Omnidirectional vision system for mini-humanoid robots.

Félix R. Cañadillas, Martin F. Stoelen, Alberto Jardón and Carlos Balaguer

Abstract— **The paper presents preliminary work towards an omnidirectional vision system for mini-humanoid robots. Such robots are currently in use in several research centers around the world, and are also widely used in robot competitions, for example in the yearly Spanish CEABOT contest. Mini-humanoid robot platforms require sensing for among other obstacles avoidance, navigation and physical interaction, and have limited onboard information processing capabilities. The work presented here explores different ways in which the information from an omnidirectional camera can be used in such a robotic system, and what advantages can be gained in the context of a mini-humanoid robot contest. The navigation task from the CEABOT competition is used as a case study. Early ideas for implementation are also presented, specifically using a Blackfin camera with onboard processing mounted on a Bioloid mini-humanoid robot.**

I. INTRODUCTION

One of the main challenges in robotics is understanding the environment through the use of its sensors, to help give the robot the ability to operate autonomously.

One way to inspire young engineering students to enter into such a field is through robotics competitions, as was emphasized in [1] and [2]. These competitions are an easy and active way of being introduced to the development of systems for the recognition of the environment, in addition to a way to compare development with the other systems that participate in these competitions. One example is the Spanish CEABOT competition [10], held yearly, where student teams can enter mini-humanoid robots in three different tests. This is further described in section II.

Different types of sensors are available for this type of competitions, which opens up a wide range of possibilities for investigation into how to improve this area of mini-humanoid robotics. Typically however, the sensors used in mini-humanoid robot competitions are limited in their coverage of the environment around them. For example infrared sensors, which measure the distance to a point in the environment, usually requiring up to 5-10 sensors for an acceptable coverage, as we can see in [3].

The project described here surfaced from the motivation to create a vision system with a more complete coverage for the robot. This system should improve the techniques of interaction with the environment that the mini-humanoid robot already has, complementing ultrasonic, infrared and pressure sensors.

Computer vision systems are important tools for detection and recognition tasks in robotic systems. There are several approaches to using vision system however. Among the different configurations commonly used in mini-humanoid robots we can highlight two; the stereo vision systems and the omnidirectional vision systems. Among stereo systems, we can highlight those used by the mini-humanoid robot QRIO [4] and DARwinOP [5], that through the stereo image processing is able to avoid obstacles, as can be seen in [6]. In addition it is worth mentioning other stereo vision systems that are not subject to commercial platforms. For example, the one developed in [7], where a small stereo vision system was implemented which can be incorporated into any mini-humanoid platform.

Omnidirectional vision systems are especially interesting, in that they can obtain a 360 degree view of the environment. For the omnidirectional vision systems, see for example [8] and [9], which through a reflector gets a full view of the complete environment. This configuration gives a very good result, but requires a study of how to create this reflector, which can be quite complicated. Another option is to acquire a commercial reflector although their prices are often high. An alternative option is to use a wide-angled lens, with the camera pointing downwards. This is further described in section III.

The remainder of this paper describes the work environment assumed, the CEABOT competition, followed by early ideas for implementation of a omnidirectional camera in the Bioloid mini-humanoid robot, and a set of simple computer vision algorithms suitable for such a system.

II. DESCRIPTION OF THE WORK ENVIRONMENT

This project is based on the creation of a vision system for a mini-humanoid robot to compete in the CEABOT competition. In [10], we can find more information in this championship, although here we describe the main features of the different tests of this championship.

Before choosing a mini-humanoid platform on which we base our project, we must consider the restrictions that are determined in the regulations CEABOT, especially the height and weight restrictions. These are a maximum weight of 3 kg and a maximum height of 50 cm with the robot fully extended. In addition all the computational calculation must be performed on board.

Observing the platforms on the market, we see that not all mini-humanoid platforms meet these requirements. For example, platforms like DARwinOP [5] and HOAP-3 [11], which have on-board image processing, could not participate in this tournament as they exceed the weight and/or height limitations. There are other platforms in the market that do meet these requirements however, and that are commonly used in tournaments of this type. Platforms like Robonova [12] and Bioloid [13] are the most used in previous editions

ISBN: 978-84-695-3472-4

of CEABOT. In the work presented here the use of a Bioloid robot is assumed, see fig. 1 below.

