowledge which would lead the player next moves, active view event. (bottom) Whites have a checkmate, task event goal, in three moves. (right) Football game just after the snap. A professional QB can read the defensive coverage, world event, to instantly recognize favourable matchups. (top) Field is reduced and only the position of the marked players is important for the play. (bottom) Having a right read on the defence leads to a successful completion of a pass, task event goal, to an open player for a first down.*

When an agent encounters a new situation a viewed event scene is analysed by assembling and instantiating frames, the system should watch for certain kinds of events and inject proposed reasons, motives, and explanations for them [Minsky, 1975]. In computer vision systems images seem to change so quickly, as fast as the scene does, that performing the computations for instantiating the representations at such pace does not seem to be computationally efficient. [Minsky, 1975] theory proposes that changes in the frame-structure representation proceed at their own pace. The system makes small changes whenever possible. In such a complex problem it is not possible to cope with many details at once. At each moment, one must work within a reasonably simple framework, and the illusion of continuity is due to the persistence of assignments to terminals common to different view-frames [Minsky, 1975]. Almost any event, action, change, flow of material, or even flow of information can be represented to a first approximation by a two-frame generalized event [Minsky, 1975]. While the different viewpoints help to insulate the parts of the potential contradiction from one another [Hayes, 1979].

Since computational resources are limited, what is important to consider here is an agent capacity for discrimination and focusing attention. Humans do not process the whole of a scene, one constantly discriminates information from a scene, categorizing, grouping and discarding chunks of information. An engage worker would generally focus all of its attention into a very small region of features deemed important for its labour. Hence it is desirable to have some indication as to which parts of the world to focus attention on, and to be able to discriminate from the whole information of the world only the important features of the current situation toward the current action. Here, questions arise as to what is relevant, where must attention go, what point of view to take, how to construct this focus view that would drive what is taken

Event Frame: Example of the task and world event frame and instances of an active view event frame	
⟨Task-event⟩ gTask	⟨World-event⟩ Env1
⟨Goal⟩ ... ⟨/Goal⟩	⟨Time⟩ 1 ⟨/Time⟩
⟨ActionSet⟩ act1 act2 ⟨/ActionSet⟩	⟨ObjectSet⟩ obj1 obj2 obj3 ⟨/ObjectSet⟩
⟨execState⟩ 0 ⟨/execState⟩	⟨Places⟩ ... ⟨/Places⟩
⟨Conditions⟩ ... ⟨/Conditions⟩	⟨Relationship⟩ ... ⟨/Relationship⟩
⟨ActiveView-event⟩ fview	
⟨Action⟩ act2 ⟨/Action⟩	
⟨Objects⟩ obj1 obj3 ⟨/Objects⟩	
⟨Conditions⟩ ... ⟨/Conditions⟩	

Tab. 4.3: *Event frame example for a generic task event frame.*

from the world to furnish one's thinking and acting. To determine what would be the agent's active view, its focus on executing attention, we propose to start from the two event frames, representing the task and world knowledge, and build from them a single frame of what constitutes the relevant aspects of the current event of the world, focusing on the knowledge for task execution. This event frame, called here an active view event frame, consists of knowledge from objects and relationships in the environment taken from the world event frame according to what the task event frame requires towards a frame of active focus that would drive the agent execution.

Figure 4.4 presents two examples of human ability to discriminate from world knowledge of a scene, an appropriate simpler frame that focus on only the relevant parts to achieve a desired goal. In the leftmost images a chess board is depicted at some stage of a game, which would form a world event frame of the situation of the chess environment. A chess master can recognize from this state, the pieces and patterns important to its objective, which in a chess task is clearly the goal of checkmating your opponent represented in a task event frame. In the right top image the board is reduced to present only relevant knowledge which would lead to the player's next moves, that is, the active view event. With the information from its world and task event, a player recognizes its patterns for action, in this case a suffocation mate, attacking with the bishop at b2 and the knight at d4 [Weteschnik, 2006]. In the right bottom image, the player with whites has a checkmate in three moves starting from the original board state in the world event. The rightmost images show a capture snapshot from a football game just after the snap, forming the world event frame for that situation. A professional quarterback can read the defensive coverage to instantly recognize favourable match-ups helping him to achieve completion of the move, which would represent a task event frame. In the right top image the field is reduced and only the position of the marked players is considered important for the execution of the move, which constitutes the active view event. With the information from its world and task event a player recognizes its patterns for action, in this case the backward position of the defensive players allows for an open space in the middle of the field for the receiver to exploit. In the right bottom image, having a right read

on the defence lead to a successful completion of the pass to an open player for a first down.

Table 4.3 shows an example of the event frames. Generic world event frame and task event frame are described and an instance of the active view event frame derived from the world and task frames is also presented. A world event frame holds a set of object frames instantiated from the environment, and of places or special locations of the environment, like an exit door etc., and frames describing relationships between them. The task event frame bears knowledge of the set of action frames required for achieving the task goals, the state of execution of the task and the conditions for invoking, executing, and ending execution of actions. The active view event frame is focused towards the knowledge necessary for instantiating execution, the frame only has the action and object frames that relate to the execution of the current activity the agent is engaged with.

4.7 Structure of the Robot Skills Knowledge Base

The final aim is to populate a knowledge base of the robot available skills for reproduction. The knowledge base would need to hold all necessary information for reproduction of the skills. A robot task would be considered to be of the form ⟨ *robot pick blue ball* ⟩, ⟨ *robot place cup on plate* ⟩, etc. in which an action is described requesting an operation upon an object for a goal oriented task. Therefore, a direct link between objects and skill actions can be intuitively established.

The first attempt at building a knowledge base of robot skills consisted on the pairing of objects and actions. The elements in the knowledge database were represented in two principal directions of objects and skill actions [Hernández et al., 2009]. The task contemplated in [Kheddar et al., 2009b] required for a robot performing actions over an object that is found in the environment. A knowledge base of the robot skill was proposed, where the models of the skill would reside and a humanoid robot could access the necessary learning to perform different motor skills. By linking actions to manipulatable objects in the representation of the skills knowledge, robot systems would be capable of generalizing learned motions of manipulation.

In [Hernández et al., 2009] an object was represented by any necessary information for the recognition and identification of that object, and any constraint related to it, such as, *Tag, Color, Size, Shape*, etc. Similarly an action would correspond to the necessary information from the model of the skill to reproduce said action. The elements in the database were, in an analogy to object-oriented programming, instances of a class object, defined by its characteristic attributes and available skill actions. The knowledge database contained a series of known objects that the robot could identify in its environment. Linked to any instance of an object there were one, several or no skill models' operations associated with it.

$$\texttt{Object1}[attributes] \Longrightarrow \texttt{Action} \begin{cases} SkillModelA \\ SkillModelB \end{cases}$$

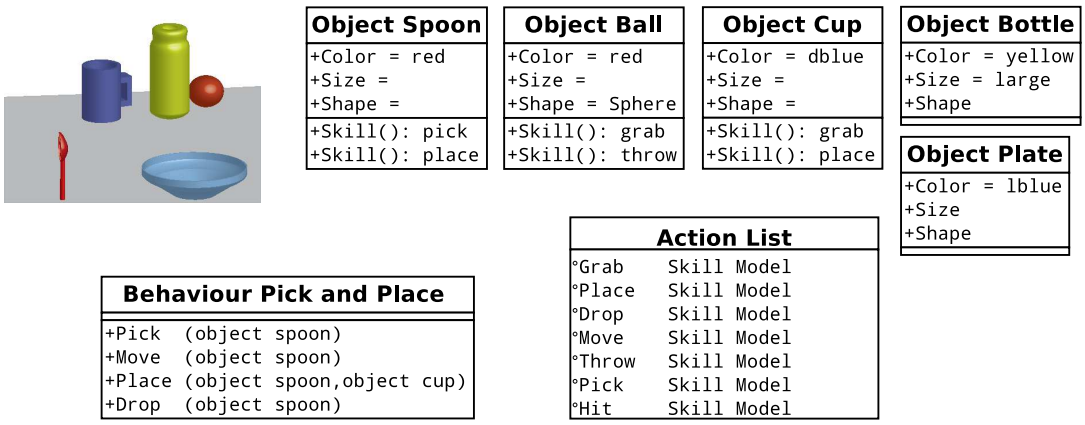


Fig. 4.5: *Instance of the Object-Action Skill Knowledge Database representing objects and actions in the scene filled with various instances and behaviours.*

It was also possible to build sequences of actions to be performed, therefore, expanding the functionalities of the Skills Database implementing "behavioural" instances, Figure 4.5.

$$\texttt{Skill1} \mapsto \texttt{Skill2} \mapsto \texttt{Skill3} \vdash \texttt{Behaviour}$$

A behaviour, consisted of a sequence of actions, with their associated objects, that need to be executed to achieve a goal.

$$\texttt{Behaviour} \begin{cases} \texttt{SkillA[object]} \\ \texttt{SkillB[object]} \end{cases}$$

Thinking in terms of objects and actions is not only intuitive but also convenient for a representational undertaking in robotics. Object and actions are at the basis of robot performance, and manipulating and reasoning with them is important for robots, as can be seen from the efforts in object action complexes [Krüger et al., 2009]. However, representing the manipulation task as pure action sequences is not flexible and also not scalable [Zoliner et al., 2005b]. Sections 4.4, 4.5 and 4.6 have shown that representational attributions must also include information about the world and situations, events and goals, for effective situated performance.

From our earlier attempts [Hernández et al., 2009], it was clear that information of objects and actions alone was not sufficient to capture the entire state of the world. Since for a single task or behaviour there could be more than one pairing $\langle$ *object, skill model* $\rangle$ the addition of at least one more dimension could be required in order to prevent ambiguities. See Figure 4.6. The objects and actions frames don't provide sufficient and complete information for a robot situated in its environment to be able to perform its task adequately. To resolve this problem, as has been shown from section 4.6, considering two more representational directives is suggested: one for the task goal, and one for the configuration of the current state of the world, mainly objects position and relations with themselves, the robot and a human operator.

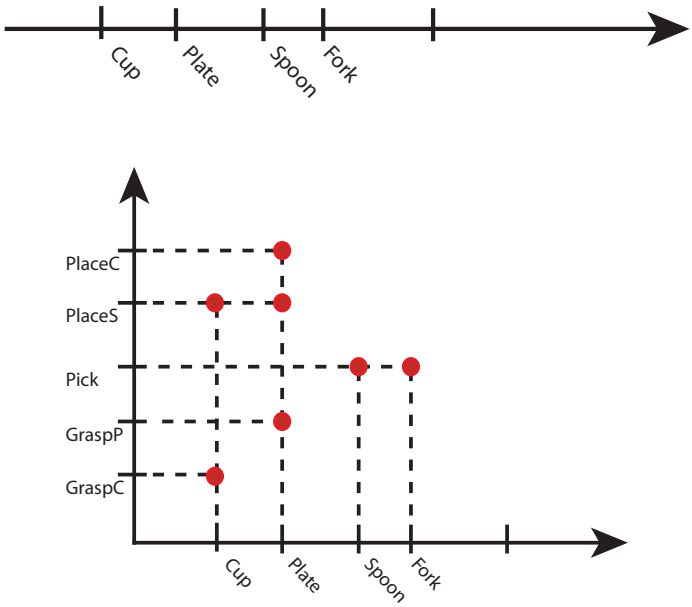


Fig. 4.6: *Representations of the robot skills in the knowledge base in the object and action directions*

An agent's knowledge must consist of skills, concepts and facts about the world. From what has been discussed in this chapter, world knowledge is thought to consist of objects, actions, and task, situations and events. The representations perform as functional abstractions for the perceived environment encoding the agent knowledge about its objects, actions, and events, into manageable internal structures standing, in for things outside the system.

In addition to the presence of objects and actions representations, as established above, the representational attributes need to take into account the state of the world, grounding the representations to the environment, the task at hand and the current situation or present events. The discussion in Section 4.2 established the importance of grounding representations to the environment and cognition to situated activity. The cognitive processes in the real world occur in particular environments employed to achieve a particular practical end, and must exploit the possibilities of interaction and manipulation in the environment. It is necessary to work with symbols and modes of reasoning related to the perception and action of a particular system [Anderson, 2003]. Embodied agents interacting with the real world must develop predictive models that capture the dynamics of the world in order to achieve its goals [Krüger et al., 2009]. The dynamical system approach works on the hypothesis that through increasingly sophisticated use of internal states to mediate between perception and action, more cognitive behaviours emerge from the dynamics of situated action [Beer, 2000].

The representations carry information about the objects or events being represented. The function of a representation is in the carrying of specific information oriented toward the needs of the given agent. Representations must be highly selective,

related to their eventual purpose, and physically grounded. Different mechanisms could very well be needed for high level and low level representations, such as the representations of high level abstract concepts of object frames, and the low level dynamic motion control of action execution; plus the coordination of their behaviour within the events of the environment. The system representations must include objects, actions, and events configurations as stated in Sections 4.4, 4.5 and 4.6.

Objects are all entities that exist in the world, only real physical perceived entities are being considered but the approach could be extended to take into account abstract and imaginary entities whose existence lies outside the world plane. Actions are all processes, transformations, etc., that can be performed or operated over an object. Here, "actions" refers to robot skills expressed in terms of a dynamical system. Actions must provide real effects on the world yet they could be generalized to include abstract and imaginary actions, like the act of thinking. As "events", one thinks of all situations, states, scenarios and configurations of the world that one can be in and recognize one's self to be in. The state of the world instantiates the world event with all that can be perceived in it; the pairing of the world event and a task event lead to recognition of the relevant features of the world, in term of its task, to instantiate a focused active view frame where thought can take place and actions are invoked.

Section 4.3 presented various approaches aimed at building libraries or databases of learned motion primitives as ways of having comprehensive repertoires of robot skills, allowing a robotic system to deal with a vast range of situations. Most of these approaches, while providing information on how the movement primitives can be learned and generated, generally offered little advice on how the library of skills could be used in the environment to select and adapt the primitives to deal with different conditions.

The knowledge base needs to hold all necessary information for reproduction of the skills in the environment. Knowledge of the task would be distributed among the representation of objects, actions and events of the goal and the state of the world. A task is then represented by the phrase ***"Do an Action (A), To an Object (O), For achieving Goal (G), When State of the World is (W)"***. Therefore, the tuple formed by $\langle$ *Do = Action(A), To = Object(O), For = Goal(G), When = World State(W)*$\rangle$ holds all necessary information for the reproduction of the task. The skill knowledge module representation presented in this chapter, see Figure 4.1, would allow the robot to extract from the received perceptual input knowledge about objects, goals and current state of its working environment. The robotic system would be able to retrieve an appropriate skill action from the knowledge base by finding the answer to the phrase ***"Do Action (?) ... "*** for its current task constraints when being presented with the triple $\langle$*Object, Goal, World State*$\rangle$.

Figure 4.7 shows the representation of the skills in the knowledge database in a three dimensional space defined by the $\langle$*Object, Goal, World State*$\rangle$ triple. Selecting from their intersection an adequate model of the skill for the reproduction of the task.

For example, let us consider, as a general typical task for humanoid robots that operate in a domestic environment together with other human agents, a kitchen setting and the cooperative labours that could arise from it such as the setting up or clearing of a table, the cleaning of dishes, the storing of groceries, etc. Tasks behaviours could

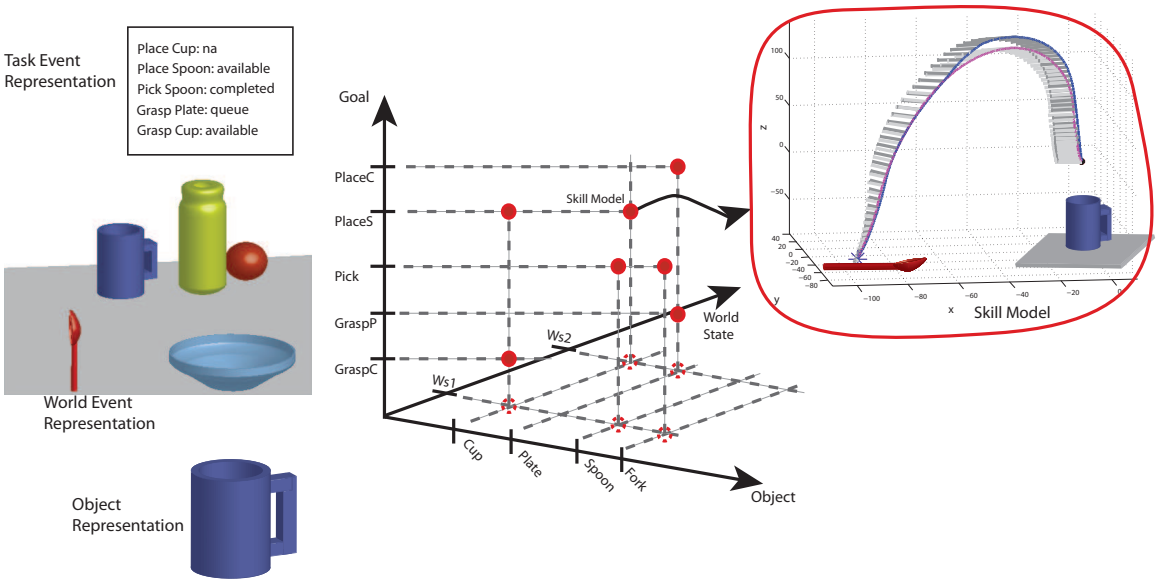


Fig. 4.7: *Representation of the skills in the knowledge base. The intersection of the triple* $\langle Object,\ Goal,\ World\ State \rangle$ *allows to select the adequate model of the skill for reproduction.*

be $\langle$ *robot pick red spoon* $\rangle$ and $\langle$ *robot place spoon in blue cup* $\rangle$ in which we'll have $\langle Do = action, To = Spoon_{red}, For = task\_event, When = world\_event \rangle$, where the world situation and the state of execution of the task will help choose whether the proper action would be $\langle Do = pick \rangle$ or $\langle Do = place \rangle$ in performing the *pick and place* behaviour.

The structure of knowledge can be of various kinds, such as programming by example, Hebbian neural network, probabilistic look-up table, behavioural cloning, etc. [Nehaniv and Dautenhahn, 2001]. In [Geib et al., 2006] object action complexes are defined as instantiated state transition fragments of a situated pairing of an object and an action generalized by the tuple $\langle s_i, mp_j, Obj_k, s_{i+1} \rangle$ comprised of two abstracted states $\langle s_i$ and $s_{i+1} \rangle$ a set of motor programs mp_j and an object class Obj_k. Here, a somewhat similar organization, in terms of the elements that construct the tuple, is used. A structure built on frames, as proposed by [Minsky, 1975], has been adopted.

Frames are a computational device for organizing stored representations in memory, and for organizing the processes of retrieval and inference which manipulate these stored representations [Hayes, 1979]. The theory of frames is an effort to move away from attempts trying to represent knowledge as collections of separate simple fragments [Minsky, 1975]. Frames' data structures are used to represent recognizable situations. The frame approach has been extraordinarily influential, with wide applications in relationship recognition, data monitoring, and propagation and enforcement of constraints [Brachman and Levesque, 2004]. The hold idea of frames is based on a theory for structuring chunks of memory grouped into packets of related facts, which can contain other packets, where any number of packets can be activated or made

available for access at once. The invocation of few appropriate packets creates an execution environment tailored to contain only the relevant portion of the system's global knowledge [Minsky, 1975]. The major force is not at a representational level, but rather at the level of implementation, the frames theory works as a suggestion about how to organize large memories, mainly in a non-clausal form [Hayes, 1979]

In order for a process to use a representation, the process must be coordinated with the format of the representation, only states appropriate to the process will count as representations [Bechtel, 1998]. The process for using the representations begins by instantiating the appropriate frames. Once a frame is proposed to represent a situation, a matching process tries to assign values to each frame's terminals [Minsky, 1975]. When fillers for all the slots of a particular frame are discovered then it means one has found a frame of such class. The data structure of a frame is made up of slots filled with attributes, which can be made of other frames, as organized in terms of a class hierarchy, analogous to an object-oriented programming paradigm. When instantiating a frame its slots will be filled with the values present in the system, any slots with unavailable information will be filled by default attributes associated with the class categories. Default values are assumptions reasonably made when the state of knowledge holds no information to the contrary. Default assumptions involve an implicit reference to the whole state of knowledge at the time the assumption was generated, any event which alters the state of knowledge is liable therefore to upset these assumptions [Hayes, 1979]. Reliance on default values in [Minsky, 1975] is based upon the realization that thinking begins with defective networks that are slowly, if ever, refined and updated.

Figure 4.8 presents the control data flow for the process of using the representations in the knowledge base for extracting the task constraints and the appropriate *Robot Skills Models* within the knowledge module presented in Figure 4.1 and the framework of Figure 2.6. The knowledge of the environment and goals is represented in terms of the World Event Frame and Task Event Frame, with Object and Action Frames representing knowledge about available objects and actions in the knowledge base respectively. From the knowledge of these frames an Active View Event Frame is built of the focused knowledge promoting the agent's execution. Looking up the knowledge base for the given object and action affordance frames yields the needed models of the skill, $\bar{\mathcal{M}}_{RS}$, for building the task model.

To serve adequately the demands of a constantly changing environment, it is necessary not only to pick items out of their general setting, but to know what parts of them may flow and alter without disturbing their general significance and functions [Bartlett and Bartlett, 1995]. The process for using the representations begins with the reception of perceptual input. From a given scene the system instantiates frames, generally governed by the precedence of visual evidence. From the perceived given input the first step for extracting a task constraint is the matching of the world to an instance of the World Event Frame and the instantiation of the Task Event. The matching process which decides the suitability of a proposed frame is partly controlled by knowledge of the system's current goals and partly by information attached to the frame [Minsky, 1975]. From information collected in the World and Task event frames, which in turn are made up of other object and action frames, the system

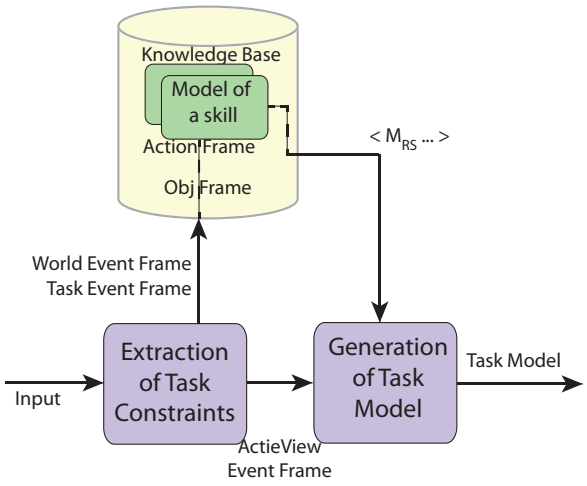


Fig. 4.8: *Knowledge base control flow for using the knowledge representations. The World Event Frame and Task Event Frame are instantiated, and an Active View Event Frame is built from them with the constraints of the task. From object and action affordance frames in the knowledge base the needed models of the skill are taken for building the task model.*

would have information about its current goals and the situation of the environment; yet this is not enough to ground the representations in order to effectively use them for supporting its performance. As has been discussed previously, the representations must be selective, physically grounded and leading to situated activity. Extending the notion of selective representations leads to closing the gap between perception and action. The perception field is always already an action field, the perceived world is always known in terms directly related to an agent's current possibilities for future action [Anderson, 2003]. Out of the perceived knowledge of the world, which we collect in the World Event Frame, the required models for action must be invoked for the system's operation. [Minsky, 1975] imagined that thinking and understanding, be it perceptual or problem-solving, was concerned with finding and instantiating a frame, breaking large problems down into many smaller jobs to be done. Maintaining a full model of the world is a large problem and one that contributes to the failure of planning approaches dealing with changing environments, but a problem with which it is not necessary to deal with, in such complex environments one can never cope with many details at once. At each moment, one must work within a reasonably simple framework. [Minsky, 1975] contend that any problem that a person can solve at all, is worked out at each moment in a small context and that the key operations in problem solving are concerned with finding or constructing these working environments.

Therefore motivation for creating an Active View Event Frame is clear from the need to focus attention and discriminate from the information of the world and task events and the important features of the current situation toward the current action. To construct this focus view that would drive what is taken from the world to furnish

one's thinking and acting, we build a single frame of what constitutes the relevant aspects of the current event of the world, focusing on knowledge for task execution. The Active View Event Frame, consists of knowledge from objects and relationships in the environment taken from the world event frame according to what the task event frame requires to drive the agent execution, and constitutes the system's output for the current world and task constraints. The frames representation is envisioned as packets of data and processes, and so are the high level goals [Minsky, 1975]. When a frame is proposed, its packet is added to the current knowledge so that its processes have direct access to what they need to know, without being choked by access to the entire knowledge of the whole system. One must choose from one's collection of clustering methods by using the goals in a micro world context [Minsky, 1975].

For an agent working in an unstructured environment, the focus of its perception must be directed towards its executing action. Knowledge of its environment and task would be collected into their appropriated frames and a focused active view frame would be built, taken from their global knowledge and breaking it down into a simpler framework from which computations and knowledge take place.

Revisiting the kitchen setting, and the task $\langle$ *robot place spoon in cup* $\rangle$. A prototypical scene is given in which the objects relevant to the task are recognizable together with clusters of other currently unimportant objects. The frames of objects and actions' knowledge are instantiated along with an event frame for the configuration of the environment and an event frame invoked with the knowledge of the task from the desired given task behaviour. Knowledge from the event frames is reduced into a simplified active frame ignoring information not pertaining to the execution of the current task.

Figure 4.9 presents the organization of the knowledge base in terms of the frames described in Sections 4.4, 4.5 and 4.6. To help better understand these points, we review an example, as before, considering a simple case in which a humanoid robot is requested to place a spoon inside a cup, and place the cup on top of a saucer plate on top of a table, as if it serving a cup of tea or coffee. The robot would begin is operation in a kitchen setting scene in front of a table with various identifiable objects typical of the tasks which would be collected into the World Event Frame and the necessary instances of Object Frames for the objects present in the scene. Naturally, in our example we would have spoon, cup and plate objects, as the robot explores its environment it will recognize any object as it finds them, relevant to its task or not, and will fill the World Event Frame with its respective Object Frames instances. Additionally the system is provided with the Task Event Frame representing the knowledge of the task and the instances of the Action Frames. In this example to complete the requested task, the robot would be required to perform several simpler tasks or subtasks, such as picking up the spoon, grasping the cup and placing the cup on top of the saucer plate, etc. The Task Event Frame holds knowledge of the state of execution of the task and Action Frames instances for the knowledge of the robot skills for reproduction.

The Task Event Frame and the World Event Frame represent the knowledge of the state of execution of the task and the environment, with Object and Action Frames representing the available objects and actions. The Active View Event is created from

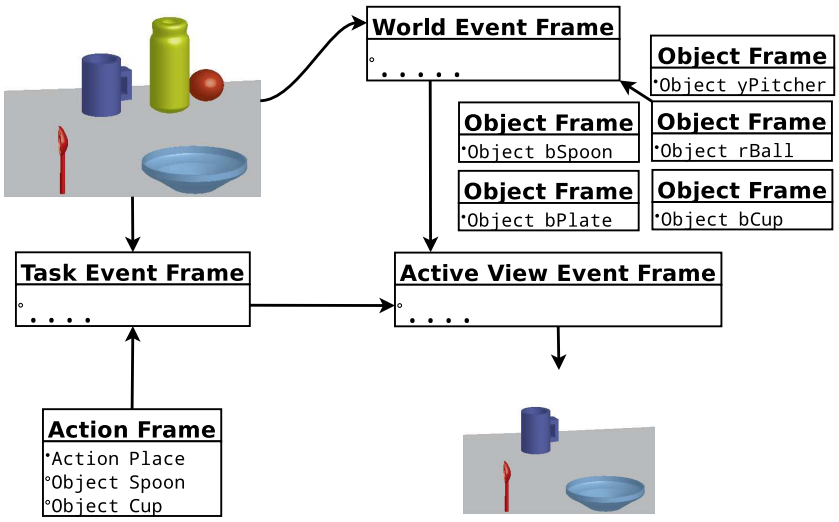


Fig. 4.9: *Knowledge base structure and organization of the knowledge representations. World Event Frame and Task Event Frames represent the knowledge of the state of the environment, with Object and Action Frames representing the available objects and actions. From the knowledge of this frames an Active View Event Frame is built of the focused knowledge required to drive the agent execution, and the system's output for the current world and task constraints.*

all these knowledge frames reducing the view of the world into a simpler scene with the important Object Frames for the task, with their proper roles assigned for the execution of the action. The full scene in Figure 4.9 is broken down into the reduced scene of the Active View Event Frame which promotes the execution of the action as in the Figure 4.7. Now let us return to our example at some point during the task execution. Imagine for instance the execution of the ⟨ *robot place spoon inside cup* ⟩ tasks. To perform the task it is assumed that the spoon object has already been picked by the robot and is in one of its hands, so the target object for the task is the cup. The World Event Frame and the Task Event Frame are filled as discussed above and as in Figure 4.9. The Active View Frame is built form these knowledge frames to create a focused, simplified frame with only the knowledge pertaining to the action being performed. In our case this means a reduced frame where only the relevant Object instances are included, the spoon, the cup and the plate, and only the executing Action frame instance is included to extract the necessary *Robot Skill Model* for the action execution.

Different approaches on related topics focused on the management of knowledge by robotic system exists, such as KnowRob, [Tenorth and Beetz, 2013] or RoboEarth, [Waibel et al., 2011]. However these systems lie at a higher more abstract level of the cognitive hierarchy while our framework lies at a lower level of action execution. Further research requires study and comparison of other systems, in particular the ones that may be used to complement the framework developed in this work.

4.8 Summary of the Chapter

Throughout this chapter the development of a knowledge base for the storing and retrieval of the learned models of the skills has been described. In section 4.2 an introduction to the topic of knowledge was presented, and its importance for development of cognitive robotics. The embodied view of cognition and its challenges to the traditional approaches of symbolic representations were studied. Also, basic notions and concepts in the field of knowledge representation and reasoning were reviewed. Section 4.3, presented a review of similar approaches aimed at building repertoires of basic units of action, also known as movement primitives, which can represent a basic set of elementary robot motor skills. Learned motion primitives can be used as ways of having comprehensive repertoires of robot skills. Most of these approaches generally offered little advice on how the library of skills could be used in the environment to select and adapt the primitives to deal with different conditions or their mechanism for representing their knowledge. In sections 4.4, 4.5 and 4.6 the approaches and problems for building representation of objects, actions, and events knowledge were presented, respectively. Finally, section 4.7, presented the development of a knowledge base for the storing and retrieval of the learned models of the skills, and the representational structure of the robot skills' knowledge base developed in this chapter. The embodied view of cognition calls for representations to be limited, physically grounded to the environment and oriented towards a particular use. The principal aim for the humanoid robot is to take actions, as situated agents, that are appropriate to its circumstances. Fitting representations are essential for this goal. Our representational framework focuses on a lower level of abstract representation aiming at action execution, while most other systems lie at a higher more abstract level of the cognitive hierarchy, however this could allow both approaches to complement each other. In this chapter, the developed representations for our robot were presented. Object and actions are at the basis of robot performance, therefore thinking in terms of objects and actions was not only intuitive but also convenient for a representational undertaking in robotics. However, representational attributions must also include information about the world and situations, events and goals, for effective situated performance. A structure built on frames has been adopted in this work; the frames are a computational device for organizing stored representations in memory, and for organizing the processes of retrieval and inference which manipulate these stored representations. In our system the knowledge of the environment and goals is represented in terms of World Event Frame and Task Event Frames, with Object and Action Frames representing knowledge about available objects and actions respectively. From the knowledge of these frames, an Active View Event Frame is built of the focused knowledge promoting the agent's execution. Figure 4.9 presents the organization of the knowledge base in terms of the World Event, Task and Active Event Frames, Object and Action Frames as described in Sections 4.4, 4.5 and 4.6.

5. GENERATION AND ADAPTATION OF ROBOT SKILLS

5.1 Outline of the Chapter

This chapter presents the algorithms developed for the generation and adaptation of robot skills. Humanoid robots are required to perform a wide repertoire of tasks working beside humans in complex dynamic environments. While *Learning from Demonstration (LfD)* approaches provide adequate methods used for learning and encoding the models of the robot skills for every conceivable scenario the robot may encounter would be a daunting undertaking, therefore, mechanisms for the generation and adaptation of new robot skills from previously learned skill models are needed. Figure 5.1 shows the framework proposed throughout this work for the adaptation of learned skills to task constraints, highlighting the generation of task models discussed in this chapter. This chapter describes the process by which, using the already learned model of a robot skill and the extracted constraints knowledge of the current task, the model of a skill is adapted to reproduce a new task. Different modes are presented for the adaptation, update, merger, combination and transition between the *Robot Skills Models*. The organization of this chapter goes as follows:

- Section 5.2, presents developments for the generation and adaptation of the robot skills. A review of related approaches aiming at the adaptation of learned skill models is given. Also, the framework employed through this work to adapt and generate the task models is presented.

- Section 5.3, presents the adaptation of a task model by operations on its inherent dynamical properties. The *Robot Skills Models* are learned in a *Dynamical Systems (DS)* approach. *DS* are intrinsically robust to spatio-temporal perturbations, do not explicitly depend on time and can be generalized to unseen initial conditions.

- Section 5.4, presents the adaptation of a task model by updating a robot skill. Models of a skill must be updatable, when given new information for the representation of a skill, the system must allow for the models to be improved. This section describes methods by which *Robot Skills Models* can be updated.

- Section 5.5, presents the generation of a task model by merging robot skills. Skills can be generated by merging two or more models into a new skill, multiple

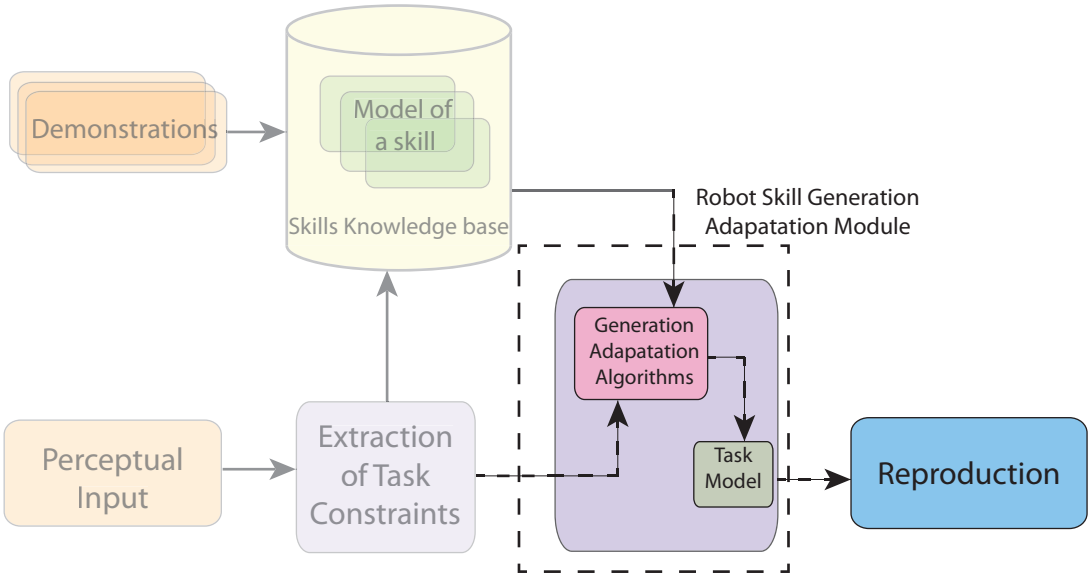


Fig. 5.1: *Module for the generation of task models from the robot skills in the knowledge base and the constraints of the task, highlighted over the proposed cognitive framework for learning and adaptation of robot skills.*

desired robot skills may be composed from superposition of various models. This section describes methods by which *Robot Skills Models* can be merged.

- Section 5.6, presents the generation of a task model by combining robot skills. Models of a skill can be combined to generate new models that encompass a larger spectrum of the attractor dynamics. This section describes methods for the combination of *Robot Skills Models*.

- Section 5.7, presents the generation of a task model by transitioning between robot skills. To generate complex behaviours, the system must sequence and transition between models of the robot skills. This section describes methods by which the system can shift smoothly among the reproduction of different *Robot Skills Models*.

5.2 Generation and Adaptation of Robot Skills

Humanoid robots are thought to collaborate and interact together with humans, sharing the same space, tools and activities. For humanoid robots to act fluently in unstructured environments, interacting with different objects and people, they must be able to perform dynamically changing tasks that require great adaptations. Flexible and generic control methods that can adapt to various tasks and robots constraints are necessary. Learning systems are required to acquire skills and mechanisms are

needed to endow systems with the capacities to adapt their acquired skills, expanding the system's knowledge and ability to act in the environment.

Traditionally available control algorithms are not nearly versatile, robust or flexible enough to achieve the complexity levels of the biological systems which are to be emulated [Peters et al., 2003]. In order to benefit from the full potential of humanoid robots a learning approach is required [Schaal, 1999].

One of the aims of this work is learning models of robot skills which are then used to build a knowledge base of the robot skills knowledge for a humanoid robot reproduction. To teach and learn the robot skills a *LfD* framework was implemented in Chapter 3. The motivations for adopting an *Imitation Learning* approach were stated in the previous chapters, the most important characteristics are that it provides intuitive and user-friendly methods to teach tasks to a robot by demonstrating the skills without requiring the user to have expert programming skills, it reduces the cost of developing automated planning and manual programming of robot control, and speeding up the learning process, as opposed to reinforcement learning methods, reducing complexity of search spaces, giving prior knowledge of task performance.

The focus of the *LfD* approaches is the development of algorithms that are generic in their representation of the skills and in the way they are generated. *LfD* methods allow a human user to teach a robot how to accomplish a given task simply by demonstrating the task and generalizing the demonstrated movements across a set of demonstrations [Gribovskaya et al., 2010]. The robot cannot simply reproduce a skill by copying an observed behaviour, it must have the capability to generalize it. One common approach for generalizing a skill consists of creating a model of the skill based on several demonstrations, performed in slightly different conditions exploiting the variability inherent to the various demonstrations [Calinon, 2009]. *Imitation Learning* focuses on three important issues: efficient motor learning; the connection between action and perception; and modular motor control in the form of movement primitives [Schaal, 1999].

To achieve the complex behaviours, such as those needed for a humanoid robot to work alongside humans, it would be necessary to have inclusive and comprehensive repertoires of robot skills. For these purposes movement primitives, basic units of action to complete a goal, are promoted. The assumption that complex movement skills are composed from smaller units of action is well accepted for these approaches. The insight that human activity is decomposed into building blocks of smaller elementary actions is an established belief which can help to cope with the complexity of motor skills learning for robots. There are many theories about motor primitives suggesting human motion be divided into its elementary trajectories [Fod et al., 2000].

To learn such basic units of actions is considered a useful approach for generating libraries of motor skills. Endowing a robotic system with a library of movement primitives filled with a sufficient number of skills can provide it with an adequate repertoire of actions to deal with a vast range of situations.The motor controller components of movement primitives could be manually derived or learned. It is important to allow their generalization and applicability to different scenarios that the primitives be characterized in parametric form and be provided with adequate representations. Chapter 4 reviewed approaches to deal with complex motions a

library of movement primitives and presented the development of the knowledge base for the storing and retrieval of the *Robot Skills Models* learned in Chapter 3 as basic primitives of action.

Therefore, it is necessary to extend the classical *LfD* approach of learning a skill model in a way that allows the adaptation of a robot previously learned motion skills to new unseen contexts.

The *Learning from Demonstration (LfD)* approaches previously reviewed offered natural, fast and implicit means of teaching a robot new skills. However, despite its clear advantages, it would be impractical for the human operator to teach the robot the skills for every needed task and for every foreseen situation, since the number of demonstrations the human must provide to the robot to generate a new model of a skill could turn it into a tiresome and time-consuming process and it wouldn't be possible to cover every necessary task and every unforeseen situation. For this reason, enhancing the *LfD* with the capacity to adapt and generate new skill models is important. Additionally, despite the fact that the *LfD* offers the capability to generalize the learned model, the generalization is relatively limited to changes in initial conditions or to rather small perturbations during the execution. To expand the scope of a learned model to areas unexplored by the demonstration would require different mechanism. Hence, to extend the classical *LfD* approach of learning a skill model in a way that allows the adaptation of a robot previously learned motion skills to new unseen contexts is necessary.

To reproduce a task adapted for an unseen context the robot is provided with knowledge of the state of the environment and the constraints of the task extracted from its perceptual input and other high level orders it could possess. Using both, the already learned model of a skill, and the extracted constraints information of the current task, the model of the skill is adapted to reproduce the task. Figure 5.2, illustrates the process for enhancing a classical *LfD* approach to generalize a skill to allow adapting a robot's previously learned skills models. The traditional approach in *LfD* from [Billard et al., 2008] as represented by the top scheme in Figure 5.2 presents a somewhat static control scheme, akin to an open loop controller, and it won't be sufficient to reproduce a task when the state of the environment is too dissimilar from how it was when the demonstrations were given to encode the model of the skill. By adding environmental and task knowledge as input to the scheme, in the bottom of Figure 5.2, the control diagram could be thought of as a close loop controller, with a feedback signal from the constraints of the task and the environment, which allows the model of a skill to be adapted accordingly to its context.

