

# An Omnidirectional Camera Simulation for the USARSim World

Tijn Schmits and Arnoud Visser

Universiteit van Amsterdam, 1098 SJ Amsterdam, the Netherlands  
[tschmits,arnoud]@science.uva.nl

**Abstract.** Omnidirectional vision is currently an important sensor in robotic research. The catadioptric omnidirectional camera with a parabolic convex mirror is a common omnidirectional vision system in the robotics research field as it has many advantages over other vision systems. This paper describes the development and validation of such a system for the RoboCup Rescue League simulator USARSim.

After an introduction of the mathematical properties of a real catadioptric omnidirectional camera we give a general overview of the simulation method. We then compare different 3D mirror meshes with respect to quality and system performance. Simulation data also is compared to real omnidirectional vision data obtained on an 4-Legged League soccer field. Comparison is based on using color histogram landmark detection and robot self-localization based on an Extended Kalman filter.

**Keywords:** RoboCup, USARSim, Omnidirectional Vision, Simulation, Catadioptric Omnidirectional Camera, Landmark Detection, Kalman Filter.

## 1 Introduction

Agents operating in a complex physical environment more often than not benefit from visual data obtained from their surroundings. The possibilities for obtaining visual information are numerous as one can vary between imaging devices, lenses and accessories. Omnidirectional vision, providing a  $360^\circ$  view of the sensor's surroundings, is currently popular in the robotics research area and is currently an important sensor in the RoboCup.

Omnidirectional views can be obtained using multiple cameras, a single rotating camera, a fish-eye lens and or a convex mirror. A catadioptric vision system consisting of a conventional camera in front of a convex mirror with the center of the mirror aligned with the optical axis of the camera, is the most generally applied technique for omnidirectional vision. Mirrors which are conic, spherical, parabolic or hyperbolic all are able to provide omnidirectional images [1].

An omnidirectional catadioptric camera has some great advantages over conventional cameras, one of them being the fact that visual landmarks remain in the field of view much longer than with a conventional camera. Also does the imaging geometry have various properties that can be exploited for navigation



**Fig. 1.** The catadioptric omnidirectional camera, real and simulated.

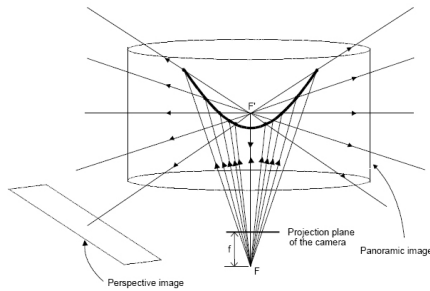
or recognition tasks, improving speed and efficiency of the tasks performed by the robot. The main disadvantage of omnidirectional cameras over conventional ones is the loss of resolution in comparison with standard images [2].

Omnidirectional vision has played a major part in past and present research. At the University of Amsterdam, the Intelligent Autonomous Systems Group uses omnidirectional vision for Trajectory SLAM and for appearance based self-localization using a probabilistic framework [3, 4]. Related to the RoboCup, Heinemann et al. use a novel approach to Monte-Carlo localization based on images from an omnidirectional camera [5]. Lima et al. have used a multi-part omnidirectional catadioptric system to develop their own specific mirror shape which they use for soccer robot self-localization [6].

At the RoboCup Rescue Simulation League one of the simulators which is used and developed is USARSim [7], the 3-D simulation environment of Urban Search And Rescue (USAR) robots and environments, built on top of the Unreal Tournament game and intended as a research tool for the study of human-robot interaction and multi-robot coordination [8]. When we started this project, USARSim did not offer a simulation model of an omnidirectional vision sensor. The Virtual Robots Competition, using USARSim as the simulation platform, aims to be the meeting point between researchers involved in the Agents Competition and those active in the RoboCup Rescue League. As the omnidirectional catadioptric camera is an important sensor in the robotics field, we decided to develop this camera for the USARSim environment. In this paper we will describe the elements which simulate a catadioptric omnidirectional camera in USARSim.

## 2 Method

This section will describe the model of the real omnidirectional vision system on which our simulation model is based and it will explain the rationale behind our simulation method. Figure 1 is an image of the real omnidirectional vision system next to a simulated version of that camera.