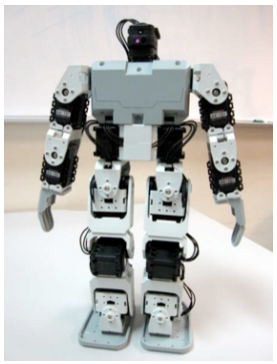

Fig. 1. Bioloid Robot.

The championship CEABOT has four tests to determine the capabilities of the mini-humanoid robots participating. These tests are:

- Obstacle race: This test is based on navigating in a partially known environment with obstacles, having to cross this area from one line to another, and returning. The fixed number of obstacles used can appear in any configuration during the competition. The lines are yellow, the ground is green and the obstacles are white. See fig. 2. This makes differentiating between them easier, provided the vision system can distinguish colors.
- Stairs: This test demonstrates the ability of the robot to climb stairs. A good way to support the robot in achieving this test will be the location of the stairs, which can be determined by edge detection.
- Sumo: In this test a battle by sumo is performed between two robots opponents. One technique from computer vision that may be of use here is the detection of movement, with which we can detect if the robot opponent is moving, at what speed and in what direction it is moving. It is noteworthy that the area of the combat zone is marked by the regulations of CEABOT and determined by a yellow circle line. It is thus important to locate this line so as to avoid crossing it.
- Free exhibition: The exhibition test is based on a free demonstration of the abilities of each robot, for example performing a series of choreographed moves. As this is a test in which the robot typically does not interact with the environment the use of a camera is likely not needed. However, if in this demonstration were to use any external object such as a ball or a cube, a camera could help the robot to interact with that object.

In view of the above tests and observing the robotic systems commonly used in this type of tournament, it was determined to focus on developing a vision system to support navigation in the obstacle race test and to determine the opponent's position in the sumo test.

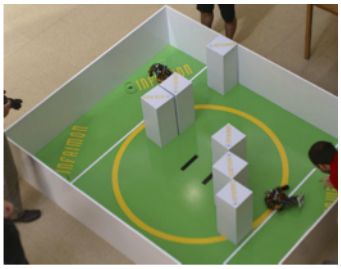

Fig. 2. Work environment in CEABOT during an obstacle race test.

III. OMNIDIRECTIONAL VISION SYSTEMS

As mentioned inn the introduction, there are several alternatives to consider when creating an omnidirectional vision system. One is the proposal in [2], where through the reflection of the image on a conical mirror positioned vertically relative to the robot it is possible to get a omnidirectional view of the environment. This is a good solution, but typically requires a conical mirror, which can be quite complicated to manufacture and expensive to buy. Another possible configuration is the use of a normal lens with a large viewing angle, so that focusing the camera to the ground from the top of the robot, we provide an omnidirectional view of the environment sufficient for observing objects close to the robot.

This omnidirectional configuration is simpler to implement, as it only requires a structure for separating the robot and the camera. For example a transparent plastic tube, as was used here. To test the feasibility of this configuration, a study of the effect of a given lens viewing angle on the height at which the camera would have to be mounted was performed. Taking into account that the mini-humanoid robot used (the BIOLOID) has a height of 31 cm, it was determined that the camera would be positioned at 40 cm above the ground, i.e. with a distance of 9 cm between the camera and robot. This height provided a sufficient view of the surroundings with a commercially available lens with a viewing angle of 120 degrees. In fig. 3 , fig. 4 and fig. 5, we see the study of these conditions.

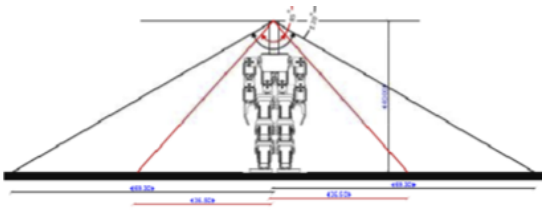

Fig. 3. Front view of the configuration omnidireccional.

The increased height of the center of mass of the robot will reduce the stability of the robot however. Thus it is important to attempt to keep the mass of the camera low. For the work performed here it was assumed that a camera with onboard processing was used, to minimize the load on the robot controller. For example the Surveyor Blackfin Camera [15], which can be obtained with a lens with a

ISBN: 978-84-695-3472-4

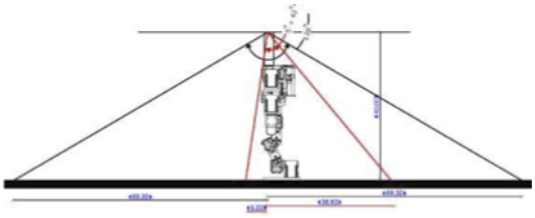

Fig. 4. Profile view of the configuration omnidireccional.