Reproduction of robot skills, if they are to be general enough, needs to present the capacity for adaptation and to generate new skills when the current situation of the world and its constraints of the task demand them. Working in dynamically changing environments, it is necessary to adjust the desired trajectories appropriately, or to generate new ones by generalizing from previously learned knowledge [Schaal et al., 2007]. The robot skills learned with the methodology described in Chapter 3 would present stable trajectories that accurately reproduce the demonstrated motion dynamics, however, there is no guarantee that outside the area of the demonstrations the reproduction of these trajectories would provide a meaningful or

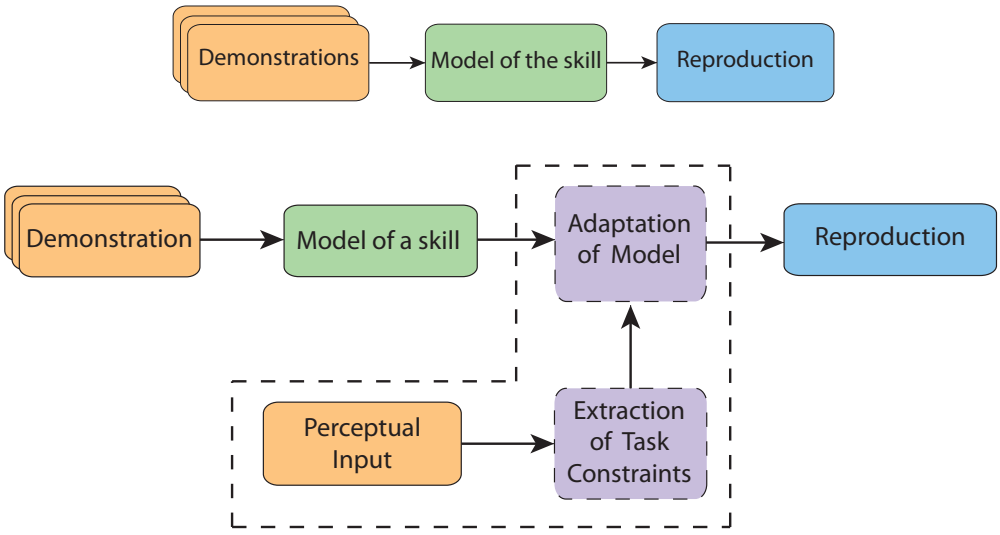


Fig. 5.2: *Augmenting the LfD approach for the generalization of a skill to allow adapting a robot previously learn skills models. (top) Generalization of a skill by extracting the statistical model across multiple observations. (bottom) Adaptation of a learn skill to new context by extracting the task constraints with a new observation and using the environment information to modified previously learned models of the skill.*

proper behaviour in accordance to what would be expected from the task. As an example, consider a case in which the robot has been taught motion skills in order to grasp a cup approaching from the left side; and later it's requested to grasp a cup positioned to its right, it would be the impulse of the robot, governed by its model of the skill, to approach the object from the left as the demonstrations showed it. However, such behaviour would not only be unnatural to achieve the task but potentially unsafe for the robot or other entities in the environment.

The *Robot Skills Models* were learned by employing a *Dynamical Systems (DS)* approach. The *DS* framework allows to comply with the attractor dynamics of the desired behaviour, modulating it with a set of non-linear dynamic systems that form an autonomous control policy for motor control. A *DS* approach was chosen because it allowed certain desirable properties. *DS* are intrinsically robust in the face of spatio-temporal perturbations. *DS* do not explicitly depend on time and are able to model arbitrary non-linear dynamics. *DS* can also be easily modulated to generate new trajectories that have similar dynamics. By learning the skills under a probabilistic approach employing *Gaussian Mixture Model (GMM)* the parameters governing the attractor dynamics of the motion are fully encoded into the parameters defining the Gaussian functions. The learned *Robot Skills Models* would form a set of basic primitives of action from which a knowledge base of skills was built in Chapter 4.

An approach based on movement primitives relies on possessing available sequences of motor commands, executed in a certain order, to accomplish a given

motor task. Movement primitives are biological structures that organize the underlying mechanism of complete movements [Fod et al., 2000]. It is generally believed that humans employ basic motor primitives as an underlying mechanism of biological motor control. Evidence exists from human and animal experiments supporting the believe that sets of motor primitives are used to build a basis for voluntary motor control [Konczak, 2005].

By working under a theory, based on the existence of basic primitives, from which full human motions are made, it seems clear that following the issue of how to create, build and learn these basic units of action primitives, the next question would be focused on how these primitives can be manipulated; how primitives can be combined to form higher level movement primitives; how sequencing and recognition of sequences of movement primitives can be accomplished [Schaal, 1999]. The idea is that actions can be decomposed into sequences of reusable primitives; primitives might be preserved in memory and adequate primitives might be retrieved from it. Humans can generate diverse actions by combining behaviour primitives [Arie et al., 2012].

To generate complex human like motions from a learned set of basic primitives units, the *Robot Skills Models*, and be able to reproduce various complex task behaviours, methods for operating and manipulating upon the primitives must be developed. The robot skills must be adaptable to conditions of its operating environment even when differing substantially from its original demonstrations. Also, the models of a robot skill must be updatable, when given new information for the representation of a skill the system must allow for the models to be improved. Additionally, the action primitives approach must be able to generate new skills by merging two or more primitives into a new skill, multiple desired robot skills may be composed from superposition of various primitives. Another important property is the combination of the *Robot Skills Models* to generate new models that encompass a larger spectrum of the attractor dynamics. A final desirable operation over the basic set of primitives skills consist of sequencing and transition between models of robot skills in order to generate complex behaviour with smooth transformation among the reproduction of different skill motions.

The bulk of the work in *LfD* or *RPbD* and movement primitives approaches has been centred on the development and validation of algorithms that would allow the learning and encoding of the skill motions, which would constitute the movement primitives, to take place. Little work has been focused on the development of techniques that would endow the system with the ability to operate upon its movement primitives and generate new and more complex behaviours. Yet some examples of these efforts can be found.

[Muelling et al., 2013] presented a framework to learn cooperative skills from interaction with a human. First, a set of elementary movements are learned from a human teacher by kinaesthetic teaching. Subsequently, the system generalizes these movements to a wider range of situations using our mixture of motor primitives approach. The resulting policy enables the robot to select appropriate motor primitives as well as to generalize between them.

The work of [Shukla and Billard, 2012] focused on combining several learned *DS*, with distinct attractors, resulting in a multi-stable *DS*. Their work presented an

Augmented-SVM model, which inherits region partitioning ability of well know *Support Vector Machine (SVM)* classifiers and is augmented with novel constraints derived from the individual *DS*.

In [Khansari-Zadeh and Billard, 2012] a novel approach is presented to real-time obstacle avoidance based on *DS*, that ensures impenetrability of multiple convex shaped objects. Obstacle avoidance proceeds by modulating the original dynamics. The modulation is parametrizable and allows to determine a safety margin and to increase the robot's reactiveness in the face of uncertainty in the localization of the obstacle.

[Kulvicius et al., 2012] focused on an approach for joining movement sequences by modifying the learned DMP exemplified on handwritten application. The method is based on the modification of the original DMP formulation. The new method can reproduce the target trajectory with high accuracy regarding both the position and the velocity profile and produces smooth and natural transitions in position space, as well as in velocity space.

In [Gomez et al., 2012a] a novel robotic learning technique based on Fast Marching Square is presented. The method assumes that the task taught to the robot can be codified into a path planning problem. Their method takes into account the environment, since it modifies the path planning algorithms of the system instead of modifying the motion control.

The work of [Palm and Iliev, 2010] records the operator's motions by a data-capturing system; they are then modelled via fuzzy clustering and a Takagi-Sugeno modelling technique. The resulting skill models use time as input and the operator's actions as outputs. The robot executes the recognized skill by using the corresponding reference skill model. Drastic differences between learned and real world conditions which occur during the execution of skills by the robot are eliminated by using the Broyden update formula for Jacobians. This method was extended for fuzzy models especially for time cluster models.

[Schaal, 1999] discussed a set of primitives for generating control commands for any given motion by modifying trajectories appropriately, or generating entirely new trajectories from previously learned knowledge.

[Calinon et al., 2012] derive a task-parametrized model that can adapt motion and impedance behaviours in real-time with respect to the current position/orientation of frames. The proposed extension is built upon the product properties of Gaussian functions.

[Tani and Ito, 2003] investigated the self-organization of behavioural primitives in a neural network model in the context of robot imitation learning. The model is characterized by the parametric biases which adaptively modulate for embedding different behaviour patterns in a single recurrent neural net, in a distributed way. Diverse behaviour patterns other than learned patterns were generated because of self-organization of non-linear map between the parametric biases and behaviour patterns.

[Arie et al., 2012] describes a model dealing imitation learning generalization by focusing on the problem of action compositionality. A robot was trained with a set of different actions concerning object manipulations which can be decomposed into sequences of action primitives. Then the robot was asked to imitate a novel

compositional action, composed of prior-learned action primitives. The results showed that the novel action can be successfully imitated by decomposing and composing it with the primitives by means of organizing unified intentional representation hosted by mirror neurons, even though the trajectory level appearance show difference between those observed and those self-generated.

The robot would receive from the different modules of perception and interaction the required appropriate commands ordering the reproduction of a skill, and would extract the constraints of the task and its environmental configuration to instantiate the appropriate knowledge frames as described in Chapter 4. With this information taken from the knowledge base, together with the pertinent *Robot Skills Models* corresponding to the requested task, the module for the generation and adaptation of robot skills is called to adapt the robot skills accordingly and to generate the task models for the robot reproduction of the task, Figure 5.1. For the operation of the module, two distinct processes are required. A first step calls for a skill model to be adapted, if necessary, to comply with the conditions of the task in which it will be reproduced or to be updated with new information. A second step requires the generation of a task model, allowing the reproduction of the encoded knowledge of the skills by the robot in order to perform a requested task. The versatility and usability of a robot skill approach depend on the capacity to manipulate the skills. These manipulations of the skills must allow for the adaptation, update, merger, combination, and transition between the *Robot Skills Models* as necessary.

Methods for model combination or joining can be found in the field of machine learning and pattern recognition. Performance improvement can be obtained by combining multiple models together in some way, instead of just using a single model in isolation [Bishop, 2006]. One method involves the learning of different models and then using the average of the predictions made by each model. An alternative form of model combination is choosing one of the models to make the prediction as a function of the input variables, in this way different models become responsible for making predictions in different regions of input space [Bishop, 2006]. These methods are very dependent on the decision process. A way of softening the weights in the decision process can be done by moving to a probabilistic framework for combining models. These methods are known as mixtures of experts; models can be viewed as mixture distributions conditioned by the input variables. A mixture of experts can be given:

$$p(t|x) = \sum_{k=1}^{\mathbf{K}} \pi_k(x) p_k(t|x) \tag{5.1}$$

In which the mixing coefficients $\pi_k(x)$ are known as gating functions and the individual component densities $p_k(t|x)$ are called experts. The idea behind this is that different components can model the distribution in different regions of input space and the gating functions determine which components are dominant in which region. The efforts in *LfD* approaches and the theory of generating movement primitive robotic skills can only have a real implementation value for developing humanoid robotic systems if the models of the skill can be operated upon to generate new behaviours of increasing levels of complexity.

5.3 Operations with Robot Skills

A knowledge base of robot skills was developed in the previous chapter. The *Robot Skills Models* in this work has been learned by employing a *Dynamical Systems (DS)* approach, built as basic primitives of movements encoding within the model the motion dynamics of a demonstrated skill. Autonomous dynamical systems were proposed as an approach for representing movements as mixtures of non-linear differential equations with well-defined attractor dynamics [Ijspeert et al., 2001]. The *DS* framework allows it to comply with the attractor dynamics of the desired behaviour, modulating it with a set of non-linear dynamic systems that form an autonomous control policy for motor control.

The *DS* framework provides a effective means to encode trajectories through time-independent functions that define the temporal evolution of the motions. The motion dynamics are estimated through a set of first order non-linear dynamical system equations. It is assumed that the motion is governed by a first order autonomous ordinary differential equation, $\dot{\xi} = f(\xi)$, as in Eq. 3.3.

A *DS* approach to skill learning can offer a fast, simple and powerful formulation for representing and generating movement plans learned from demonstration. The *DS* framework allows to comply with the attractor dynamics of the desired behaviour, modulating it with a set of non-linear dynamic systems that form an autonomous control policy for motor control. *DS* provide efficient and clean means for encoding a skill and fulfilling various desirable properties.

The *DS* framework presents three advantages, i) *DS* can be easily modulated to generate new trajectories that have similar dynamics, performing in areas that were not covered during demonstrations, [Khansari-Zadeh and Billard, 2011]; ii) *DS* are intrinsically robust and can adapt their trajectories instantly in the face of spatio-temporal perturbations [Khansari-Zadeh and Billard, 2010a]; iii) *DS* do not explicitly depend on time indexing and provide closed loop control and are able to model arbitrary non-linear dynamics, [Gribovskaya et al., 2010].

The concept of a dynamical system is quite general. *Dynamical Systems* are mathematical objects that unambiguously describe how the state of some system evolves over time [Beer, 2000]. *DS* theory offers a wide variety of tools for visualizing and analysing the temporal behaviour of such systems. There are two types of dynamical systems differential equations and iterated maps or difference equations [Strogatz, 1994]. Differential equations define a vector field, which assigns an instantaneous direction and magnitude of change to each point in the state space. The sequence of states generated by the action of the dynamics is called a solution trajectory. The set of all possible solution trajectories is called the phase portrait.

Of particular interest is the possible long-term behaviour of a dynamical system. Over time, the state of many dynamical systems eventually ends up in a small subset of the state space called a limit set. Two simple types of limits sets are equilibrium points and limit cycles. For stable limit sets or attractors, all nearby trajectories converge on the limit set, so that small perturbations away from the limit set will return there. In contrast, any perturbation from an unstable limit set will not return to that limit set, but will instead be carried elsewhere by the dynamics [Beer, 2000].

The appearance of phase portraits is controlled by the fixed points $\bar{\xi}$ defined by $f(\bar{\xi}) = 0$ representing equilibrium point solutions [Strogatz, 1994]. The qualitative structure of the flow can change as parameters are varied; fixed points can be created or destroyed, or their stabilities changed. These changes in the dynamics are called bifurcations. Bifurcations provide models of transitions and instabilities as some control parameter is varied [Strogatz, 1994].

5.3.1 Stability Concerns

A key matter when adopting a *Dynamical System* approach to modelling the robot skills for creating control policy of the movement primitives is the requirement to ensure the stability of learned *DS* [Khansari-Zadeh and Billard, 2011]. Falling into an unstable behaviour or a divergence from the desired trajectory would be a potentially dangerous occurrence when controlling a robot, more specially a humanoid robot which may be performing together with other humans. Therefore, analysing the behaviour of the system is essential, as is determining whether it is stable. Stable *DS* would benefit from inherent properties crucial to modelling movement primitives robot skills.

The stability analysis of *DS* is usually around its equilibrium points. In this work the notation $\bar{\xi}$ defines an equilibrium point, a point $\bar{\xi}$ is an equilibrium point if $\xi(0) = \bar{\xi}$ initially and then $\xi(t) = \bar{\xi}$ for all time, an equilibrium is defined to be stable if all sufficiently small disturbances away from it damp out in time [Strogatz, 1994]. The equilibrium points can be determined by computing the real roots of Eq 3.3. The stability of a given equilibrium point $\bar{\xi}$ can be defined as follows.

$\xi = \bar{\xi}$ is a locally stable equilibrium point if for each $R > 0$, there is $r = r(R) > 0$ such that if the initial state $\xi(0) - \bar{\xi} < r$, then the evolution of the system in time satisfies $\xi(t) - \bar{\xi} < R$ for all $t > 0$.

$\xi = \bar{\xi}$ is a locally asymptotically stable equilibrium point if it is stable and r can be chosen such that if $\xi(0) - \bar{\xi} < r$, then it implies $\lim_{t\to\infty} \xi(t) = \bar{\xi}$.

$\xi = \bar{\xi}$ is a globally asymptotically stable equilibrium point if the asymptotic stability holds for any initial point, $\lim_{t\to\infty} \xi(t) = \bar{\xi}$, for all $\xi(0) \in Rd$.

Studying the stability of *DS* can generally be divided into linear and non-linear systems. An autonomous linear *DS* has the following general form:

$$\dot{\xi} = \mathbf{A}\xi + \mathbf{b} \tag{5.2}$$

where $\mathbf{A} \in \mathbb{R}^{d\times d}$ and $\mathbf{b} \in \mathbb{R}^d$ are a constant matrix and a vector. The stability of linear dynamics in Eq. 5.2 is global asymptotic at the unique equilibrium point $\bar{\xi} = -\mathbf{A}^{-1}\mathbf{b}$ if and only if the real part of all eigenvalues of the matrix $\mathbf{A}$ are strictly negative. Eq. 5.2 correspond to the linear terms of Eq. 3.24 for the *GMR*.

Stability analysis of linear dynamical systems is well studied, in contrast, there is no unique method to analyse the stability of non-linear *DS*, and theoretical solutions exist only for particular cases [Gribovskaya et al., 2010]. The Lyapunov methods are the most common and general approaches for studying the stability of non-linear *DS*. The stability analysis via the standard Lyapunov Stability theorem requires first

finding a non-negative energy function $V(\xi) \geq 0$, and second verifying if it always decreases in a neighbourhood around the equilibrium point $\bar{\xi}$.

The non-linear *DS* given by Eq. 3.3 is asymptotically stable at $\bar{\xi}$, if a continuous and continuously differentiable Lyapunov function $V(\xi)$ can be found such that

$$\begin{cases} V(\xi) > 0 & \forall \xi \in \Omega \subset \mathbb{R}^d \& \xi \neq \bar{\xi} \\ \dot{V}(\xi) < 0 & \forall \xi \in \Omega \subset \mathbb{R}^d \& \xi \neq \bar{\xi} \\ V(\bar{\xi}) = 0 \& \dot{V}(\bar{\xi}) = 0 \end{cases} \tag{5.3}$$

In this case, the domain Ω is called the stability domain or region of attraction, and should correspond to a level set of $V(\xi)$. If the stability domain is expanded to the whole state-space $\Omega = \mathbb{R}^d$ and $V(\xi) \to \infty$ as $\|\xi\| \to \infty$ then $\bar{\xi}$ is globally asymptotically stable.

[Khansari-Zadeh and Billard, 2011] algorithm *SEDS* was employed to learn the estimates for the *DS*, as described in Chapter 3. *SEDS* computes the optimal values for the parameter θ while ensuring the estimate $\hat{f}$ to be globally stable in $\mathbb{R}^d$ given sufficient stability conditions.

An arbitrary non-linear *DS* given by Eq. 3.24 is globally asymptotically stable at the target $\bar{\xi} \in \mathbb{R}^d$ by ensuring the following stability conditions,

$$\begin{cases} \mathbf{b}^k = -\mathbf{A}^k \bar{\xi} \\ \mathbf{A}^k + (\mathbf{A}^k)^\top \prec 0 \end{cases} \qquad \forall k = 1 \dots \mathbf{K} \qquad \text{as in Eq. 3.34}$$

where $\mathbf{A}^k$ and $\mathbf{b}^k$ are defined according to Eq. 3.23, and $\prec 0$ refers to the negative definiteness of a matrix.

A proof and details can be found on [Khansari-Zadeh and Billard, 2010a]. Eq. 3.34 provides us with sufficient conditions to make *DS* globally asymptotically stable at the target $\bar{\xi}$. Such a model is advantageous in that it ensures that starting from any point in the space the trajectory always converges on the target.

5.3.2 Generalizing to Unseen Conditions

As we have reviewed adopting a *DS* framework was advantageous because it offered several valuable properties inherent to the nature of the *Dynamical Systems*. *DS* could be modulated to generate trajectories that have similar dynamics, performing in areas that were not covered during demonstrations [Khansari-Zadeh and Billard, 2011].

Generalization of the motion to an unobserved part of the space results immediately from the application of the function to the new set of input variables [Gribovskaya et al., 2010]. The ability to generate a trajectory from an arbitrary initial position to the target with a relevant velocity profile is a strong point of encoding motion with *DS*. This generalization process consists of exploiting the variability inherent in the various demonstrations and to extract the essential components of the task. These essential components should be those that remain unchanged across the various demonstrations [Calinon, 2009].

DS can encode movements and replay them in various conditions, Figure 5.3 present results of the generalization of the motion in different starting conditions

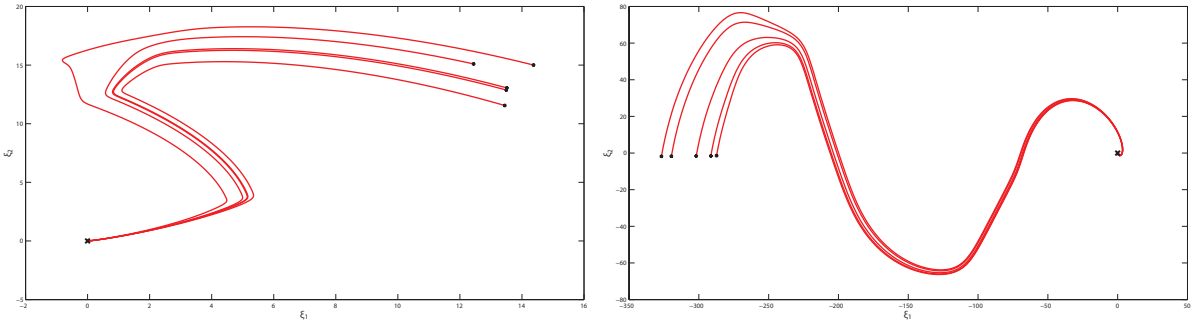


Fig. 5.3: *Generalize the motion to different starting conditions. (left) The result of starting one of the motion tested in Chapter 3, 5Curve, at different starting points. (right) The result of starting one of the motion tested in Chapter 3, Sine, at different starting points.*

tested on the reproduction of a couple of motions learned in Chapter?. Generalization properties allow the system to adapt a robot reproduction of the skill to changes in the environment relating to the position of the targets at the onset of motion.

The *DS* do not define a single trajectory but a family of solutions within the attractor landscape of our system, therefore adapting to different starting positions comes naturally under the *DS* framework, just as in the potential field approaches, *Dynamical Systems* approaches in motor control create vector fields according to which movement system is supposed to move. *DS* trajectory based thinking creates simpler, although less flexible, attractor landscapes, but scales easily to higher dimensions and enables machine learning to shape the landscapes [Ijspeert et al., 2009].

5.3.3 Robustness to Perturbations

Another advantage of adopting a *DS* framework is its inherent robustness to perturbations. *DS* are intrinsically robust and can adapt their trajectories instantly in the face of spatio-temporal perturbations [Khansari-Zadeh and Billard, 2010a].

A major strength of the *DS* approach is its ability to cope with perturbations in real-time. As a perturbation is understood, the unexpected changes the position of the attractor or the robot's joints could present during motion. The learned dynamics with a position of an object mapped into an attractor can successfully track the object. Such flexibility combined with the guarantee of ultimately reaching the object is one of the major advantages of the proposed method in comparison with traditional planners [Gribovskaya et al., 2010].

The *DS* framework provides a robust robot controller in the face of perturbations during the reproduction of a learned robot skill. Perturbations are produced by displacement of the target object or of the robot trajectory during reproduction attempts. The robot can smoothly adapt its generalized trajectory to handle dynamic perturbations during the reproduction. We must distinguish between spatial and temporal perturbations. Temporal perturbations are produced when the robot

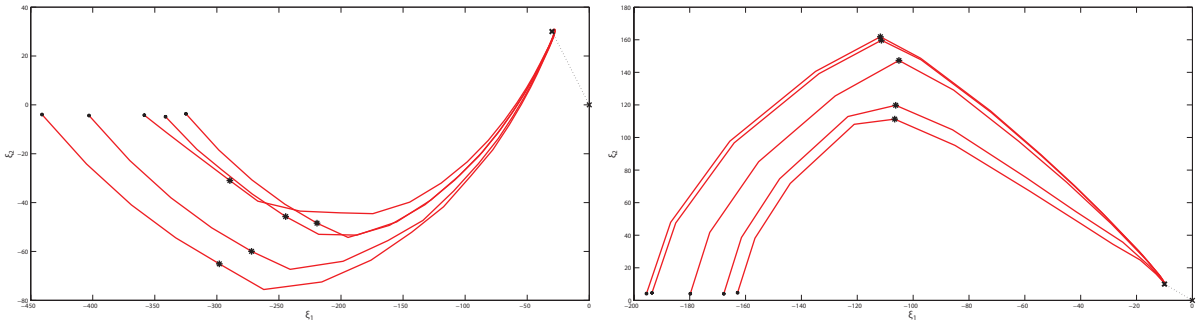


Fig. 5.4: *Robustness to a perturbation of the target position during reproduction. (left) The result of perturbation with one of the motion tested in Chapter 3, Arc, moving the target during execution. (right) The result of perturbation with one of the motion tested in Chapter 3, Angle, moving the target during execution. Trajectories are drawn in red with different starting points. A black star marks the target position and the black dotted line the displacement from the perturbation. The moment when the perturbation takes place is marked by a black asterisk.*

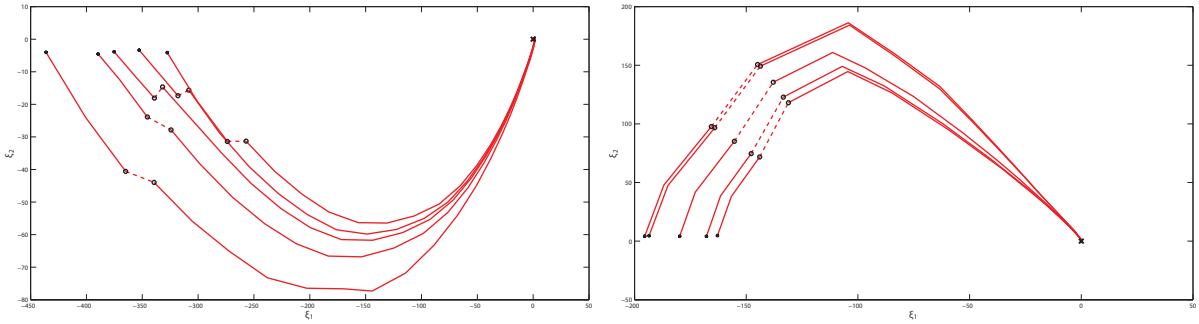


Fig. 5.5: *Robustness to a perturbation of the robot trajectory during reproduction. (left) The result of perturbation with one of the motion tested in Chapter 3, Arc, moving the robot during execution. (right) The result of perturbation with one of the motion tested in Chapter 3, Angle, moving the robot during execution. Trajectories are drawn in red with different starting points. A black star marks the target position. The displacement of the robot trajectory from the perturbation is marked by the red dashed line enclosed by black circles at the moment of perturbation.*

trajectory is momentarily stopped because of a safety issues or an object blocking its path. Spatial perturbations typically derive from the dynamic interaction with an environment in which an object or robot's arm could be displaced by an external perturbation.

Figure 5.4 presents results of the adaptation of the motion to a spatial perturbation in the position of the target tested on the reproduction of a couple of motions learned in Chapter 3. The moment when the perturbation takes place is marked by a black asterisk. The trajectories are instantly recovered from the perturbation and don't display noticeable changes in their execution towards their new target position. Figure 5.5 shows results of the adaptation of the motion to perturbations in the robot trajectory during execution tested on the reproduction of a couple of motions learned in Chapter 3. The moment when the perturbation takes place is marked by a black circle enclosing the displacement of the robot trajectory marked by the red dashed line. The *DS* recovers its trajectory instantly from the perturbation and doesn't display noticeable changes in its execution towards its new target position.

Dynamical systems offer a particularly interesting solution to an imitation process aimed at being robust to perturbations which are robust to dynamical changes in the environment [Billard et al., 2008]. Learning the robot skills as *DS* which are time-invariant and globally stable, the system is able to handle both temporal and spatial perturbations, while performing the motion as close to the demonstrations as possible. A controller driven by a *DS* is robust to perturbations because it embeds all possible solutions to reach a target in one single function [Khansari-Zadeh and Billard, 2010a]. The system is generic regarding tasks it may reproduce; furthermore, it may work with limited and inaccurate information about the environment, as it does not require any costly replanning.

5.3.4 Obstacle Avoidance

Working with humanoids in the natural environment requires that the robotic systems work in cluttered environments, where they may face several objects during the task execution. Collision avoidance capabilities would have to be present for these systems.

Obstacle avoidance is a classic problem in robotics and many approaches have been proposed to solve it. One may distinguish between local and global methods, depending on whether the obstacle influences the behaviour either locally or everywhere. Global methods, such as those dealt with by path planning algorithms, ensure finding a valid solution, if it exists. However, these methods cannot offer the reactivity sought for swiftly avoiding obstacles that appear suddenly [Khansari-Zadeh and Billard, 2012].

In Artificial Potential Fields, obstacles are modelled as repelling potential fields which are designed to automatically push a control system to circumnavigate them in an on-line reactive way which prevents the robot from colliding with the obstacle. An appropriate repulsion force should be computed so that it repels the trajectory sufficiently away from the obstacle while avoiding getting stuck in local minima. Such reactive behaviour assumes that obstacles may appear in an unforeseen and sudden way, such that pre-planning is not possible or useful [Ijspeert et al., 2009]

The approaches based on attractor dynamics are another variant of the potential field method, which uses heading direction, rather than the Cartesian position of the robot. The Dynamic Potential Field approach extends the potential field principle by also considering the velocity along the path [Khansari-Zadeh and Billard, 2012].

[Ijspeert et al., 2009] suggested a model for obstacle avoidance with the use of a coupling term in their *DS* approach. On the way to the goal state, an obstacle is positioned at $\mathbf{o} = [o_1 o_2 o_3]^T$ and needs to be avoided. A suitable coupling term $C_t = [C_{t,1} C_{t,2} C_{t,3}]^T$ for obstacle avoidance can be formulated as $C_t = \gamma R \dot{y} \phi exp(-\beta\phi)$. The angle ϕ is interpreted as the angle between the velocity vector $\dot{y}$ and the difference vector $(\mathbf{o} - y)$ between the current position and the obstacle. R is a rotation matrix which causes a rotation of 90 degrees about the vector r perpendicular to the plane spanned by $\dot{y}$ and $(\mathbf{o} - y)$.

In [Khansari-Zadeh and Billard, 2012] a novel approach is presented to real-time obstacle avoidance based on *DS* that ensures impenetrability of multiple convex shaped objects. Their approach induced a modulation on the generic motion due to the presence of an obstacle. First, start with an object centred on ξ^o and provide a convex bounding volume formulation of the outer surface of the obstacle and define $\tilde{\xi} = \xi - \xi^o$ to simplify notation. A continuous function $\Gamma(\tilde{\xi})$, which projects $\mathbb{R}^d$ into $\mathbb{R}$, has continuous first order partial derivatives and increases monotonically. By construction, the relation $\Gamma(\tilde{\xi}) = 1$ holds at the surface of the obstacle. A modulation matrix is given by $M(\tilde{\xi}) = E(\tilde{\xi}) D(\tilde{\xi}) E(\tilde{\xi})^{-1}$ The dynamic modulation matrix $M(\tilde{\xi})$ propagates the influence of the obstacle on the motion flow. The effect of the dynamic modulation matrix is maximum at the boundaries of the obstacle, and vanishes for points far away from it [Khansari-Zadeh and Billard, 2012]. The modulation matrix can be applied to the original dynamics given by f so as to have,

$$\dot{\xi} = M(\tilde{\xi}) f(\xi) \tag{5.4}$$

From [Khansari-Zadeh and Billard, 2012] a motion that starts outside the obstacle, $\Gamma(\tilde{\xi(0)}) \geq 1$, and evolves according to Eq. 5.5 does not penetrate the obstacle. Therefore, the dynamic modulation matrix $M(\tilde{\xi})$ can be used to deform a robot motion such that it does not collide with an obstacle. The magnitude of the modulation can be tuned by modifying the eigenvalues of the dynamic modulation matrix.

When in the presence of multiple obstacles, the single modulation matrix is ineffective and should be modified to include the effect of all the obstacles. Considering K obstacles with associated reference points $\xi^{o;k}$ and boundary functions $\Gamma^k(\tilde{\xi^k})$, for $k = 1..K$. the dynamic modulation matrix for each obstacle can be expressed by $M^k(\tilde{\xi^k}) = E^k(\tilde{\xi^k}) D^k(\tilde{\xi^k}) E^k(\tilde{\xi^k})^{-1}$. The combined modulation matrix that considers the net effect of all the obstacles is then given by,

$$\bar{M}(\tilde{\xi}) = \prod_{k=1}^{K} M^k(\tilde{\xi^k}) \tag{5.5}$$

The modulation is parametrizable and allows to determine a safety margin. For proof and further explanations see [Khansari-Zadeh and Billard, 2012]. The approach should be incorporated as part of the *Robot Skills Models* in future work.

5.4 Update of Robot Skills

In previous chapters the techniques to learn the models of robot skills and the development of a knowledge base to store and access them to have a set of basic primitive actions on which to generate complex human like motions have been presented and developed. Now it is necessary to come up with methods to operate upon the *Robot Skills Models*. A first desirable manipulation over the learned robot skills would naturally be the ability to update and refine the models in order to adapt the skill with new information. As outlined at the beginning of this chapter, learning and encoding the models of the robot skills for every conceivable scenario the robot may encounter would not be feasible, therefore, updating previously learned models is a key mechanism for generating new models and expanding the application and versatility of the robot skills. A robot skill must be updatable; when given new information for the representation of a skill the system must allow for the models to be refined. The update approaches are related to those efforts to develop incremental learning techniques. Unlike other approaches which assume that data comes in blocks, the incremental learning approaches work for the case when novel data points arrive one by one. Incremental learning approaches that gradually refine the task knowledge as more examples become available pave the way towards *LfD* systems suitable for real-time interactions between humans and robots [Billard et al., 2008].

Intuitively, a new model of the skill could be generated by including the new demonstrations $\mathcal{D}_{new}$ of the skill with the previous dataset $\mathcal{D}_{orig}$ and just retrain the model of the skill, with the complete dataset $\mathcal{D} = \mathcal{D}_{orig} + \mathcal{D}_{new}$, as it was described in Chapter 3. This would produce a new model, yet, several issues arise. First, for this approach to be possible it would be required that all the training information from the demonstrations' dataset be stored in memory. Storing all this information should not be required. New available data must allow to refine a model of the motion without the need for keeping the whole training demonstrations data in memory [Calinon, 2009]. Secondly, trying to update the skill model with all the previous demonstrations could present a problem of diluting the influence at the new demonstrations if is paired with a much bigger dataset. Intuitively, it can be seen that if all the information that was available is the current *GMM* estimate, then a single novel point would never carry enough information to cause significant change in the Gaussian components [Arandjelovic and Cipolla, 2005]. Also, by using the combined dataset of old and new demonstrations there could be issues on the compatibility of the recorded demonstrations, which would need to be adjusted before training with the possible loss of information.

The system must be capable of updating and refining its model of a skill when presented with new relevant information for the skills, taking only the stored knowledge of the *Robot Skill Model* in memory by updating its learned parameters θ based on the new demonstration.

The problem of incrementally updating a *GMM*, taking only into account new incoming data and a previous estimation of *GMM* parameters has been proposed for on-line data stream clustering. [Song and Wang, 2005] suggested an approach to incrementally updating the estimate taking only the newly arrived data and the pre-

viously estimated model. Their approach first creates a new *GMM* from the new incoming data, and then creates a compound model by merging the components of the old and the new *GMM*. Their incremental Gaussian mixture model estimation algorithm merges Gaussian components that are statistically equivalent. For each cluster in the new *GMM*, it is determined if there is a statistically equivalent covariance and mean with any of the components of the old *GMM*, then a new component is created by merging them. If not it will add the remaining components adjusting their weights accordingly. The main drawback of the suggested approach is that it is computationally expensive and tends to produce more components than the standard *EM* algorithm [Calinon, 2009]. Also, they fail to exploit the available probabilistic information by failing to take into account the evidence for each component at the time of merging [Arandjelovic and Cipolla, 2005].

Other approaches suggest the use of the temporal coherence properties of data streams to update the *GMM* parameters. [Arandjelovic and Cipolla, 2005] propose a method consisting of a three-stage model update each time a new data point becomes available. First, model parameters are updated under the constraint of fixed complexity. Then new Gaussian components are postulated by model splitting and components are merged to minimize the expected model description length. Their model assumes that data varies smoothly in time, which allows the *GMM* parameters to be adjusted when new data is observed.

[Calinon, 2009] proposes two approaches to deal with these problems, where the need is for adjusting an already existing model when new data points are given. A first method proposed a reformulation of the problem in [Arandjelovic and Cipolla, 2005] for a generic observation of multiple data points. The idea is that an adaptation of the *EM* algorithm in Eq. 3.21 by separating the parts dedicated to the data already used to train the model and the one dedicated to the newly available data, with the assumption that the set of posterior probabilities remain the same when the new data is used to update the model. The model is first created with N data points ξ_j and updated iteratively during T *EM-steps* until convergence to the set of parameters $(\pi_T^k, \mu_T^k, \Sigma_T^k)_{k=1}^k$. When a new demonstration is given, $\tilde{T}$ *EM-steps* are again performed to adjust the model to the new $\tilde{N}$ data points $\tilde{\xi}_j$, starting from an initial set of parameters $(\tilde{\pi}_0^k, \tilde{\mu}_0^k, \tilde{\Sigma}_0^k)_{k=1}^k = (\pi_T^k, \mu_T^k, \Sigma_T^k)_{k=1}^k$ and iterating until a new updated model is estimated.

It is important to note that for this approach to work, it is assumed that the cumulated posterior probability does not change much with the inclusion of the novel data in the model; this is only true if the new data is close to the model [Calinon, 2009]. This restriction is important because it cannot always be guaranteed and more importantly most times it is not even wanted, since the desirable refinement of the model requires sufficient departure from the original skill for the update process to be meaningful. To illustrate the result of this method Figure 5.6 shows the result of updating a learned model of a skill with a new demonstration.

[Calinon, 2009] also presented an alternative to the above method using a stochastic process to update the parameters. An initial *GMM* model $(\pi^k, \mu^k, \Sigma^k)_{k=1}^k$ is first created using the *EM* algorithm in Eq. 3.21. When an update is required with new given data a process of *GMR* regression is performed over the learned model to

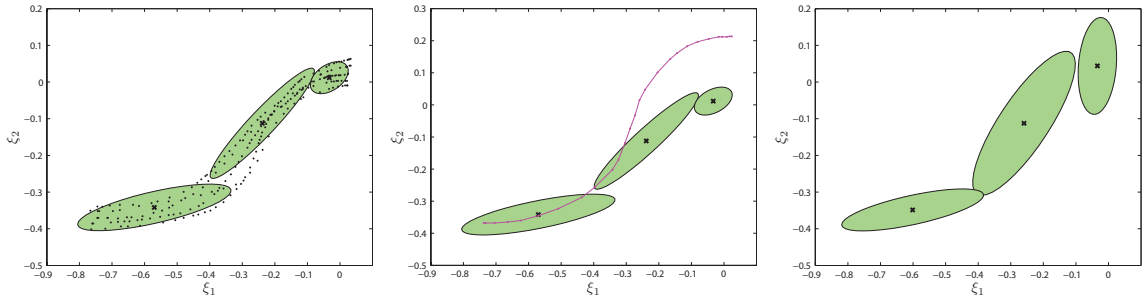


Fig. 5.6: *Direct incremental method update of a skill. (left) Model of the learned skill, with demonstrations in black. (center) New demonstration, in magenta, over the learned skill model. (right) Updated model of the skill.*

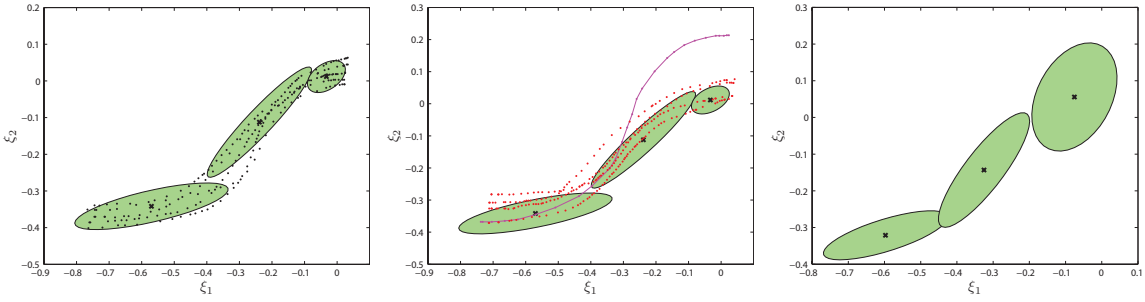


Fig. 5.7: *Generative method update of a skill. (left) Model of the learned skill, with demonstrations in black. (center) Model of the learned skill, with the new demonstration in magenta and the generated trajectories in red. (right) Updated model of the skill.*

stochastically generate a dataset from the model. Therefore a new dataset is created composed of this generated demonstration and the new observed dataset; the *GMM* parameters of the updated model are then retrained with the *EM* algorithm. For this purpose a learning rate α was defined, along with the number of samples, $n = n_1 + n_2$ used for the learning procedure, where n_1 and n_2 are respectively the number of examples from the new observation and number of examples generated stochastically by the current model. The training set of n trajectories is used to refine the model by updating the current set of parameters using the *EM* algorithm [Calinon, 2009]. The value for $\alpha \in [0; 1]$ can be set to a fixed learning rate or depend on the current number of demonstrations used to train the model, also α can be computed recursively for each newly available demonstration.

To illustrate the result of this method Figure 5.7 shows the updating a learned model of a skill in the same way as the one that was presented for the previous method in Figure 5.6.

For the adaptation of a task model by updating a robot skill in this work a method similar to the one presented above from [Calinon, 2009] is employed. Models of a skill must be updatable when given new information for the representation of a

Algorithm: Update the learned robot skill

Input: Learned *Robot Skill Model*, $\mathcal{M}_{RS}$, with parameters $\theta^k = (\pi^k, \mu^k, \Sigma^k)$.

1. Record new demonstration trajectory for the update of the skill.
2. Generate stochastically n_gen trajectories from the current model by the *GMR*.
3. Determine the parameter $\alpha = \alpha^k \in [0; 1]; k = 1..\mathbf{K}$
4. Create a new update demonstration dataset $\{\xi, \dot{\xi}\}_{update}$
5. Generate the new update model of the skill.
6. **END**

Output: Updated *Robot Skill Model*, $\mathcal{M}_{RS_{update}}$, with parameters $\theta^k_{update} = (\pi^k, \mu^k, \Sigma^k)$.