**Fig. 2.** A schematic of the Single Effective Viewpoint

## 2.1 The Real Camera

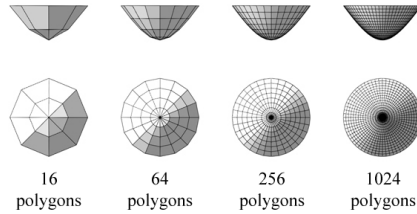
The virtual omnidirectional camera model is based on the catadioptric omnidirectional vision system shown in Figure 1. It is employed by the Intelligent Systems Lab Amsterdam (ISLA) and consists of a Dragonfly<sup>®</sup> 2 camera made by Point Grey Research Inc. and a Large Type Panorama Eye<sup>®</sup> made by Accowle Company, Ltd. The Dragonfly<sup>®</sup> 2 camera is an OEM style board level camera designed for imaging product development. It offers double the standard frame rate, auto-iris lens control and on-board color processing<sup>1</sup>. It captures omnidirectional image data in the reflection of the Panorama Eye<sup>®</sup> parabolic convex mirror.

The parabolic surface is designed to satisfy the Single Viewpoint Constraint (SVC). For a catadioptric system to satisfy the SVC, all irradiance measurements must pass through a single point in space, called the effective viewpoint. This is desirable as it allows the construction of geometrically correct perspective images which can be processed by the enormous amount of vision techniques that assume perspective projection: as the geometry of the catadioptric system is fixed, the directions of rays of light measured by the camera are known a priori. This allows reconstruction of planar and cylindrical perspective images, as is depicted in Figure 2 [1, 9, 10]. Baker and Nayar constructed surface equation which comprises a general solution of the single viewpoint constraint equation for parabolic mirrors [1]:

$$\left(z - \frac{c}{2}\right)^2 - r^2\left(\frac{k}{2} - 1\right) = \frac{c^2}{4}\left(\frac{k-2}{k}\right) (k \geq 2) \quad (1)$$

with  $c$  denoting the distance between the camera pinhole and the effective viewpoint,  $r$  is defined by  $r = \sqrt{x^2 + y^2}$  and  $k$  is a constant.

<sup>1</sup> See <http://www.ptgrey.com/products/dragonfly2/>  
Last accessed: December 20th, 2007.



**Fig. 3.** Mirror Surfaces with increasing polygon counts.

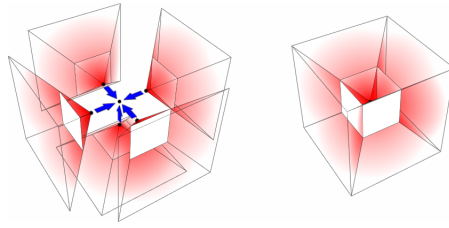
## 2.2 The Virtual Camera Mesh

To recreate this catadioptric camera in simulation first the parabolic convex has to be built with small polygons as building blocks. The 3D mesh was modeled to scale in Lightwave<sup>®</sup> and imported into the Unreal Editor<sup>™</sup>. To investigate the influence of the mirror polygon count, four mirror meshes were created. The mirror surfaces are defined by Equation 1 with  $k = 11.546$  and  $c = 2.321$ , and have  $2^4$ ,  $2^6$ ,  $2^8$  and  $2^{10}$  polygons respectively (Figure 3).

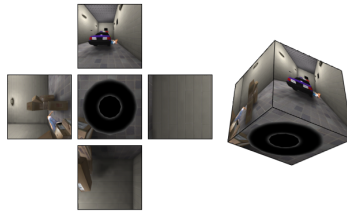
## 2.3 Cube Mapping in Unreal

A mirror should reflect its surroundings. Unfortunately, the Unreal Engine is only capable of rendering a limited number of planar reflective surfaces. Instead of using many reflecting polygons, a limited virtual cameras are placed at the effective viewpoint. Multiple virtual cameras are needed to see the surroundings (omnidirectional). This idea is equivalent to the approach of Beck et al., who developed a catadioptric camera meeting the SVC for the Gazebo simulator. They mapped images of the environment to the faces of a cube and then applied this texture to the surface of a three-dimensional object, i.e. a mesh object resembling the surface of a hyperbolic mirror [11]. In the USARSim environment a similar approach is followed based on CameraTextureClients.