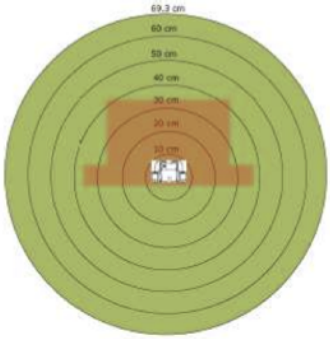

Fig. 5. Top view of the configuration omnidireccional.

120 degrees viewing angle. This camera can be a good option when implementing the vision system onboard the Bioloid platform. It incorporates a Blackfin BF537 processor of 500MHz with a camera with a optical sensor of 1.3 megapixel. In addition, it incorporates an external I/O header of 32pins. It has a a total mass of 0.046 kg, and size 50 x 65 x 50 mm.

IV. VISION ALGORITHMS

In this section we will explain the vision algorithms that was explored for the purpose of this project. All these algorithms have been implemented in Matlab, as it is a complete tool when working with matrix computation, and in addition has a large amount of toolboxes available. This simplifies the prototyping of the algorithms, and allows them to be assessed and compared, before implementing them on the camera itself. It is assumed that the robot is stationary when obtaing images.

A. Floor-Obstacle Detection

One of the simplifying characteristics of the working environment assumed, at least for the obstacle race and sumo tests, is that the robot is moving in a planar world. Another is that the different components of the environment have different colors. If we capture an image on the CEABOT environment from an omnidirectional perspective, we note that the image consists of two elements defined by its color, green for the floor, and white for the obstacles. In the fig. 6 we can see an example of such an image. The omnidirectional camera is here simulated by a normal web camera mounted on a structure above the BIOLOID robot.

As the camera will be mounted directly above the robot, and will be moving with the robot, calculating the distance from the center of the image to white areas along the border

Fig. 6. Example image captured in CEABOT environment.

of the image, it is possible to estimate the distance between the robot and the different obstacles. The angle with respect to the robot can also easily be determined. Through the preprocessing of the image and its later binarization[16], we get an image on which these calculations can be more easily be performed. See fig. 7.

To calculate the distances between the center of the image and the obstacles, we generated a series of lines with origin in the center, and simply look for the first white pixel. The number of lines to generate depends on the coverage that we want or need for our system. For the example shown here, the recognition of obstacles is performed each 15 degrees to generate 24 equally spaced lines. If we change the number of lines to generate, we must calculate the angle of each straight line in response to the following equation.

$$Ang(n) = \frac{2\pi}{n_{total}} * n \tag{1}$$

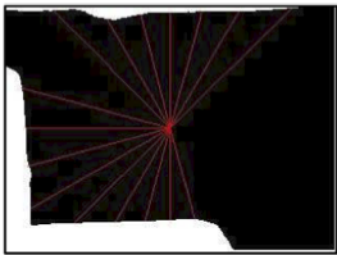

Fig. 7. Representation of the distance between the obstacle and the robot.

Considering the general equation of a line $y = ax + b$ where b is the initial y_0, for our case 0, where a is the slope of the line defined for our case as $a = tan(Ang(n))$, therefore we can determine that:

$$y = x * tan(Ang(n)) \tag{2}$$

Using this equation, we will cross the x values by the straight line of angle n to get a y value. Having calculated x and y, we check the value of the binary image for the $pixel(x, y)$. If the value of $pixel(x, y)$ is equal to 0 means no objects and therefore will not be necessary to calculate the distance, but if the value of the $pixel(x, y)$ is equal to 1 means we have found the subject and therefore no longer need to continue along the line. To calculate the distance to the object we calculate the magnitude of the vector generated by x and y, which we can determine how:

ISBN: 978-84-695-3472-4

$$Dist(n) = \sqrt{x^2 + y^2} \tag{3}$$

where $Dist(n)$ is the distance to the object for line n. Repeating the process for all lines one can obtain mapped distances all around the robot. This mapping of all distances is then stored in a vector $Dist(n)$ which contain all the distances calculated. Finally, note that these calculated distances are not real and represent the distance in pixels in the image. To change the distances to real-world distances, the perspective projection [16] of the camera can be used. This equation is as follows:

$$x = f\frac{X}{Z} \qquad y = f\frac{Y}{Z} \tag{4}$$

being the coordinates in the image variables (x, y) and real variables (X, Y). Furthermore, f is the focal length and Z is the height at which the camera is mounted. To get from pixel coordinates to the image coordinates, the intrinsic parameters of the camera can be used, as shown in the following equation:

$$x = -(x_{im} - o_x) * s_x \qquad y = -(y_{im} - o_y) * s_y \tag{5}$$

where (o_x, o_y) are the coordinates of the center pixel of the image, and (s_x, s_y) the pixel size in millimeters in the horizontal direction and vertical image respectively. Radial distortions are here ignored, but should be included in the final application if exact distances are required.