Tab. 5.1: *Procedure for updating a learned model of a robot skill.*

skill without having to store the training demonstrations data in memory. A *Robot Skill Model*, $\bar{\mathcal{M}}_{RS}$, is first learned by means of the *SEDS* algorithm presented by [Khansari-Zadeh and Billard, 2011] as described in Chapter 3 with learned parameters $\theta^k = (\pi^k, \mu^k, \Sigma^k)$. We are considering only the case when the model is refined after receiving one update demonstration for the skill, so therefore, the number of samples n, used for the learning procedure would be $n = n_gen+1$, with n_gen being the number of examples generated stochastically from the current model by the *GMR*. For our method the new update demonstration dataset $\{\xi, \dot{\xi}\}_{update}$ would be grouped into $\mathbf{K}$ clusters according to the number of Gaussian functions determined for the original *Robot Skill Model*, and the parameter α would be defined as $\alpha^k \in [0; 1]; k = 1..\mathbf{K}$, and it would determine a measure of the relative importance of the area in cluster k the update demonstration should have for refining the model over the stochastic demonstrations generated from the learned model. When generating the stochastic demonstrations sampling out of the *GMR* with the learned parameters θ of the robot skill random samples are taken starting around the given demonstration. To induce more weight on the update dataset or the generated dataset as determined by the parameter α^k, data points of the generated dataset which are too far removed from the update dataset according to a threshold dependant of α will be discarded.

To illustrate the result of this method Figure 5.8 shows the result of updating a learned model of a skill. The process for updating a learned *Robot Skill Model* is summarized in Table 5.1.

5.5 Merger of Robot Skills

As it has been stated throughout this chapter, efforts at learning and generating movement primitives robotic skills can only have a real implementation value for developing humanoid robotic systems if the models of the skill can be operated upon to generate new behaviours of increasing levels of complexity. It is necessary to come

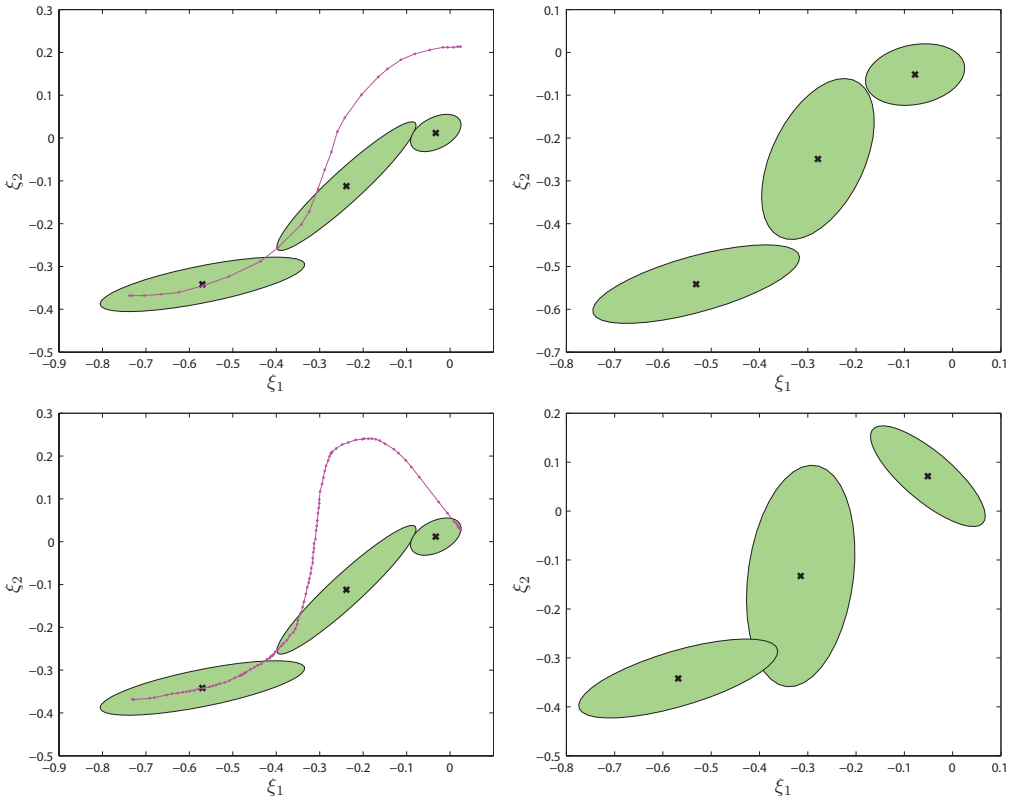


Fig. 5.8: *Update process of a robot skill. Model of the learned skill with new demonstrations in magenta. Updated model of the skill. (top) this example show the update of the skill for the same demonstration as the above examples, the parameter α^k are define to govern the influence of new data on the update process. (bottom) Appropriate selection of α^k allows the updated model to reproduce the curve at the top of the trajectory.*

up with methods to manipulate the *Robot Skills Models*. Encoding models of the robot skills for every conceivable need the robot may find itself in is not plausible, hence, it is key for approaches to be able to generate new skills by merging different skill primitives into a new skill. The ability to generalize skills and adapt them to a new situation is fundamental for the *LfD* concept; performing the task under different circumstances from those present during demonstrations, given appropriate adaptation, can allow an acquired skill to carry out more complex task than the teacher is capable of demonstrating [Khansari-Zadeh and Billard, 2011].

The learned *DS* models encode specific motion skills, which can be seen as building blocks used to generate more complex motions. Multiple desired robot skills may be composed from sequencing or superposition of various primitives skills. The modularity of the *DS* approach is essential as it would allow to control a wider repertoire of movements from a smaller set of basic skills [Schaal, 1999]. Intuitively, one could consider an approach to merging two or more models of a skill simply by adding and averaging together their learned parameters $\theta = (\pi, \mu, \Sigma)$ in order to obtain a new skill model through a linear superposition. The models would represent the distributions $f^1(\xi)$ and $f^2(\xi)$ respectively as from Eq. 3.17,

$$f^1(\xi) = \sum_{i=1}^{\mathbf{N}} \pi^i \mathcal{N}^i(\xi; \mu^i, \Sigma^i)$$
$$f^2(\xi) = \sum_{j=1}^{\mathbf{M}} \pi^j \mathcal{N}^j(\xi; \mu^j, \Sigma^j)$$

A weighted sum of these densities would give the merged model

$$f(\xi) = \alpha f^1(\xi) + \beta f^2(\xi) \tag{5.6}$$

The weights α and β scale the prior of the components to give the new *GMM*, with $N + M$ components, by simply concatenating the descriptions of each *GMM*,

$$f(\xi) = \sum_{k=1}^{N+M} \pi^k \mathcal{N}^k(\xi; \mu^k, \Sigma^k)$$

the first N components terms are specified from $f^1(\xi)$, while the remaining components come from $f^2(\xi)$. The prior π^k are exactly the π^i or π^j of f^1, f^2 scaled by *alpha* or β accordingly, while the μ^k centres and Σ^k covariance matrix are copied from their source *GMM*. While this approach may work in some cases it is important to note that direct superposition of the skills does not allow to control the manner in which the new model is generated. Also the non-linear sum of two or more stable *DS* would not necessarily generate a stable new model and special attention should be considered in this regard [Khansari-Zadeh and Billard, 2011].

The work of [Hall et al., 2005] presents a modified approach to merging a pair of *GMM* to produce a third *GMM*; this closely approximates the *GMM* which would

be constructed by a standard algorithm for fitting the data, having the concatenation of data sets of the two mixture models as input.

$$f(\xi^1) \oplus f(\xi^2) \approx f(\xi^1 : \xi^2) \tag{5.7}$$

There are three main steps to their approach, first concatenating the models by trivially combining the *GMM* into a single model, as per Eq. 5.6, to produce a model with $M + N$ components. Then simplifying the *GMM* of the concatenated model by merging components using a weighted summation of their parameters. [Hall et al., 2005] merged components by combining their parametric descriptions, not by adding the density functions. Finally, selecting the optimal number of components $1 \leq K \leq M + N$ for the *GMM* that best explains the distribution.

[Muelling et al., 2013] presented a framework to generalize learned motor primitives to a wider range of situations using a mixture of motor primitives approach. The resulting policy enables the robot to select appropriate motor primitives as well as to generalize between them. The goal was to acquire a library of movement primitives from demonstrations and to select and generalize among these movement primitives to adapt to new situations. The primitives are associated with a set of parameters referred to as the augmented state. A new movement is generated for a new augmented state selecting a primitive to use as components of the mixture of motor primitives algorithm [Muelling et al., 2013]. The algorithm activates components using a gating network based on the augmented state and generates a new movement using the activated components.

The mixture of motor primitives generates a new movement for the current situation triggered by the augmented state by computing the weighted average of all movement primitives in the library, the resulting policy $f(\xi)$ generated by the algorithm is given by

$$f(\xi) = \frac{\sum_{i=1}^{\mathbf{L}} \gamma^i(\delta) f^i(\xi)}{\sum_{j=1}^{\mathbf{L}} \gamma^i(\delta)} \tag{5.8}$$

where the function $\gamma^i(\delta)$ generates the weight of f^i given the augmented state ξ. The sum of all weights $\sum_{j=1}^{\mathbf{L}} \gamma^i(\delta)$ form the gating network of the mixture of motor primitives algorithm [Muelling et al., 2013]. The gating network weights the movement primitives based on their expected performance within the current context, ensuring only appropriate movement primitives can contribute. The weights are modelled by an exponential family distribution. The resulting motor policy $f(\xi)$ is composed of several primitives weighted by their suitability in the given context of the task; the weights are adapted to the task based on the outcome of previous trials [Muelling et al., 2013].

In this work, in order to generate a new skill based on the merger of several *Robot Skills Models*, previously learned and stored in the knowledge base, we developed a method taking from the above approaches. First we review a couple of useful

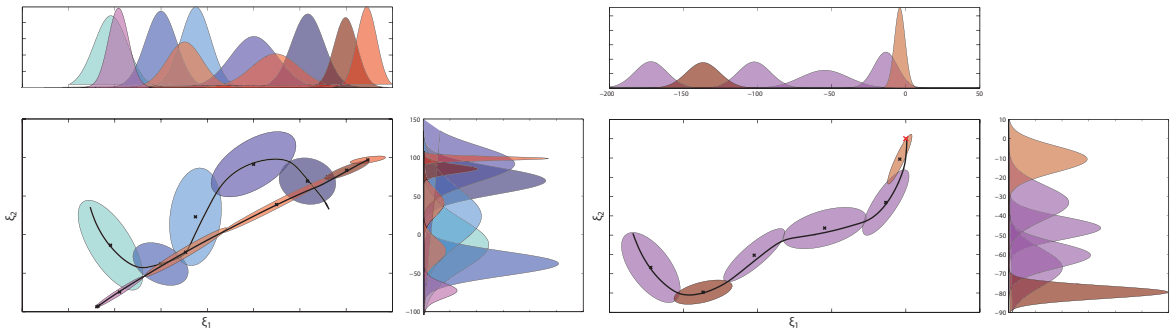


Fig. 5.9: *Process of merging two robot skills to generate a new model. (left) Representation of the two learned models of the robot skill and their non-linear terms h^k. (right) Representation of the new robot skill model generated from the merger process and its non-linear terms $\tilde{h}^k$.*

mathematical properties from the *SEDS* formulation chosen to learn the skills,

$$\begin{gathered} \text{if} \quad f(\xi) \quad \text{is } \textit{SEDS}, \text{ and} \quad \alpha > 0 \in \mathbb{R} \\ \dot{\xi} = \alpha f(\xi) \quad \text{is } \textit{SEDS} \\ \text{consider} \quad \mathbf{M} \quad \textit{SEDS} \text{ models} \quad f^i(\xi), i \in 1..\mathbf{M} \\ \dot{\xi} = \sum_{i=1}^{\mathbf{M}} \alpha^i f^i(\xi); \alpha^i > 0 \quad \text{is } \textit{SEDS} \end{gathered} \tag{5.9}$$

The models of the robot skills can be expressed as a non-linear sum of linear dynamical systems of the form

$$\dot{\xi} = \hat{f}(\xi) = \sum_{k=1}^{K} h^k(\xi)(\mathbf{A}^k\xi + \mathbf{b}^k) \qquad \text{as in Eq. 3.24}$$

Here, recalling the expression of the non-linear weighting function $h^k(\xi)$, as in Eq. 3.23, it can be found that it shares a similar formulation with the expression of the weights $\gamma^i(\delta)$ for the gating function of Eq. 5.8. The merger of the *Robot Skills Models* can be carried out with a model combination approach expressed in mixtures of experts model as from Eq. 5.1, in which the mixing coefficients $\pi_k(x)$ of the gating function are given by the non-linear weighting function $h^k(\xi)$, and the $p_k(t|x)$ density is given by the linear *DS* $\mathbf{A}^k\xi + \mathbf{b}^k$.

The process for the merging of robot skills would proceed as in the above approaches; first, the *GMM* of the robot skills are joined into a single model. Then a new weighting function $\tilde{h}(\xi)$ for the single model must be built out of the original weighting terms $h^k(\xi)$ from the merged models, ensuring the Gaussian with the biggest weight in every region of the trajectory provides the largest influence over the new *GMM* model in that region and that the new weighting function $\tilde{h}(\xi)$ still meets the constraint of the mixing coefficient as in Eq. 3.22, $0 > h^k(\xi) > 1$ and $\sum h^k(\xi) = 1$.

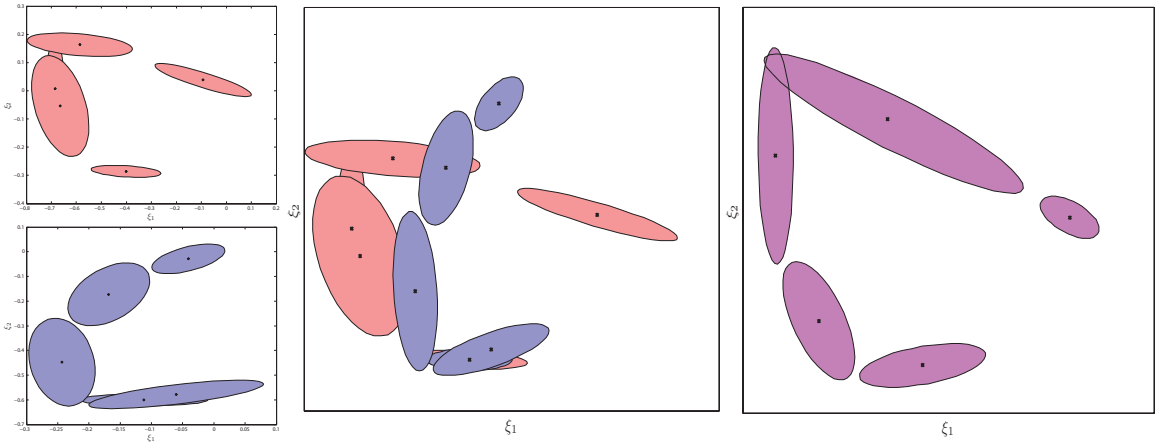


Fig. 5.10: *Merger of two learned GMM skill models to generate a new skill. (left) Learned models of the robot skill. (center) Merging process of the two models to generate a new one. (right) Merged skill model.*

Then a new weighting function $\tilde{h}^k(\xi)$ would be given by $\tilde{h}^k(\xi) = \alpha^k(\xi, h)h^k(\xi)$ where $\alpha^k(\xi, h)$ is a scalar function that weights the original $h^k(\xi)$ of the models, and ensures the constraints of $\tilde{h}(\xi)$. Figure 5.9 illustrates the process of merging two robot skills to generate a new skill model.

Figure 5.10 illustrates the result of merging two robot skills to generate a new skill model. The process for updating a learned *Robot Skill Model* is summarized in Table 5.3.

5.6 Combination of Robot Skills

Operations over the models of the robot skills must include the capacity to generate skills in order to allow carrying out more complex task than those the teacher is capable of presenting during demonstrations. The *Robot Skills Models* must be combinable into new models capable of generating skills, encompassing a larger spectrum of the attractor dynamics. One important gain from the combination of robot skills comes from increasing the accuracy of the generalized behaviour. The convergence of the motion to the target is ensured, yet, due to the lack of information for points far from demonstrations, a model may reproduce some trajectories that are not consistent with the usual way of doing the task. The presented behaviours of the robot may not be optimal in these cases; however, such results are inevitable, given that the information from demonstrations is incomplete and the inference for points too far from them is not reliable. The generation of a model by combining robot skills is necessary in order to improve the task execution.

The more direct and intuitive approach would rely on providing the robot with more demonstrations over regions not covered before. By showing the robot more demonstrations and re-training the model with the new data, the robot should be able to successfully accomplish the task [Khansari-Zadeh and Billard, 2011]. However,

Algorithm: Merger of the learned robot skill

Input: Learned *Robot Skill Models*, $\mathcal{M}^1_{RS}$, $\mathcal{M}^2_{RS}$, ..., $\mathcal{M}^n_{RS}$.

1. Compute the new model as $\sum_{k=1}^{K} h^k(\xi)(\mathbf{A}^k\xi + \mathbf{b}^k)$.
2. Compute the parameters α^k for the new model.
3. Build the weighting function $\tilde{h}$, as $\tilde{h}(\xi) = \alpha^k(\xi, h)h^k(\xi)$.
4. Generate the new merged model of the skill.
5. **END**

Output: Merged *Robot Skill Model*, $\mathcal{M}_{RS_{merged}}$, given by $\sum_{k=1}^{K} \tilde{h}^k(\xi)(\mathbf{A}^k\xi + \mathbf{b}^k)$.

Tab. 5.2: *Procedure for merging learned models of a robot skill.*

this approach would not seem to be the most flexible and general, and also robots performing tasks in the real world cannot reliably expect to have an available teacher to provide them with more demonstrations whenever their knowledge of a task doesn't suffice.

The work of [Chatzis et al., 2012] reformulate *GMR* models, introducing the concept of quantum states, which can be constructed by superposing conventional *GMR* states by means of linear combinations; their approach is especially suitable for learning complex demonstration trajectories. In [Shukla and Billard, 2012] the focus is on combining several learned *DS*, with distinct attractors, resulting in a multi-stable *DS*, as could be the case of different attractors representing several grasp points of a single object. Their work presented an *Augmented-SVM* model, which inherits region partitioning ability of well know *Support Vector Machine (SVM)* classifiers and is augmented with novel constraints derived from the individual *DS*. A *DS* composed of multiple stable attractors provides an opportunity to encode multiple dynamics, directed towards different attractors, into a single *DS*. Restricting the motion dynamics to a single attractor constrains considerably the applicability of these methods to realistic grasping problems. From a robotics viewpoint, a robot controlled using a *DS* with multiple attractors would be able to switch online across grasping strategies [Shukla and Billard, 2012].

The stability at multiple targets is an important concern; this problem has been addressed largely through neural networks approaches. For instance, Hopfield networks can offer a powerful means of encoding several stable attractors in the same system. However, the dynamics to reach these attractors was not controlled for; nor was the partitioning of the state space that would send the trajectories to each attractor. A naive approach to building a multi-attractor *DS* would be to first partition the space and then learn a DS in each partition separately this would however rarely result in the desired compound system [Shukla and Billard, 2012]. Due to the influence of non-linear dynamics, trajectories that initialize in one region could cross the boundary and converge to the attractor of the other region. In a real scenario, cross-

ing over may take the trajectories towards unreachable regions. Also, trajectories that encounter the boundary may switch rapidly between different dynamics leading to jittery motion [Shukla and Billard, 2012].

To ensure the trajectories remain within the region of attraction of their respective attractors, an approach can be adapted in which each of the original *DS* is modulated so that the generated trajectories always move away from the classifier boundary. [Shukla and Billard, 2012] developed a system that ensured strict classification across regions of attraction for each *DS*, closely following the dynamics of each *DS* and ensuring that trajectories in each region reached their desired attractors. [Shukla and Billard, 2012] presented the *Augmented-SVM* model for combining non-linear *DS* through a partitioning of the space. The resulting model behaves as a multi-stable *DS* with attractors at the desired locations.

[Khansari-Zadeh and Billard, 2011] presents an embedding of different ways of performing a task in one single model. As stated above, sometimes it may be necessary to execute a single task in different ways starting from different areas in the space and a single *DS* driving the motion is not sufficient. Their work uses *SEDS* to integrate different motions into one single dynamic. The robot follows distinct trajectories starting from different points in the workspace. Two different *SEDS* models, $\bar{\mathcal{M}}^1_{RS}, \bar{\mathcal{M}}^2_{RS}$ can be combined just by concatenating their parameters, such that the parameter of the new model can defined as $\pi = \frac{[\pi^1; \pi^2]}{(\pi^1 + \pi^2)}$, $\mu = [\mu^1 \mu^2]$ and $\Sigma = [\Sigma^1 \Sigma^2]$. While reproductions locally follow the desired motion around each set of demonstrations, they smoothly switch from one motion to another in areas between demonstrations [Khansari-Zadeh and Billard, 2010a]. The proposed method offers a simple but reliable procedure to teach a robot different ways of performing a task; however, a more complex method is required in order to provide a better fit for the multiple dynamics and prevent possible interference among models when switching between different dynamics in trajectories close to the border of each attractor region.

In this work, in order to generate a new skill made of the combination of several *Robot Skills Models* previously learned and stored in the knowledge base, we developed a method taking from the above approaches. Two different *SEDS* models are first combined by concatenating their parameters. Then, an area of influence for the *DS* attractor is defined based on the non-linear weighting function $h^k(\xi)$ of the *SEDS* models expressed as a non-linear sum of linear dynamical systems as in Eq. 3.24. A new weighting function $\tilde{h}(\xi) = \alpha^k(\xi, h)h^k(\xi)$ for the single model must be built out of the original weighting terms $h^k(\xi)$, as in the merging of the models, however in this case the $h^k(\xi)$ terms must be strongly biased such as that the influence over the trajectory comes at any time from only one model, therefore, the $\alpha^k(\xi, h)$ function must have a completely different form that for the merging of the robot skill models.

Figure 5.11 and 5.12 illustrate the results for combining two and three robot skills to generate a new skill model. The process for updating a learned *Robot Skill Model* is summarized in Table 5.3.

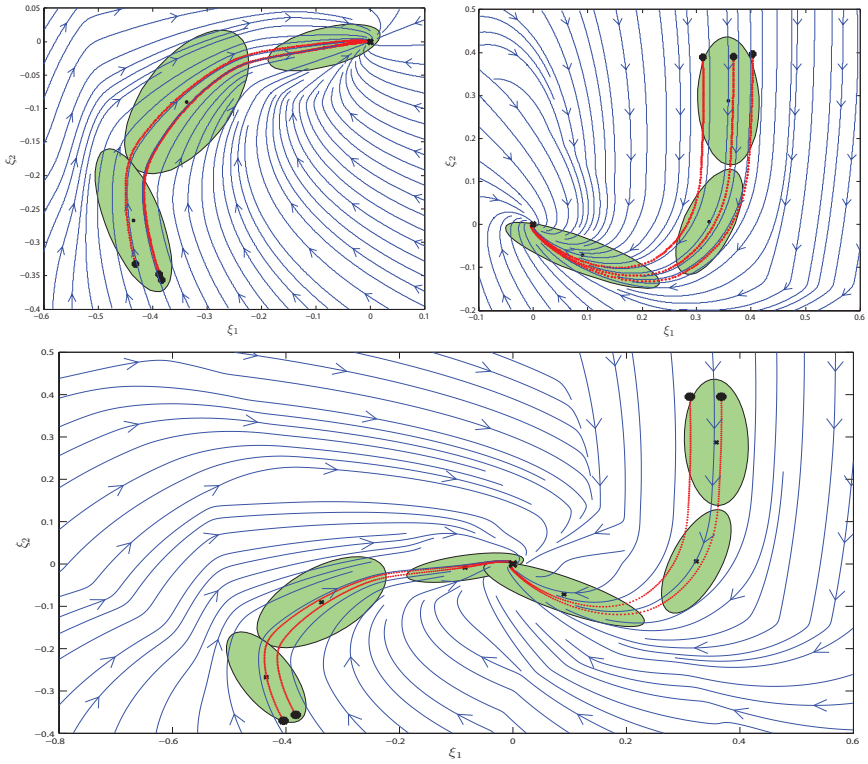


Fig. 5.11: *Combining the dynamics of two skills into a single task model.*

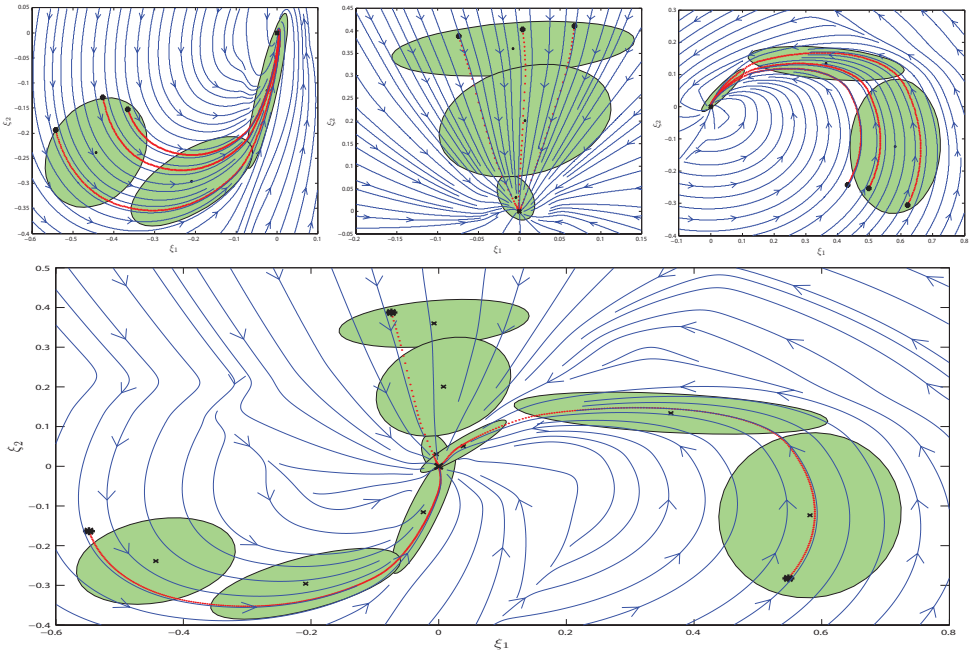


Fig. 5.12: *Combining the dynamics of three skills into a single task model.*

Algorithm: Combination of learned robot skill

Input: Learned *Robot Skill Models*, $\mathcal{M}^1_{RS}$, $\mathcal{M}^2_{RS}$, ..., $\mathcal{M}^n_{RS}$.

1. Calculate the prior $\tilde{\pi}$, as $\tilde{\pi} = \frac{[\pi^1; \pi^2; ...; \pi^n]}{(\pi^1 + \pi^2)}$
2. Calculate the mean $\tilde{\mu}$, as $\tilde{\mu} = [\mu^1 \mu^2 ... \mu^n]$
3. Calculate the covariance $\tilde{\Sigma}$, as $\tilde{\Sigma} = [\Sigma^1 \Sigma^2 ... \Sigma^n]$
4. Build the weighting function $\tilde{h}$, as $\tilde{h}(\xi) = \alpha^k(\xi, h) h^k(\xi)$.
5. **END**

Output: Combine *Robot Skill Model*, $\mathcal{M}_{RS_{combine}}$, given by $\sum_{k=1}^{K} \tilde{h}^k(\xi)(\mathbf{A}^k \xi + \mathbf{b}^k)$.

Tab. 5.3: *Procedure for combining learned models of a robot skill.*

5.7 Transition between Robot Skills

A desirable operation over the basic set of primitive skills consists of the sequencing and transition between robot skill models in order to generate complex behaviours with smooth transformation among the reproduction of different skill motions.

The simplest way to join several *DS* would be just to perform one robot skill until it reaches the end point of the motion and then, continue with the reproduction of the next *DS* starting at that point; that is, the end point of the first *DS* is used as the starting point of the second *DS* and so on. This approach is very simple, but it clearly has certain drawbacks, mostly stemming from the unnatural slowing and restarting behaviour that the close-to-zero velocities at the end of the movement trajectory in the original *DS* would produce.

[Kulvicius et al., 2012] focused an approach for joining movement sequences modifying the learned *DMP* exemplified in a handwritten application. The method is based on the modification of the original *DMP* formulation. The new method can reproduce the target trajectory with high accuracy regarding both the position and the velocity profile and produces smooth and natural transitions in position space, as well as in velocity space.

Smooth transitions between the *DS* representing the robot skills could be produced by modifying the parameters of the *DS* to generate a transcritical bifurcation at the moment the first *DS* reaches its attractor, pushing the system dynamics towards the attractor of the second *DS*. In a transcritical bifurcation a fixed point interchanges its stability with another fixed point as the parameter is varied [Strogatz, 1994]. In this type of bifurcation an attractive stable fixed point is exchanged, when they collide, so the unstable fixed point becomes stable and vice versa. A robot skill would reproduce the trajectory as in a normal case towards its target attractor, when the first motion is close to reaching the attractor, the bifurcation would change the stable nature of the attractor in order to move the system from this state towards the target attractor of the following skill *DS*.

These types of approaches, as well as others, are important case studies. Performing different tasks and executing different robot skills with smooth, natural and stable transitions between them is an important goal for humanoid robotics.

5.8 Summary of the Chapter

Throughout this chapter a review of the algorithms developed for the generation and adaptation of the robot skills has been given. Humanoid robots working beside humans in complex dynamic environments are required to perform a wide repertoire of tasks. Efforts to generate robotic skills can only have a real implementation value for developing humanoid robotic systems if the models of the skill can be operated upon to generate new behaviours of increasing levels of complexity. Section 5.2 presents a review of related approaches aiming at the adaptation of learned skill models and the developments for the generation and adaptation of the robot skills. Section 5.3 presented dynamical properties inherent to the models of a robot skill, such as robustness to spatio-temporal perturbations, independence on time, and generalizable to unseen initial conditions. Stability conditions required for generating stable *DS* representations of the skill were reviewed as well as a method to expand applicability of the *DS* approach with mechanism for obstacle avoidance. In this chapter, processes by which, using the already learned model of a robot skill and the extracted constraints knowledge of the current task, the model of a skill can be adapted to reproduce a new task were described. Different modalities were developed and implemented that allow for the adaptation and generation of new skill models based on the already learned models of skills stored in the knowledge base. Different modes are presented for the adaptation, update, merger, and combination of the *Robot Skills Models*. Section 5.4 presented the adaptation of a task model by updating a robot skill. Updating previously learned skills is a very important ability for humanoid robots, allowing them to increase and improve their available skill set. A developed method for updating a robot skill was presented in Table 5.1. Section 5.5 presented the generation of a task model by merging robot skills. Skills can be generated by merging two or more models into a new skill. New models of a skills can be generated by merging two or more models into a new skill in order to expand the robot skill set and increase its range of action. A developed method for merging robot skills was presented in Table 5.2. Section 5.6 presented the generation of a task model by combining robot skills. Models of a skill can be combined to generate new models that encompass a larger spectrum of the attractor dynamics and allowing to generalize the models of the skills to regions outside their original demonstrations. A developed method for combining robot skills was presented in Table 5.3. Section 5.7 discussed the generation of a task model by transitioning between robot skills. For humanoid robots to be capable of working successfully in the capacity in which they are envisioned, it is of vital importance that they present ample and robust skill sets. The ability to learn robot skills is a key aspect, yet learning by itself is not sufficient, the capacity to operate over the learned robot skill, such as the merger, update and combination of skills, is necessary.

6. REPRODUCTION OF ROBOT SKILLS

6.1 Outline of the Chapter

This chapter presents the reproduction of the generated task models by a humanoid robot platform operating under task constraints. The robot reproduction of skills follows the framework presented in Chapter 2, employing the systems developed for learning robot skills in Chapter 3, the representation of robot skill in a knowledge base in Chapter 4, and the generation and adaptation of robot skills in Chapter 5. Figure 1.3 shows the framework proposed throughout this work for the robot skills' adaptation of learned models to task constraints. In this chapter the implementation of the various systems developed in the framework for learning and adaptation of skills to task constraints are also presented, see Figure 6.1. Finally, experimental results and analysis validating the framework proposed throughout this work are presented; different evaluation scenarios are described to test the performance of the various modules implemented in our framework and to provide separate validation for the operation of the system for storing and retrieving robot skills from the knowledge base; the system for generating and adapting the robot skills to the constraints of the task, and the evaluation of the complete developed framework. The organization of this chapter is as follows:

- Section 6.2 describes the development of the proposed framework for learning and adaptation of the robot skills and the experimental set up for the validation of the framework. Here, the robotic platform used in this work is presented with a description of its structure, joints and sensor distribution. Also, a description of the evaluation scenarios to test the performance of the framework and the implemented modules is given. To validate the proposed framework and modules, the experiments would be performed over different scenarios.

- Section 6.3, presents the implementation of the learning system. The robot skill learning module collects the learning processes and algorithms used for learning and encoding the models of the skills. The development and operation of the module for learning the robot skills is described in this section.

- Section 6.4, presents the implementation of the knowledge base system. The robot skill knowledge module controls the developed knowledge base for the storing and retrieval of the learned models of the skills. A description of the development and the process for building and navigating the knowledge base is given in this section.

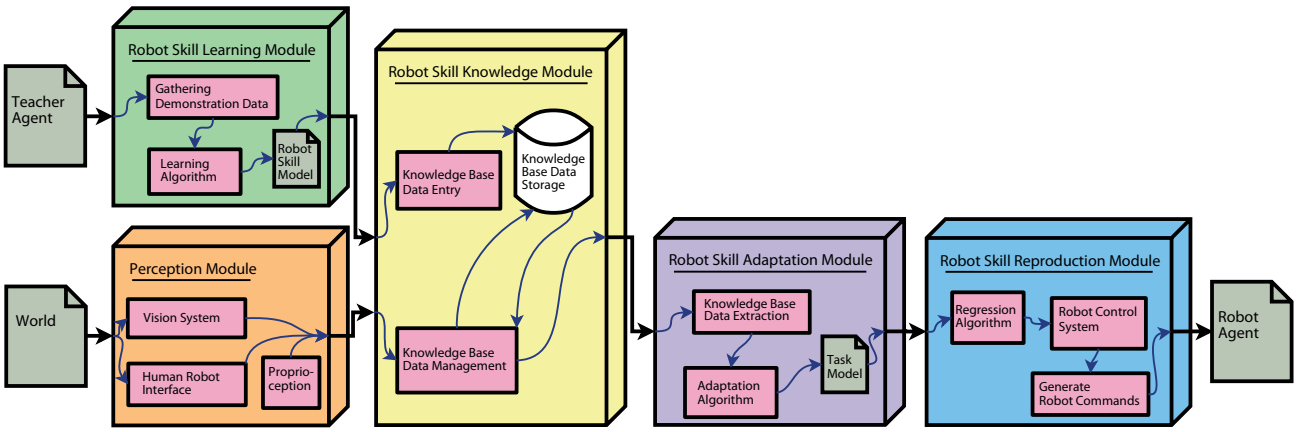


Fig. 6.1: *Deployment diagram for the proposed cognitive framework for learning and adaptation of robot skills. The framework is formed by a robot skill learning module, a perception and interaction module, a robot skill knowledge module, a robot skill generation and adaptation module, and a robot skill reproduction module.*

- Section 6.5, presents the implementation of the task model generation and adaptation system. The robot skill generation and adaptation module governs the process by which the learned model of a skill can be operated to reproduce a new task, including the adaptation, update, merger, combination, or transition of the skill models. The development and operation of the module for adapting robot skills is described in this section.

- Section 6.6, presents the implementation of the reproduction system. The robot skill reproduction module is in charge of producing the adequate control signals to the robot for the reproduction of robot skills. A description of the development and operation of the module for the robot reproduction of skills is given in this section.

- Section 6.7, presents the experimental results and analysis for validation of the proposed framework over the evaluation scenarios described in the previous section of this chapter. Different evaluation scenarios are employed to test the performance of the various modules implemented in our framework. Demonstrations are organized over three major scenarios to provide separate validation for the knowledge base system, the task model generation and adaptation system, and the complete developed framework.

6.2 Development of the Robot Skills Framework

The framework proposed in this thesis is meant to allow the following: for an operator to teach and demonstrate to the robot the motion of a task skill it must reproduce; to build a knowledge base of the learned skills knowledge allowing for its

storage, classification and retrieval; to adapt and generate learned models of a skill to new contexts for compliance with the current task constraints.

The framework proposed here was developed as a cognitive model intended to provide the robot with an essential cognitive ability for learning and adaptation of skills. Though it is not a primary consideration in this work, our framework can be thought of as part of a level in the hierarchy of a more complex architecture, or as a first stepping stone upon which to incrementally build more complex cognitive processes. The goal of the developed framework is to provide a minimum degree of intelligence for the humanoid robot, that is, the ability to sense the environment, learn, and adapt its actions to perform successfully under a set of circumstances.

The framework provides humanoid robots with systems that allow them to continuously learn new skills, represent their skills' knowledge and adapt their existing skills to new contexts, as well as to robustly reproduce new behaviours in a dynamical environment. The cognitive framework for learning and adaptation of robot skills is made up of several modules, as is represented by the diagram on Figure 6.1. The framework is formed by modules for the learning of robot skills, the perception and interaction with the environment, the management and representation of skill knowledge, the generation and adaptation of skill models, and the reproduction of robot skills.

The robot skill learning module collects the learning processes and algorithms used for learning and encoding the models of the skills. The perception and interaction module is in charge of processing the outside information of the robot's working environment to use in the other modules. The robot skill knowledge module controls the developed knowledge base for the storing and retrieval of the learned models of the skills. The robot skill generation and adaptation module govern the process by which the learned model of a skill can be operated to reproduce a new task, whether the adaptation, update, merger, combination, or transition of the skill models. The robot skill reproduction module is in charge of producing the adequate control signals to the robot for the reproduction of robot skills.

Robotic Platform

In order to test the proposed systems the *HOAP-3 Humanoid Robot* was used as a test platform, Figure 6.2. The *HOAP-3* was designed to resemble the human shape, on a small scale, with a complete humanoid configuration with two legs and arms, a head with vision and sound capacities, and grip-able hands. The small humanoid robot *HOAP-3* is about 60 cm in height, and weighs about 8 kg, so that it becomes quite easy to control and move while maintaining the whole stability [Pierro et al., 2009]. The *HOAP* robots were designed for a broad range of applications for Research and Development of robot technologies.

In 2001 Fujitsu produced its first commercial humanoid robot named *HOAP-1*, the first in its series of humanoid robots, *HOAP* stands for "Humanoid for Open Architecture Platform" [Riezenman, 2002], its successor, *HOAP-2* was announced in 2003. It has a height of 48 cm and weight of 6.8 kg. The model used in this work is an evolution from the previous *HOAP* and *HOAP-2* robot family. The *HOAP-3* robot

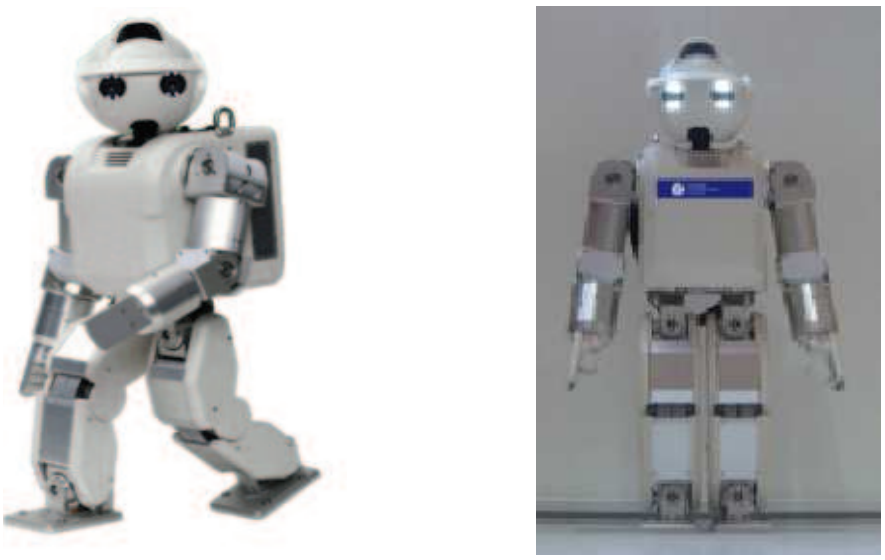


Fig. 6.2: *The HOAP-3 robot platform. HOAP robots were designed for a broad range of applications for Research and Development of robot technologies. HOAP-3 is about 60 cm in height, and weight about 8 kg. It can perform walks on flat terrain, sumo movements and dancing and grasp thin objects.*

was announced in 2005, adding movable axes for the head and hands, CCD cameras, a microphone, a speaker and LEDs to show expression. *HOAP* can successfully perform walks on flat terrain, sumo movements and dancing and grasp thin objects, such as pens, brushes, etc.

The control architecture operates on RT-Linux mounted on a embedded PC-104 computer, Pentium 1.1 GHz processor with 512 Mb of RAM memory and a Compact Flash drive of 1 Gb capacity. The communications with the robot platform could be done via a USB interface or by means of an on board Wi-Fi IEEE802.11g communication. The robot electronics are mounted on the robot's back and protected with a backpack casing. Additionally, a container on the robot's chest allows for a rechargeable 24V NiMH battery to be loaded on to it, the battery pack allows for approximately a 30 min autonomy operation.

The *HOAP-3* structure is made out of a total of 28 degrees of freedom (DOF), powered by DC motors for the legs, waist and arms DOF, and servo motors for the operation of the hands and legs. The distribution of the DOF is as follows:

- 6 DOF for each robot arm, 4 DOF for the arm, 2 DOF for the hand.
- 6 DOF for each leg.
- 3 DOF in the head, for the pitch, yaw, and roll.
- 1 DOF for the waist.