**Virtual Camera Placement.** CameraTextureClients are Unreal Tournament objects which project virtual camera views on textures. Originally designed to create security camera monitor screens in the Unreal Tournament game, they can be used to obtain images of the environment which then can be mapped on the surface of a hyperbolic mirror. If done correctly, the mapping creates a perspective distortion typically for catadioptric cameras. To create the effect of a proper cube mapping, five virtual cameras with a  $90^\circ$  field of view (FOV) need to be placed on the Effective Viewpoint in  $90^\circ$  angles of each other as depicted in Figure 4. A  $90^\circ$  FOV for all the cameras will result in the projection planes of the cameras touching each other at the edges, which means all viewing angles are covered by exactly one camera. For this particular application five cameras are needed instead of six, as a camera looking at the top side of the cube will register image data which should not be reflected by the virtual mirror. It can



**Fig. 4.** Placement of the 5 virtual cameras.



**Fig. 5.** 5 Virtual Camera views and a cube mapping

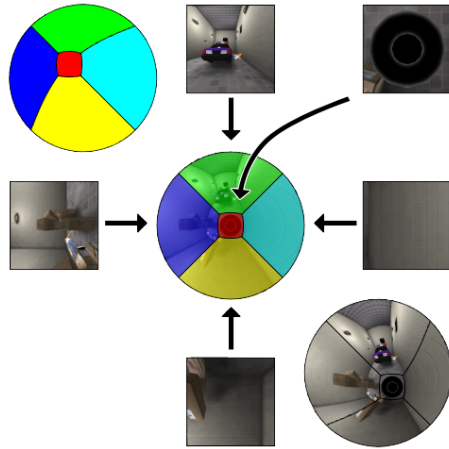
therefore be omitted to save computational requirements. Figure 5 shows five images produced by virtual cameras placed in this position and how they relate to the mapping cube.

**Mapping the Camera Textures.** For a real catadioptric omnidirectional camera satisfying the SVC using a parabolic mirror, the relation between the effective single viewpoint pitch incidence angle  $\theta$  and the radius of the correlating circle in the omnidirectional image  $r_i$  is known to be defined by the following equation

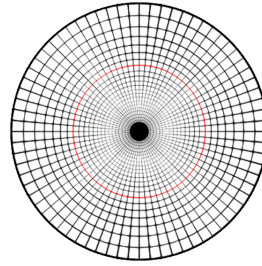
$$r_i = \sin(\theta) \frac{h}{(1 + \cos(\theta))} \quad (2)$$

where  $h$  is the radius of the  $90^\circ$  circle [12] and, as the shape of the mirror defines the location of the  $90^\circ$  circle,  $h$  is directly related to  $c$  and  $k$  in Equation 1.

The proper UV-mapping of the simulated mirror surface produces omnidirectional image data displaying the same relation between  $\theta$  and  $r_i$  as defined by Equation 2. A point projected UV-mapping of the mapping cube from the Effective Viewpoint to the 3D parabolic mirror surface, depicted in Figure 6, has been verified to produce data which concurs with Equation 2. A simulation rendering of the relation between  $\theta$  and  $r_i$  is depicted in Figure 7.



**Fig. 6.** UV Mapping the Virtual Camera Views



**Fig. 7.** A simulation mirror rendering of the relation between incidence angles and image pixel locations, correlating to Equation 2. The red line depicts the  $90^\circ$  incidence angle and each line depicts a five-degree difference.

## 2.4 Simulation Architecture

To set up the simulation of the catadioptric camera, three groups of elements need to be added to a robot configuration in the `USARBot.ini` file.

First, the static mesh needs to be added. When this is spawned with the robot, it provides a surface on which a mirror reflection is projected. The Static Mesh placement coordinates relate to the SVC center of projection.

Second, the `CameraTextureClients` and virtual cameras need to be added which create the projections on the dynamic textures, mapped on the mirror surface, effectively simulating the reflection. The virtual cameras need to be placed on the SVC center of projection in a manner depicted in Figure 4.