The results of this algorithm are interesting, as it seems to be relatively reliable in the real-world images used, mainly due to the simplified planar environment and the different colors used for floor and obstacles. This is of course assuming that the lighting conditions are relatively constant during the competition. A possible improvement would be the use of edge detection as a complement to this algorithm, although this may increase the computational load of the system.

B. Movement Detection

The strategy that we have used for the detection of the opponent robot in the Sumo test is based on detecting the movement of this robot.

As shown in the book [16], any perceptible movement in the scene results in the sequence of images taken from that scene, so, if such changes are detected, you can analyze the characteristics of this movement. In the general case of three-dimensional motion of an object is only possible to obtain qualitative estimates of the movement. In fact, when an object moves in the direction of your line of observation, movement is not noticeable, although there is some evidence of movement, as the variation of its size in the image, the movement of the shadow cast, etc. However, when the object moves in a plane parallel to the image plane can obtain good estimates of the components of motion.

One theory is used for motion detection is the use of the image difference. This is based on subtraction of images, which we can calculate the amount of movement of the object because for a single moving object, the difference between images will be the movement of said object. Therefore, the image difference is defined as $Image_d$:

$$Image_d(p, t_1, t_2) = Image(p, t_2) - Image(p, t_1) \tag{6}$$

where $p = (x, y)$ is a generic pixel of the image and t_1, t_2 are the time instants of two consecutive images. Note that the values or intensities that are obtained through (6) can be negative. The most appealing aspect of this technique is its simplicity. The information provided, however, is not very descriptive about the shape or motion of objects, although it can detect the image area where changes are occurring, and being able to concentrate the later computational effort in the detected area.

To determine the magnitude and direction of motion of the robot opponent, we apply this algorithm to three consecutive images, so you get two results from the image difference, different in the time. We can see one of these binary images difference in the fig. 8, for a soda-can example.

Fig. 8. Example of the image difference in binary.

By having a single opponent and therefore a single moving element, we can calculate the total centroid from all regions of motion captured and assume that they belong to the same element in motion. By then creating a vector between the centroids of the two object-differences obtained previously, we get a representative vector of movement of the robot opponent. To calculate the centroid of all regions of motion captured in an image difference, the contribution of each was set to be proportional to the area of each region of movement within the total set of motion captured. This algorithm is performed by applying the following equation:

$$Centroid_{total}(x, y) = \sum_{i=1}^{n} \frac{Area(i)}{Area_{total}}) * Centroid(i) \tag{7}$$

With these two centroids a vector can be created, which provides the magnitude and direction moving object. This can be seen in fig. 9, where the vector is overlaid on the original image. Note that the the speed of movement is in the image frame only, and without information about the distance to the moving robot. However, this algorithm is sufficient for determining in what direction the enemy robot is currentl, and wether the enemy robot is moving tangentially with respect to our robot.

ISBN: 978-84-695-3472-4

Fig. 9. Representation of the velocity vector in the original image.

This algorithm is a simple way to calculate the position of the robot opponent when this is moving. Because of its simplicity, is not affected by the illumination, making it more robust to different lighting conditions, for example caused by the shadows of the spectators.

C. Centered Through Optical Flow

This section describes a vision algorithm implemented to support the first algorithm in navigation tasks. In [17] the optical flow from a camera system was used to allow for a centering behavior of a mobile robot when navigating in-between obstacles. This was based on the maximum optical flow observed on the left and right peripheral visual field as an indicator of the proximity of obstacles. If the maximum optical flow is greater on the right, the objects are closest on this side and therefore, the robot should turn to the right in order to avoid collision with the obstacle. This could also be of interest for the obstacle race test considered here.