In addition the robot platform sensory system is equipped different sensors endowing the robot with:

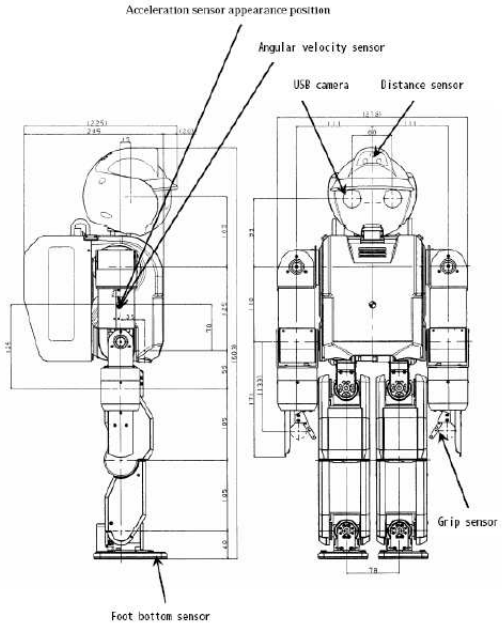


Fig. 6.3: *The HOAP-3 robot dimensions and distribution of joints and sensors capabilities. The HOAP robot is equipped with 28 DOF, additionally it have a gyroscope, an accelerometer, and various force sensors, and two cameras.*

- Posture sensors (a gyroscope sensor and acceleration sensor).
- Contact sensors (force sensor in every corner of each foot).
- Grip sensors (force sensors in the thumbs).
- Vision sensors (Two USB cameras in the head).

Figure 6.3 shows *HOAP-3* robot structure and sensor distribution. Its structure and sensor system allows to try different control architecture, thought to be used in a collaborative system.

Perception System

A perception system was developed for the operation of the *HOAP-3* robot. The perception system consists of a stereo vision system, making use of both robot cameras, and an interaction system, making use of a human-robot interface for high-level communication with the robot.

Since the *HOAP-3* robot platform is equipped with two cameras, stereo vision is used when it is possible, given the disposition and angle of view of cameras. When objects cannot be perceived by both cameras, i.e. inside the workspace area of arms (close to robot), the estimations are made by monocular vision.

Recognition of the objects is based on blob detection by color filtering and area comparison. Objects are recognized based on their color properties and blob size.

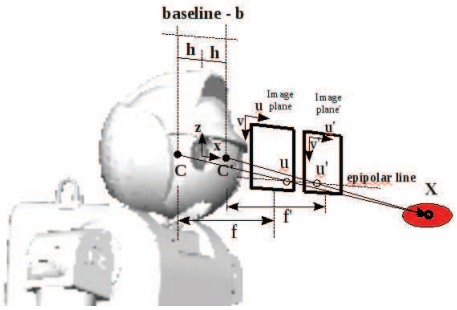


Fig. 6.4: *Stereo vision with cameras disposed in rectified configuration.*

First of all the images are filtered by colors. The color segmentation method consists of selecting a prism for each channel in the HSV color domain. The pixels are color labelled and similar regions are grouped into blobs. Sanity checks are applied to every blob to avoid wrong detections and correspondences. For example, the dimension of every blob is checked and when they are too small, such blobs are rejected. Another important sanity check is the horizon view, which consists of calibrating maximum height of objects in the camera plane. When blobs are not in the feasible zone for grasping, they are filtered out. When the blobs satisfy sanity checks and match with the color properties of some object, they are considered as the detection of an object.

Because the humanoid platform is equipped with two cameras emulating human eyes, these inputs can be used for estimating depth information. The typical steps for determining depth using a two-camera vision system, stereopsis, are i) calibration of cameras, ii) establishment of correspondences between features of both cameras and iii) reconstruction of 3D coordinates of detections in the scene. The basis of stereopsis is epipolar geometry, which states that the line connecting optical centres of both cameras, baseline, intersects the image planes in the epipoles. A simplified case of stereopsis is the rectified configuration of cameras, which reduces the dimensionality of search space for a correspondence from 2D to 1D. This configuration consists of both image planes being parallel, and hence, the baseline also being parallel to image planes, sending the epipoles to infinity. In addition, epipolar lines of all possible detections coincide with the images' rows, and correspondences between detection of both images can be found by matching pixels linewise. Considering the rectified configuration, depth can be recovered by using the notion of disparity, Figure 6.4. The stereo vision system implemented in the robot uses the weighted center of the color labelled blobs. The epipolar line is stated as the weight average of blobs using the number of pixels of the blobs as the weight factor.

The human-robot interface is user friendly and it gives an intuitive way for a non-expert user to interact with the humanoid robot HOAP-3. Main functionalities in the HRI user interface are the graphical control component, allowing the operator to move the robot in several directions at different speeds, rotate it and stop it. And the high-level button controls which allow the operator to request the robot to perform several high-level actions. The overall performance of the algorithms in the

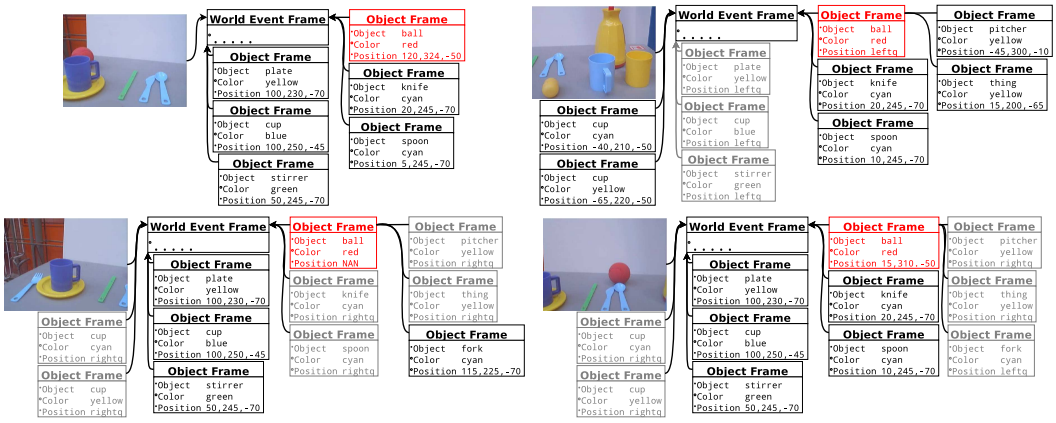


Fig. 6.5: *Description of experiment A.1 in the knowledge base scenario. (upper-left) World and object frames are instantiate from the robot view. (upper-right) As it explores the scene new objects are added, and information for existing ones are updated. Out of view objects (grayed out) remain in the knowledge base and their position are changed to a relative value reflecting their expected location. (bottom-left) Robot is tasked with finding the red ball, and defaults to looking for it starting in its last known location. (bottom-right) Robot continues its search looking for the red ball until it is found in a new position or it is not found and the object instance is removed from the knowledge base.*

perception system is not an element of this thesis, and the selection of the perception system components was made on the criteria of availability and easy integration with the rest of the framework. For further explanations of the perception system see [Pierro et al., 2012a].

Description of Knowledge Base Scenario

Here we provide a general description of a demonstrator for the evaluation of the knowledge base scenario performance. Quantitative evaluation of knowledge processing systems is hardly possible since many of its features are difficult to reflect in numbers. However the system can be evaluated in a qualitative form. Several experiments were conducted to prove the validity of the system and to test the operation of the developed knowledge representation and the knowledge base module.

A first experiment involves the *HOAP-3* robot operating in a kitchen setting. The *HOAP-3* robot would stand in front with a top view, from the cameras in its head, of an assortment of objects commonly expected in a kitchen or dinner scene, i.e., cups, plates, forks, knifes, etc., see Figure 6.5. The available objects come from a toy set and were chosen so their size and shape can fit properly with *HOAP-3* robot structure; also, the available object present bright, solid colors facilitating the recognition of objects by their color properties in the perception system.

This experiment would go as follows, the *HOAP-3* robot would look around the

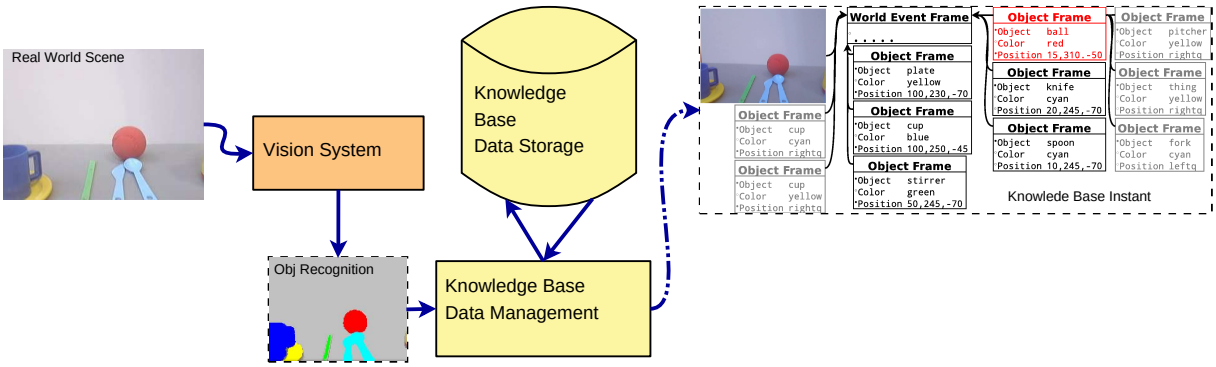


Fig. 6.6: *Schematic view of the Knowledge Base Scenario experiment A.1.*

scene as the perception system instantiates object frames from the recognized objects in its way, and modify and upkeep its world event frame in the knowledge base. The objects in the environment would be taken in and out the robot field of view or moved around the scene by an agent; also the robot would be asked to focus on different objects throughout the demonstration as its task directive is changed by the agent. The purpose of this scenario is to prove the performance of the knowledge representation and developed knowledge base to maintain relevant information in the knowledge base in a dynamic environment with changing world and task events during its operation.

This demonstrator highlights the operation and interaction of the perception and knowledge systems to instantiate the different frames in the knowledge base and build the active view event frame out of the extracted knowledge and constraints from the current task and world events. There are 2 modules involved for this scenario:

- The perception system: for detection and tracking of objects in the table.
- The knowledge base system: for instantiating the different frames in the knowledge base accordingly.

Figure 6.6 shows a schematic view of the overall knowledge base scenario experiment described above. The perception system, through the vision system, is in charge of analysing the environment of the robot as captured with the robot cameras, recognizing objects that are present and computing their location. The knowledge base system would receive this information from the perception system and would instantiate object frames from the recognized objects present, and build the knowledge representation of the scene in the knowledge base. As the robot moves around, the environment or the scene is changed by adding, moving, or removing objects the contents on the knowledge base are updated.

This scenario is meant to provide proof of concept of how frames in the knowledge base are instantiated from the perception of the environment and how the knowledge base maintains and upkeeps its knowledge representation over time in a changing environment. Further development of this scenario would add more functionality in

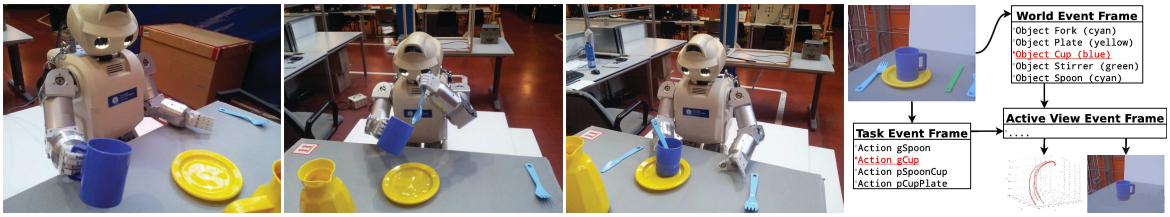


Fig. 6.7: *Description of experiment A.2 in the knowledge base scenario. At the beginning the task can be started by either picking the cup or the spoon. (left) The HOAP-3 robot starts the task by grasping the blue cup. (center-left) With the cup and the spoon in its hands the robot performs the action skill to put the spoon inside the cup. (center-right) Finally, the HOAP-3 robot places the blue cup on the saucer plate. (right) State of the knowledge base at some step during execution. From the Task and World Events the Active View Event is built to drive the action execution.*

the following subsections to highlight the operation of other systems in the developed framework.

A second experiment would have an agent and the *HOAP-3* robot interacting to complete a simple task. The task in this case requires the robot to pick up a cup and a spoon in each hand and then to put the spoon inside the cup; then finally it would put down the cup in front of it. The agent would provide the robot with the cup and spoon objects so it can pick them up; also the agent would indicate to the robot where to put down the cup, see Figure 6.7.

Execution of the demonstration could vary depending on the actions of both the human agent and the *HOAP-3* robot. At the start of the demonstration the robot is given the task event frame knowledge for the desired behaviour containing the knowledge of the 4 action skills needed to complete the action, pick spoon, pick cup, place spoon in cup, place cup down. Extracting the adequate action would depend on the agent interaction and the content of the rest of the knowledge base. The purpose of this demonstration is to validate the performance of the developed knowledge base in a dynamic interaction with an agent where the invocation of an action skill is controlled by the representations in the knowledge base as described in Chapter 4.

This demonstration highlights the operation of the knowledge base and how the representations of object, action, task event, world event and active view event frames are used to command the robot execution of the desired task. There are 4 modules involved in the operation of this scenario:

- The perception system: for detection and tracking of the objects.
- The learning system: for teaching the robot a set of robot skills.
- The knowledge base system: for representing the object, action, task event, world event and active view event frames, used to command the robot execution of the desired task.

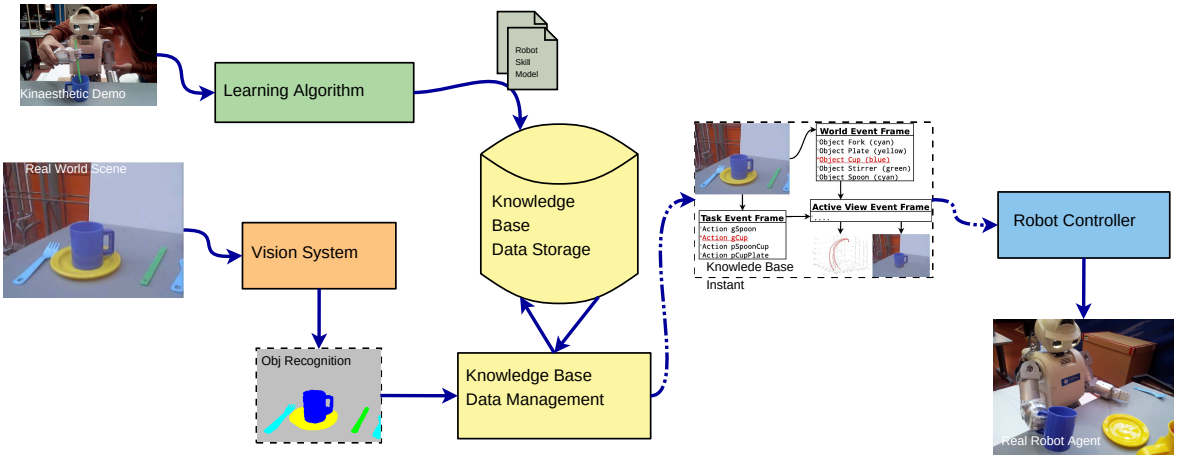


Fig. 6.8: *Schematic view of the Knowledge Base Scenario experiment A.2.*

- The robot reproduction system: for controlling the robot execution.

Figure 6.8 shows a schematic view of the overall knowledge base scenario experiment described above. The perception system handles the interaction with the user and the detection of objects in the environment. The knowledge base system would receive this information from the perception system and would instantiate the frames and built the knowledge representation of the scene in the knowledge base. The knowledge base system would select and activate an action skill when the conditions in the knowledge representation afford such action. Once an action is selected, the *HOAP-3* robot controller would execute the robot commands required for the skill reproduction.

This demonstrator scenario is meant to provide proof of how action execution is invoked by the state of the representation frames present in the knowledge base. Further development of this scenario would add more functionality in the following subsections to highlight the operation of other systems in the developed framework.

Description of Skill Generation and Adaptation Scenario

In this subsection, a general description of a demonstrator for the evaluation of the performance of the robot skill generation and adaptation scenario is provided. To this end, several experiments were conducted to prove the validity of the system and to test the operation of the developed robot skill generation and adaptation module. Experiments were designed to test the performance of the different robot skills' operations described in Chapter 5. The demonstrators in this subsection were chosen as very simple scenarios in which to have proving ground in which to test different robot skills and skill generation and adaptation mechanisms.

As a first scenario, we'll considerer a table tennis robot task. In this setting the *HOAP-3* humanoid robot would stand equipped with a table tennis paddle, from a table tennis toy set, of an appropriate size and handle shape to fit the *HOAP* frame and the grasp capabilities of its hands, see Figure 6.9.

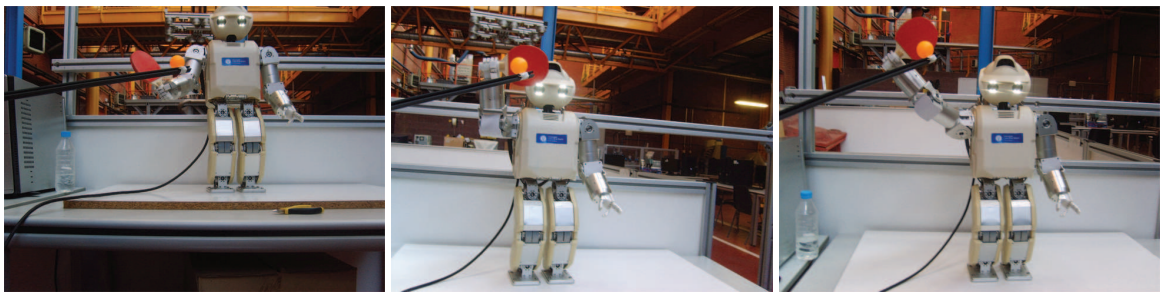


Fig. 6.9: *Description of experiment B.1 in skill generation and adaptation scenario. HOAP-3 robot performs different tennis shots: (left) HOAP-3 robot performs a forehand shot. (center) HOAP-3 robot performs a smash shot. (right) HOAP-3 robot performs a forehand-smash shot generated from the merger of the forehand and smash shots.*

The limitations of the robot systems and vision tracking do not allow for real time reproduction of a robot table tennis game, however that is not the intended goal of the demonstrator. The focus in this experiment would be on the learned *Robot Skill Models* and on the operation of the algorithm in Chapter 5 for the merger of robot skills in order to generate new, more complex, skills given the robot's additional action for performing tennis shots from the ones that are previously learned by the robot.

The *HOAP* robot is required to execute different tennis shot actions to hit a table tennis ball moved towards the robot. Originally 3 robot skills' models are taught to the *HOAP-3* robot to hit an approaching ball coming from its left, right, or above, to provide the robot with the skills to perform a forehand, a backhand, or a straight smash shot. To expand the robot skill set, two learned skill models are merged to obtain a new *Robot Skill Model*. In this case a forehand and a backhand will be merged with the smash shot skill to generate two more skills for forehand-smash and backhand-smash shots.

This demonstrator highlights the operation of the proposed algorithms in Chapter 5 for the merging of *Robot Skills Models*. There are 4 modules involved in the operation of this scenario:

- The perception system: for the detection and tracking of the table tennis ball's position.
- The learning system: for teaching the robot a set of skills for reproducing different tennis shots.
- The generation and adaptation system: for generating new robot skills from the merger of two learned models of a skill.
- The robot reproduction system: for controlling the robot skill execution of a tennis shot.

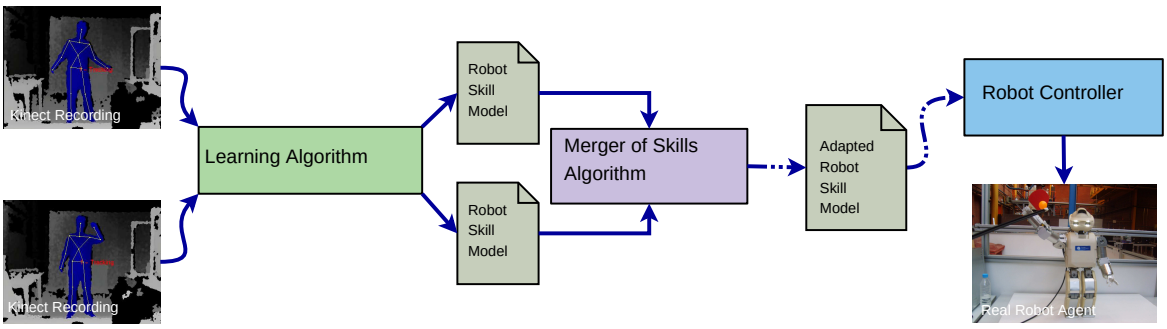


Fig. 6.10: *Schematic view for experiment B.1 in the robot skill generation and adaptation scenario.*

Figure 6.10 shows a schematic view of the overall generation and adaptation scenario experiment described above. This scenario is a demonstrator for the merger of robot skills presented in Chapter 5. First, demonstrations are given, recorded with a Microsoft Kinect sensor, to the learning module to encode the models of the robot skills for the forehand and smash tennis shots. Then the learned robot skills are fed to the skill merger algorithm to generate a new robot skill model for a forehand-smash shot from the merger of the two previous skills. Finally, the model of the skill is given to the *HOAP-3* robot controller for the execution of the skill.

This demonstrator scenario is meant to provide proof of concept of how the generation and adaptation system can operate over previously learned robot skills for generating new, more complex, skill actions and for increasing the scope of operation in the given available skills to expand the range of task which can be performed by the *HOAP-3* humanoid robot. Further development of this scenario would strive to bring more functionality in the following subsections to highlight the operation of other systems in the developed framework.

A second demonstrator was designed to test the performance and evaluate the developed methods for the update and combination of *Robot Skill Models* as described in Chapter 5. To validate the proposed methods for generating new skills from previously learned models, by updating or combining the *Robot Skills Models* a very simple scenario was chosen in which the robot would be required to grasp a plastic cup, from the kitchen toy set used in the previous scenario, see Figure 6.11.

The contemplated task requires that the robot be able to grasp the plastic cup located in any possible place in a "cupboard", which consists of two shelves, a bottom and a top shelf. The *HOAP-3* robot must be able to grasp the cup, as long as it is inside the robot arm's workspace, in any of six possible general locations in relation to the robot arm; three on the bottom shelf and three on the top shelf, for example, a cup could be placed at the left-bottom, right-bottom, center-bottom, or left-top, right-top, center-top, of the robot. Initially, only the skills for learning to grasp the cup placed on the bottom shelf are taught to the robot by the methods described in Chapter 3. The complete task would be unachievable with the robot skills learned so far, since the skill reproduction would not generalize well to the target's new position

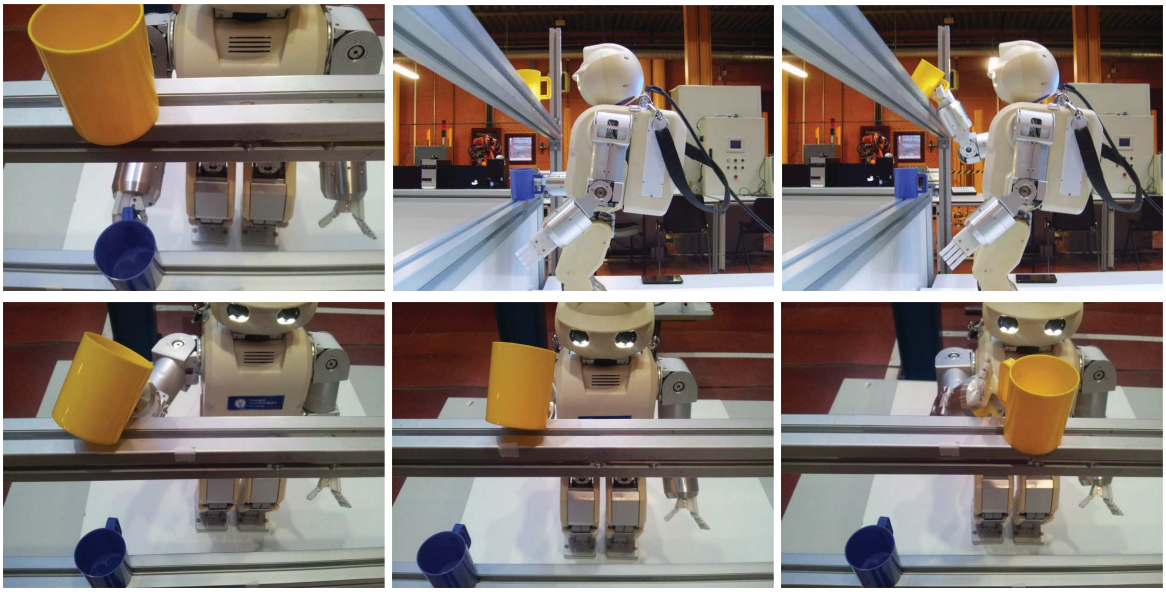


Fig. 6.11: *Description of experiment B.2 in skill generation and adaptation scenario. The HOAP-3 robot must grasp a cup placed at any position in either of the two shelves of the "cupboard" scenario. Initially only skills for grasping the cup in the bottom shelf are taught to the robot. To generalize the skill to the target's new position at the top shelf the skills learned to grasp the cup at the bottom shelf must be updated. To generalize across the whole working space the three models of the robot skill, for right-, left- and center-, are combined into a single model of the attractor dynamics. (top-row) Robot is taught a grasp skill motion for the cup place in the bottom shelf. By the update of the robot skill a new model is generate to allow the HOAP-3 robot to grasp the cup place at the top shelf. (bottom-row) By the combination of various robot skills the HOAP-3 robot can grasp the cup place at its right, center or left, using a single model of the skill.*

on the top shelf. To grasp the cup, placed on the top shelf, at either side of the robot the skills learned to grasp the cup on the bottom shelf must be updated to generate the required new robot skill models. Finally, to generalize across the whole working space the three models of the robot skill, for right-, left- and center-, are combined into a single model of the attractor dynamics. Figure 6.11 illustrates the scenario.

This demonstrator highlights the operation of the proposed algorithms in Chapter 5 for the update and combination of *Robot Skills Models*. There are 3 modules involved in the operation of this scenario:

- The learning system: for teaching the robot a set of skills for reproducing different grasp actions.

- The generation and adaptation system: for generating new robot skills from the

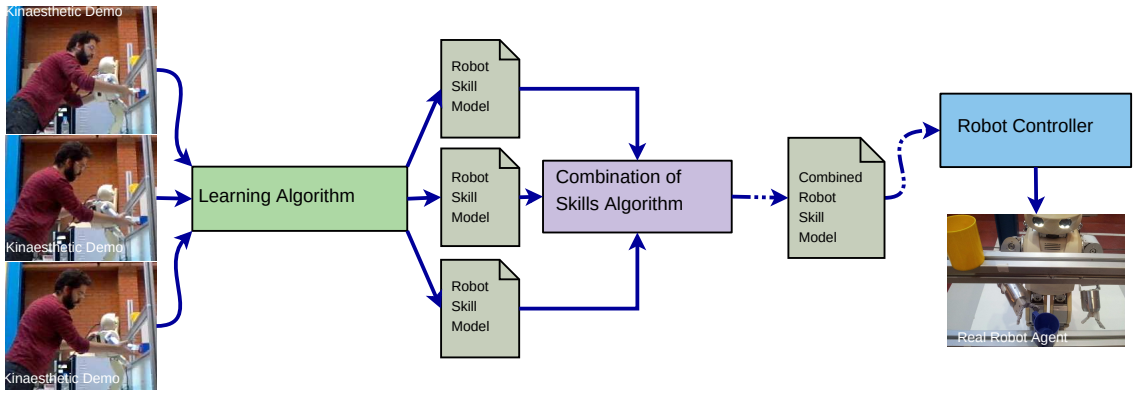


Fig. 6.12: *Schematic view for experiment B.2 in the robot skill generation and adaptation scenario.*

update of the learned models of a skill and the combination of different learned skill models into a single model of the attractor dynamics.

- The robot reproduction system: for controlling the *HOAP-3* execution of the robot skills.

Figure 6.12 shows a schematic view of the overall generation and adaptation scenario experiment described above. This scenario is a demonstrator for the combination of the robot skills method presented in Chapter 5. First demonstrations are given, recorded by kinaesthetic teaching, to the learning module to encode the models of the robot skills for grasping trajectories at the possible locations, left, right and in front of the robot. Then the learned robot skills are fed to the skill combination algorithm to generate a new robot skill model from the combination of the previous skills. Finally, the model of the skill is given to the *HOAP-3* robot controller for execution of the skill.

This demonstrator scenario is meant to provide proof of how the generation and adaptation system can operate over learned robot skills for increasing the scope of available skills for the performance of the *HOAP* humanoid robot. Further development of this scenario would add more functionality in the following subsections to highlight the operation of other systems in the developed framework.

Description of Robot Skill Reproduction Scenario

As a final evaluation a couple of general demonstrators' scenarios were implemented for the validation of the robot skill reproduction and to test the complete developed framework for the learning and adaptation of robot skills. Several experiments were conducted to prove the validity of the system and to test the operation of the developed framework. Experiments were designed, requiring the humanoid robot to reproduce different *Robot Skill Models* throughout the unfolding of the task in order to test the performance of the overall system and the operation and interaction

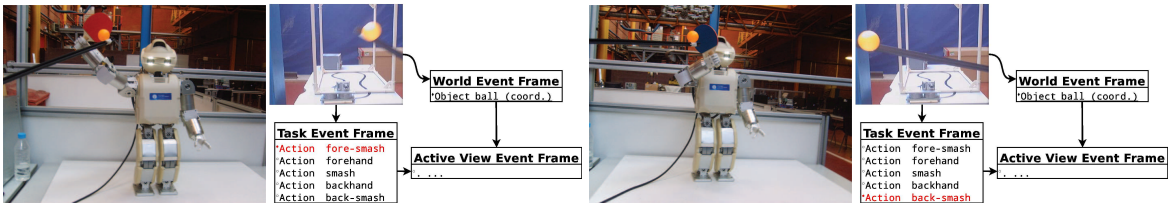


Fig. 6.13: *Description of experiment C.1 in the robot skill reproduction scenario. HOAP-3 robot performs different tennis shots, as learned and generated from the previous experiment. (left) HOAP-3 robot performs the forehand smash shot. (right) HOAP-3 robot performs a backhand-smash shot. The knowledge base selects the appropriate action to execute the proper tennis shot out of the instantiated frames knowledge.*

of the different modules in the framework for learning skills, representing knowledge, generation and adaptation of models and robot skill reproduction.

As a first scenario we'll review the table tennis robot task described in the skill generation and adaptation scenario. The setting is the same as before with the *HOAP-3* humanoid robot equipped with a table tennis paddle, and a set of learned robot skills to perform different tennis shots, namely a backhand, a forehand, and a smash shot, plus the generated merged forehand-smash and backhand-smash shots. The purpose of this scenario is to prove the viability of the developed representations and knowledge base system in Chapter 4 for selecting the appropriate robot skills for a tennis shot to hit the table tennis ball from its available action frames and perceived world state knowledge.

With the *HOAP* robot, paddle in hand, in a resting position, the perception system detects a table tennis ball that is moved towards the robot. The system computes the relevant information from recognition of the ball, and extracts from the knowledge base, the appropriate learned robot skill models to reproduce the action to hit the ball under the current circumstances, see Figure 6.13.

This demonstrator highlights the operation of the perception system and the knowledge base system to instantiate the proper frames in the knowledge base and extract from this information the needed *Robot Skills Models*. Additionally, it is meant to highlight as well, the operation of the systems for generation and adaptation and for robot reproduction of the *Robot Skills Models*. There are 5 modules involved in the operation of this scenario:

- The perception system: for detection and tracking of the table tennis ball.
- The learning system: for teaching the robot a set of skills for reproducing different tennis shots.
- The knowledge base system: for selecting the appropriate robot skills for a tennis shot out of the instantiated frames knowledge.

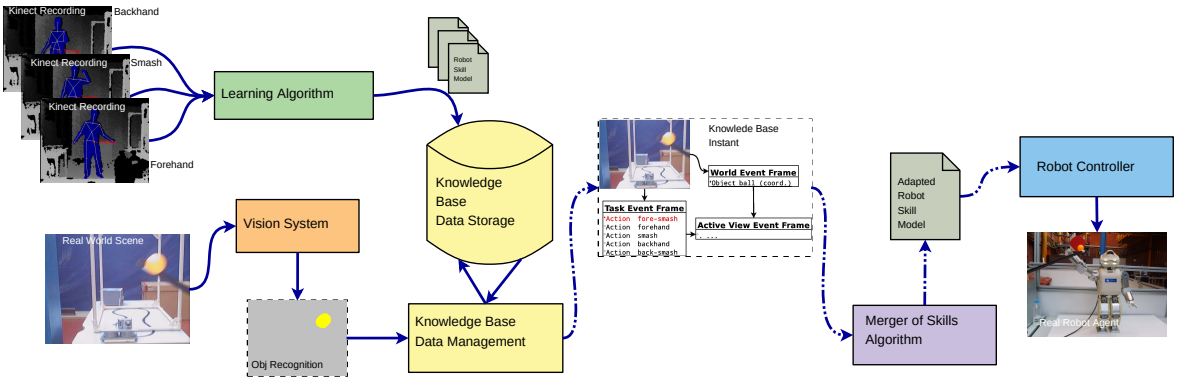


Fig. 6.14: *Schematic view for experiment C.1 in the robot skill reproduction scenario.*

- The generation and adaptation system: for generating new robot skills from the learned models of a skill.

- The robot reproduction system: for executing the robot skill for an appropriate tennis shot.

Figure 6.14 shows a schematic view of the overall skill reproduction scenario experiment described above. This scenario is a demonstrator for the evaluation of the robot skill reproduction, and the operation of the complete developed framework, with the main focus on the performance of the knowledge base system to extract from its information, the needed skill models for the robot's successful execution. For this scenario, demonstrations are first given to the learning module, recorded with a Microsoft Kinect sensor, to encode the models of the robot skills for three tennis shots, forehand, backhand, and a smash. Subsequently, the learned robot skills are stored by the knowledge base system. During operation, a table tennis ball will move towards the *HOAP-3* robot, with the perception system and the knowledge base system, the information of the position and trajectory of the ball is used to recover and select the needed robot skills for action reproduction and perform the proper tennis shot skill in the current situation.

This demonstrator scenario is meant to provide proof of concept of how the knowledge base system recovers and selects robot skills for action reproduction based on the instantiated knowledge frames, stored and represented by the developed knowledge base. Together with the previous experiment, evaluating the generation and adaptation system, the proposed demonstrator validates the performance of the developed framework to learn, store and adapt the robot skill for executing different actions, complying with the task constraints, with the *HOAP-3* humanoid robot.

As a final experiment we'll revisit the kitchen or dinner table scenario and expand the demonstrators presented in the previous sections. In this scenario the *HOAP-3* robot is required to complete a setting up a dinner service task behaviour in conjunction with a human agent. The purpose of the demonstrator is to test the overall

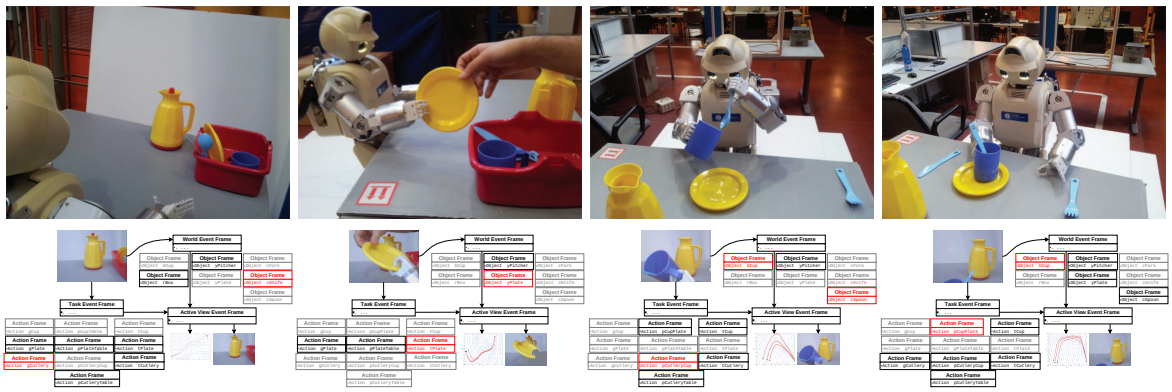


Fig. 6.15: *Description of experiment C.2 in the robot skill reproduction scenario. HOAP-3 robot sets a "dinner service" consisting of a fork, a knife, a saucer plate, a cup and a spoon. (top-row) Still captures from the HOAP-3 robot performing in the robot skill reproduction scenario. (bottom-row) State of the knowledge base at the execution step from the top-row. Objects and actions not in the Active View Event are grayed out. Objects and Actions key to the current robot reproduction are highlighted in red.*

operation of the developed framework, as well as to validate the performance of every individual module and interaction between themselves.

The sequence of execution of the task could vary depending on the actions of both the human agent and the *HOAP-3* robot. The plan for the demonstrator requires the robot to set up a "dinner service" consisting of a fork, a knife, a saucer plate, a cup and a spoon, see Figure 6.15. Robot skills to grasp the different object are taught to the robot by the methods described in Chapter 3. Execution of the task is instigated by the agent when putting on the table a yellow pitcher object. The robot would set the rest of the objects on the table, their positions in relation to the pivot pitcher object. The objects to place are provided to the robot by the agent, and could be in any possible place, therefore the learned *Robot Skill Models* must also be updated, merged, and combined by the methods described in Chapter 5 as in the scenario in the previous section. The invocation of robot action skills is controlled by the representations in the knowledge base described in Chapter 4 as in the scenario in previous sections.

This demonstrator highlights the operation of the individual modules as well as the overall performance of the overall framework for learning and adaptation of skills to task constraints; involving the perception of objects and interaction with the agent, the learning of various robot skills, the representation of knowledge in the knowledge base, the generation and adaptation of the skill models and the adequate reproduction of the robot skills. There are 5 modules involved in the operation of this scenario:

- The perception system: for detection the objects involved in the task.
- The learning system: for teaching the robot a set of robot skills.

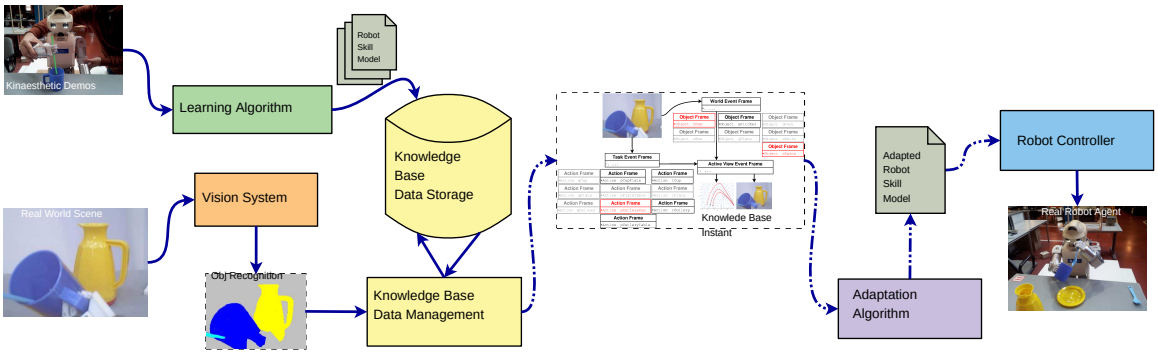


Fig. 6.16: *Schematic view for experiment C.2 in the robot skill reproduction scenario.*

- The knowledge base system: for representing the object, action, task event, world event and active view event frames used to control the robot execution.

- The generation and adaptation system: for the update and merger of learned models of a skill and the combination of different learned skill models into single models of the attractor dynamics.

- The robot reproduction system: for executing the robot skill for an appropriate completion of the task.

Figure 6.16 shows a schematic view of the overall skill reproduction scenario experiment described above. This scenario is a demonstrator for the evaluation of the robot skill reproduction and the overall operation of the complete developed framework; involving the usage of the perception, the learning, the knowledge base, the adaptation, and the reproduction systems. For this scenario, various demonstrations of skills, recorded with the *HOAP-3* robot, are first given to the learning module, to encode the models of the robot skills for the different actions required for the "dinner service" task. Subsequently the learned robot skills are stored by the knowledge base system. During operation, the user would provide objects to the robot, by placing them in its action field, both of vision and manipulation. The perception system would handle the interaction with the user and the detection of objects in the environment. The knowledge base system would receive this information from the perception system and would instantiate the frames and build the knowledge representation of the scene in the knowledge base. Through this interaction with the user and the environment, the knowledge base system would select the corresponding skills to activate them as the conditions in the knowledge representation afforded such actions. Once the necessary robot skills are selected, the generation and adaptation system would be in charge of building the appropriate task model satisfying, the desired command and constraints of the environment for reproducing the appropriate skill action. Finally the *HOAP-3* robot controller would execute the robot commands required for skill reproduction.

This demonstrator scenario is meant to provide proof of concept of how the knowledge base operates to instantiated frames from the perception of the environment, and how the knowledge base maintains and upkeeps its knowledge representation over time in a changing environment, as well as how action execution is invoked by the state of the representation frames present in the knowledge base. Additionally, the demonstrator scenario provides validation for the generation and adaptation system and how it operates over learned robot skills for increasing the scope of available skills for the performance of the *HOAP* humanoid robot.

6.3 Learning the Robot Skills

The capability to learn and teach a robot the necessary robot skills is clearly a crucial part of the developed framework. Therefore the robot skill learning module has a central importance in our framework. In Chapter 3 the methods employed for learning the models of a robot skill have been described. In this section the development and operation of the robot skill learning module will be presented.