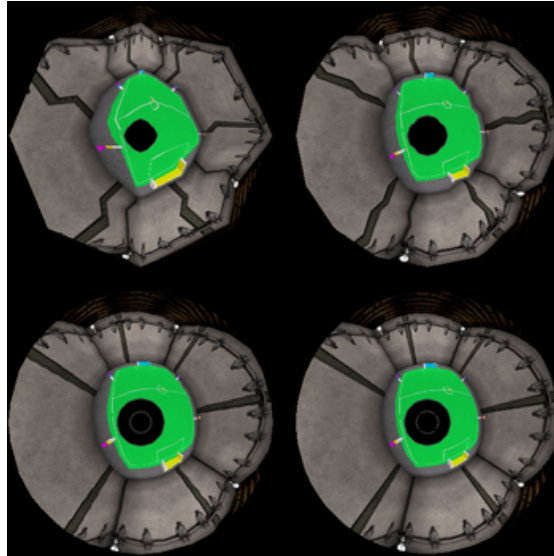
Finally, a camera needs to be added with a fixed relative position and orientation to capture the reflection and provide actual image data in an Unreal Client.

## 3 Experimental Results

The performance of the simulation method is demonstrated from two perspectives. First, sensor data quality is compared to the influence of the mirror polygon count on system performance. Second, self localization on a RoboCup 2006 four-legged league soccer field is performed within the simulated environment, using color histogram landmark detection and an extended Kalman filter.

### 3.1 Mirror Polygon Size Reflection Influence

As the UV-texture mapping method linearly interpolates texture placement on a single polygon, improper distortions occur when mirror polygons are too big.



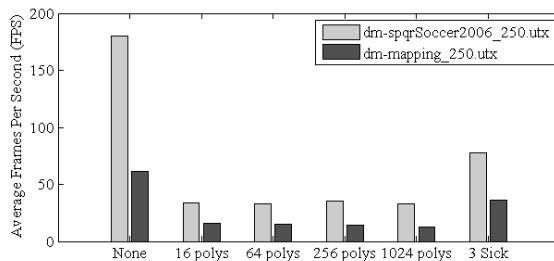
**Fig. 8.**  $360 \times 360$  Pixel mirror surface view comparison. Top left: 16 polygons. Top right: 64 polygons. Bottom left: 256 polygons. Bottom right: 1024 polygons.



**Fig. 9.** Mirror Surface Comparison Detail. Left: 256 polygons. Right: 1024 polygons.

Figure 8 shows omnidirectional images obtained from mirror surfaces described in subsection 2.2. The first two reflection images, produced by the mirrors based on the 16 and 64 polygon surfaces respectively, show heavy distortion of the reflection image due to the linear interpolation. The data provided by these surfaces does not relate to a proper reflection simulation. The 256 polygon surface does provide a proper reflection image, though Figure 9 shows a detail of the two surfaces with 256 and 1024 polygons respectively, which depicts a slight quality difference: The 256 polygon mirror still shows minor distortion, jaggings of a straight line, which in the 1024 polygon image mirror does not occur. Augmenting the number of polygons to 4096 resulted in images of a quality identical to those produced by the 1024 polygon surface, making the 1024 polygon surface the most proper surface for this simulation method.

Figure 10 shows the influence of mirror surface polygon count on system performance. All four mirror surfaces were mounted on a P2AT robot model and spawned in two RoboCup maps:



**Fig. 10.** Mirror surface polygon count related to USARSim performance.

- DM-Mapping\_250.utx and
- DM-spqrSoccer2006\_250.utx.

The test was run on a dual core 1.73 Ghz processor 1014MB RAM system with a 224MB Intel GMA 950 video processor. It is clear that FPS is influenced by the presence of an omnidirectional camera, though this influence does not depend on the number of polygons. The influence is of the omnidirectional camera is comparable with an other often used sensor, the LMS200 laser scanner.

### 3.2 Real Data Comparison

To compare the simulated omnidirectional camera to the real omnidirectional system on which it was based, we decided to do color histogram landmark detection on cylindrical projections of the omnidirectional camera data and perform self-localization using an Extended Kalman Filter in these two environments:

- on a RoboCup 2006 Four-Legged League soccer field of the Lab of Intelligent Autonomous Systems Group, Faculty of Science, University of Amsterdam, using the omnidirectional camera described in Subsection 2.1;
- in the USARSim DM-spqrSoccer2006\_250.utx map [13], using the developed omnidirectional camera simulation.