The optical flow plays an important role in the estimation and description of movement in an image set, which is used for the detection, segmentation and tracking of moving objects in a scene. As seen in [18] optical flow can be described as a vector field subject to the condition of the constant brightness equation and is defined as the apparent motion of the pattern of image brightness. The constant brightness equation is shown below:

$$(\nabla E)^T * \upsilon + E_T = 0 \tag{8}$$

As noted in the equation for an image $E = E(x, y, t)$, and a vector v motion field, the sum of the product of the image gradient multiplied by the vector representative of the movement, plus the time-dependent image taken E_T must be equal to zero for an environment with no change in brightness, ie constant brightness. Therefore, the optical flow field is the approximation of the motion which can be calculated from sequences of timevarying images, as long as we assume the following conditions [18]:

- Lambertian surfaces: A perfectly diffusing surface is one that emits or reflects the light output in a form that it presents the same luminance regardless of viewing angle. Such a surface is called Lambertian because it responds to Lamberts law.
- Point source of light at infinity: Refers to the light source by placing the long distance from our catchment area movement.
- Without photometric distortion: The photometric distortion represents the variation in contrast between images, so for the proper use of optical flow we have an environment in which the contrast variation over time is zero.

Therefore, to implement our vision algorithm we make the following assumptions.

- Constant brightness equation produces a good approximation of the normal component of the motion field.
- The field motion vector field approximates well to the constant within any small portion of the image plane.

Taking the first case, for each point pi in a region of small Q, of size NxN, can be written as equation (3), where the spatial and temporal derivatives of image brightness is calculated in p_1, p_2 ... p_{NxN}. Typically uses a small region of 5x5. Therefore, the optical flow can be estimated with Q as the constant vector, υ, which minimizes the functionality of the following equation:

$$\Psi[\upsilon] = \Sigma_{Pixel \in Q}[((\nabla E)^T * \upsilon + E_T)]^2 \tag{9}$$

The solution to this least squares problem can be determined by passing a linear system, this will do as follows:

$$A^T A * \upsilon = A^T * b \tag{10}$$

The i-th row of the matrix A (N2x2) is the spatial image gradient evaluated at point p_i,

$$A = \begin{pmatrix} \nabla E(p_1) \\ \nabla E(p_2) \\ \vdots \\ \nabla E(p_{NxN}) \end{pmatrix} \tag{11}$$

and b is the dimension N of the vector of temporary partial brightness of the image, evaluated at p_1,...p_{NxN}, after a change of sign:

$$b = -[E_t(p_1), ..., E_t(p_{NxN}]^T \tag{12}$$

The least squares solution of the system through restrictions on (10) can be obtained as:

$$\upsilon = (A^T A)^1 * A^T * b \tag{13}$$

where υ is the optical flow (the estimated motion field) in the center of the region Q; repeating this process for all image points, we get a complete optical flow. As we can see in the fig. 11.

Fig. 10. Real environment test.

ISBN: 978-84-695-3472-4

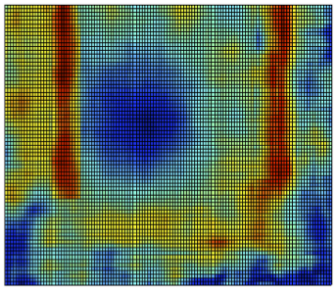

Fig. 11. Graphical representation of optical flow module.

To reduce the computational requirements a strategy similar to the one developed in [19] was followed, to focus the application of the optical flow algorithm on determined areas of the image, in our case, on both sides of the robot. Fig. 12 shows the representation of the optical flow computation in the areas most relevant of our image.

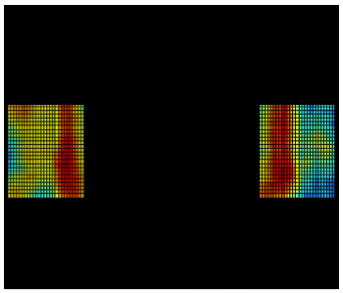

Fig. 12. Graphical representation of optical flow module in the relevant areas.

Using the optical flow algorithm, we can give support to the two previous algorithms, making the whole system more robust and reliable. Note that this algorithm is sensitive to changes in lighting, but that by averaging the optical flow over large areas a more robust result can be obtained.

V. CONCLUSIONS

In the beginning of this project we have focused on studying the regulation of CEABOT, the required features of a omnidirectional camera, and the characteristics of mini-humanoid robots. Through these three studies, we have obtained determining factors when developing this project, as for example the height of the robot with the camera, and the dimensions of the CEABOT environment. From this a camera with a 120 degree wide-angle lens pointed downwards, and mounted at 40 cm, was chosen.