Humanoid robots working alongside humans must deal with continuously changing environments and a huge variability of tasks; therefore, algorithms for learning and extracting important features of task actions are fundamental. The robot skill learning module is naturally responsible for allowing the humanoid robot to learn the models of robot skills. This requires the module to provide the mechanism needed for gathering the demonstration data from a teacher agent and for encoding the motions into a model of the robot skill. The robot skill learning module collects motion data from demonstrations, processes it and builds the demonstration data set that feeds the learning algorithms. The *SEDS* algorithm is employed to learn an estimate of the motion through a set of first order non-linear multivariate dynamical systems in a statistical approach. Figure 3.8 illustrates the control flow for the operation of the robot skill learning module. The learning systems are required to acquire skills and developed task knowledge of how to act in order to provide a robot with a sufficient number of skills that permit it to perform autonomously in an unstructured environment.

The robot skill learning module collects the learning processes and algorithms used for learning and encoding the models of the skills. There are three subsystems in this module; a subsystem for gathering demonstration data; a subsystem for building an estimate of the demonstration with the learning algorithm; and a subsystem for encoding the robot skill model. Figure 6.17 shows the deployment diagram for the robot skill learning module.

The subsystem for gathering demonstration data is made up of three processes. At first a teacher agent input data is collected, Chapter 3 presented different modalities from which the teacher demonstration could be collected. Secondly, a preprocessing step is performed to transform the collected data to ensure correspondence with the robot system. A final third step process the raw data from the previous step to build the demonstration data set as required to feed the learning algorithm. The operation of the subsystem for gathering demonstration data is handled by an external proces-

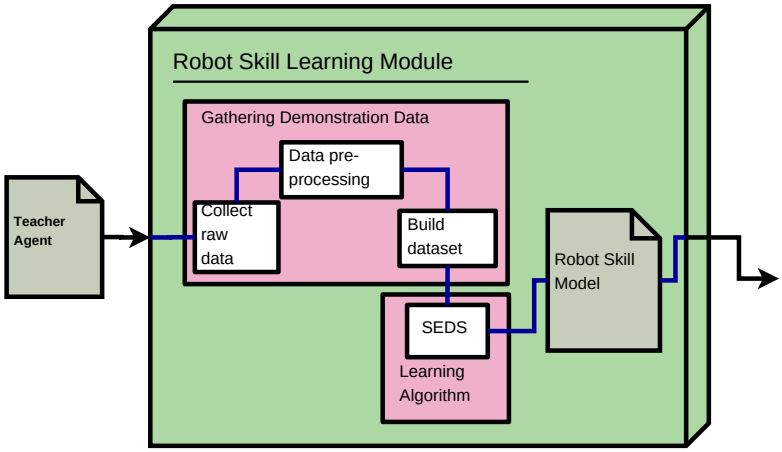


Fig. 6.17: *Deployment diagram for the robot skill learning module. There exist three subsystems in this module, a subsystem for gathering demonstration data, a subsystem for building an estimate of the demonstration with the learning algorithm, and a subsystem for encoding the robot skill model.*

sor with different implementations for the recording of the teacher demonstrations. Three modalities were presented in Chapter 3. For the teleoperation of the robot by means of kinaesthetic teaching, the robot encoders are used and each joint motion is recorded at a rate of $1000Hz$, and saved in an appropriate file, storing the data for a given demonstration; these are then re-sampled to a fixed number of points to process the raw data into the required demonstrations dataset. For the OpenRave simulated environment, the process goes as before, but a simulated model of the robot is implemented and it is used instead of the real robot sensor. For the recording of visual demonstrations, a Microsoft Kinect sensor is used, a software system was implemented to make use of the skeleton tracking capabilities provided by the OpenNI api, the motions of a teacher in front of the sensors are recorded, later the teacher recorded joints are transformed to match the corresponding robot joints.

The learning algorithm subsystem handles the learning of the robot skill as described in Chapter 3; the algorithm for the building of the demonstration with *SEDS* can be found in Table 3.3. The subsystem for encoding the robot skill model is in charge of preparing and expressing the learned estimates of the motions as *Robot Skill Models* for the rest of the framework. The learning algorithm process is carried out off-line. The implemented system is derived from the *SEDS* library provided by [Khansari-Zadeh and Billard, 2011]. The file with the recorded demonstration data is previously provided by the subsystem for gathering demonstration data. A first pre-processing step is carried out to build the adequate dataset needed by the algorithm. The MATLAB numerical computing environment is used for the implementation of the learning algorithm subsystem in our framework, implementing the *GMM*, *GMR*, and *SEDS* algorithms. A model is obtained with the θ parameters encoding the robot motion dynamics. In the final step a file is outputted, storing the learned *Robot Skill Models*.

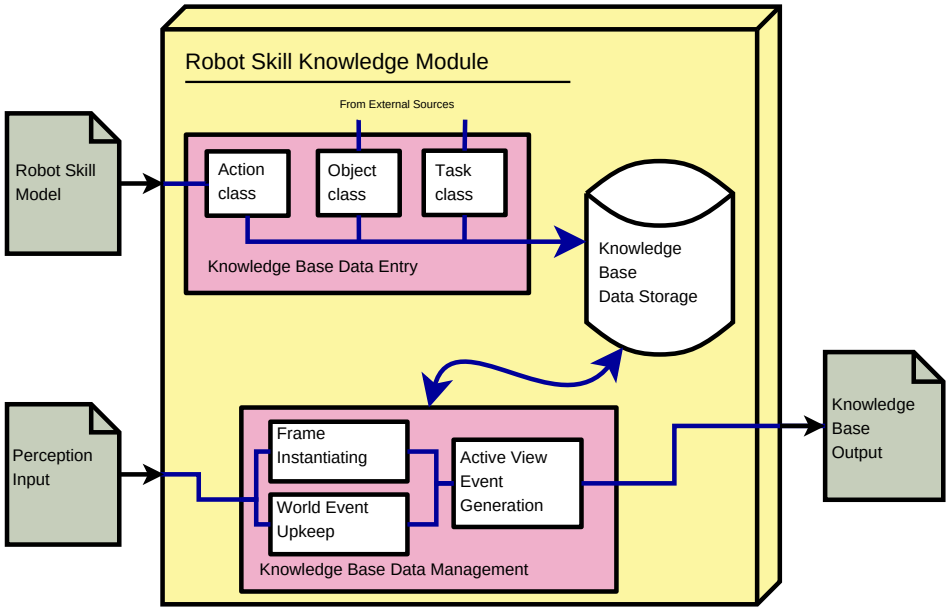


Fig. 6.18: *Deployment diagram for the robot skill knowledge module. There exist three subsystems in this module, a subsystem for the data entry to the knowledge base, a subsystem for the knowledge base data storage, and a subsystem for the knowledge base data management.*

6.4 Navigating the Robot Skill Knowledge Base

In Chapter 4 the knowledge base for the storing and retrieval of the learned models of the skills was described. In this section the development and the process for building and navigating the knowledge base is given. For a robotic system to perform different skills and tasks in a changing and unstructured scenario, it is important to have mechanisms to organize the acquired knowledge in a manner that allows it to be retrieved in order to use it to drive its actions. The robot skill knowledge module is in charge of managing the knowledge base and the processing of the knowledge represented within it. This requires the module to provide the mechanism by which acquired knowledge about objects, actions and events of the task and the state of the world is represented in the knowledge base, and also how this knowledge is operated to extract from it necessary information for the robot's successful completion of its tasks. Figure 4.8 illustrates the control flow for the operation of the robot skill knowledge module. Developing appropriate structures in which to organize the acquired knowledge, to allow the retrieval of it to use it in fulfilling the system goals is key if humanoid robots are to be capable and flexible enough to handle the challenges of working alongside humans in complex natural environments.

The robot skill knowledge module governs the operation of the knowledge base and the instantiation and maintenance of the different frames in the developed knowledge representational structure. Task and World Event Frames are instantiated, from the information provided by the perception module, and the Active View Event Frame

is built from them with the constraints of the task. There are three subsystems in this module, a subsystem for the data entry to the knowledge base, a subsystem for the knowledge base data storage, and a subsystem for the knowledge base data management. Figure 6.18 shows the deployment diagram for the robot skill knowledge module.

The knowledge base data entry subsystem works as a middleware between the knowledge base data storage subsystem and the robot skill learning module for uploading robot skills models and action, object and task classes for storage into the knowledge base. The knowledge base holds all necessary information for reproduction of the skills in the environment; knowledge of the task would be distributed among the representation of objects, actions and events of the task. Operations of the knowledge base data entry subsystem are made off-line. Entries into the knowledge base are made to stored the needed frames for the task. *Robot Skill Models* are generated as explained in the Robot Skill Learning Module and stored in the knowledge base. The objects and task frames entries are made beforehand by a human operator to ensure the appropriate knowledge for the task execution is stored in the knowledge base. Some approaches exist for on-line autonomous generations of this knowledge's data structures, such as in the RoboEarth project [Waibel et al., 2011], which could be studied for future implementation.

The knowledge base data storage subsystem works as a database collecting and organizing the robot skill knowledge as per the representational structure discussed in Chapter 4. Entries in the knowledge base are implemented using the *XML* markup language, following the structure and tag labels as necessary for the different knowledge frames as presented in Chapter 4. The physical implementation of the knowledge base is on an accompanying PC outside of the robot main system. Communications with the robot on-board computer are carried out using a WLAN network.

The knowledge base data management subsystem is at the heart of the robot skill knowledge module. The knowledge base data management subsystem handles the operation and performance of the knowledge base, presented in Chapter 4; the knowledge of the environment and goals taken form the perception module is represented in terms of the World Event Frame and Task Event Frame, with Object and Action Frames representing knowledge about available objects and actions respectively. From the knowledge of these frames an Active View Event Frame of the focused knowledge promoting the agent's execution is built. Looking up the knowledge base storage for the given object and action affordance frames yields the needed models of the skill, $\bar{\mathcal{M}}_{RS}$, required by the module for its operation. In the knowledge base data management subsystem, search and reasoning operations over the stored knowledge are carried out. The implementation of the knowledge base data management subsystem was made using SWI-Prolog and the Python high-level programming language. A YARP layer was implemented for the communications between the robot skill knowledge module and the rest of the systems.

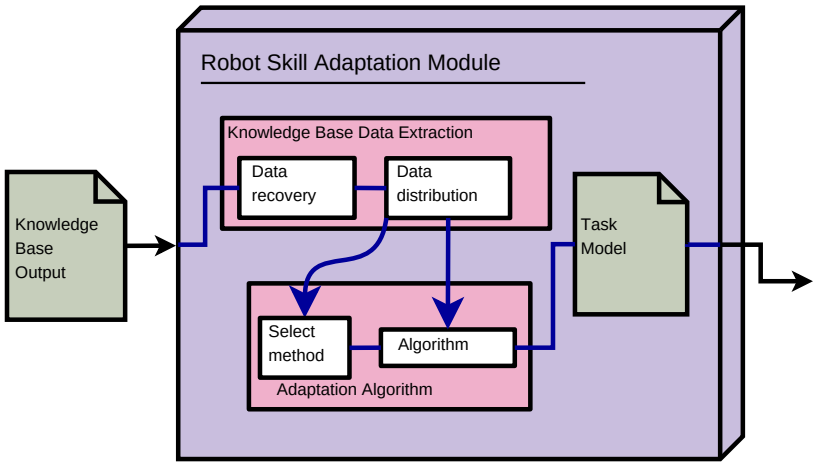


Fig. 6.19: *Deployment diagram for the robot skill generation and adaptation module. There exist three subsystems in this module, a subsystem for extracting data from the knowledge base, a subsystem for operating upon the robot skill with the adaptation algorithm, and a subsystem for generating the task models.*

6.5 Generating the Robot Skills Task Models

The robot skill generation and adaptation module is a vital part of the developed framework. In Chapter 5, the algorithms developed for the generation and adaptation of the robot skills were described. In this section the development and operation of the robot skill generation and adaptation module will be presented. For humanoids to cope with working in continuously changing environments and performing a wide variability of tasks, it is imperative to endow them with mechanisms that support the adaptation of their skills and behaviours to generate new ones fitting their context.

The robot skill generation and adaptation module is in control of handling the process by which learned models of a skill are adapted for an unseen context. The robot skill generation and adaptation module is provided with knowledge of the state of the environment and the constraints of the task extracted from the robot skill knowledge module; using both, the already learned model of a skill, and the extracted constraints information of the current task, the model of the skill is adapted to reproduce the task. Figure 5.2 illustrates the control flow for the operation of the robot skill generation and adaptation module. Mechanisms are needed to endow systems with the capacities to adapt their acquired skills expanding the system's knowledge and ability to act in the environment.

The robot skill generation and adaptation module supervises the process by which the learned model of a skill can be operated to reproduce a new task, including the adaptation, update, merger, combination, or transition of the skill models. There are three subsystems in this module, a subsystem for extracting data from the knowledge base, a subsystem for operating upon the robot skill with the adaptation algorithm, and a subsystem for generating the task models. Figure 6.17 shows the deployment

diagram for the robot skill generation and adaptation module.

The subsystem for extracting data from the knowledge base is made up of two processes; first it recovers data from the robot skill knowledge module and secondly it distributes appropriately this data to the rest of the subsystems for their operations. This subsystem implements a middleware between the knowledge base and the rest of the systems.

The adaptation algorithm subsystem handles the process of operating upon the learned robot skills, a first step from the information received from the previous subsystem would help it decide which type of method is required for adaptation; afterwards the chosen algorithm would work on the given robot skill models as described throughout Chapter 5. The adaptation algorithms were implemented using the MATLAB numerical computing environment.

The subsystem for generating the task models is in charge of preparing and expressing the adapted *Robot Skill Models* in a form suitable for robot reproduction. As a final step, a file is outputted storing the computed task model.

6.6 Reproducing the Robot Skills Task Models

Obviously all efforts in our framework would be useless if the robot were not equipped with proper mechanisms for the motor control of the robot skill reproduction. The robot skill reproduction module is in charge of producing the adequate control signals to the robot for the reproduction of robot skills. In this section, the development and operation of the robot skill reproduction module will be presented. The robot reproduction module is assigned with the task of providing suitable controllers that convert kinematic variables into appropriate motor commands. The robot skill reproduction module is given as input from the previous modules in the framework; the model of a robot skill as a *GMM*, as explained in Chapter 3. The first step is to compute the desired target value $\dot{\xi}$ through the *GMR* process, as given in Chapter 3. This would compute the desired target values for reference of the *HOAP* robot control system. Figure 6.20 presents the control strategy of the robot skill reproduction module, for details see [Pierro et al., 2009].

This scheme considers several blocks. Once a command has been received, the robot distinguishes if it is a command for the walking generation or for the arms movement. The walking patterns of the robot have been designed based on the theory of the 3D Linear Inverted Pendulum Mode presented in [Kajita et al., 2001b]. [Monje et al., 2008] presents studies for the posture stability control. If the received command requires a movement of the arms, as in the case of a grasping task, the selection of the suitable arm is first considered. Finally, the trajectory of the arm is evaluated online through the algorithm of kinematic inversion [Siciliano et al., 2009], once the command provides the distance and the orientation from the object. The orientation reference for the object is calculated with the support of the unit quaternion presented in [Chiaverini and Siciliano, 1999].

In order to decide the best arm to perform the grasping, the reachable workspace is divided into three areas: in particular, the two areas that can only be reached by one

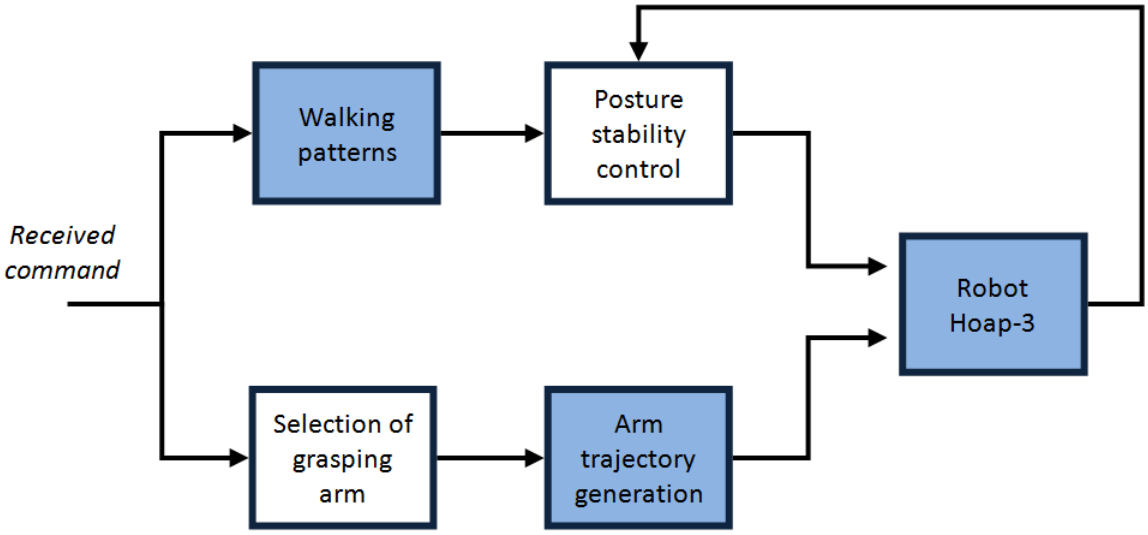


Fig. 6.20: *Control strategy of the robot skill reproduction module.*

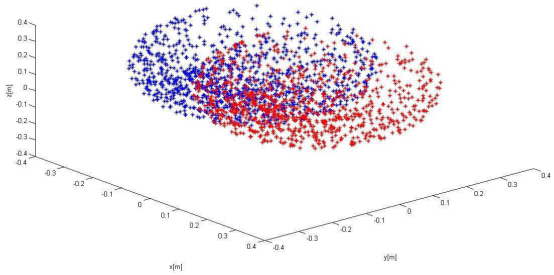


Fig. 6.21: *Workspace of Hoap-3 arms. Zone of service of right arm is depicted in blue while red are represent the zone reachable by left arm.*

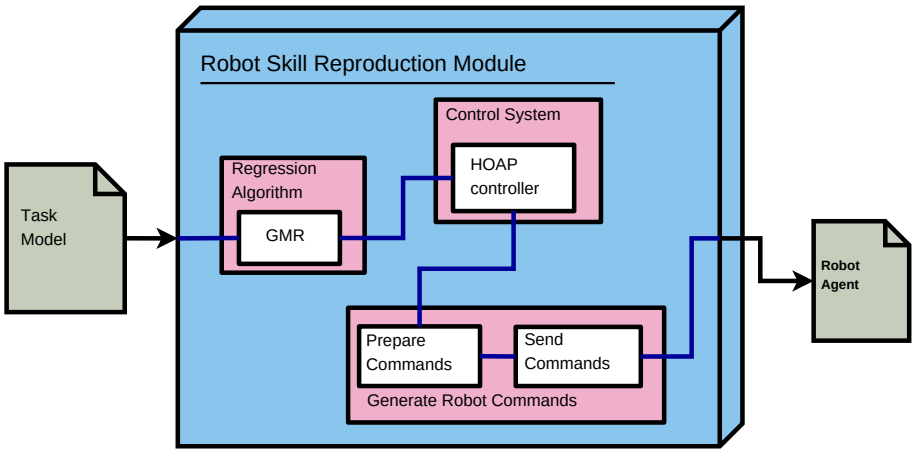


Fig. 6.22: *Deployment diagram for the robot skill reproduction module. The module has three subsystems, a subsystem for computing the regression of the model with GMR to obtain the desired target commands, a subsystem for producing the adequate control signals from the target commands, and a subsystem to communicate the control signals to the robot and monitor the HOAP-3 robot execution.*

of the arms and the workspace that can be reached by both arms. Figure 6.21 shows the three areas. Since there is only one arm that can reach the first two areas, we don't have to decide anything. In the case of work areas that can be reached by both arms, the system should decide the one whose manipulability is higher, considering the definition of manipulability stated in [Siciliano et al., 2009]

$$(\mathbf{q}) = \sqrt{\det\left(\mathbf{J}\left(\mathbf{q}\right)\mathbf{J}^{T}\left(\mathbf{q}\right)\right)} \tag{6.1}$$

where $\mathbf{J}$ is the Jacobian matrix of the corresponding arm and $\mathbf{q}$ the joint positions of that arm.

The robot skill reproduction module controls the execution of robot skills. The module has three subsystems, a subsystem for computing regression of the model with *GMR* to obtain the desired target commands, a subsystem for producing the adequate control signals form the target commands, and a subsystem to communicate the control signals to the robot and monitor the *HOAP-3* robot execution. Figure 6.22 shows the deployment diagram for the robot skill reproduction module.

The *HOAP-3* control systems is in charge of computing the appropriate command to control the execution in real-time of the humanoid robot, the control system has been presented above in Figure 6.20. The robot command subsystem handles the communications from the developed framework and its subsystems and the real robot agent for the actual reproduction and execution by the *HOAP-3* robot. The physical implementation of the robot control system is made on three PCs; an on-board PC implements the robot control systems; an auxiliary PC implements the knowledge and learning systems; and a laptop computer implements the **??** and perception systems. A YARP layer was implemented for the communications between processes.

6.7 Experimental Evaluation

Previously, in Section 6.2, a series of experimental evaluation scenarios were described. The evaluation scenarios were designed with the intent to present a demonstration of the overall performance of the framework developed through this work and the operation of its different modules. The evaluation of robotic systems, and knowledge base robotics systems in particular, is a complicated issue in which there are not readily available standardized evaluations or established benchmarks [Tenorth and Beetz, 2013]. The experimental evaluations presented in this section are aimed at providing proof of concept for the developed framework. Here the major focus of interest lies not in the measurement of performance and efficiency metrics but in the validation of the viability of the proposed system and the capabilities of the framework in dealing with a range of different and increasingly complex situations. The demonstration will test the operation of the humanoid robot and the developed framework as it is required to complete distinct tasks. Different scenarios are presented in order to highlight how the components of our framework contribute to achieving realistic tasks, and that the implementation of the capabilities for learning, knowledge manipulation and adaptation of skills are fundamental for the development of viable humanoid robots.

Several experiments were conducted to validate the proposed systems. A first scenario evaluates the performance of the perception and knowledge base modules. A later scenario deals with the performance of the robot skill generation and adaptation module. The final scenario is made to evaluate the performance of the robot skill reproduction and the complete developed framework for learning and adaptation of robot skills.

Evaluation of Knowledge Base Scenario

The first demonstrators were devised for testing the operation of the knowledge base scenario. The aim of the knowledge base scenario is to demonstrate how the humanoid robot employs the knowledge base module for the instantiation and upkeep of information from its environment perception and the objects that are present in it, as they are relevant for its task. It also presents the performance of knowledge base modules for storing *Robot Skill Models* and for retrieving and invoking the skills knowledge from the knowledge base when the information is needed to perform the robot skill in the completion of a task.

Two main experiments were carried out with the *HOAP-3* humanoid robot in this scenario, as described in Section 6.2. In the first demonstrator a humanoid robot would look around its environment as a human agent moves and manipulates various objects under the robot's field of view. The robot would instantiate and upkeep the object's knowledge as they become present and modified through the human agent interaction with the environment, keeping up to date information of the known objects to answer queries from the human agent about the state of certain objects. The second demonstrator presents a humanoid robot with the directive to complete a given task. The knowledge of the task and world state in the knowledge base would

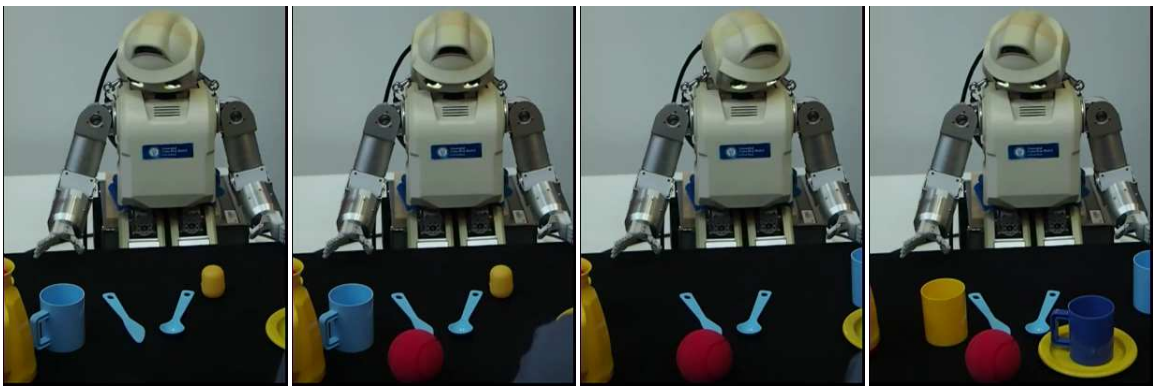


Fig. 6.23: *Knowledge Base Scenario Experiment A.1: different snapshots from the execution of the task in the demonstrator. The robot looks around the environment keeping in the knowledge base information of the objects state. The human agent moves and takes in and out of view the different objects at will. At different stages of the demonstrator the robot is ask to locate an object.*

afford the robot the possibility of completing its task by extracting the necessary robot skill models to perform the required skill motions needed to successfully complete the desired task.

For the first demonstrator, the *HOAP-3* robot looks around the scene as the perception system recognizes objects and instantiates or upkeeps their object frames in the knowledge base. Objects in the environment are taken in and out of the robot field of view or moved around the scene; also the robot would be asked to focus on different objects throughout the demonstration as its task directive is changed by the agent. Figure 6.6 shows a schematic view of the first demonstrator experiment in the knowledge base scenario. The focus of this demonstrator is on instantiation and upkeeping of object frames in the knowledge base.

Figure 6.23 shows different snapshots from the execution of the task in the first demonstrator. The experiment execution in this demonstration scenario would develop as follows, the experiment starts with the *HOAP-3* robot standing looking down at a table in front of it. The human agent arranges different objects on the table for the robot to recognize. In the first step the *HOAP-3* robot scans the scene from left to right, instantiating objects it can recognize. After the scan step is completed in the subsequent steps, the human agent rearranges any number of objects, while additionally, the robot is asked to locate one of the objects. In this stage the robot would look up the object's last known location information from the knowledge base and begin to look for the object from there; assuming the requested object is one that the human agent moved around. In this state the robot's main focus is to locate the requested object while a background process is still in charge of instantiating and upkeeping the rest of the objects in the robot field of view. The robot would either locate the object or complete one scan of the scene and assume the object has been removed and delete its instance from the knowledge base. The human agent would

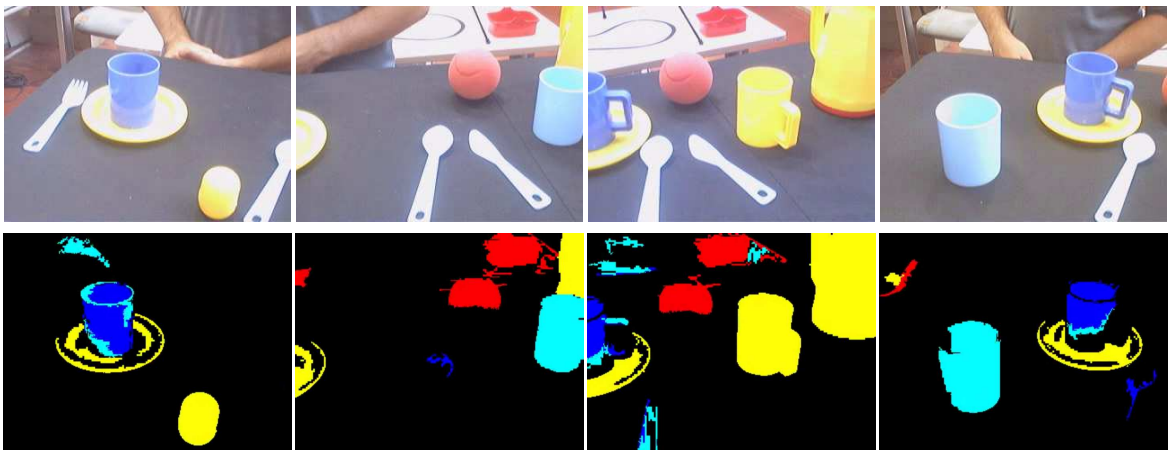


Fig. 6.24: *Knowledge Base Scenario Experiment A.1: different snapshots from the perception system during execution of the task in the demonstrator.*

repeat this for various objects during the experiment. In Figure 6.23 the robot is first seen looking for the red ball which it locates and later the yellow 'egg' which it can't find.

Figure 6.24 presents the operation of the perception system during the execution of the demonstrator experiments. Recognition of the objects is based on blob detection by color filtering and area comparison. The performance of the computer vision algorithms is not an element of this thesis, and the selection of the perception system components was made on the criteria of availability and easy integration with the rest of the framework. In general, the perception system works adequately for what it is needed, and there were only problems recognizing the knife and spoon objects; see center images in Figure 6.24, that were too bright and didn't accurately reflect their colours making them invisible for the recognition algorithm.

Figure 6.25 presents the operation of the knowledge base system during the execution of the demonstrator experiments. As new objects are being recognized, instances of the objects are created in the knowledge base storing information of their properties, in this case their colour and location. When objects are moved by the human agent interacting with the environment, objects' instances of the knowledge base update their information. The system focuses on objects that are in its current field view and that are important to its goals. The Figure 6.25 shows the contents of the knowledge base; in the lower row images they correspond to the state of the system at the moment of the images from the perception system in the above row. Objects' instances for objects that are out view are grayed out and their location property is changed to a relative value to reflect loss of certainty of their position; this value is then used as an indication of where to expect the object to be and the starting point to begin an exploration to look for it. Object instances for objects that are key to the robot goal are shown in blue; here the robot was asked to find the red ball, the second image, and the cyan cup, the fourth image. When an object can't be found again in the environment, in this case the yellow 'egg' in the third image, its object

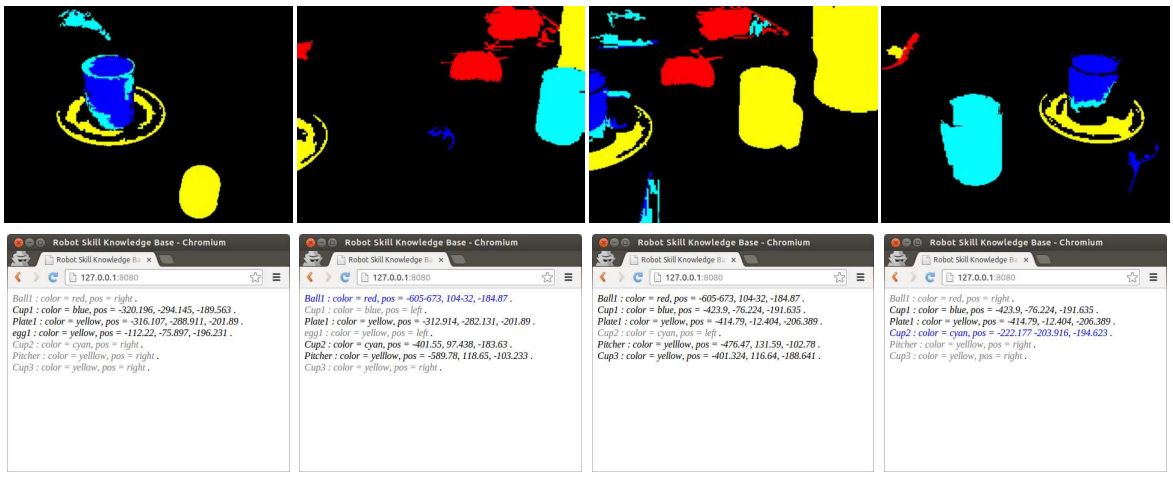


Fig. 6.25: *Knowledge Base Scenario Experiment A.1: different snapshots from the knowledge base system execution of the task in the demonstrator.*

instance is removed from the knowledge base.

The goal of this demonstrator scenario is to show how frames in the knowledge base are instantiated from the perception of the environment and how the knowledge base maintains and upkeeps its knowledge representation over time, in a changing environment. The capacity to manage the knowledge of the environment is an important affair for humanoid robots. While for industrial robotics or robots working in controlled environments where knowledge of objects and events around them are known and can be planned for in advance, for a humanoid robot working in a dynamic setting the state of the environment can have almost an unlimited number of configurations and can change unexpectedly at any moment. The knowledge base system allows the robot to build representations of objects in its environment and to keep track of changes that may occur. Also the knowledge base system is needed to help overcome some faults from the perception system and the problem of not always having available complete and reliable information from the environment.

For the second demonstrator the *HOAP-3* robot and a human agent interact to complete a simple task requiring the robot to pick up a cup and a spoon in each hand and then to put the spoon inside the cup; then finally it puts down the cup in front of itself. Execution of the demonstration could vary depending on the actions of both the human agent and the *HOAP-3* robot. Figure 6.8 shows a schematic view of the second demonstrator experiment in the knowledge base scenario.

Figure 6.26 shows different snapshots from the execution of the task in the second demonstrator. The experiment requires for the *HOAP-3* robot to complete the task of putting a spoon inside a cup and putting the cup down on a plate. Completion of this task has different steps, picking up the cup, picking up the spoon, placing the spoon inside the cup and placing the cup on top of the plate; selection of which skills is executed and when depends on the environment state and the interaction with the human agent. Execution of the experiment would develop as follows: first the robot

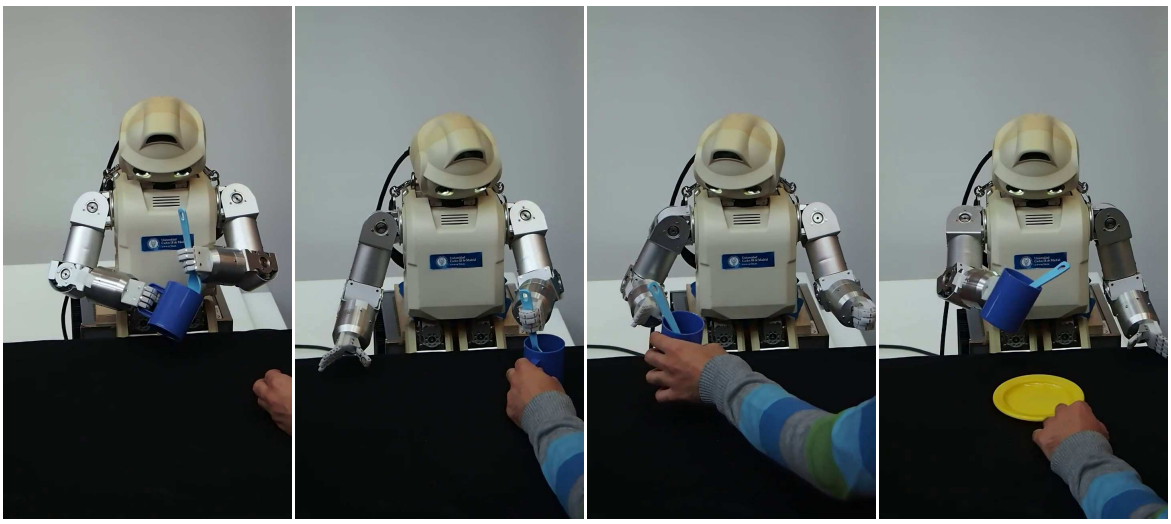


Fig. 6.26: *Knowledge Base Scenario Experiment A.2: different snapshots from the execution of the task in the demonstrator. Depending on the state of the environment and the human agent interaction the robot perform different robot skills.*

picks up the spoon in its left hand, then the human agent brings the cup near to the robot's workspace, depending on the position of the cup, the robot would either invoke from the knowledge base the "pick up cup" or "place spoon" skills. Eventually the robot's right hand is in possession of the cup with the spoon inside it, and the knowledge base invokes the execution of the skill motion for putting the cup down on the plate. In Figure 6.26 the robot can be seen executing different skills.

Figure 6.27 presents the operation of the perception system during the execution of the demonstrator experiments. Objects are recognized based on their colour properties and blob size. From the images it can be seen that some problems can take place when the human agent or the robot platform arm enter the camera's field of view, as occlusions and false recognitions can happen. Typically, these issues can be taken care of by the blobs' size and area inconsistency with expected objects' properties, or by their failed instances being removed from the knowledge base since their constant movement made them disappear too quickly for them to affect the operation of the system.

The operation of the knowledge base system during the execution of the demonstrator experiments can be seen in Figure 6.27. The knowledge base presents information for the environment and the task execution. The task frame holds knowledge of the actions to carry out by the robot for the execution of the task. Actions highlighted in blue reflect the current invocation of that action's knowledge for the robot reproduction of the skill. Actions that have been completed are deactivated and highlighted in grey. The selection and activation of which skill motion to carry out next, is completely determined by the skill initial conditions being matched to the state of the environment. Therefore, the sequence of execution of the task is controlled by the

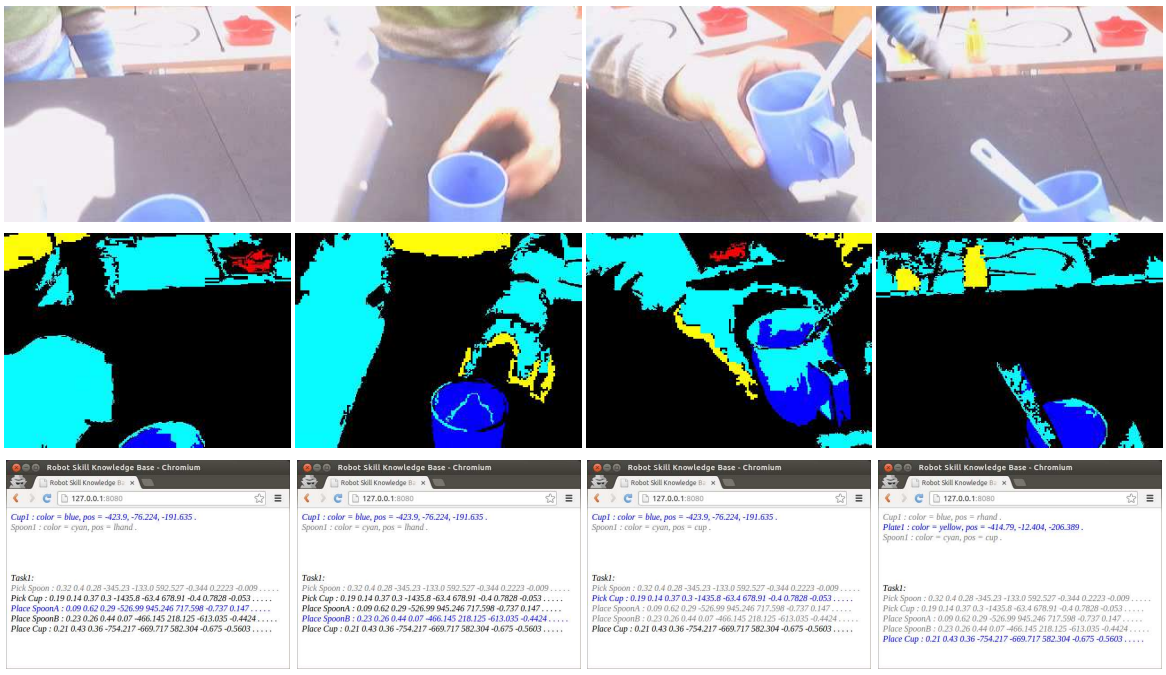


Fig. 6.27: *Knowledge Base Scenario Experiment A.2: different snapshots of the execution of the demonstration illustrating the operation of the perception system and the knowledge base system.*

human agent as it interacts with the robot and the environment and facilitates the objects and conditions needed for the robot to fulfil the task. A potential problem is determining which action has precedence when many of them can satisfy their conditions at the same time. The tasks considered in the demonstrator didn't present this issue, since the robot's limited workspace prevented the conditions for picking up the cup and placing the spoon to be satisfied at the same time. This issue has not been fully explored so far, and as a first simplification precedence is determined by the order of the actions in the task frame as determined by the programmer of the task; although not satisfactory for every scenario, this solution is probable enough for many common tasks. The use of some form of long time planner could be effective to solve this issue by assigning precedence by determining how the decision of performing one action over another could affect the execution of the task several steps ahead.

The goal of this demonstrator scenario is to show how action execution is invoked by the state of the representation frames present in the knowledge base. Figure 6.28 presents a storyboard of the performance of the system during the execution of the demonstrator experiments with snapshots taken at various stages. A knowledge base approach for robots working in unstructured environments, where the execution of the task cannot be scripted beforehand is fundamental if they are to be able to work successfully. Without such a system the robot would be unfit to respond to any unforeseen deviation from the plan, and be largely ineffective to perform in all but the most ideal of situations. The knowledge base system allows the robot to keep

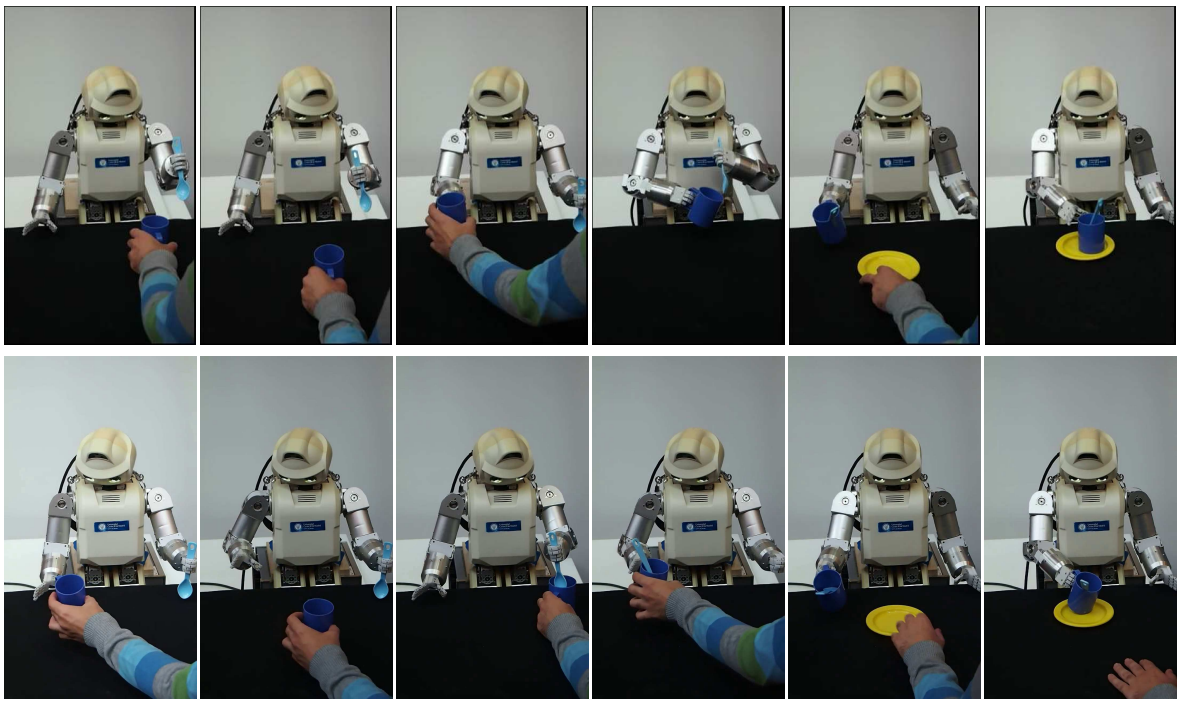


Fig. 6.28: *Knowledge Base Scenario Experiment A.2: different snapshots during the execution of the demonstration. The top row and the bottom row represent two different reproduction of the experiment.*

track of the environment and the state execution of the task, this provides the system with flexibility to deal with different states at a particular point with out losing focus of the global task objective.