Four 3-dimensional RGB color histograms were created, one for each landmark color (i.e. *cyan*, *magenta* and *yellow*) and one for all non-landmark environment colors (*neg*). These four histograms  $H_{cyan}$ ,  $H_{magenta}$ ,  $H_{yellow}$ , and  $H_{neg}$  were used to define three pixel classifiers based on the standard likelihood ratio approach [14], labeling a particular  $rgb \in RGB$  value a specific landmark color if

$$\frac{P(rgb|landmark\ color)}{P(rgb|neg)} \geq \Theta \quad (3)$$

where  $\Theta$  defines a threshold value which optimizes the balance between the costs of false negatives and false positives.

The Extended Kalman Filter (EKF), a recursive filter which estimates a Markov process based on a Gaussian noisy model, is based on formulas provided

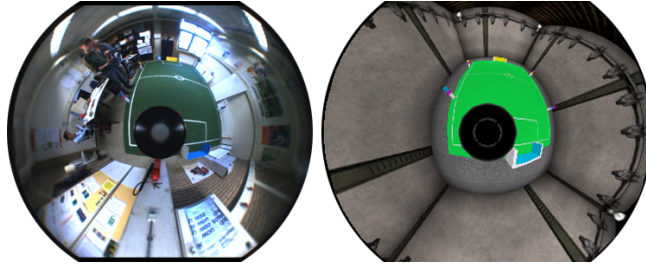


Fig. 11. Omnidirectional Image Data, Real and Simulated.

by Thrun and Burgard in [15]. The robot position  $\mu_t$  is estimated on the control input  $u_t$  and landmark measurement  $m_t$ , as defined by Equations 4 to 6 respectively.

$$\mu_t = (x_t, y_t, \phi_t) \quad (4)$$

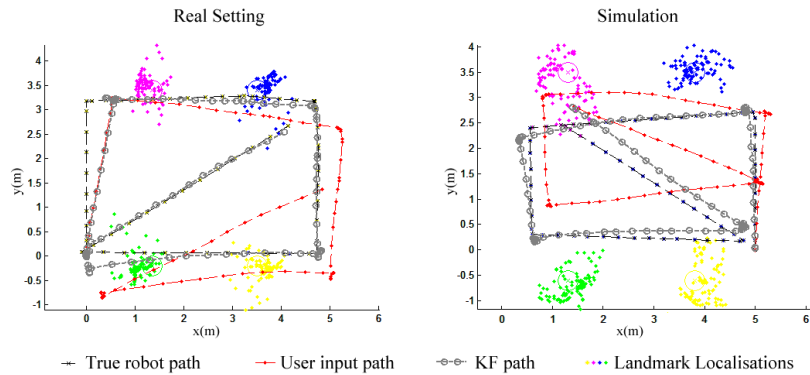
$$u_t = (v_t, \delta\phi_t) \quad (5)$$

$$m_t = (r_t, \gamma_t) \quad (6)$$

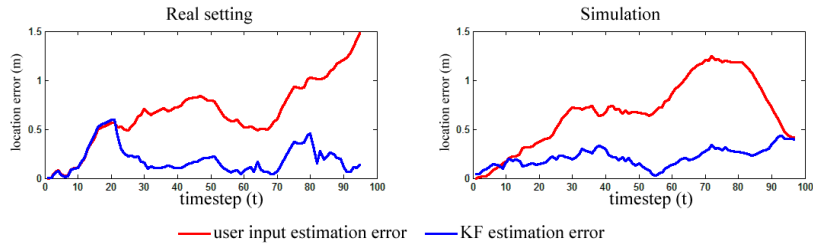
where  $x_t$  and  $y_t$  define the robot location in world coordinates,  $\phi_t$  defines the robot direction,  $v_t$  defines the robot velocity,  $\delta\phi_t$  defines robot rotation angle,  $\gamma_t$  defines the landmark perception angle. User input  $u$  is defined by noisy odometry sensor data and the measured distance to a landmark,  $r_t$ , is calculated based on the inverse of Equation 2,  $\theta \leftarrow f(r_i)$ , and camera effective viewpoint height.