Regarding the three vision algorithms, initial prototypes have so far been implemented in Matlab, and tested on realistic images from the real CAEBOT environment. All three are relatively simple to implement, and have a low computational requirement. Especially the movement detector for the Sumo test. The navigation required for the obstacle race test can likely benefit from the floor-obstacle detection algorithm, while the optical flow algorithm may require more work to be useful. In general, the wide view provided by an omnidirectional camera provides a significant increase in the resolution of sensing that is possible with respect to the typical proximity sensing used for mini-humanoid robots.

ACKNOWLEDGMENT

The authors of this paper want to thank the support of COMANDER CCG10-UC3M/DPI-5350 and ROBOCITY2030 projects funded by Comunidad de Madrid and UC3M. Thanks to the Society of Robotics at the University Carlos III (ASROB) [20] for giving us the means to develop it.

REFERENCES

[1] M. González, A. Jardón, S. Martínez, M.F. Stoelen, J.G. Victores and C. Balaguer. Educational initiatives related with the CEABOT contest. In Proc. of 2nd Int. Conf. on Simulation, Modeling and Programming for Autonomous Robots (SIMPAR 2010), Darmstadt, Germany, pp. 649-658.

[2] H. Kitano, M. Asada, Y. Kuniyoshi, I. Noda and E. Osawa. RoboCup: The Robot World Cup Initiative. IJCAI-95 Workshop on Entertaiment and AI/Alife, 1995.

[3] S. Pons, N. Tolós and S. Troyano. Algorismes i modifications hardware a la robot Dorami per participar al campionat nacional de robotica humanoide CEABOT 2010. Technical Report, Institut de Robotica i Imformatica Industrial, 2010.

[4] QRIO, Sony Dream Robot, 2012. [Online]. Available: http://sonyaibo.net/aboutqrio.htm

[5] DARwinOP, Open Plataform Humanoid Robot for Research and Education, 2012. [Online]. Avaliable: romela.org

[6] K. Sabe, M. Fukuchi, J.S. Gutmann, T. Ohashi, K. Kawamoto and T. Yoshigahara. Obastacle avoidance and path planing for humanoid robots using stereo vision. In Int. Conf. on Robotics and Automation (ICRA), Seoul, Korea, 2001.

[7] K. Konolige. Small Vision Systems: Hardware and Implementation. In Eighth International Symposium on Robotics Research. Hayama, Japan, 1997.

[8] N. Winters, J. Gaspar, G. Lacey and J. Santos. Omni-directional Vision for Robot Navigation. IEEE Workshop on Omnidirectional Vision, 2000.

[9] J. Kim and Y. Suga. An Omnidirectional Vision-Based Moving Obstacle Detection in Mobile Robot. International Journal of Control, Automation and Systems, vol. 5, no. 6, 2007. pp. 663-673.

[10] A. Jardón, P. J. Sanz, F. Gómez, J. Felip and J. C. García. CEABOT'10 Normativa. Comité Español de Automática (CEA), Universidad de Jaén. Spain, 2010.

[11] HOAP-3, Miniature Humanoid Robot. Fujitsu, 2012. [Online] Avaliable: home.comcast.net/jtechsc/HOAP-3-SpecSheet.pdf

[12] Robonova, Hitec Robotics, 2012. [Online] Avaliable: robonova.de/store/home.php

[13] Bioloid, ROBOTIS, 2012. [Online] Avaliable: robotis.com/xe/bioloid-en

[14] E. Menegatti, E. Pagello and M. Wright, Using Omnidirectional Vision within the Spatial Semantic Hierarchy. Conference on Robotics and Automation. Proceedings. ICRA 02. IEEE International, vol.1. 2002. pp. 908-914.

[15] Surveyor SRV-1 Blackfin Camera, Surveyor Corporation, 2012. [Online] Avaliable: surveyor.com/blackfin/

[16] Javier González Jiménez, Visión por computador. Madrid, PARANINFO, 1999.

[17] D. Cooms and K. Roberts, Centering behavior using peripheral vision, Computer Vision and Pattern Recognition. Proceedings CVPR 93, IEEE Computer Society Conference, New York, 1993. pp. 440-445.

[18] E. Trucco and A. Verri, Introductory Techniques for 3-D Computer Vision. New Jersey: Prentice Hall, 1998, ch. 8.

[19] J. van der Blij, Omnidirectional Active Vision in Evolutionary Car Driving, M.Sc. Graduation Thesis in Artificial Intelligence, University of Groningen (RuG), Groningen, The Netherlands, 2005.

[20] Society of Robotics at the University Carlos III (ASROB), 2012. [Online]. Avaliable: http://asrob.uc3m.es