Evaluation of Skill Generation and Adaptation Scenario

Here the demonstrators are oriented to the evaluation of the performance of the robot skill generation and adaptation scenario. The aim of the skill generation and adaptation scenario is to demonstrate how the operations of the humanoid robot can be expanded from an original set of learned robot skills by operating over the *Robot Skill Models* as presented in Chapter 5, in order to generate new models of robot skills. In this scenario the performance of the learning module for learning and encoding *Robot Skill Models* is presented first. The human agent would provide different teaching demonstrators to the robot, gathered from the methods described in Chapter 3 to build a first set of skill models. Secondly, the methods for merger, update and combination of *Robot Skill Models* are validated by applying them in different situations allowing the humanoid robot to achieve its task objectives, unreachable with its original skill set, by employing newly generated robot skills.

Two main experiments were carried out with the *HOAP-3* humanoid robots in this scenario, as described in Section 6.2. In the first demonstrator, a humanoid robot is equipped with a table tennis paddle and taught to perform different tennis

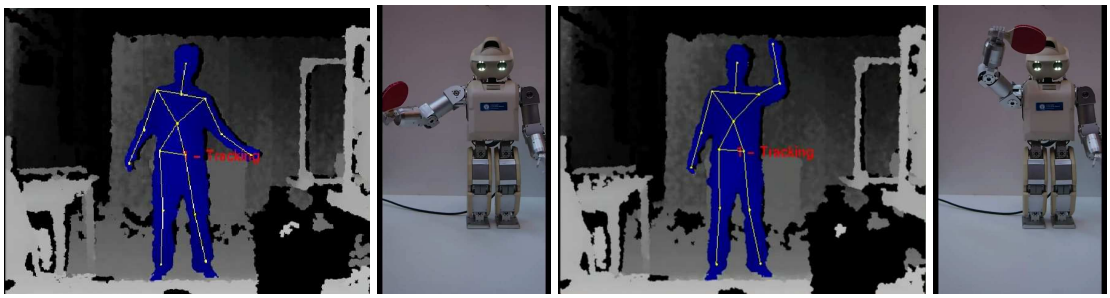


Fig. 6.29: *Generation and Adaptation Scenario Experiment B.1: different snapshots from the execution of the task in the demonstrator. Recording of the teacher demonstrations for the forehand and smash shot skill with a kinect camera (the kinect images are mirrored). HOAP-3 robot reproduction of the learned forehand and smash shot skill.*

shots, then additional tennis shots skills are generated by the merger of different learned skill shots. For the second demonstrator, the humanoid robot is required to grasp an object from various possible initial locations, while being taught to perform the skill motions to grasp it for only a limited number of locations. The complete task would be unachievable with the limited robot skills set learned at first since the skill reproduction would not generalize well to every target's location. To generalize, across the whole working space models of the robot skill are combined into a single model of the attractor dynamics.

For the first demonstrator in this scenario the *HOAP* robot is required to execute different tennis shot actions to hit a table tennis ball. Originally, robot skills' models are taught to the *HOAP-3* robot to hit an approaching ball, providing the robot with the skills to perform a forehand shot and a straight smash shot. To expand the robot skill set, the two learned skill models are merged to obtain a new *Robot Skill Model* for a forehand-smash shot. Figure 6.10 shows a schematic view of the first demonstrator experiment in the generation and adaptation scenario.

Figure 6.29 shows different snapshots from the execution of the task in the first demonstrator. The experiment execution in this demonstration scenario would develop as follows: first, a human teacher is recorded executing demonstrations for the forehand and smash tennis shot skill motions. The teacher demonstrations are recorded with the use of a kinect camera tracking the skeleton of the user during the demonstration. *Robot Skill Models* are encoded from the demonstrations following the *SEDS* learning mechanism reviewed in Chapter 3. With the learned robot skills, the robot is given the capacity to successfully perform a forehand tennis shot and straight up smash shot skill motion.

Figures 6.30 and 6.31 summarizes the process of encoding the tennis shot skill motions presented above. The figures show a 3D reproduction of the learned skills trajectories, the training data from the recorded demonstrations of the skill and the encoded *SEDS* models of the robot skill.

In order to expand the robot skill set and increase its range of action for the table

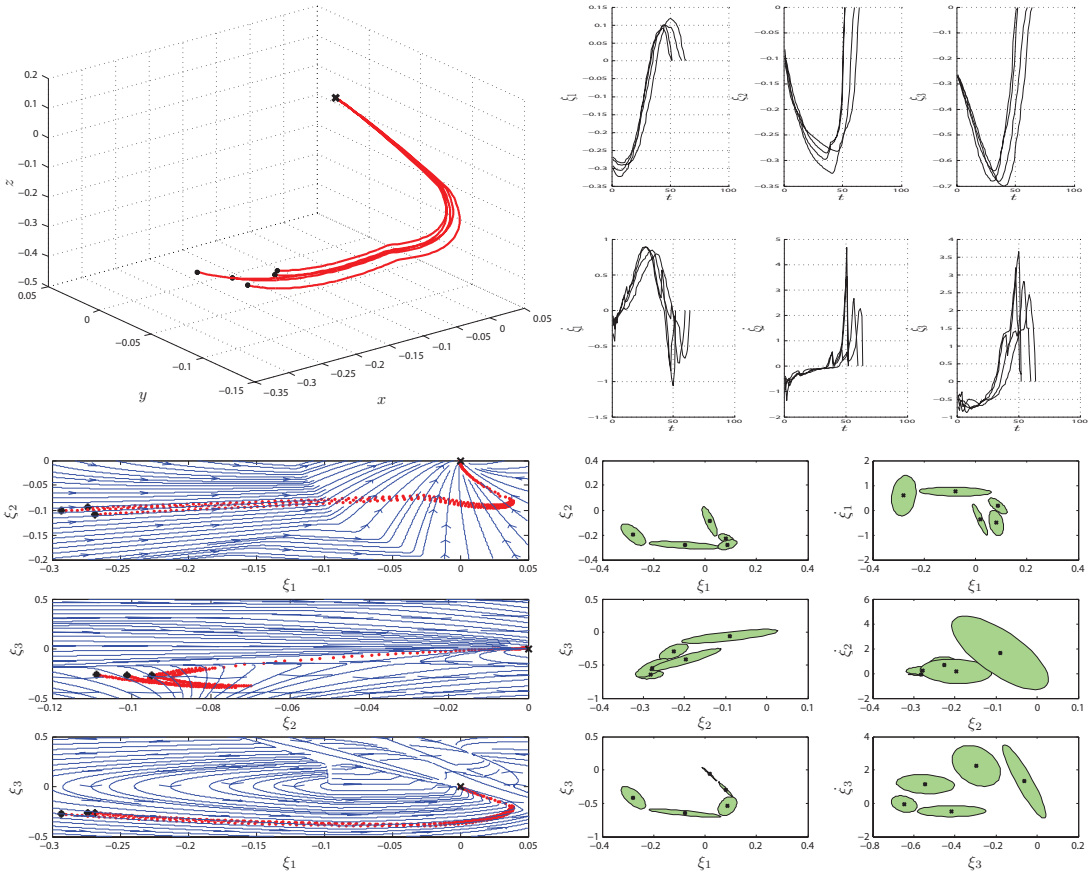


Fig. 6.30: *Generation and Adaptation Scenario Experiment B.1: learning the forehand shot skill. 3D reproductions of the learned skill. Demonstrations trajectories, positions and velocities plotted over time. Streamlines of the learned dynamics with various reproductions of the motion. The learned GMM of the robot skill.*

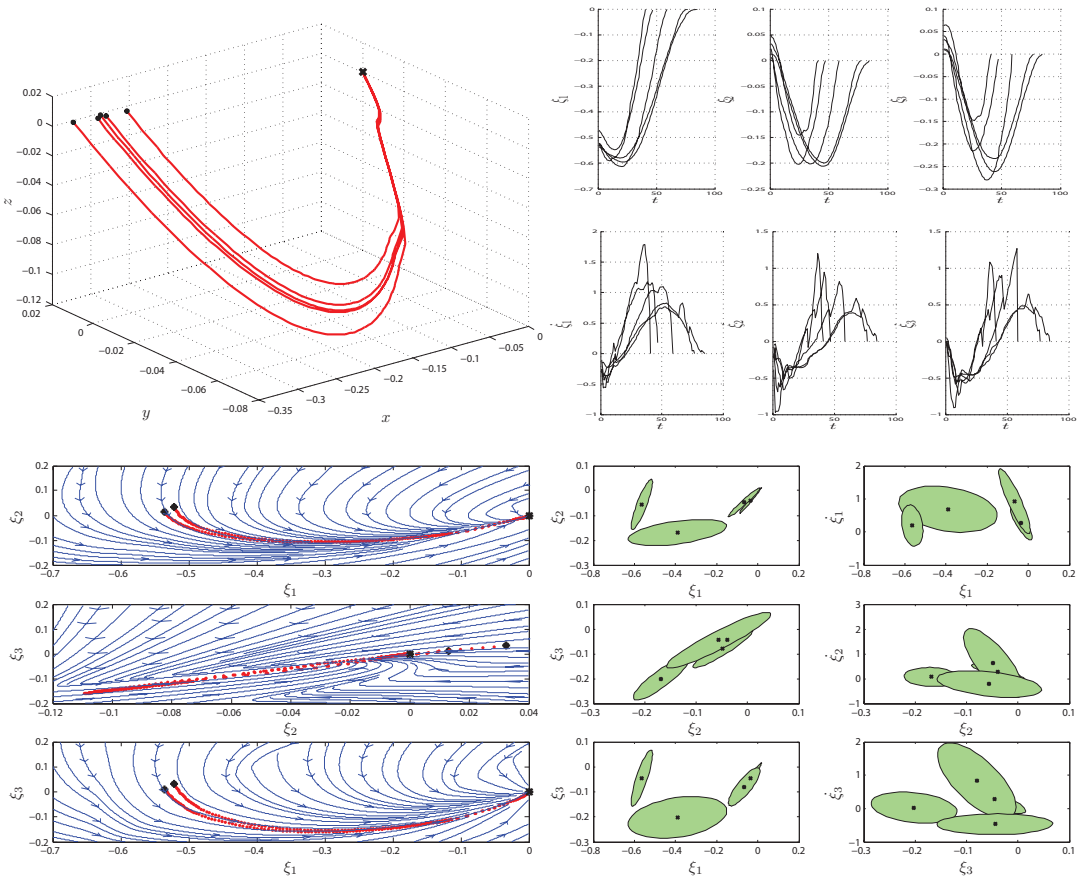


Fig. 6.31: *Generation and Adaptation Scenario Experiment B.1: learning the smash skill. 3D reproductions of the learned skill. Demonstrations trajectories, positions and velocities plotted over time. Streamlines of the learned dynamics with various reproductions of the motion. The learned GMM of the robot skill.*

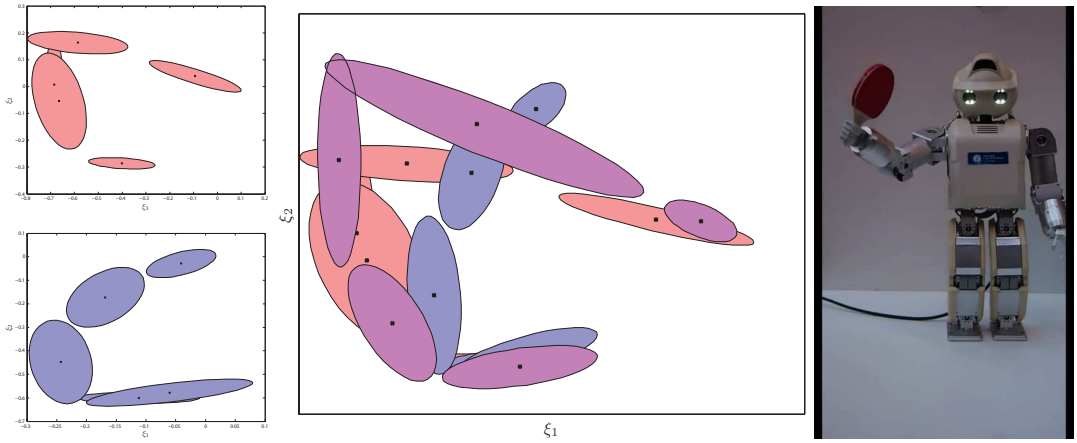


Fig. 6.32: *Generation and Adaptation Scenario Experiment B.1: Generating the forehand-smash shot skill from the merger of the learned forehand and smash shot skill models. HOAP-3 robot reproduction of the forehand-smash shot skill.*

tennis task, the two learned skill motions are merged through the methods presented in Chapter 5. Figure 6.32 illustrates the process of generating the forehand-smash skill model from the merger of the forehand and smash robot skills learned in Figures 6.30 and 6.31. Being capable of expanding a robot set of learned skills is clearly an important issue as robots will be asked to perform an increasing number of activities and learning and programming every possible skill into the robot is infeasible. As stated in the previous chapters, the properties of the learned *Robot Skill Models* encoded with the *SEDS* method form *Khansari* will hold for the merged *Robot Skill Models* generated here. Learning the robot skills with *SEDS* as a model of the motions dynamics has several desirable properties that have been stated before in previous chapters. This allows the robot to have an encoded model, generalizing the dynamics of the motion, that can respond to perturbations on the execution of the task and changes to the initial conditions.

The second demonstrator requires that the *HOAP-3* robot grasps a cup object located in any possible place in a "cupboard", which is made up of two shelves, a bottom and a top shelf. The *HOAP-3* robot must be able to grasp the cup, as long as it is inside the robot arm's workspace, in any of six possible general locations in relation to the robot arm: three on the bottom shelf and three on the top shelf. At first, the only skills learned by the robot are for grasping the cup placed on the bottom shelf. To grasp the cup, placed on the top shelf, at either side of the robot, the skills learned to grasp the cup on the bottom shelf must be updated to generate the required new robot skill models. Figure 6.12 shows a schematic view of the second demonstrator experiment in the generation and adaptation scenario.

The experiment execution in this demonstration scenario would develop as follows: first a human teacher is recorded executing demonstrations for grasping a plastic cup object located on the bottom shelf of a "cupboard". The teacher demonstrations are

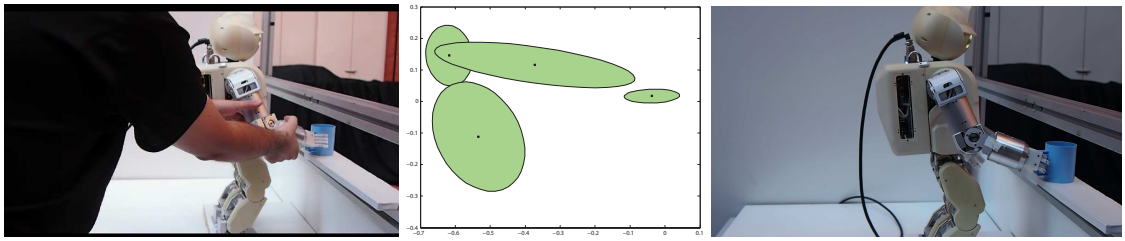


Fig. 6.33: *Generation and Adaptation Scenario Experiment B.2: teaching and learning the skill motion for grasping a cup in the bottom shelf of the "cupboard".*

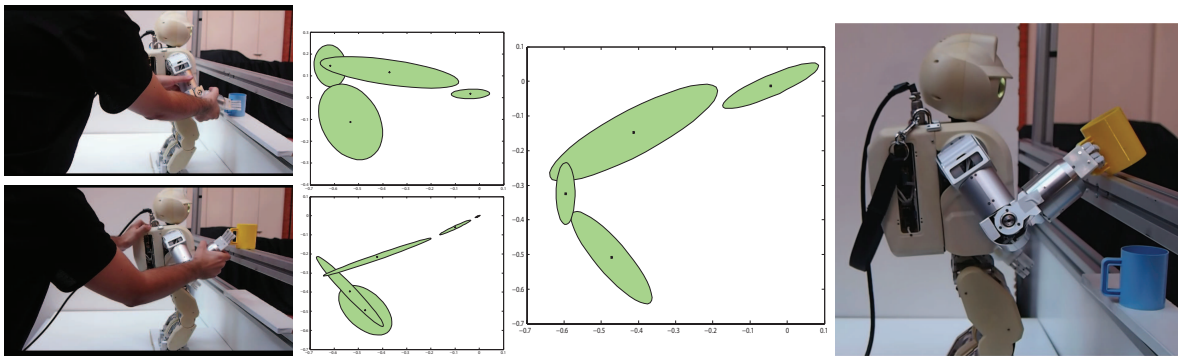


Fig. 6.34: *Generation and Adaptation Scenario Experiment B.2: updating the skill motion for grasping a cup in the top shelf of the "cupboard".*

recorded by means of kinaesthetic teaching, with a human agent moving the *HOAP-3* robot arm through the demonstration of the skill. Figure 6.33 shows the process of teaching and learning the skill motion in the first demonstrator. *Robot Skill Models* are encoded from the demonstrations following the *SEDS* learning mechanism reviewed in Chapter 3. The desired goal is for the robot to have the capacity to successfully grasp the cup out of the "cupboard" regardless of its possible position inside it. That is, the cup could be placed to the left, right or in front of the robot on either the top or bottom shelf. Trying to generalize the learned skill for grasping on the bottom shelf to perform a grasp on the top shelf would not be successful. To grasp the cup when placed on the top shelf, the learned *Robot Skills Models* must be updated, employing the method presented in Chapter 5. Figure 6.34 shows the process of updating the skills of the bottom shelf grasping for performing the grasp skill for placements of the cup on the top shelf of the "cupboard".

Figure 6.35 summarizes the process for generalizing the learned skill for grasping the cup object out of the "cupboard" regardless of its possible location. The figure shows the performance of the system during the execution of the demonstrator experiments successfully grasping the cup when placed to the left-bottom, right-bottom, center-bottom, left-top, right-top, center-top, of the robot.

In order to expand the robot skill set and increase its range of action to encompass

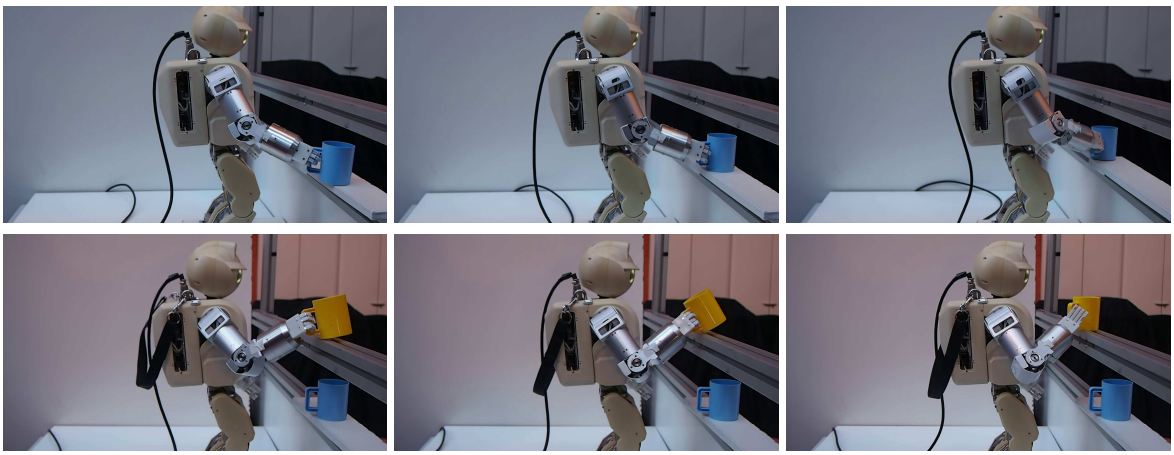


Fig. 6.35: *Generation and Adaptation Scenario Experiment B.2: different snapshots from the execution of the task in the demonstrator. Grasping the cup when placed to the left-bottom, right-bottom, center-bottom, left-top, right-top, center-top, of the robot.*

a larger spectrum of the attractor dynamics, the *Robot Skills Models* must be combinable into new models. This allows more complex tasks than those presented during demonstrations to be carried out, generalizing the models of the skills to regions outside their original demonstrations. To generalize the skill across the whole working space of the shelves in the "cupboard", the three models of the robot skill, for right, left and center, grasping motion on a shelf, are combined into a single model of the attractor dynamics. Figure 6.36 illustrates the complete behaviour of the generated new skills models. For humanoid robots to be capable of working successfully in the capacity in which they are envisioned, it is of vital importance that they present ample and robust skill sets. The ability to learn robot skills is a key aspect to achieving this, yet learning by itself is not sufficient; the capacity to operate over the learned robot skill, such as the merger, update and combination of skills, is necessary. Updating previously learned skills is a very important ability for humanoid robots, allowing them to increase and improve their available skill set. Combining different robot skills allows the expansion of the scope of application of the learned skills and generalizes them to new contexts. One important gain from the combination of robot skills comes from increasing the accuracy of the generalized behaviour. The generation of a model by combining robot skills is necessary in order to improve the task execution.

Evaluation of Robot Skill Reproduction Scenario

For the final scenario, two general demonstrators were implemented for validating the complete developed framework for learning and adaptation of robot skills. The goal for the robot skill reproduction scenario is to demonstrate the operation and integration of all the overall systems in the framework for the performance of a humanoid robot in a complex unscripted environment interacting with a human agent.

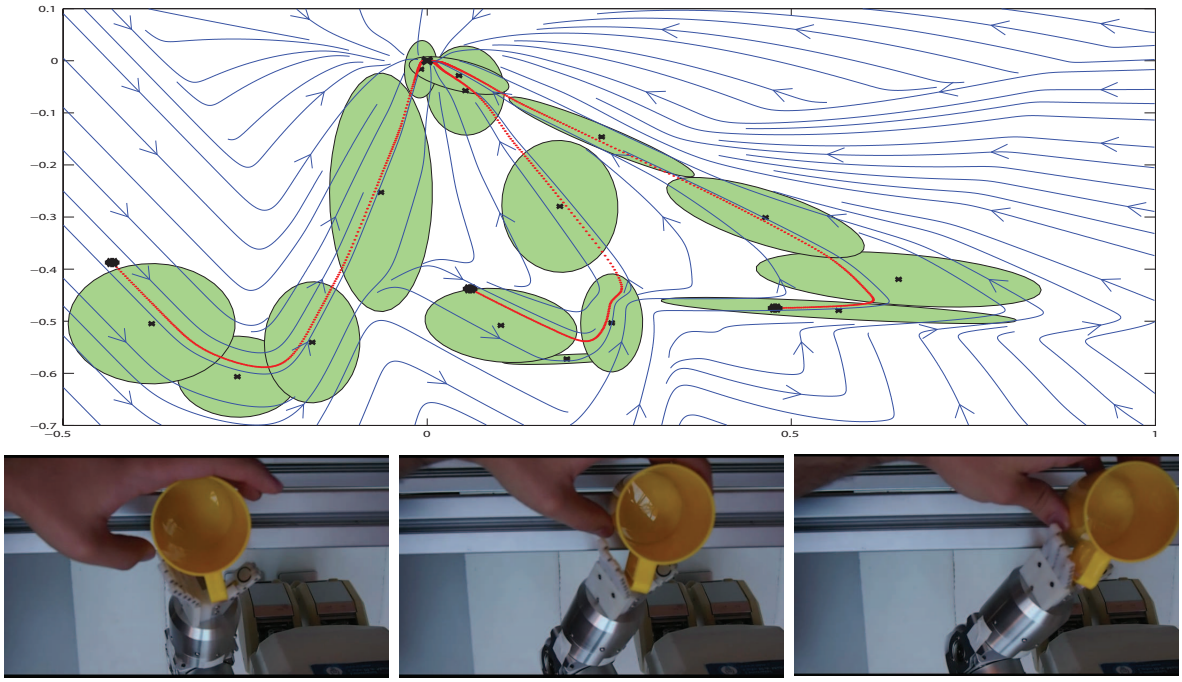


Fig. 6.36: *Generation and Adaptation Scenario Experiment B.2: combine skill model allows to encompass a larger spectrum of the attractor dynamics. Different executions of the task grasping the cup at different locations.*

Two main experiments were carried out with the *HOAP-3* humanoid robot in this scenario, as described in Section 6.2. In the first demonstrator we complete the table tennis scenario from the previous subsection. Here, the *HOAP-3* robot would stand equipped with a table tennis paddle waiting for an approaching ball to hit with an appropriate tennis shot skill. For the second demonstrator the previous scenarios involving the *HOAP-3* robot employing kitchen objects are expanded. Here, the robot is required to complete a setting up of a dinner service task behaviour with assistance from a human agent.

For the first demonstrator in this scenario the *HOAP* robot is expected to simulate a game of table tennis. The humanoid robot stands, paddle in hand, expecting a table tennis ball to be moved towards it. The perception system would recognize the ball and extract the appropriate learned robot skill models to reproduce the action from the knowledge base. Figure 6.14 shows a schematic view of the first demonstrator experiment in the robot skill reproduction scenario.

Figure 6.37 shows a set of different snapshots captured during the execution of the task in the first demonstrator. A successful execution of the experiment in this demonstrator would develop as follows: the *HOAP-3* robot starts the experiment standing at a rest position, with a table tennis paddle in its right hand, waiting for an approaching ball to hit with an appropriate tennis shot skill. The limitations of the perception and of the robot itself don't allow for a real-time reproduction of the task, therefore, the ball is handled by a human agent who approaches it to the robot

at a controlled speed. The perception system recognizes the ball at a certain distance from the *HOAP-3* robot position, since providing complete accurate estimation of the ball's position when moving is not possible by the perception system, we simplify things and divide the space into quadrants and make a rough estimate of what the ball final position will be, based on which quadrant the ball was travelling in at the recognizing step.

After the perception system recognizes the ball an instance of the ball object is created on the knowledge base. Since the precise position of the ball is not needed, the ball object instance holds only the estimate for which quadrant the ball is in. The knowledge base also holds the task event frame for the demonstrator consisting of the *Robot Skill Models* for performing the tennis shots. There are 4 *Robot Skill Models* in the task event frame from the skill learned in the previous scenario, we have a forehand and smash shot skill, and also from the previous scenario we have a forehand-smash shot skill generated from the merger of the other two skill models. One additional skill was learned for the demonstrator for the performance of a backhand shot employing the same methodology as it was for learning the other *Robot Skill Models*. When the ball is recognized by the perception system crossing one of the quadrants the appropriate skill action is invoked from the task event frame for the robot reproduction. Figure 6.37 shows the *HOAP-3* robot performing the different tennis shot skills as it interacts within the demonstrator; the central image is at the onset of the motion, and the right image is at the end of the motion; the left image depicts the state of the system leading to the reproduction of the tennis shot skill. For the experimental run illustrated by Figure 6.37 the 'point' begins with the *HOAP-3* robot returning a backhand shot (first row), followed by two successful forehand shots (only one is depicted, second row), then the *HOAP-3* robot performs a smash shot (third row), and finally the 'point' concludes with a forehand-smash shot return for a score of "love, 15".

The second demonstrator requires the robot to set a "dinner service" consisting of a fork, a knife, a saucer plate, a cup and a spoon, in conjunction with a human agent. The purpose of the demonstrator is to test the overall operation of the developed framework, as well as validating the performance of every individual module and interaction between themselves. The sequence of execution of the task could vary depending on the actions of both the human agent and the *HOAP-3* robot. Figure 6.16 shows a schematic view of the second demonstrator experiment in the robot skill reproduction scenario.

Figure 6.38 depicts a storyboard of the performance of the second demonstrator taken from several snapshots, captured from the execution experiment. A standard run-through the demonstration scenario would develop as follows: first the robot is given the task of setting up "dinner service" at the table in front of it, and all necessary robot skill actions and task event frames are stored in the knowledge base. The task begins with the robot standing in front of the empty table. The final set-up of the table requires a plate to be placed in the center, a cup is placed on top of the plate, and a spoon is placed inside the cup, a fork and knife flank the plate at its left and right sides respectively. Completing the task requires the performance of several different skills, the selection of which skill is to be carried out by the robot at each

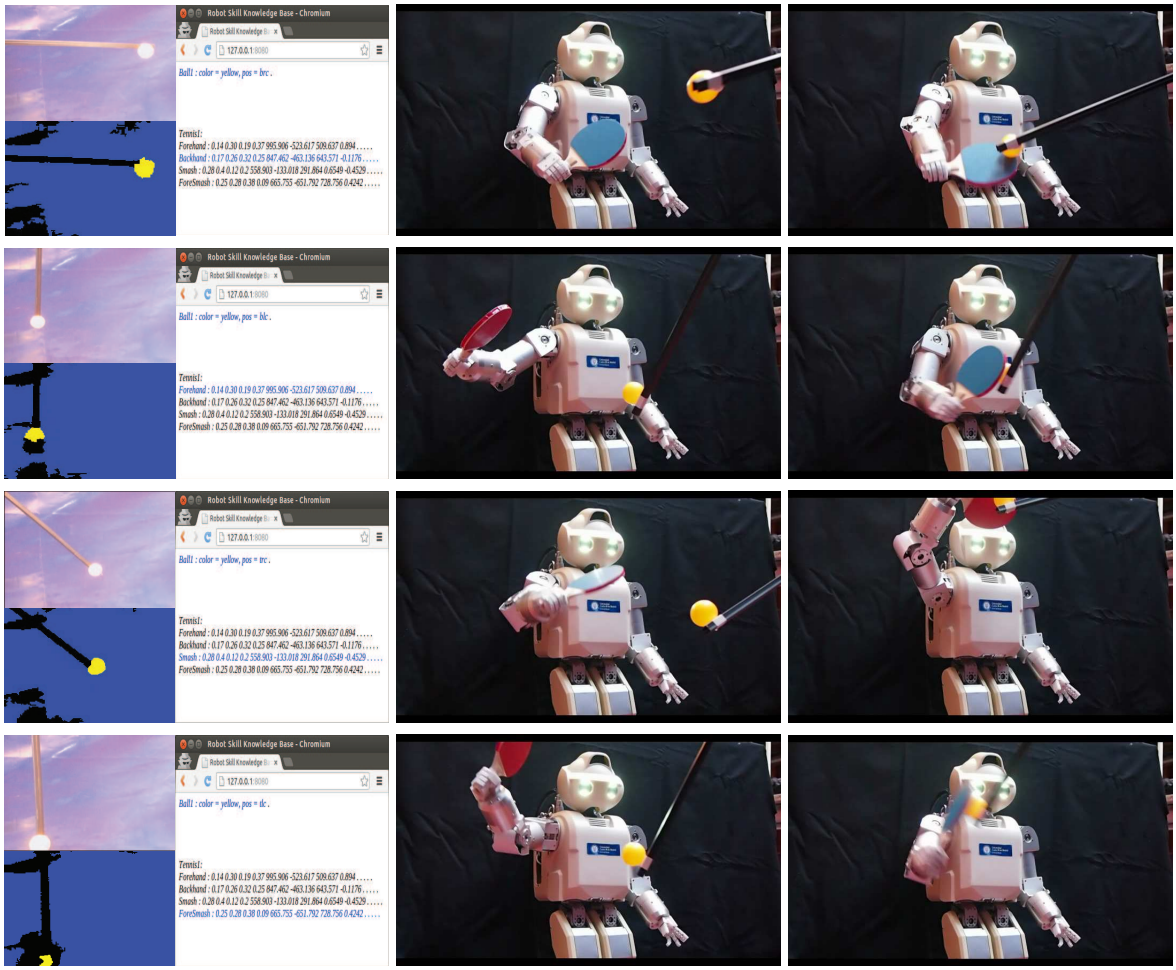


Fig. 6.37: *Robot Skill Reproduction Scenario Experiment C.1: different snapshots from the execution of the task in the demonstrator.*

time comes from the actions being afforded to the robot by the environment, through the interaction with a human agent, in the knowledge base. Therefore, the sequence of execution of the task is governed by the human agent as it is him who chooses the order in which to provide the robot with the needed objects. Certain items, however, have precedence over others, i.e. the plate must be placed on the table before the cup, since the cup goes on top of it.

The first object to be placed on the table is a 'red' container box, from which the *HOAP-3* robot picks up the objects, when available. A human agent would then choose from the pool of objects of the task one object to be placed by the robot, and would set it on the container box, Figure 6.38 top left image shows the instance where the human agent sets the first object for that run of the task, in that case a plate. The *HOAP-3* robot perception would recognize the object in the container box, when this happens an object frame instance is created in the knowledge base, and the action frames in the task event frame are checked out to find which, if any, match is invoking conditions from the current state of the world frame, in order to begin reproduction of a skill. Once an action is chosen, the *Robot Skill Model* parameters $\theta = \{\pi, \mu, \Sigma\}$ are recovered from the knowledge base system and provided to the robot skill reproduction model for performing the actual reproduction of the skill, as in the *GMR* process described in Chapter 3. The robot will pick up the given object and carry out the required operations with it to place the object where ever it will be appropriate.

The rest of the task will continue in this way, with the human agent initiating the performance of action skills to an object as determined by his interaction with the robot agent by presenting it with the objects. Figure 6.38 shows the *HOAP-3* robot performing a different skill from this interaction in completing the setting up "dinner service" task. From left to right, starting at the top row, the human agent first presents the robot with the yellow plate, then the robot picks the plate up, the robot transports the plate to the position where it must be placed, and finally the robot puts the plate down on the table; a little assistance is required by the human agent in that instance as the robot configuration of the wrist DOF makes it difficult for the robot to orientate the plate for dropping it gently on the table. The second row begins with the human agent presenting the robot with the fork; then the robot picks it up, the robot then switches the fork to its left hand, and finally drops it on the table. The third row begins with the human agent presenting the robot with the knife, then the robot picks it up, the robot then transports it to the position where it must be placed, and finally the robot puts the knife down on the table at the side of the plate; a little assistance is also required from the human agent. The fourth row depicts the operations with the spoon object which goes the same as with the fork; the skill actions to handle them are the same, except for the final step in which the spoon is not set down on the table as it needs to go inside the cup. The fifth row begins with the robot picking up the cup from the human agent; it then transport it towards its left, and transport the spoon towards the cup; finally the robot places the spoon inside the cup and return to rest with the cup with the spoon in its right hand. The final row depicts the *HOAP-3* robot performing the skill action to place the cup on top of the plate and complete the task experiment.

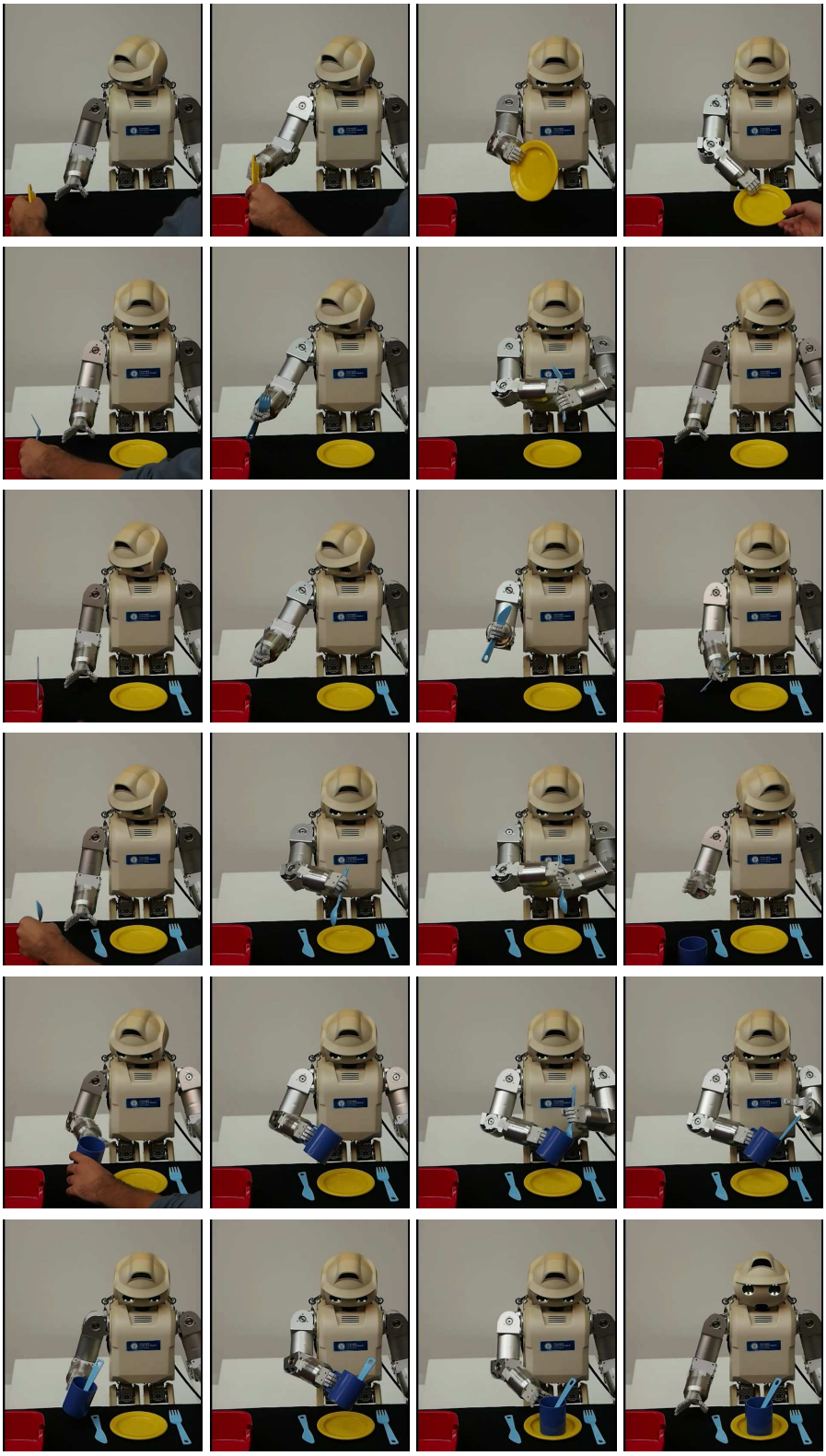


Fig. 6.38: *Robot Skill Reproduction Scenario Experiment C.2: different snapshots from the execution of the task in the demonstrator.*

6.8 Summary of the Chapter

Throughout this chapter the development and implementation of the framework and the different modules that compose it have been described. Also, the experimental scenarios are described and results and analysis are presented for the validation of the framework proposed throughout this work. Different evaluation scenarios were developed to test the performance of the various modules implemented in our framework and to provide separate validation for the operation of the system. Section 6.2 described the development of the framework as well as the experimental set up for validating it, and the robotic platform used in this work, complete with a description of its structure, joints and sensor distribution. In Section 6.3 the implementation of the robot skill learning module was described. Section 6.4 presented the implementation of the knowledge base system. In Section 6.5 the development and operation of the robot skill generation and adaptation module is described. Section 6.6 presents the implementation of the robot skill reproduction module in charge of producing the adequate control signals to the robot for the reproduction of robot skills. Finally in Section 6.7, a description of the experimental results and analysis for validation of the proposed framework over the evaluation scenarios is given. Different evaluation scenarios are employed to test the performance of the various modules implemented in our framework. Demonstrations are organized over three major scenarios to provide separate validation for the knowledge base system, the generation and adaptation system, and the complete developed framework.

7. DISCUSSION AND FUTURE WORK

Work on this thesis first started under the scope of a European project for the development of humanoid robots for collaborative working environments. It then continued, through various other projects, always linked to the issues and challenges of designing and creating humanoid robots which are capable enough of working and aiding their human partners during its everyday tasks. The prime motivation of this work, and of the humanoid robotics field in general, is in the development of humanoid robots, and their control mechanisms, with comparable skills and behaviours to those of humans. The main idea being that human-like robots would be favoured to perform in the real world and that the use of humanoid robots that can give support in performing human daily activities would significantly help people in work sites, homes and in dangerous or emergency situations.

Before this vision can become a reality, many important challenges need to be addressed. These challenges encompass a whole range of issues from locomotion and motor control, to perception, interaction and cognitive behaviour and intelligence. In Chapter 2 a review of the developments and challenges in humanoid robotics research, and of different proposals for intelligent agents' architectures for robotic systems, was presented. There is much work to be done to improve the capabilities of humanoid robots for locomotion, perception, interaction, cognitive behaviour and competence at performing tasks. Progress in all of these aspects is vital and separate efforts at improving each one of these issues is of crucial importance; however, true breakthroughs in the development of fully functional humanoid robots can only occur when advances in all of these issues can be done concurrently.