**Real Setting** - In the real setting a Nomad Super Scout II, mounted with the omnidirectional vision system described in 2.1, was placed at a corner of the soccer field and it was driven on a clockwise path along the edges of the field until it reached its starting point, where it then crossed the field diagonally. The real omnidirectional camera suffers from many image degradation artifacts (defocus and motion blur) not yet modeled in simulation. The most important factors to reliably detect landmarks based on color histograms appear to be the constant lighting conditions and a clean laboratory. Under realistic circumstances detection algorithm also detects a large amount of false positives. For a fair comparison with the clean and constant simulation world, landmark locations in the recorded omnidirectional images were determined manually. Figure 11, left, shows omnidirectional image data obtained during this run.

Figure 12 and 13 on the left show results obtained in this run. The robot starts in the lower left of Figure 12 and moves up. The EKF estimate relies on the user input error (odometry) and mostly ignores the observations of the landmark ahead. This is the result of propagation of the high certainty of the initial robot location formulated by an a priori estimate error covariance matrix containing solely zero valued entries combined with a strong deviation of the user input, which leads to sensor readings to be discarded by the validation gate [16]. But as uncertainty about robot location increases, in the 23rd timestep measurements start to pass the validation gate, which leads to adjustments of the gain matrix



**Fig. 12.** 2006 Four Legged League Soccer Field Self Localization Results in a real and simulated environment.



**Fig. 13.** Absolute differences between true and predicted robot locations in meters.

*K*. The difference between the predicted and actual measurements of landmarks becomes now important, and the estimation error significantly lowers as can be seen in Figure 13, left. When the robot returns to the lower left corner, estimation error increases again as user input deviation to the left without enough landmark observations to correct this error. The estimation error drops again after timestep 80 as the robot rotates to face the center of the soccer field.

**Simulation** - In the simulated environment, a P2AT robot model mounted with the 1024 polygon mirror omnidirectional camera was spawned in an opposite where the real robot started its path and it followed a similar path on the simulated Four Legged League soccer field, though counter-clockwise. Landmark locations were determined using the color histogram classifier, and the EKF algorithm was run using the same parameters as with the real run. Figure 11, right, shows omnidirectional image data obtained during this run.

In Figure 12 on the right, the run in the USARSim environment, several differences can be observed. First of all, landmark measurements are less accurate, due to the fact that these were produced by the automatic color histogram based landmark location method. Secondly, the initial pose estimate is better as landmark measurements pass the validation gate instantly. This can be seen

because the KF estimation and the user input differ. In the first five steps KF estimation error is higher than with the user input, though as user input error increases the KF estimation error remains well below 0.5 meters over the course of the whole run. The most striking difference is the diagonal path at timesteps 84 to 100 in which the KF estimate describes a parallel path to the true path with an average location error of only 0.349 meters.

A general similarity between both runs can be observed as well. The EKF location estimate error is accurate within 0.6 meters for both the real and simulation run, while the location error based on user input which rises well above 1 meter in at several timesteps. The simulation results are not perfect, but show the same variations as can be obtained in a real setting.

## 4 Discussion and Further Work

Based on the theory described in Section 2 and regarding the results in the previous section we can conclude that USARSim can now simulate a Catadioptric Omnidirectional Camera which meets the Single Viewpoint Constraint. Like any other camera in the USARSim environment, it does not simulate all effects that can degrade image quality. The result is that a color histogram based pixel classifier can be used in the simulated environment to accurately locate landmarks where this classifier fails to do so in a real setting. A valid simulation model should implement these degradation models and USARSim as a whole could benefit from such an addition.

The omnidirectional camera simulation model does provide image data which has been verified to concur with realistic omnidirectional image transformation equations. The omnidirectional camera is also a valuable addition to the RoboCup Rescue League as test results show the benefits of utilizing omnidirectional vision. As it is known that landmarks are in line of sight regardless of robot orientation, it implies that victims will be as well. This will improve robot efficiency and team performance. More importantly, the omnidirectional camera simulation adds to the realism of the high fidelity simulator USARSim, bringing the Simulation League one step closer to the Rescue League.