The field of robotics has certainly seen some advances in these issues over the years, with great usage of robotics for industry, surveillance, entertainment and manufacture applications. However, the performance of humanoid robotics remains hindered by these issues, in particular the requirement for intelligent cognitive behaviour. Humanoid robots must present intelligent, natural, predictable and reasonable behaviours, and development of intelligent controls to resemble this is a major challenge. Research into cognitive architectures constitutes a solid basis for building intelligent systems, but even though some attempts in the field have been made for providing cognitive processes for humanoid robots, there are not fully developed cognitive architectures readily available with the capabilities of endowing robots with the needed functional intelligence. The cognitive approaches are centred on the mechanism that allows for the generation of thought and the interior workings of cognition; this calls for an organization of intelligence in terms of cognitive models. Models of cognition must be embodied processes that capture the unfolding of cognition in time, mindful of the associated sensory and motor surfaces embedded in the environment.

One major contribution of this work is the development of a framework, as presented in Figure 2.6, for a cognitive model for the generation and adaptation of learned robot skill models for complying with task constraints. In the developed framework a knowledge base of the skills is built with the models of the skills learned through demonstrations. During the execution, the constraints of a requested task are extracted from the perceptual system from the working environment and the models of an appropriate skill are retrieved from the skills knowledge base. With all available information, a new adapted task model is generated for reproduction.

The framework developed in this work was proposed as a cognitive model intended to provide the robot with an essential cognitive ability for learning and adaptation of skills. Though it is not a primary consideration of this work, our framework can be thought of as one module level in the hierarchy of a more complex architecture, or as a first stepping stone upon which to incrementally build more complex cognitive processes. The goal of the developed framework is to provide a minimum degree of intelligence for the humanoid robot. The ultimate goal of the field, as stated before, calls for fully functional humanoid robots capable of performing any type of task as a human agent would, and capable of working, collaborating and interacting with humans, sharing the same space, tools, and activities. This vision requires for robots to present full level cognitive and intelligent architectures, however, current developments are not yet even nearly close to these capacities, and our discussion needs to start at some point in a basic functional level of intelligence. The review of intelligence, on Chapter 2, lead to recognizing as a minimal desirable level of intelligence for our humanoid robots the ability to sense the environment, learn, and adapt its actions to perform successfully under a set of circumstances.

The developed framework provides humanoid robots with systems that allow them to continuously learn new skills, represent their skill's knowledge, and adapt their existing skills to new contexts, as well as to robustly reproduce new behaviours in a dynamical environment. The cognitive framework for learning and adaptation of robot skills is made up from several modules, including modules for the learning of robot skills, the perception and interaction with the environment, the management and representation of skill knowledge, the generation and adaptation of skill models, and the reproduction of robot skills.

A skill in our context has been defined as a motor trajectory motion learned by the agent, an acquired ability for the execution of a task. *Imitation Learning* approaches were used to teach a robot how to accomplish a given task. To learn the skills motion a time independent model of the motion dynamics was estimated through a set of first order non-linear multivariate dynamical systems. Despite *Imitation Learning* clear advantages, it would be impractical to teach the robot skills for every task and situation, therefore, it was necessary to extend the approach in a way that allows the adaptation of previously learned motion skills to new contexts. The models of a skill are adapted to generate a new task by operating over the given robot skill models. The system must be able to store and latter retrieve and use their knowledge of learned skills. The knowledge base holds all necessary information for reproduction of the skills in the environment. Knowledge of the task is distributed among the representation of objects, actions and events of the task and the state of the world.

7.1 Discussion on Learning Robot Skills

The ability to learn robot skills is one of the most important for developing humanoid robots. While programming robots to perform a series of required tasks is certainly possible for industrial robotics, humanoid robots are required to perform a wide repertoire of tasks working beside humans in complex dynamic environments, making a learning approach a necessity. Learning systems are required to acquire skills and develop task knowledge of how to act. Algorithms for learning and extracting important features of task actions are fundamental in order to build intelligent behaviours. Chapter 3 reviewed the field of *Learning from Demonstration (LfD)* and the process and methods used for learning and encoding the models of the robot skills. *LfD* formulates user-friendly methods by which a human user can teach a robot how to accomplish a given task, simply by demonstrating this task, and generalizing the demonstrated movements across a set of demonstrations. Also, different methodologies for gathering the demonstrations were reviewed, various techniques for teaching and building the demonstrations datasets were presented like, kinaesthetic teaching, visual demonstrations, motion capturing systems to record demonstrations, or generating robot trajectories with virtual reality or simulated environments.

For teaching and learning the different sets of skills *LfD* algorithms and modalities were implemented and evaluated. In this thesis the robot skills were learned in a *Dynamical System* approach. The approach is based on learning time independent models of the motion dynamics estimated through a set of first order non-linear multivariate dynamical systems.

Through the work on this thesis, a number of *Imitation Learning* techniques have been studied and implemented in teaching and learning with the robot, the different sets of skills employed in the rest of the framework. Three algorithms to learn the dynamics of demonstrated motions were studied. A first approach was implemented learning the skills with multivariate Gaussian functions; however, this formulation could not guarantee the learning of a stable estimate of the dynamics. The *BM* method was implemented next; this method could produce a model of *DS* with local asymptotic stability at the target. Finally the *SEDS* method was reviewed with two objective functions: *SEDS-likelihood* and *SEDS-MSE*. The *SEDS* formulation to learn the underlying dynamics of a motion can guarantee that estimates of the dynamics are globally asymptotically stable at the target.

Methodologies used for the reproduction of the learned motion dynamics of the robot skills were reviewed, comparing the performance of the methods presented through this work. Validation was performed of the performance of the methods were compared across the demonstrated motions, the estimates of several 2-D and 3-D motions were learned. For learning the *Robot Skill Models* through the experiments presented in this work, the *SEDS-likelihood* was employed. Learning the robot skills with *SEDS* as a model of the motions dynamics has several desirable properties that have been outlined before in other chapters. This allows the robot to have an encoded model generalizing the dynamics of the motion, that can respond to perturbations on the execution of the task and changes to the initial conditions.

Future Work

This work reviewed various topics in the area of learning robot skills and *Imitation Learning*. A module was successfully implemented allowing the robot to learn skills from demonstrations. However, there remain some issues and pertinent consequent studies as follows:

- Different techniques were examined for gathering the teachers' demonstrations in this work, an interesting possible topic requires the study of how demonstrations of the same skill recorded by different techniques can be used jointly for the training of a robot skill with an *LfD* algorithm.
- Also, more in-depth studies and comparisons of the techniques for gathering demonstrations with a user experienced focus on mind would be relevant and helpful for deciding the mechanism by which a skill would better be demonstrated to the teacher.
- The role of the teacher and how the quality of the provided demonstrations influences and determines the robot behaviour has not been fully explored.
- This work reviewed several learning algorithms and settled on using the *SEDS-Likelihood* algorithms since it had better overall results and facilitated implementation. However, many different algorithms exist in the literature, any one of them with their strengths and weaknesses. Mechanism for determining which algorithms could be better suited for the learning of a skill out of the demonstrated data would be an interesting topic for future research.
- Employing different learning algorithms in the same system naturally complicates the interactions that different skills, with different encodings, could have with the rest of the system. This leads to the need to research mechanisms by which different encodings of a skill could be transform from one to another.
- An interesting topic of research, not sufficiently explored, in this work is how the information encoded within the model of a robot skill can be used for the categorization and recognition of skills.

7.2 Discussion on Representation of Robot Skills

For a robotic system to perform different skills and tasks in a changing and unstructured scenario, it is important to endow them with a framework in which to organize their acquired knowledge in a manner that allows it to be retrieved it in order to use it to deal with the current context constraints. In Chapter 4, a knowledge base of skills was developed and implemented. The knowledge base allows for the storage, classification and retrieval of learned models of skills. A knowledge base is populated with robot available skills, learned by demonstration, for later reproduction. A method for the representation of the knowledge of the skills and task constraints needed for reproduction was developed.

The learned motion primitives can be used as a way of having comprehensive repertoires of robot skills. Chapter 4 reviewed similar approaches aimed at building repertoires of basic robot motor skills which can represent a basic set of elementary movement primitives. Most of these approaches generally offered little advice on how the library of skills could be used to select and adapt the primitives to deal with different conditions, or their mechanisms for representing their knowledge.

An important challenge for robotics, and particularly for robots acting on unstructured dynamic environments, is in dealing with internal representation and understanding of the world. The embodied view of cognition call for representations to be limited, physically grounded to the environment and oriented towards a particular use. Approaches from artificial intelligence and logic base reasoning see the world more as discrete time experiences. Yet the state and action representations are dynamic. The robot actions and thinking must be processes of interacting change in the environment. The dynamical system theory approach is an appropriate alternative to the traditional formats of representations. Dynamical systems can store knowledge and have this stored knowledge influence their behaviour.

The principal aim for the humanoid robot is to take actions, as situated agents, that are appropriate to their circumstances. Fitting representations are essential for this goal. Thinking in terms of actions, and objects, is not only intuitive but also convenient for a representational undertaking in robotics. Object and actions are at the basis of robot performance. However, representational attributions must also include information about the world and situations, events and goals, for effective situated performance. Our representations included information about objects and actions, the world and situations, events and goals, for effective situated performance. A structure built on frames has been adopted in this work. The knowledge of the environment and goals is represented in terms of World Event Frames and Task Event Frames, with Object and Action Frames representing knowledge about available objects and actions respectively. From their knowledge, an Active View Event Frame is built from the focused knowledge promoting the agent's execution.

Future Work

This work has introduced many issues in the framework of knowledge and representations for robot skills. Some possible consequent studies are as follows:

- Work on this thesis has tried to build a comprehensive set of skills knowledge, however, the sets we are able to build are still limited compared to what a robot working in a real world situation would be able to develop over time. Further research is needed for topics of decision making and conflict resolution over the selection of a proper path, when there are two or more viable choices for action.

- The topic of reasoning is a very large subject and there are several different approaches and applications for reasoning with robotics. While studying them was outside the scope of this thesis, future works would benefit from a comparative study of reasoning approaches and the application of different methods as they are best suited to a situation.

- A further point of research is on investigating how to handle inaccurate or unreliable perceptions and information in the system and mechanism by which the knowledge base could recover from erroneous and false assumptions.

- Work on this thesis has not focused on the creation or obtaining of plans, but assumed general plans to be already in the system and loaded into the task event frames; instead it has focused on mechanisms for the robot execution of the tasks' actions. Future work must review the process by which humanoid robots can learn, create, choose and modify their plans of actions.

- There are various different approaches to related topics focused on the management of knowledge by robotic systems. An interesting topic of future research is the study and comparison of these systems; in particular the ones that may be used to complement the framework developed in this work, such as KnowRob or RoboEarth, which could lie at a higher, more abstract level of the cognitive hierarchy while our framework lies at a lower level of action execution.

7.3 Discussion on Generation and Adaptation of Robot Skills

Humanoid robots are required to perform a wide repertoire of task working beside humans in complex dynamic environments. Learning mechanism are important for building up this type of repertoire of robot skills; however, despite the clear advantages of *LfD* approaches it would still be impractical for the human operator to teach the robot the skills for every necessary task and for every foreseen situation. Efforts to generate robotic skills can only have a real implementation value for developing humanoid robotic systems, if the models of the skill can be operated upon to generate new behaviours of increasing levels of complexity. Therefore, extending the *LfD* approach of learning a skill model in a way that allows the adaptation of a robot previously learned motion skills to new unseen contexts is necessary.

The algorithms developed for the generation and adaptation of the robot skills were reviewed in Chapter 5. In that chapter, the process by which the model of a skill can be adapted to reproduce a new task using the already learned model of a robot skill and the extracted constraints knowledge of the current task was described. Different modalities were developed and implemented that allow for the adaptation and generation of new skill models based on the already learned models of skills, stored in the knowledge base. Different modes are presented for the adaptation, update, merger, and combination of the *Robot Skills Models*.

Models of a skill must be updatable; when given new information for the representation of a skill, the system must allow for the models to be improved. Updating previously learned skills is a very important ability for humanoid robots, allowing them to increase and improve their available skill set.

Skills can be generated by merging two or more models into a new skill; multiple desired robot skills may be composed from superposition of various models. New models of a skill can be generated by merging two or more models into a new skill in order to expand the robot skill set and increase its range of action.

In order to expand the robot skill set and increase its range of action to encompass a larger spectrum of the attractor dynamics, the *Robot Skills Models* must be combinable into new models. This makes it possible to carry out more complex tasks than those presented during demonstrations, generalizing the models of the skills to regions outside their original demonstrations. One important gain from the combination of robot skills comes from increasing the accuracy of the generalized behaviour. The generation of a model by combining robot skills is necessary in order to improve the task execution.

For humanoid robots to be able of working successfully in the capacity they are envisioned, it is of vital importance that they present ample and robust skill sets. Being capable of expanding a robot set of learned skills is clearly an important issue as robots will be asked to perform an increasing number of activities and learning and programming every possible skill into the robot is infeasible. The ability to learn robot skills is a key aspect in achieving this; yet learning by itself is not sufficient, the capacity to operate over the learned robot skill, such as the merger, update and combination of skills is necessary.

Future Work

Throughout this work we have explored many different issues for the generation and adaptation of robot skills. Some promising, derivable topics for future research are as follows:

- Recovering and handling safely interruptions, abrupt distortions, or miss executions during skill reproduction is an important issue which has not been fully explored during this work.

- For the evaluations performed during this work, a relative limited set of skills was used in which discriminating among robot skills was not an issue. An important topic for future research is evaluating how can the system select the proper skill primitives out of different competing robot skills.

- The methods developed in this work for the update, merger and combination were evaluated off-line. Future work must focus on evaluating the viability of performing the developed methods in real time execution.

- The methods developed in this work for operating on the robot skills rely on heuristic methods with a human input in selecting certain appropriate parameters. Future research must evaluate methods by which the system could autonomously determine the proper parameters for the desired performance.

- Sequencing and transition operations between robot skill models in order to generate complex behaviours with smooth transitions is an important issue for further exploration.

7.4 Discussion on Reproduction of Robot Skills

In this work a framework has been developed for the generation and adaptation of learned models of a skill for complying with task constraints. The framework is meant to provide humanoid robots with systems that allow them to continuously learn new skills, represent their skills' knowledge, and adapt their existing skills to new contexts, as well as to robustly reproduce new behaviours in a dynamical environment. The framework for learning and adaptation of robot skills is made up from several modules, as represented by the diagram on Figure 6.1. The framework is formed by modules for the learning of robot skills, the perception and interaction with the environment, the management and representation of skill knowledge, the generation and adaptation of skill models, and the reproduction of robot skills.

The development and implementation of the framework and the different modules that compose the framework have been described throughout this work. A module for the robot skill learning based on the *LfD* paradigm was implemented. There are three subsystems in this module; a subsystem for gathering demonstration data; a subsystem for building an estimate of the demonstration with the learning algorithm *SEDS*; and a subsystem for encoding the robot skill model. A module for the knowledge base system was also implemented. There are four subsystems in this module; a subsystem for the data entry to the knowledge base; a subsystem for the data extraction from the knowledge base; a subsystem for the knowledge base data storage; and a subsystem for the knowledge base data management. Operation and development of the robot skill generation and adaptation module was also described. There are three subsystems in this module; a subsystem for extracting data from the knowledge base; a subsystem for operating upon the robot skill with the adaptation algorithm; and a subsystem for generating the task models. A robot skill reproduction module, in charge of producing the adequate control signals to the robot for the reproduction of robot skills, was implemented. This module has three subsystems; a subsystem for computing the regression of the model with *GMR* to obtain the desired target commands; a subsystem for producing the adequate control signals form the target commands; and a subsystem to communicate the control signals to the robot.

Chapter 6 presented the practical experimentation and evaluation of the reproduction of skills in the proposed framework. The experimental scenarios are described and results and analysis are presented for the validation of the framework proposed throughout this work. Different evaluation scenarios were developed to test the performance of the various modules implemented in our framework and to provide separate validation for the operation of the system. Demonstrations are organized over three major scenarios to provide separate validation for the knowledge base system, the generation and adaptation system, and the complete developed framework. The proposed framework was demonstrated with a commercial humanoid robot *HOAP-3*, endowing it with the capacity to learn skill models from a teacher demonstration and to store them in a knowledge base, and adapt the learned models of a skill to reproduce the required skills in different contexts.

Future Work

The work carried out in this thesis has led to the development and implementation of a framework for the learning and adaptation of robot skills evaluated in the *HOAP-3* humanoid robot. But some minor issues remain and other questions have arisen during the development process:

- To validate the developed framework we use the *HOAP-3* robot. The *HOAP-3* provides a readily available humanoid testing platform, however, the *HOAP-3* robot still has many limitations. First, while its small size facilitates the robots stability and control it severely limits the manipulation capabilities and range of operation of the *HOAP-3* robot. It is also unequipped to handle most objects, either because of its size, shape or weight, limiting the actual number of tasks the robot is able to perform.

- Developments in humanoid robotics have been marred by many of the same problems; different issues severely limit their operation. Current performance levels of humanoid robotic platforms are far from the expected goal of a robotic partner working alongside its human co-workers. Though many advances have been made, there is still much work to be done.

- Through this work many challenges in relation to humanoid robotics have been outlined. Perhaps the most important standing challenge is in relation to working on the integration of solutions for all the different challenges at the same time. A final answer for these challenges must come by working from the ground up on solutions that foster each other in generating the desired behaviours.

- One major problem for the development of humanoid robots is the need for the robots to replicate behaviours and performances like those of humans. These difficulties are not only in relation to the mechanical challenges, but also in the problem of comprehending human behaviour. There is no clear cut understanding about the mechanism by which humans' cognitive processes develop. This lack of knowledge and understanding of the internal workings of human intelligence makes reproducing these behaviours an extremely complicated challenge. Work on robotics, artificial intelligence and cognitive science must work out from theories and reasonable assumptions and continuously review and update them as continuous developments shed new light on the problem.

- The framework developed in this work aims at providing the robot with essential cognitive abilities for the learning and adaptation of skills. The framework has been devised as a bottom level module that could be part of the hierarchy in a more complex system, with the goal of providing a minimum functional degree of intelligence for a humanoid robot which would be continuously increased as the system develops further in a bottom-up approach. Future work will constantly focus on augmenting the framework cognitive capacities to generate better, more intelligent behaviours.

- By choosing to start from a bottom level definition of intelligence many assumptions and simplifications are made; this limits the possible scope of performance for the robots while reducing the complexity of the systems. These issues must be handled and solved in future work as we continue to improve the system and make it capable of performing ever more complex behaviours.

- The skill learning module provides effective means for teaching the robot the desired skills. However, the teaching process is not as smooth and streamlined as it could aspire to be, and a certain level of practice and familiarity with the robot platform is required from the teacher in order to be efficient at providing demonstrations. Future work must concentrate on topics of human-robot interaction to improve the demonstration approach.

- The perception module implemented in this work was very simple; it focused only on recognizing objects by their color and size. This of course is very limited; performance was also less reliable with changing lightning conditions. Future work must develop the perception system further, or better yet work to integrate existing more advance solutions with the rest of our framework.

- The skill knowledge module affords the robot mechanisms by which to select skills to reproduce in different contexts. The implemented system is capable of performing under the demonstrated scenarios. However, these demonstrations are still limited in terms of the number of possible choices and situations they have to handle. Future work must provide comprehensive evaluations of capabilities and limitations of the skill knowledge module in a larger range of scenarios.

- The skill adaptation module proves functional for the requirements under the designed demonstrated scenarios. However, the module in its current implementation requires supervision from the operating user, future work must always increase the degree of autonomy for the overall system. Also, future work would benefit from testing and user evaluations employing different users with varying levels of expertise.

- The implemented skill reproduction module allows satisfactory control of the robot performance in reproducing various task. Future work is required to enhance the performance of the robot reproductions, particularly for improving execution speed and providing more natural, human-like, movements. Additionally, future work must test and implement the developed framework on the full scale humanoid robot platform *TEO* being developed at Universidad Carlos III de Madrid.

- A final important point for future research is in the integration of our framework with other existing approaches. Working on developing our system under the *ROS (Robot Operating System)* software framework would be an advantage since *ROS* is quickly becoming a go to standard for robotics development and many existing *ROS* enabled solutions are available.

BIBLIOGRAPHY

[Agre and Chapman, 1987] Agre, P. E. and Chapman, D. (1987). Pengi: an implementation of a theory of activity. In Forbus, K. D. and Shrobe, H. E., editors, *Proceedings of the Sixth National Conference on Artificial Intelligence (AAAI-87)*, pages 268–272. Morgan Kaufmann.

[Akachi et al., 2005] Akachi, K., Kaneko, K., Kanehira, N., Ota, S., Miyamori, G., Hirata, M., Kajita, S., and Kanehiro, F. (2005). Development of humanoid robot HRP-3P. *5th IEEE-RAS International Conference on Humanoid Robots, 2005.*, pages 50–55.

[Albers et al., 2006] Albers, A., Brudniok, S., Ottnad, J., Sauter, C., and Sedchaicham, K. (2006). Body of a new Humanoid Robot - the Design of ARMAR III. 00:308–313.

[Albus, 1991] Albus, J. (1991). Outline for a theory of intelligence. *Systems, Man and Cybernetics, IEEE Transactions on*, 21(3):473 –509.

[Albus et al., 1987] Albus, J., McCain, H., Lumia, R., and of Standards, U. S. N. B. (1987). *NASA/NBS Standard Reference Model for Telerobot Control System Architecture (NASREM)*. NBS Technical Note. U.S. Department of Commerce, National Bureau of Standards.

[Albus, 1997] Albus, J. S. (1997). The nist real-time control system (rcs): an approach to intelligent systems research. *J. Exp. Theor. Artif. Intell.*, 9(2-3):157–174.

[Albus and Barbera, 2005] Albus, J. S. and Barbera, A. J. (2005). Rcs: A cognitive architecture for intelligent multi-agent systems. *Annual Reviews in Control*, 29(1):87 – 99.

[Aleotti and Caselli, 2006] Aleotti, J. and Caselli, S. (2006). Robust trajectory learning and approximation for robot programming by demonstration. *Robotics and Autonomous Systems*, 54(5):409–413.

[Alissandrakis et al., 2002a] Alissandrakis, A., Nehaniv, C., and Dautenhahn, K. (2002a). Do as i do: Correspondences across different robotic embodiments. *Proceedings 5th German Workshop on Artificial Life. Lubeck*, (GWAL5):143–152.

[Alissandrakis et al., 2002b] Alissandrakis, A., Nehaniv, C., and Dautenhahn, K. (2002b). Imitation with alice: learning to imitate corresponding actions across

dissimilar embodiments. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 32(4):482 – 496.

[Alonso-Martin and Salichs, 2011] Alonso-Martin, F. and Salichs, M. A. (2011). Integration of a voice recognition system in a social robot. *Cybern. Syst.*, 42(4):215–245.

[Ambros-Ingerson and Steel, 1988] Ambros-Ingerson, J. and Steel, S. (1988). Integrating Planning, Execution and Monitoring. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 83–88, St. Paul, MN.

[Ambrose et al., 2000] Ambrose, R., Aldridge, H., Askew, R., Burridge, R., Bluethmann, W., Diftler, M., Lovchik, C., Magruder, D., and Rehnmark, F. (2000). Robonaut: Nasa's space humanoid. *Intelligent Systems and their Applications, IEEE*, 15(4):57–63.

[American-Heritage, 2006] American-Heritage, D. (2006). *The American Heritage Dictionary of the English Language, Fourth Editon: Print and CD-ROM Edition*. AMERICAN HERITAGE DICTIONARY OF THE ENGLISH LANGUAGE. Houghton Mifflin Harcourt.

[Anastasi, 1992] Anastasi, A. (1992). What counselors should know about the use and interpreation of psychological tests. *Journal of Counseling and Development*, 70(5):610 –15.

[Anderson et al., 2004] Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., and Qin, Y. (2004). An integrated theory of the mind. *PSYCHOLOGICAL REVIEW*, 111:1036–1060.

[Anderson, 2003] Anderson, M. L. (2003). Embodied cognition: a field guide. *Artif. Intell.*, 149(1):91–130.

[Arandjelovic and Cipolla, 2005] Arandjelovic, O. and Cipolla, R. (2005). Incremental learning of temporally-coherent gaussian mixture models. In Clocksin, W. F., Fitzgibbon, A. W., and Torr, P. H. S., editors, *BMVC*. British Machine Vision Association.

[Arbulú et al., 2009] Arbulú, M., Kaynov, D., Cabas, L. M., and Balaguer, C. (2009). The rh-1 full-size humanoid robot: design, walking pattern generation and control. *Journal of Applied Bionics and Biomechanics.*, 6(3):301–344.

[Argall et al., 2009] Argall, B. D., Chernova, S., Veloso, M., and Browning, B. (2009). A survey of robot learning from demonstration. *Robot. Auton. Syst.*, 57(5):469–483.

[Arie et al., 2012] Arie, H., Arakaki, T., Sugano, S., and Tani, J. (2012). Imitating others by composition of primitive actions: A neuro-dynamic model. *Robotics and Autonomous Systems*, 60(5):729 – 741. <ce:title>Mobiligence: Intelligence for generating adaptive motor function</ce:title>.

[Arkin, 1989a] Arkin, R. C. (1989a). Motor Schema – Based Mobile Robot Navigation. *The International Journal of Robotics Research*, 8(4):92–112.

[Arkin, 1989b] Arkin, R. C. (1989b). Navigational path planning for a vision-based mobile robot. *Robotica*, 7(01):49–63.

[Arkin and Balch, 1997] Arkin, R. C. and Balch, T. (1997). Aura: Principles and practice in review. *Journal of Experimental and Theoretical Artificial Intelligence*, 9:175–189.

[Arkin and Mackenzie, 1994] Arkin, R. C. and Mackenzie, D. C. (1994). Planning to behave: A hybrid deliberative/reactive robot control architecture for mobile manipulation. In *International Symposium on Robotics and Manufacturing, Maui, HI*, pages 5–12.

[Asfour et al., 2006] Asfour, T., Regenstein, K., and Azad, P. (2006). ARMAR-III: An integrated humanoid platform for sensory-motor control. *Humanoid Robots,*, pages 169–175.

[Atkeson et al., 2000] Atkeson, C., Hale, J., and Pollick, F. (2000). Using humanoid robots to study human behavior. *Systems and their*.

[Baglini et al., 2010] Baglini, E., Cannata, G., and Mastrogiovanni, F. (2010). Design of an embedded networking infrastructure for whole-body tactile sensing in humanoid robots. In *Humanoid Robots (Humanoids), 2010 10th IEEE-RAS International Conference on*, pages 671 –676.

[Balaguer et al., 2011] Balaguer, C., Jardón, A., Monje, C., Bonsignorio, F., Stoelen, M., Martinez, S., and Victores, J. (2011). Sultan: Simultaneous user learning and task execution, and its application in assistive robotics. In WadaEditors, K., editor, *Workshop on New and Emerging Technologies in Assistive Robotics at IROS 2011*, pages 6–8.

[Balch and Arkin, 1998] Balch, T. and Arkin, R. (1998). Behavior-based formation control for multirobot teams. *Robotics and Automation, IEEE Transactions on*, 14(6):926 –939.

[Barck-Holst et al., 2009] Barck-Holst, C., Ralph, M., Holmar, F., and Kragic, D. (2009). Learning grasping affordance using probabilistic and ontological approaches. In *Advanced Robotics, 2009. ICAR 2009. International Conference on*, pages 1–6.

[Bartlett and Bartlett, 1995] Bartlett, F. and Bartlett, F. (1995). *Remembering: A Study in Experimental and Social Psychology*. Cambridge books online. Cambridge University Press.

[Basilio, 2013] Basilio, N. (2013). Mldemos visualization tool for machine learning algorithms. http://mldemos.b4silio.com/.

[Baum, 2003] Baum, W. (2003). *Understanding Behaviorism*. Behavior analysis and society series. John Wiley & Sons.

[Bechtel, 1998] Bechtel, W. (1998). Representations and cognitive explanations: Assessing the dynamicistś challenge in cognitive science. *Cognitive Science*, 22(3):295–318.

[Beer, 2000] Beer, R. D. (2000). Dynamical approaches to cognitive science. *Trends in Cognitive Sciences*, 4(3):91–99.

[Behnke, 2008] Behnke, S. (2008). Humanoid Robots-From Fiction to Reality? *Künstliche Intelligenz Heft*, 4(December):5–9.

[Belleghem et al., 1995] Belleghem, K. V., Denecker, M., and Schreye, D. D. (1995). Combining situation calculus and event calculus. In *In Proc. of the twelfth international conference on Logic Programming*, pages 83–97. MIT Press.

[Bertsch and Hafner, 2009] Bertsch, F. and Hafner, V. (2009). Real-time dynamic visual gesture recognition in human-robot interaction. In *Humanoid Robots, 2009. Humanoids 2009. 9th IEEE-RAS International Conference on*, pages 447 –453.

[Billard et al., 2008] Billard, A., Calinon, S., Dillmann, R., and Schaal, S. (2008). Robot programming by demonstration. In Siciliano, B. and Khatib, O., editors, *Handbook of Robotics*, pages 1371–1394. Springer, Secaucus, NJ, USA.

[Bishop, 2006] Bishop, C. (2006). *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer.

[Bonasso, 1991] Bonasso, R. P. (1991). Integrating reaction plans and layered competences through synchronous control. In *Proceedings of the 12th international joint conference on Artificial intelligence - Volume 2*, IJCAI'91, pages 1225–1231, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[Bonasso et al., 1995] Bonasso, R. P., Kortenkamp, D., Miller, D. P., and Slack, M. (1995). Experiences with an architecture for intelligent, reactive agents. *Journal of Experimental and Theoretical Artificial Intelligence*, 9:237–256.

[Borghi and Cimatti, 2010] Borghi, A. M. and Cimatti, F. (2010). Embodied cognition and beyond: Acting and sensing the body. *Neuropsychologia*, 48(3):763 – 773. <ce:title>The Sense of Body</ce:title>.

[Brachman and Levesque, 2004] Brachman, R. and Levesque, H. (2004). *Knowledge Representation and Reasoning*. The Morgan Kaufmann Series in Artificial Intelligence. Elsevier Science.

[Bratman et al., 1988] Bratman, M. E., Israel, D. J., and Pollack, M. E. (1988). Plans and resource-bounded practical reasoning.

[Breazeal, 2001] Breazeal, C. (2001). Socially intelligent robots: research, development, and applications. In *Systems, Man, and Cybernetics, 2001 IEEE International Conference on*, volume 4, pages 2121 –2126 vol.4.

[Brooks, 1986] Brooks, R. (1986). A robust layered control system for a mobile robot. *Robotics and Automation, IEEE Journal of*, 2(1):14 – 23.

[Brooks, 1996] Brooks, R. (1996). Behavior-based humanoid robotics. *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems. IROS '96*, 1:1–8.

[Brooks et al., 1999] Brooks, R., Breazeal, C., Marjanović, M., Scassellati, B., and Williamson, M. (1999). The Cog project: Building a humanoid robot. In *Computation for metaphors, analogy, and agents*, pages 52–87. Springer-Verlag.

[Brooks, 1990] Brooks, R. A. (1990). Elephants don't play chess. *Robotics and Autonomous Systems*, 6:3–15.

[Bueno et al., 2012] Bueno, J. G., Gonzalez-Fierro, M., Balaguer, C., and Moreno, L. (2012). Facial gesture recognition using active appearance models based on neural evolution. In *Human-Robot Interaction (HRI), 2012 7th ACM/IEEE International Conference on*, pages 133 –134.

[Burghart et al., 2005] Burghart, C., Mikut, R., and Stiefelhagen, R. (2005). A cognitive architecture for a humanoid robot: A first approach. *Humanoid Robots,*, pages 357–362.

[Calinon, 2009] Calinon, S. (2009). *Robot Programming by Demonstration: A Probabilistic Approach*. EPFL/CRC Press.

[Calinon and Billard, 2008] Calinon, S. and Billard, A. (2008). A framework integrating statistical and social cues to teach a humanoid robot new skills. Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Workshop on Social Interaction with Intelligent Indoor Robots, 2008.

[Calinon et al., 2007] Calinon, S., Guenter, F., and Billard, A. (2007). On learning, representing and generalizing a task in a humanoid robot. *IEEE Transactions on Systems, Man and Cybernetics, Part B. Special issue on robot learning by observation, demonstration and imitation*, 37(2):286–298.

[Calinon et al., 2012] Calinon, S., Li, Z., Alizadeh, T., Tsagarakis, N. G., and Caldwell, D. G. (2012). Statistical dynamical systems for skills acquisition in humanoids. In *Proc. IEEE Intl Conf. on Humanoid Robots (Humanoids)*.

[Calinon et al., 2010] Calinon, S., Sauser, E., Billard, A., and Caldwell, D. (2010). Evaluation of a probabilistic approach to learn and reproduce gestures by imitation. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 2671–2676.

[Cassimatis et al., 2004] Cassimatis, N. L., Trafton, J. G., Bugajska, M. D., and Schultz, A. C. (2004). Integrating cognition, perception and action through mental simulation in robots. *Robotics and Autonomous Systems*, 49(1 - 2):13 – 23. <ce:title>Knowledge Engineering and Ontologies for Autonomous Systems 2004 AAAI Spring Symposium</ce:title>.

[Chapman and Agre, 1987] Chapman, D. and Agre, P. E. (1987). Abstract reasoning as emergent from concrete activity. In Georgeff, M. P. and Lansky, A. L., editors, *Reasoning about Actions and Plans*, pages 411–424. Kaufmann, Los Altos, CA.

[Chatzis et al., 2012] Chatzis, S. P., Korkinof, D., and Demiris, Y. (2012). A Quantum-Statistical Approach Toward Robot Learning by Demonstration. *IEEE Transactions on Robotics*.

[Chen et al., 2007] Chen, J., Haas, E., and Barnes, M. (2007). Human performance issues and user interface design for teleoperated robots. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 37(6):1231 –1245.

[Chiaverini and Siciliano, 1999] Chiaverini, S. and Siciliano, B. (1999). The unit quaternion: a useful tool for inverse kinematics of robot manipulators. *Syst. Anal. Model. Simul.*, 35(1):45–60.

[Choi et al., 2009] Choi, D., Kang, Y., and Lim, H. (2009). Knowledge-based control of a humanoid robot. *Intelligent Robots and*, pages 3949–3954.

[Clark, 1997] Clark, A. (1997). The dynamical challenge. *Cognitive Science*, 21(4):461–481.

[Clark, 2004] Clark, A. (2004). Mind and causality. Advances in Consciousness Research, chapter Embodiment and the Philosophy of Mind. John Benjamins Pub.

[Clark and Grush, 1999] Clark, A. and Grush, R. (1999). Towards a cognitive robotics. *Adapt. Behav.*, 7(1):5–16.

[Cohen et al., 1989] Cohen, P. R., Greenberg, M. L., Hart, D. M., and Howe, A. E. (1989). Trial by fire: understanding the design requirements for agents in complex environments. *AI Mag.*, 10(3):34–48.

[Cohn et al., 1996] Cohn, D. A., Ghahramani, Z., and Jordan, M. I. (1996). Active learning with statistical models. *J. Artif. Intell. Res. (JAIR)*, 4:129–145.

[Company, 2012] Company, S. R. (2012). The shadow dextrous hand. http://www.shadow.org.uk/products/newhand.shtml.

[Connell, 1992] Connell, J. (1992). Sss: a hybrid architecture applied to robot navigation. In *Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on*, pages 2719 –2724 vol.3.

[Consortium, 2012] Consortium, T. R. (2012). icub. http://www.icub.org/.

[Coradeschi et al., 2006] Coradeschi, S., Ishiguro, H., and Asada, M. (2006). Human-inspired robots. *Intelligent Systems,*.

[Şahin et al., 2007] Şahin, E., Çakmak, M., Doğar, M. R., Uğur, E., and Üçoluk, G. (2007). To afford or not to afford: A new formalization of affordances toward affordance-based robot control. *Adaptive Behavior - Animals, Animats, Software Agents, Robots, Adaptive Systems*, 15(4):447–472.

[Dasgupta and Schulman, 2000] Dasgupta, S. and Schulman, L. J. (2000). A two-round variant of em for gaussian mixtures. In *UAI '00: Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, pages 152–159, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[Davis et al., 1993] Davis, R., Shrobe, H. E., and Szolovits, P. (1993). What is a knowledge representation? *AI Magazine*, 14(1):17–33.

[De Silva and Ekanayake, 2008] De Silva, L. and Ekanayake, H. (2008). Behavior-based robotics and the reactive paradigm a survey. In *Computer and Information Technology, 2008. ICCIT 2008. 11th International Conference on*, pages 36 –43.

[Deleuze, 1985] Deleuze, G. (1985). *Kant's Critical Philosophy: The Doctrine of the Faculties*. University of Minnesota Press.

[Dempster et al., 1977] Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B*, 39:1–38.

[Diankov and Kuffner, 2008] Diankov, R. and Kuffner, J. (2008). OpenRAVE: A Planning Architecture for Autonomous Robotics. Technical Report CMU-RI-TR-08-34, Robotics Institute, Pittsburgh, PA.

[Diftler et al., 2011] Diftler, M., Mehling, J., Abdallah, M., Radford, N., Bridgwater, L., Sanders, A., Askew, R., Linn, D., Yamokoski, J., Permenter, F., Hargrave, B., Piatt, R., Savely, R., and Ambrose, R. (2011). Robonaut 2 - the first humanoid robot in space. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 2178 –2183.

[Dreyfus et al., 2000] Dreyfus, H., Dreyfus, S., and Athanasiou, T. (2000). *Mind Over Machine*. Simon & Schuster.

[Duch et al., 2008] Duch, W., Oentaryo, R. J., and Pasquier, M. (2008). Cognitive architectures: Where do we go from here? In *Proceedings of the 2008 conference on Artificial General Intelligence 2008: Proceedings of the First AGI Conference*, pages 122–136, Amsterdam, The Netherlands, The Netherlands. IOS Press.

[Dyer et al., 2013] Dyer, S., Martin, J., and Zulauf, J. (2013). Motion capture white paper. ftp://ftp.sgi.com/sgi/AW/jam/mocap/MoCapWP_v2.0.html.

[Evrard et al., 2009] Evrard, P., Mansard, N., Stasse, O., Kheddar, A., Schauss, T., Weber, C., Peer, A., and Buss, M. (2009). Intercontinental, multimodal, wide-range tele-cooperation using a humanoid robot. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 5635 –5640.

[Fangzhen, 2007] Fangzhen, L. (2007). Situation calculus. In van Harmelen, F., van Harmelen, F., Lifschitz, V., and Porter, B., editors, *Handbook of Knowledge Representation*, pages 649–669. Elsevier Science, San Diego, USA.

[Ferguson, 1991] Ferguson, I. A. (1991). Towards an architecture for adaptive, rational, mobile agents. In Werner, E. and Demazeau, Y., editors, *Decentralized AI 3 — Proceedings of the Third European Workshop on Modelling Autonomous Agents in a Multi-Agent World (MAAMAW-91)*, pages 249–262. Elsevier Science Publishers B.V.: Amsterdam, The Netherlands.

[Fikes and Nilsson, 1971] Fikes, R. E. and Nilsson, N. J. (1971). Strips: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2(3-4):189–208.

[Fod et al., 2000] Fod, A., Matarić, M. J., and Jenkins, O. C. (2000). Automated derivation of primitives for movement classification. In *In Proc. of First IEEE-RAS International Conference on Humanoid Robots*.

[Fong et al., 2002] Fong, T., Thorpe, C., and Baur, C. (2002). Collaboration, dialogue, and human-robot interaction. In *In 10th International Symposium of Robotics Research*. Springer-Verlag.

[Forte et al., 2012] Forte, D., Gams, A., Morimoto, J., and Ude, A. (2012). On-line motion synthesis and adaptation using a trajectory database. *Robot. Auton. Syst.*, 60(10):1327–1339.

[Funge, 1999] Funge, J. (1999). Representing knowledge within the situation calculus using interval-valued epistemic fluents. *Journal of Reliable Computing*, 5.

[Galindo et al., 2005] Galindo, C., Gonzalez, J., and Fernández-Madrigal, J. (2005). An architecture for cognitive human-robot integration. Application to rehabilitation robotics. In *Mechatronics and Automation, 2005 IEEE International Conference*, volume 1, pages 329–334. IEEE.

[Gardner, 1993] Gardner, H. (1993). *Frames Of Mind: The Theory Of Multiple Intelligences*. Sección de obras de psicología, psiquiatría y psicoanálisis. Basic Books.

[Gat, 1992] Gat, E. (1992). Integrating planning and reacting in a heterogeneous asynchronous architecture for controlling real-world mobile robots. In *Proceedings of the tenth national conference on Artificial intelligence*, AAAI'92, pages 809–815. AAAI Press.

[Gat, 1997] Gat, E. (1997). On three-layer architectures. *Artificial intelligence and mobile robots*, pages 195–210.

[Gee et al., 2005] Gee, F., Browne, W., and Kawamura, K. (2005). Uncanny valley revisited. *ROMAN 2005. IEEE International Workshop on Robot and Human Interactive Communication, 2005.*, pages 151–157.

[Geib et al., 2006] Geib, C., Mourao, K., Petrick, R., Pugeault, N., Steedman, M., Krueger, N., and Wörgötter, F. (2006). Object action complexes as an interface for planning and robot control. *IEEE RAS*, Int Conf. Humanoid Robots(Genova):Dec. 4–6, 2006.

[Gibson, 1986] Gibson, J. (1986). *The Ecological Approach to Visual Perception*. Resources for ecological psychology. LAWRENCE ERLBAUM ASSOC Incorporated.

[Glassmire et al., 2004] Glassmire, J., O'Malley, M., Bluethmann, W., and Ambrose, R. (2004). Cooperative manipulation between humans and teleoperated agents. In *Haptic Interfaces for Virtual Environment and Teleoperator Systems, 2004. HAPTICS '04. Proceedings. 12th International Symposium on*, pages 114 – 120.

[Gomez et al., 2012a] Gomez, J., Alvarez, D., Garrido, S., and Moreno, L. (2012a). Kinesthetic teaching via fast marching square. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 1305–1310.

[Gomez et al., 2012b] Gomez, R., Nakamura, K., Kawahara, T., and Nakadai, K. (2012b). Multi-party human-robot interaction with distant-talking speech recognition. In *Human-Robot Interaction (HRI), 2012 7th ACM/IEEE International Conference on*, pages 439 –446.

[Gorostiza et al., 2006] Gorostiza, J., Barber, R., Khamis, A., Pacheco, M., Rivas, R., Corrales, A., Delgado, E., and Salichs, M. (2006). Multimodal human-robot interaction framework for a personal robot. In *Robot and Human Interactive Communication, 2006. ROMAN 2006. The 15th IEEE International Symposium on*, pages 39 –44.

[Gribovskaya and Billard, 2008] Gribovskaya, E. and Billard, A. (2008). Combining dynamical systems control and programming by demonstration for teaching discrete bimanual coordination tasks to a humanoid robot. In *Proceeding of IEEE/ACM International Conference on Human-Robot Interaction*.

[Gribovskaya and Billard, 2009] Gribovskaya, E. and Billard, A. (2009). Learning Nonlinear Multi-Variate Motion Dynamics for Real- Time Position and Orientation Control of Robotic Manipulators. In *Proceedings of 9th IEEE-RAS International Conference on Humanoid Robots*.

[Gribovskaya et al., 2010] Gribovskaya, E., Zadeh, K., Mohammad, S., and Billard, A. (2010). Learning Nonlinear Multivariate Dynamics of Motion in Robotic Manipulators [accepted]. *International Journal of Robotics Research*.

[Grollman and Jenkins, 2008] Grollman, D. and Jenkins, O. (2008). Sparse incremental learning for interactive robot control policy estimation. In *Robotics and*

Automation, 2008. ICRA 2008. IEEE International Conference on, pages 3315–3320.

[Gruber, 2009] Gruber, T. (2009). Ontology. In Liu, L. and Özsu, M. T., editors, *Encyclopedia of Database Systems*, pages 1963–1965.

[Guan et al., 2006] Guan, Y., Neo, E. S., Yokoi, K., and Tanie, K. (2006). Stepping over obstacles with humanoid robots. *Robotics, IEEE Transactions on*, 22(5):958 –973.

[Haasch et al., 2005] Haasch, A., Hofemann, N., Fritsch, J., and Sagerer, G. (2005). A multi-modal object attention system for a mobile robot. In *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 2712 – 2717.

[Hall et al., 2005] Hall, P. M., Hicks, Y., and Robinson, T. (2005). A method to add gaussian mixture models. Other. ID number: CSBU-2005-03.

[Harada et al., 2007a] Harada, K., Kajita, S., Kanehiro, F., Fujiwara, K., Kaneko, K., Yokoi, K., and Hirukawa, H. (2007a). Real-time planning of humanoid robot's gait for force-controlled manipulation. *Mechatronics, IEEE/ASME Transactions on*, 12(1):53 –62.

[Harada et al., 2007b] Harada, K., Kajita, S., Kanehiro, F., Fujiwara, K., Kaneko, K., Yokoi, K., and Hirukawa, H. (2007b). Real-time planning of humanoid robot's gait for force-controlled manipulation. *Mechatronics, IEEE/ASME Transactions on*, 12(1):53–62.

[Haselager et al., 2003] Haselager, W., Bongers, R. M., and van Rooij, I. (2003). *Cognitive science, representations and dynamical systems theory.* Studies of Nonlinear Phenomena in Life Sciences. World Scientific.

[Hastie et al., 2009] Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning.* Springer series in statistics. Springer-Verlag New York.

[Hasunuma et al., 2006] Hasunuma, H., Harada, K., and Hirukawa, H. (2006). The tele-operation of the humanoid robot-whole body operation for humanoid robots in contact with environment-. In *Humanoid Robots, 2006 6th IEEE-RAS International Conference on*, pages 333 –339.

[Hayes, 1979] Hayes, P. J. (1979). The logic of frames. In Metzing, D., editor, *Frame Conceptions and Text Understanding*, pages 46–61. Walter de Gruyter and Co., Berlin, Germany.

[Herman, 2007] Herman, I. (2007). *Physics of the Human Body.* Biological And Medical Physics, Biomedical Engineering. Springer.

[Hernández et al., 2009] Hernández, D., Fierro, M. G., Weiss, A., Wurhofer, D., Gribovskaya, E., and Peer, A. (2009). Deliverable 2.8-9: Final report on ist-robotic architectures and models in cwe. Technical report, ROBOT@CWE. Advanced robotic systems in future collaborative working environments.

[Hersch et al., 2008] Hersch, M., Guenter, F., Calinon, S., and Billard, A. (2008). Dynamical system modulation for robot learning via kinesthetic demonstrations. *Robotics, IEEE Transactions on*, 24(6):1463–1467.

[Hersch et al., Dec] Hersch, M., Guenter, F., Calinon, S., and Billard, A. (Dec.). Learning dynamical system modulation for constrained reaching tasks. In *Humanoid Robots, 2006 6th IEEE-RAS International Conference on*, pages 444–449.

[Hirai et al., 1998] Hirai, K., Hirose, M., Haikawa, Y., and Takenaka, T. (1998). The development of honda humanoid robot. In *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, volume 2, pages 1321 –1326 vol.2.

[HONDA, 2012] HONDA (2012). History of asimo. http://asimo.honda.com/asimo-history/.

[Huggins-Daines et al., 2006] Huggins-Daines, D., Kumar, M., Chan, A., Black, A., Ravishankar, M., and Rudnicky, A. (2006). Pocketsphinx: A free, real-time continuous speech recognition system for hand-held devices. In *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, volume 1, page I.

[Hwang et al., 2006] Hwang, Y. K., Choi, K. J., and Hong, D. S. (2006). Self-learning control of cooperative motion for a humanoid robot. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 475 –480.

[Ijspeert et al., 2001] Ijspeert, A., Nakanishi, J., and Schaal, S. (2001). Trajectory formation for imitation with nonlinear dynamical systems. In *ieee international conference on intelligent robots and systems (iros 2001)*.

[Ijspeert et al., 2009] Ijspeert, A. J., Nakanishi, J., Hoffmann, H., Pastor, P., Schaal, S., and F, E. P. (2009). Learning nonlinear dynamical systems models. *Neural Computation*, 12(12):1–33.

[Ijspeert et al., 2002] Ijspeert, A. J., Nakanishi, J., and Schaal, S. (2002). Movement imitation with nonlinear dynamical systems in humanoid robots. In *In IEEE International Conference on Robotics and Automation (ICRA2002*, pages 1398–1403.

[Ijspeert et al., 2003] Ijspeert, A. J., Nakanishi, J., and Schaal, S. (2003). Learning attractor landscapes for learning motor primitives. In *in Advances in Neural Information Processing Systems*, pages 1523–1530. MIT Press.

[Ingrand et al., 1992] Ingrand, F., Georgeff, M., and Rao, A. (1992). An architecture for real-time reasoning and system control. *IEEE Expert*, 7(6):34 –44.

[Ishida et al., 2005] Ishida, H., Nakayama, G., Nakamoto, T., and Moriizumi, T. (2005). Controlling a gas/odor plume-tracking robot based on transient responses of gas sensors. *Sensors Journal, IEEE*, 5(3):537 – 545.

[Jenkins et al., 2007] Jenkins, O., Gonzalez, G., and Loper, M. (2007). Interactive human pose and action recognition using dynamical motion primitives. *International Journal of Humanoid Robotics*, 4(2):365–385.

[Jenkins et al., 2000] Jenkins, O. C., Matarić, M. J., and Weber, S. (2000). Primitive-based movement classification for humanoid imitation. Technical Report IRIS-00-385, Institute for Robotics and Intelligent Systems, University of Southern California.

[Jennings, 1993] Jennings, N. (1993). Specification and implementation of a belief-desire-joint-intention architecture for collaborative problem solving. *Journal of Intelligent and Cooperative Information Systems*, 2:289–318.

[Johnston and Rabe, 2006] Johnston, M. D. and Rabe, K. J. (2006). *Integrated Planning for Telepresence with Time Delays*, pages 140–148. Ieee.

[Jung et al., 2007] Jung, Y., Choi, Y., Park, H., Shin, W., and Myaeng, S.-H. (2007). Integrating Robot Task Scripts with a Cognitive Architecture for Cognitive Human-Robot Interactions. *2007 IEEE International Conference on Information Reuse and Integration*, pages 152–157.

[Kajita et al., 2003] Kajita, S., Kanehiro, F., Kaneko, K., Fujiwara, K., Harada, K., Yokoi, K., and Hirukawa, H. (2003). Biped walking pattern generation by using preview control of zero-moment point. In *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, volume 2, pages 1620 – 1626 vol.2.

[Kajita et al., 2001a] Kajita, S., Kanehiro, F., Kaneko, K., Yokoi, K., and Hirukawa, H. (2001a). The 3d linear inverted pendulum mode: a simple modeling for a biped walking pattern generation. In *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, volume 1, pages 239 –246 vol.1.

[Kajita et al., 2001b] Kajita, S., Kanehiro, F., Kaneko, K., Yokoi, K., and Hirukawa, H. (2001b). The 3d linear inverted pendulum mode: a simple modeling for a biped walking pattern generation. In *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, volume 1, pages 239–246 vol.1.

[Kamio and Iba, 2005] Kamio, S. and Iba, H. (2005). Adaptation technique for integrating genetic programming and reinforcement learning for real robots. *Evolutionary Computation, IEEE Transactions on*, 9(3):318 – 333.

[Kaneko et al., 2004a] Kaneko, K., Kanehiro, F., Kajita, S., Hirukawa, H., Kawasaki, T., Hirata, M., Akachi, K., and Isozumi, T. (2004a). Humanoid robot hrp-2. pages 1083–1090. Proceedings of IEEE International Conference on Robotics and Automation.

[Kaneko et al., 2004b] Kaneko, K., Kanehiro, F., Kajita, S., Hirukawa, H., Kawasaki, T., Hirata, M., Akachi, K., and Isozumi, T. (2004b). Humanoid robot hrp-2. In *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, volume 2, pages 1083–1090.

[Kaneko et al., 2002] Kaneko, K., Kanehiro, F., Kajita, S., Yokoyama, K., Akachi, K., Kawasaki, T., Ota, S., and Isozumi, T. (2002). Design of prototype humanoid robotics platform for hrp. In *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, volume 3, pages 2431 – 2436 vol.3.

[Kawamura et al., 2008] Kawamura, K., Gordon, S. M., Ratanaswasd, P., Erdemir, E., and Hall, J. F. (2008). Implementation of cognitive control for a humanoid robot. *I. J. Humanoid Robotics*, 5(4):547–586.

[Kaynov et al., 2007] Kaynov, D., Hernandez, A., and Balaguer, C. (2007). Joint control of a humanoid robot. In *Humanoid Robots, 2007 7th IEEE-RAS International Conference on*, pages 256 –263.

[Kaynov et al., 2009] Kaynov, D., Soueres, P., Pierro, P., and Balaguer, C. (2009). A practical decoupled stabilizer for joint-position controlled humanoid robots. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 3392 –3397.

[Kelso, 1995] Kelso, J. (1995). *Dynamic Patterns: The Self-Organization of Brain and Behavior*. Complex Adaptive Systems Series. Mit Press.

[Khansari-Zadeh and Billard, 2010a] Khansari-Zadeh, S. and Billard, A. (2010a). Imitation learning of globally stable non-linear point-to-point robot motions using nonlinear programming. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 2676–2683.

[Khansari-Zadeh and Billard, 2011] Khansari-Zadeh, S. and Billard, A. (2011). Learning stable nonlinear dynamical systems with gaussian mixture models. *Robotics, IEEE Transactions on*, 27(5):943 –957.

[Khansari-Zadeh and Billard, 2012] Khansari-Zadeh, S. and Billard, A. (2012). A dynamical system approach to realtime obstacle avoidance. *Autonomous Robots*, 32(4):433–454.

[Khansari-Zadeh and Billard, 2010b] Khansari-Zadeh, S. M. and Billard, A. (2010b). BM: An iterative algorithm to learn stable non-linear dynamical systems with gaussian mixture models. In *Proceeding of the International Conference on Robotics and Automation (ICRA)*, pages 2381–2388.

[Kheddar et al., 2009a] Kheddar, A., Weber, C., Peer, A., and Hernández, D. (2009a). Deliverable 4.1-3: Final report on robot@cwe demonstrators. Technical report, ROBOT@CWE. Advanced robotic systems in future collaborative working environments.

[Kheddar et al., 2009b] Kheddar, A., Weber, C., Peer, A., and Hernández, D. (2009b). Deliverable 4.1-3: Final report on robot@cwe demonstrators. Technical report, ROBOT@CWE. Advanced robotic systems in future collaborative working environments.

[Kieras and Meyer, 1997] Kieras, D. E. and Meyer, D. E. (1997). An overview of the epic architecture for cognition and performance with application to human-computer interaction. *Hum.-Comput. Interact.*, 12(4):391–438.

[Kim et al., 2005] Kim, J.-y., Park, I.-w., Lee, J., Kim, M.-s., Cho, B.-k., and Oh, J.-h. (2005). System Design and Dynamic Walking of Humanoid Robot KHR-2. (April):1431–1436.

[Kim et al., 2010] Kim, K., Lee, J.-Y., Choi, D., Park, J.-M., and You, B.-J. (2010). Autonomous task execution of a humanoid robot using a cognitive model. *2010 IEEE International Conference on Robotics and Biomimetics*, pages 405–410.

[Kober and Peters, 2010] Kober, J. and Peters, J. (2010). Imitation and reinforcement learning. *Robotics Automation Magazine, IEEE*, 17(2):55 –62.

[Konczak, 2005] Konczak, J. (2005). On the notion of motor primitives in humans and robots.

[Kronander and Billard, 2012] Kronander, K. and Billard, A. (2012). Online learning of varying stiffness through physical human-robot interaction. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 1842 –1849.

[Krüger et al., 2009] Krüger, N., Piater, J., Wörgötter, F., Geib, C., Petrick, R., Steedman, M., Ude, A., Asfour, T., Kraft, D., Omrcen, D., Hommel, B., Agostino, A., Kragic, D., Eklundh, J., Krüger, V., and Dillmann, R. (2009). A Formal Definition of Object Action Complexes and Examples at Different Levels of the Process Hierarchy. Technical report, EU project PACO-PLUS.

[Kulvicius et al., 2012] Kulvicius, T., Ning, K., Tamosiunaite, M., and Wörgötter, F. (2012). Joining movement sequences: Modified dynamic movement primitives for robotics applications exemplified on handwriting. *Robotics, IEEE Transactions on*, 28(1):145–157.

[Laird et al., 1987] Laird, J. E., Newell, A., and Rosenbloom, P. S. (1987). Soar: an architecture for general intelligence. *Artif. Intell.*, 33(1):1–64.

[Langley and Cummings, 2004] Langley, P. and Cummings, K. (2004). Hierarchical skills and cognitive architectures. In *Proceedings of the Twenty-Sixth Annual Conference of the Cognitive Science Society*, pages 779–784.

[Langley et al., 2009] Langley, P., Laird, J. E., and Rogers, S. (2009). Cognitive architectures: Research issues and challenges. *Cognitive Systems Research*, 10(2):141–160.

[Legg and Hutter, 2006] Legg, S. and Hutter, M. (2006). A formal measure of machine intelligence. In *In Proc. 15th Annual Machine Learning Conference of Belgium and The Netherlands (Benelearn'06*, pages 73–80.

[Legg and Hutter, 2007] Legg, S. and Hutter, M. (2007). A collection of definitions of intelligence. *Galleria Rassegna Bimestrale Di Cultura*, 157(07-07):1 –12.

[Lemaignan et al., 2010] Lemaignan, S., Ros, R., Mo senlechner, L., Alami, R., and Beetz, M. (2010). ORO, a knowledge management platform for cognitive architectures in robotics. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 3548–3553. IEEE.

[Lengagne et al., 2011] Lengagne, S., Ramdani, N., and Fraisse, P. (2011). Planning and fast replanning safe motions for humanoid robots. *Robotics, IEEE Transactions on*, 27(6):1095 –1106.

[Lenser et al., 2001] Lenser, S., Bruce, J., and Veloso, M. (2001). A modular hierarchical behavior-based architecture. In *RoboCup-2001: The Fifth RoboCup Competitions and Conferences*, pages 423–428. Springer Verlag.

[Levesque and Lakemeyer, 2008] Levesque, H. and Lakemeyer, G. (2008). Chapter 23 Cognitive Robotics. In Frank van Harmelen, V. L. and Porter, B., editors, *Handbook of Knowledge Representation*, volume 6526, chapter 23, pages 869–886. Elsevier.

[MacDorman and Cowley, 2006] MacDorman, K. and Cowley, S. (2006). Long-term relationships as a benchmark for robot personhood. *ROMAN 2006 - The 15th IEEE International Symposium on Robot and Human Interactive Communication*, pages 378–383.

[Maes, 1991a] Maes, P. (1991a). The agent network architecture (ana). *SIGART Bull.*, 2(4):115–120.

[Maes, 1991b] Maes, P. (1991b). *Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back*. Bradford Books. MIT Press.

[Malfaz et al., 2011] Malfaz, M., Castro-Gonzalez, A., Barber, R., and Salichs, M. (2011). A biologically inspired architecture for an autonomous and social robot. *Autonomous Mental Development, IEEE Transactions on*, 3(3):232 –246.

[Martinez et al., 2012] Martinez, S., Monje, C. A., Jardon, A., Pierro, P., Balaguer, C., and Munoz, D. (2012). Teo: Full-size humanoid robot design powered by a fuel cell system. *Cybernetics and Systems*, 43(3):163–180.

[Mataric, 1992] Mataric, M. (1992). Integration of representation into goal-driven behavior-based robots. *Robotics and Automation, IEEE Transactions on*, 8(3):304 –312.

[Mataric, 2000] Mataric, M. (2000). Getting humanoids to move and imitate. *Intelligent Systems and their Applications, IEEE*, 15(4):18–24.

[Mataric, 1997] Mataric, M. J. (1997). Behavior-based control: Examples from navigation, learning, and group behavior. *Journal of Experimental and Theoretical Artificial Intelligence*, 9:323–336.

[Mclachlan and Peel, 2000] Mclachlan, G. and Peel, D. (2000). *Finite Mixture Models*. Wiley Series in Probability and Statistics. Wiley-Interscience.

[Meier et al., 2011] Meier, F., Theodorou, E., Stulp, F., and Schaal, S. (2011). Movement segmentation using a primitive library. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 3407–3412.

[Meystel, 1986] Meystel, A. (1986). Planning in a hierarchical nested controller for autonomous robots. In *Decision and Control, 1986 25th IEEE Conference on*, volume 25, pages 1237 –1249.

[Meystel, 1988] Meystel, A. (1988). Intelligent module for planning/control of master-dependent systems. In *IEA/AIE (Vol. 1)*, pages 620–628.

[Microsoft, 2013] Microsoft (2013). Kinect for xbox 360. http://www.xbox.com/en-US/kinect/.

[Minsky, 1975] Minsky, M. (1975). A framework for representing knowledge. In Winston, P., editor, *The psychology of computer vision*, pages 211–277. McGraw-Hill, New-York.

[Monje et al., 2011] Monje, C., Martinez, S., Jardon, A., and Balaguer, C. (2011). Full-size humanoid robot TEO: Design attending mechanical robustness and energy consumption. *IEEE International Conference on Humanoid Robots*, pages 325–330.

[Monje et al., 2008] Monje, C. A., Pierro, P., and Balaguer, C. (2008). Humanoid robot rh-1 for collaborative tasks: a control architecture for human-robot cooperation. *Applied Bionics and Biomechanics*, 5(4):225–234.

[Moravec et al., 1986] Moravec, H., Kadonoff, M., Benayad-Cherif, F., Franklin, A., Maddox, J., Muller, L., and Sert, B. (1986). Arbitration of multiple control strategies for mobile robots. In *SPIE Proceedings: Advances in Intelligent Robotics Systems*, volume 727.

[Mori et al., 2012] Mori, M., MacDorman, K., and Kageki, N. (2012). The Uncanny Valley [From the Field]. *Robotics Automation Magazine, IEEE*, 19(June):98–100.

[Mueller, 2007] Mueller, E. (2007). Event calculus. In van Harmelen, F., van Harmelen, F., Lifschitz, V., and Porter, B., editors, *Handbook of Knowledge Representation*, pages 649–669. Elsevier Science, San Diego, USA.

[Muelling et al., 2013] Muelling, K., Kober, J., Kroemer, O., and Peters, J. (2013). Learning to select and generalize striking movements in robot table tennis. (3):263–279.

[Müller and Pischel, 1994] Müller, J. P. and Pischel, M. (1994). Modelling interacting agents in dynamic environments. In *ECAI*, pages 709–713.

[Münch et al., 1994] Münch, S., Kreuziger, J., Kaiser, M., and Dillmann, R. (1994). Robot programming by demonstration (rpd) - using machine learning and user interaction methods for the development of easy and comfortable robot programming systems. In *In Proceedings of the 24th International Symposium on Industrial Robots*, pages 685–693.

[Murphy, 2000] Murphy, R. (2000). *An Introduction to AI Robotics*. Intelligent Robotics And Autonomous Agents. MIT Press.

[Nehaniv et al., 1998] Nehaniv, C., Dautenhahn, K., and Press (1998). The Correspondence Problem.

[Nehaniv and Dautenhahn, 2001] Nehaniv, C. L. and Dautenhahn, K. (2001). Like me?- measures of correspondence and imitation. *Cybernetics and Systems*, 32(1):11–51.

[Neo et al., 2008] Neo, E. S., Sakaguchi, T., and Yokoi, K. (2008). A humanoid robot that listens, speaks, sees and manipulates in human environments. In *Multisensor Fusion and Integration for Intelligent Systems, 2008. MFI 2008. IEEE International Conference on*, pages 419–425.

[Neo et al., 2007] Neo, E. S., Yokoi, K., Kajita, S., and Tanie, K. (2007). Whole-body motion generation integrating operator's intention and robot's autonomy in controlling humanoid robots. *Robotics, IEEE Transactions on*, 23(4):763 –775.

[Nicolescu and Mataric, 2002] Nicolescu, M. N. and Mataric, M. J. (2002). A hierarchical architecture for behavior-based robots. In *In Proc., First International Joint Conference on Autonomous Agents and Multi-Agent Systems*, pages 227–233.

[Nicolescu and Mataric, 2003] Nicolescu, M. N. and Mataric, M. J. (2003). Linking perception and action in a control architecture for human-robot domains. In *Proceedings of the 36th Annual Hawaii International Conference on System Sciences (HICSS'03) - Track 5 - Volume 5*, HICSS '03, pages 126.1–, Washington, DC, USA. IEEE Computer Society.

[Nilsson, 1984] Nilsson, N. J. (1984). Shakey the robot. Technical Report 323.

[Nilsson, 2007] Nilsson, N. J. (2007). 50 years of artificial intelligence. chapter The physical symbol system hypothesis: status and prospects, pages 9–17. Springer-Verlag, Berlin, Heidelberg.

[Norman, 1988] Norman, D. (1988). *The Design of Everyday Things*. Number n.º 842 in The Design of Everyday Things. Bantam Doubleday Dell Publishing Group.

[Norman et al., 1980] Norman, D., Shallice, T., and PROCESSING., C. U. S. D. L. J. C. F. H. I. (1980). *Attention to Action: Willed and Automatic Control of Behavior*. Center for Human Information Processing, University of California, San Diego.

[Ogura et al., 2006] Ogura, Y., Aikawa, H., Kondo, H., Morishima, A., ok Lim, H., and Takanishi, A. (2006). Development of a new humanoid robot wabian-2. pages 76–81. International Conference on Proceedings of IEEE Robotics and Automation.

[Ohashi et al., 2007] Ohashi, E., Aiko, T., Tsuji, T., Nishi, H., and Ohnishi, K. (2007). Collision avoidance method of humanoid robot with arm force. *Industrial Electronics, IEEE Transactions on*, 54(3):1632 –1641.

[Okada et al.,] Okada, M., Tatani, K., and Nakamura, Y. Polynomial design of the nonlinear dynamics for the brain-like information processing of whole body motion. In *Robotics and Automation, 2002. Proceedings. ICRA '02. IEEE International Conference on*, volume 2, pages 1410–1415 vol.2.

[O'Reilly et al., 1998] O'Reilly, R. C., Braver, T. S., and Cohen, J. D. (1998). A biologically based computational model of working memory. In Miyake, A. and Shah, P., editors, *Models of Working Memory: Mechanisms of Active Maintenance and Executive Control*. Cambridge University Press, New York.

[Ott et al., 2006] Ott, C., Eiberger, O., Friedl, W., Bauml, B., Hillenbrand, U., Borst, C., Albu-Schaffer, A., Brunner, B., Hirschmuller, H., Kielhofer, S., Konietschke, R., Suppa, M., Wimbock, T., Zacharias, F., and Hirzinger, G. (2006). A humanoid two-arm system for dexterous manipulation. In *Humanoid Robots, 2006 6th IEEE-RAS International Conference on*, pages 276 –283.

[Pack et al., 1997] Pack, R., Wilkes, D., and Kawamura, G. (1997). A software architecture for integrated service robot development. In *Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference on*, volume 4, pages 3774 –3779 vol.4.

[Palm and Iliev, 2010] Palm, R. and Iliev, B. (2010). Learning and adaptation of robot skills using fuzzy models. In *Fuzzy Systems (FUZZ), 2010 IEEE International Conference on*, pages 1–8.

[Park et al., 2004] Park, I., Kim, J., and Park, S. (2004). Development of humanoid robot platform KHR-2 (KAIST Humanoid Robot-2). *Humanoid Robots, 2004*, 2:292–310.

[Pastor et al., 2009] Pastor, P., Hoffmann, H., Asfour, T., and Schaal, S. (2009). learning and generalization of motor skills by learning from demonstration. In *international conference on robotics and automation (icra2009)*.

[Payton, 1986] Payton, D. (1986). An architecture for reflexive autonomous vehicle control. In *Robotics and Automation. Proceedings. 1986 IEEE International Conference on*, volume 3, pages 1838 – 1845.

[Peters et al., 2003] Peters, J., Vijayakumar, S., and Schaal, S. (2003). Reinforcement learning for humanoid robotics. In *Proceedings of the third IEEE-RAS international conference on humanoid robots*, pages 1–20.

[Pierro et al., 2012a] Pierro, P., Hernandez, D., Fierro, M., and Balaguer, C. (2012a). Perception system for working with humanoid robots in unstructured collaborative scenarios. In *Workshop V Perception in Robotics at International IEEE Intelligent Vehicles Symposium 2012*.

[Pierro et al., 2009] Pierro, P., Hernandez, D., Gonzalez-Fierro, M., Blasi, L., Milani, A., and Balaguer, C. (2009). Humanoid teleoperation system for space environments. In *Advanced Robotics, 2009. ICAR 2009. International Conference on*, pages 1 –6.

[Pierro et al., 2012b] Pierro, P., Herrero-Perez, D., Hernandez Garcia, D., Gonzalez-Fierro, M., Blasi, L., and Balaguer, C. (2012b). A DH-DR Collaborative Architecture for a Humanoid Robot in an Unstructured Scenario. *Robotics and Autonomous Systems UNDER REVIEW*.

[Poole et al., 1998] Poole, D., Mackworth, A., and Goebel, R. (1998). *Computational Intelligence: A Logical Approach*. Oxford University Press, USA.

[Pradesh, 2006] Pradesh, U. (2006). Futuristic Humanoid Robots : An Overview. (August):8–11.

[Raibert, 2010] Raibert, M. (2010). Dynamic legged robots for rough terrain. In *Humanoid Robots (Humanoids), 2010 10th IEEE-RAS International Conference on*, page 1.

[Rasmussen and Williams, 2006] Rasmussen, C. and Williams, C. (2006). *Gaussian Processes for Machine Learning*. Adaptive Computation And Machine Learning. Mit Press.

[Riezenman, 2002] Riezenman, M. (2002). Robots stand on own two feet. *Spectrum, IEEE*, 39(8):24 – 25.

[Righetti and Ijspeert, 2006] Righetti, L. and Ijspeert, A. (2006). Programmable central pattern generators: an application to biped locomotion control. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 1585 –1590.

[Rockmore, 2005] Rockmore, T. (2005). *On Constructivist Epistemology*. Rowman & Littlefield Publishers.

[Russell, 2012] Russell, B. (2012). *History of Western Philosophy*. Routledge Classics. Taylor & Francis.

[Russell and Norvig, 2010] Russell, S. and Norvig, P. (2010). *Artificial Intelligence: A Modern Approach*. Prentice Hall Series in Artificial Intelligence. Prentice Hall.

[Sakagami et al., 2002] Sakagami, Y., Watanabe, R., Aoyama, C., Matsunaga, S., Higaki, N., and Fujimura, K. (2002). The intelligent asimo: System overview and integration. pages 2478–2483. IEEE/RSJ international conference on intelligent robots and systems.

[Schaal, 1999] Schaal, S. (1999). is imitation learning the route to humanoid robots? (6):233–242.

[Schaal et al., 2000] Schaal, S., Atkeson, C., and Vijayakumar, S. (2000). Real-time robot learning with locally weighted statistical learning. In *Robotics and Automation, 2000. Proceedings. ICRA '00. IEEE International Conference on*, volume 1, pages 288 –293 vol.1.

[Schaal and Atkeson, 1998] Schaal, S. and Atkeson, C. G. (1998). Constructive incremental learning from only local information. *Neural Computation*, 10(8):2047–2084.

[Schaal and Atkeson, 2010] Schaal, S. and Atkeson, C. G. (2010). learning control in robotics – trajectory-based opitimal control techniques. (2):20–29.

[Schaal et al., 2003] Schaal, S., Ijspeert, A., and Billard, A. (2003). Computational approaches to motor learning by imitation. *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences*, 358(1431):537–547.

[Schaal et al., 2007] Schaal, S., Mohajerian, P., and Ijspeert, A. (2007). Dynamics systems vs. optimal control - a unifying view. In Paul Cisek, T. D. and Kalaska, J. F., editors, *Computational Neuroscience: Theoretical Insights into Brain Function*, volume 165 of *Progress in Brain Research*, pages 425 – 445. Elsevier.

[Schmidt-Rohr et al., 2010] Schmidt-Rohr, S., Loandsch, M., and Dillmann, R. (2010). Learning flexible, multi-modal human-robot interaction by observing human-human-interaction. In *RO-MAN, 2010 IEEE*, pages 582 –587.

[Schneider and Ertel, 2010] Schneider, M. and Ertel, W. (2010). Robot learning by demonstration with local gaussian process regression. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 255–260.

[Schöner, 2008] Schöner, G. (2008). Dynamical Systems Approaches to Cognition. In Sun, R., editor, *Cambridge Handbook of Computational Cognitive Modeling*. Cambridge University Press.

[Schwarz, 1978] Schwarz, G. (1978). Estimating the Dimension of a Model. *The Annals of Statistics*, 6(2):461–464.

[Shukla and Billard, 2012] Shukla, A. and Billard, A. (2012). Augmented-svm: Automatic space partitioning for combining multiple non-linear dynamics. In *NIPS*, pages 1025–1033.

[Siciliano and Khatib, 2008] Siciliano, B. and Khatib, O., editors (2008). *Handbook of Robotics*. Springer.

[Siciliano et al., 2009] Siciliano, B., Sciavicco, L., Villani, L., and Oriolo, G. (2009). *Robotics: Modelling, Planning and Control*. Advanced Textbooks in Control and Signal Processing. Springer, second edition.

[Siino et al., 2008] Siino, R., Chung, J., and Hinds, P. (2008). Colleague vs. tool: Effects of disclosure in human-robot collaboration. In *Robot and Human Interactive Communication, 2008. RO-MAN 2008. The 17th IEEE International Symposium on*, pages 558 –562.

[Simmons, 1994] Simmons, R. (1994). Structured control for autonomous robots. *IEEE Transactions on Robotics and Automation*, 10(1):34–43.

[Song and Wang, 2005] Song, M. and Wang, H. (2005). Highly efficient incremental estimation of Gaussian mixture models for online data stream clustering. In Priddy, K. L., editor, *Intelligent Computing: Theory and Applications III*, volume 5803, pages 174–183. SPIE.

[Spiegelhalter et al., 2002] Spiegelhalter, S. D., Best, N. G., Carlin, B. P., and Linde, A. V. D. (2002). Bayesian measures of model complexity and fit. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 64(4):583–639.

[Stasse et al., 2009] Stasse, O., Verrelst, B., Vanderborght, B., and Yokoi, K. (2009). Strategies for humanoid robots to dynamically walk over large obstacles. *Robotics, IEEE Transactions on*, 25(4):960 –967.

[Sternberg, 2000] Sternberg, R. (2000). *Handbook of Intelligence*. Cambridge University Press.

[Stiefelhagen et al., 2007] Stiefelhagen, R., Ekenel, H., and Fugen, C. (2007). Enabling Multimodal Human - Robot Interaction for the Karlsruhe Humanoid Robot. *Robotics, IEEE*, 23(5):840–851.

[Stiefelhagen et al., 2004] Stiefelhagen, R., Fugen, C., Gieselmann, R., Holzapfel, H., Nickel, K., and Waibel, A. (2004). Natural human-robot interaction using speech, head pose and gestures. In *Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 3, pages 2422 – 2427 vol.3.

[Stilman et al., 2008] Stilman, M., Nishiwaki, K., and Kagami, S. (2008). Humanoid teleoperation for whole body manipulation. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 3175 –3180.

[Stoytchev and Arkin, 2001] Stoytchev, A. and Arkin, R. (2001). Combining deliberation, reactivity, and motivation in the context of a behavior-based robot architecture. In *Computational Intelligence in Robotics and Automation, 2001. Proceedings 2001 IEEE International Symposium on*, pages 290 – 295.

[Strogatz, 1994] Strogatz, S. (1994). *Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry, and Engineering*. Studies in Nonlinearity Series. Perseus Books Group.

[Stulp et al., 2010] Stulp, F., Buchli, J., and Theodorou, E. (2010). Reinforcement learning of full-body humanoid motor skills. *Robots (Humanoids),*, pages 405–410.

[Sugano and Kato, 1987] Sugano, S. and Kato, I. (1987). Wabot-2: Autonomous robot with dexterous finger-arm–finger-arm coordination control in keyboard performance. In *Robotics and Automation. Proceedings. 1987 IEEE International Conference on*, volume 4, pages 90 – 97.

[Sugisaka, 2009] Sugisaka, M. (2009). An approach for soft humanoid robot with artificial muscles. In *Industrial Electronics, 2009. ISIE 2009. IEEE International Symposium on*, pages xiii –xviii.

[Sun, 2009] Sun, R. (2009). Multi-agent systems for society. chapter Cognitive Architectures and Multi-agent Social Simulation, pages 7–21. Springer-Verlag, Berlin, Heidelberg.

[Sun et al., 2001] Sun, R., Merrill, E., and Peterson, T. (2001). From implicit skills to explicit knowledge: a bottom-up model of skill learning. *Cognitive Science*, 25(2):203–244.

[Sung, 2004] Sung, H. G. (2004). *Gaussian mixture regression and classification*. PhD thesis, Rice University.

[Tan, 2012] Tan, H. (2012). *The future of humanoid robots - research and applications*, chapter Implementation of a Framework for Imitation Learning on a Humanoid Robot Using a Cognitive Architecture, pages 191–210. InTech.

[Tani and Ito, 2003] Tani, J. and Ito, M. (2003). Self-organization of behavioral primitives as multiple attractor dynamics: A robot experiment. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 33(4):481–488.

[Tani et al., 2008] Tani, J., Nishimoto, R., Namikawa, J., and Ito, M. (2008). Codevelopmental learning between human and humanoid robot using a dynamic neural-network model. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 38(1):43 –59.

[Tenorth and Beetz, 2013] Tenorth, M. and Beetz, M. (2013). Knowrob: A knowledge processing infrastructure for cognition-enabled robots. *I. J. Robotic Res.*, 32(5):566–590.

[Thagard, 2005] Thagard, P. (2005). *Mind: Introduction To Cognitive Science*. Bradford Books. Mit Press.

[Thelen and Smith, 2007] Thelen, E. and Smith, L. B. (2007). *Dynamic Systems Theories*. John Wiley and Sons, Inc.

[Tsagarakis et al., 2007] Tsagarakis, N., Becchi, F., Righetti, L., Ijspeert, A., and Caldwell, D. (2007). Lower body realization of the baby humanoid - icub;. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 3616 –3622.

[Tso and Liu, 1996] Tso, S. and Liu, K. (1996). Hidden markov model for intelligent extraction of robot trajectory command from demonstrated trajectories. In *Industrial Technology, 1996. (ICIT '96), Proceedings of The IEEE International Conference on*, pages 294 –298.

[Tsuchiya et al., 2003] Tsuchiya, K., Aoi, S., and Tsujita, K. (2003). Locomotion control of a biped locomotion robot using nonlinear oscillators. In *Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, volume 2, pages 1745 – 1750 vol.2.

[Tsuji et al., Nov] Tsuji, T., Tanaka, Y., Morasso, P., Sanguineti, V., and Kaneko, M. (Nov.). Bio-mimetic trajectory generation of robots via artificial potential field with time base generator. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 32(4):426–439.

[Ude, 1993] Ude, A. (1993). Trajectory generation from noisy positions of object features for teaching robot paths. *Robotics and Autonomous Systems*, 11(2):113–127.

[Ude et al., 2007] Ude, A., Riley, M., Nemec, B., Kos, A., Asfour, T., and Cheng, G. (2007). Synthesizing goal-directed actions from a library of example movements. In *Humanoid Robots, 2007 7th IEEE-RAS International Conference on*.

[van Gelder, 1995] van Gelder, T. (1995). What Might Cognition Be, If Not Computation? *The Journal of Philosophy*, 92(7):345–381.

[van Gelder and Port, 1995] van Gelder, T. and Port, R. F. (1995). Mind as motion. chapter It's about time: an overview of the dynamical approach to cognition, pages 1–43. Massachusetts Institute of Technology, Cambridge, MA, USA.

[Varadarajan and Vincze, 2012] Varadarajan, K. and Vincze, M. (2012). Afrob: The affordance network ontology for robots. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 1343–1350.

[Vecchio, 2002] Vecchio, D. (2002). Primitives for human motion: a dynamical approach. In *Proceedings of the 15th IFAC World Congress on Automatic Control*.

[Vecchio et al., 2003] Vecchio, D. D., Murray, R. M., and Perona, P. (2003). Decomposition of human motion into dynamics-based primitives with application to drawing tasks. *Automatica*, 39(12):2085–2098.

[Veloso et al., 1995] Veloso, M., Carbonell, J., PÃ©rez, A., Borrajo, D., and Blythe, J. (1995). Integrating planning and learning: The prodigy architecture. *Journal of Experimental and Theoretical Artificial Intelligence*, 7:81–120.

[Vernon et al., 2007] Vernon, D., Metta, G., and Sandini, G. (2007). The icub cognitive architecture: Interactive development in a humanoid robot. In *Development and Learning, 2007. ICDL 2007. IEEE 6th International Conference on*, pages 122 –127.

[Vijayakumar and Schaal,] Vijayakumar, S. and Schaal, S. Locally weighted projection regression: An o(n) algorithm for incremental real time learning in high dimensional space. In *in Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000*, pages 1079–1086.

[Vukobratovic and Borovac, 2004] Vukobratovic, M. and Borovac, B. (2004). Zero-moment point - thirty five years of its life. *International Journal of Humanoid Robotics*, 1(1):157–173.

[Waibel et al., 2011] Waibel, M., Beetz, M., Civera, J., D'Andrea, R., Elfring, J., Galvez-Lopez, D., Haussermann, K., Janssen, R., Montiel, J. M. M., Perzylo, A., Schiessle, B., Tenorth, M., Zweigle, O., and van de Molengraft, R. (2011). Roboearth. *Robotics Automation Magazine, IEEE*, 18(2):69–82.

[Weteschnik, 2006] Weteschnik, M. (2006). *Understanding Chess Tactics*. Quality Chess Europe.

[Wheeler et al., 1994] Wheeler, M., of Sussex. School of Cognitive, U., and Sciences, C. (1994). *For Whom the Bell Tolls?: The Roles of Representation and Computation in the Study of Situated Agents*. Cognitive science research papers. School of Cognitive and Computing Sciences, University of Sussex.

[Wood, 1993] Wood, S. (1993). *Planning and Decision Making in Dynamic Domains*. Ellis Horwood Series in Artificial Intelligence. Ellis Horwood.

[Wooldridge and Jennings, 1995] Wooldridge, M. and Jennings, N. R. (1995). Intelligent agents: Theory and practice. *Knowledge Engineering Review*, 10:115–152.

[Yang et al., 1997] Yang, J., Xu, Y., and Chen, C. (1997). Human action learning via hidden markov model. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 27(1):34 –44.

[Yokoi et al., 2008] Yokoi, K., Sian, N. E., Sakaguchi, T., Stasse, O., Kawai, Y., and Maruyama, K.-i. (2008). Humanoid robot hrp-2 with human supervision. *Experimental Robotics*, 39:513–522.

[Yoshida et al., 2008] Yoshida, E., Esteves, C., Belousov, I., Laumond, J.-P., Sakaguchi, T., and Yokoi, K. (2008). Planning 3-d collision-free dynamic robotic motion through iterative reshaping. *Robotics, IEEE Transactions on*, 24(5):1186 –1198.

[Zoliner et al., 2005a] Zoliner, R., Pardowitz, M., Knoop, S., and Dillmann, R. (2005a). Towards cognitive robots: Building hierarchical task representations of manipulations from human demonstration. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, number April, pages 1535–1540. IEEE.

[Zoliner et al., 2005b] Zoliner, R., Pardowitz, M., Knoop, S., and Dillmann, R. (2005b). Towards cognitive robots: Building hierarchical task representations of manipulations from human demonstration. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 1535 – 1540.