Further work could also be done to create omnidirectional vision systems based on other real-life cameras used in the RoboCup. Not only catadioptric systems with different mirror shapes, but also omnidirectional cameras based on for instance fish-eye lenses, could be explored.

## 5 Acknowledgments

The authors would like to thank Bas Terwijn for helping out with providing the real omnidirectional image data at the Intelligent Systems Laboratory Amsterdam. A part of the research reported here is performed in the context of the Interactive Collaborative Information Systems (ICIS) project, supported by the Dutch Ministry of Economic Affairs, grant nr: BSIK03024.

## References

1. Baker, S., Nayar, S.K.: A theory of single-viewpoint catadioptric image formation. *International Journal of Computer Vision* **35** (1999) 1–22
2. Gaspar, J., Winters, N., Santos-Victor, J.: Vision-based navigation and environmental representations with an omnidirectional camera. *IEEE Transactions on Robotics and Automation* **16** (2000) 890–898
3. Bunschoten, R., Kröse, B.: Robust scene reconstruction from an omnidirectional vision system. *IEEE Transactions on Robotics and Automation* **19** (2003) 351–357
4. Booij, O., Terwijn, B., Zivkovic, Z., Kröse, B.: Navigation using an appearance based topological map. In: *Proceedings of the International Conference on Robotics and Automation ICRA'07, Roma, Italy* (2007) 3927–3932
5. Heinemann, P., Haass, J., Zell, A.: A combined monte-carlo localization and tracking algorithm for robocup. In: *Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '06)*. (2006) 15351540
6. Lima, P., Bonarini, A., Machado, C., Marchese, F.M., Marques, C., Ribeiro, F., Sorrenti, D.G.: Omnidirectional catadioptric vision for soccer robots. *Journal of Robotics and Autonomous Systems* **36** (2001) 87–102
7. Carpin, S., Lewis, M., Wang, J., Balakirsky, S., Scrapper, C.: Bridging the Gap Between Simulation and Reality in Urban Search and Rescue. In Lakemeyer, G., Sklar, E., Sorrenti, D., Takahashi, T., eds.: *RoboCup 2006: Robot Soccer World Cup X. Volume 4434 of Lecture Notes on Artificial Intelligence.*, Berlin Heidelberg New York, Springer (2007) 1–12
8. Jacoff, A., Messina, E., Evans, J.: A standard test course for urban search and rescue robots. In: *Proceedings of the Performance Metrics for Intelligent Systems Workshop*. (2000) 253–259
9. Geyer, C., Daniilidis, K.: Catadioptric projective geometry. *International Journal of Computer Vision* **45** (2001) 223–243
10. Grassi Jr., V., Okamoto Jr., J.: Development of an omnidirectional vision system. *Journal of the Brazilian Society of Mechanical Sciences and Engineering* **28** (2006) 58–68
11. Beck, D., Ferrein, A., Lakemeyer, G.: A simulation environment for middle-size robots with multi-level abstraction. In: *Proceedings CD of the 11th RoboCup International Symposium*. (2007)
12. Nayar, S.K.: Catadioptric Omnidirectional Camera. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (1997) 482–488
13. Zaratti, M., Fratarcangeli, M., Iocchi, L.: A 3D Simulator of Multiple Legged Robots Based on USARSim. In Lakemeyer, G., Sklar, E., Sorrenti, D., Takahashi, T., eds.: *RoboCup 2006: Robot Soccer World Cup X. Volume 4434 of Lecture Notes on Artificial Intelligence.*, Berlin Heidelberg New York, Springer (2007) 13–24
14. Fukunaga, K.: *Introduction to statistical pattern recognition* (2nd ed.). Academic Press Professional, Inc., San Diego, CA, USA (1990)
15. Thrun, S., Burgard, W., Fox, D.: *Probabilistic Robotics*. MIT Press, Inc., Cambridge, MA, USA (2005)
16. Kristensen, S., Jensfelt, P.: An experimental comparison of localisation methods, the mhl se ssions. In: *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'03)*. (2003) 992–997