Testing omnidirectional vision-based Monte-Carlo Localization under occlusion

E. Menegatti, A. Pretto and E. Pagello†
Intelligent Autonomous Systems Laboratory
Department of Information Engineering
The University of Padua, Italy
†*also with:* Institute ISIB of CNR Padua, Italy
{emg, albe76, epv}@dei.unipd.it

***Abstract*— One of the most challenging issue in mobile robot navigation is the localization problem in densely populated environments. In this paper, we present a new approach for vision-based localization able to solve this problem. The omnidirectional camera is used as a range finder sensitive to the distance of color transitions, whereas classical range finders, like lasers or sonars, are sensitive to the distance of the nearest obstacles. The well-known Monte-Carlo localization technique was adapted for this new type of range sensor. The system runs in real time on a low-cost pc. In this paper we present experiments, performed in a crowded RoboCup Middle-size field, proving the robustness of the approach to the occlusions of the vision sensor by moving obstacles (e.g other robots); occlusions that are very likely to occur in a real environment. Although, the system was implemented for the RoboCup environment, the system can be used in more general environments.**

I. Introduction

Localization is the fundamental problem of estimating the pose of the robot inside the environment. In a dynamic multi-agent world a precise localization is necessary to effectively perform high level coordination behaviors or information sharing techniques: at the same time the presence of other robots makes localizations harder to perform. In fact, if the density of moving obstacles in the environment is high, occlusion of the robot's sensors is very frequent. A localization system for a densely populated environment like the Middle-Size RoboCup field, must be proved robust against occlusion. In this paper, we explicitly discuss the robustness of our system against occlusion that was just hinted in [7].

To manage the uncertainty on the localization coming from the measurements we used the well-know Monte-Carlo localization technique. Usually two kinds of sensors have been used in the Monte-Carlo localization approach: range finder devices performing scan matching on a metric map of the environment [2], [13] or vision sensors either to recognize landmarks matched within a map [3], [9], [5] or to find the reference image most similar to the image currently grabbed by the robot [14], [8], [6].

In this work, we use an omnidirectional vision system as sensor to emulate and enhance the behavior of range-finder devices. The most significant differences are: (i) a conventional range-finder device senses the obstacles in the environment, while our sensor is sensitive to the chromatic transitions in the environment, and (ii) the sensor can reject some scans if along the ray a wrong chromatic transition is detected: we called this technique *ray discrimination*. To take into account the new information given by this sensor, we had to slightly modify MCL (Monte-Carlo Localization) technique proposed in [13]. The experiments presented in this paper aim to shown the robustness of our system in a densely populated environment. They were carried on in our training RoboCup Middle-Size field of 8×4 meters. However the algorithm proposed is very general and applicable also outside of the RoboCup domain. The only assumption made in this work is that a geometric map of the environment, augmented with the information on the color transition, is available.

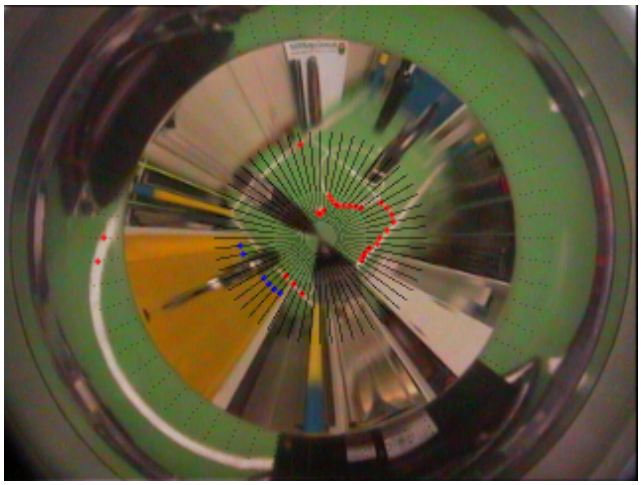


Fig. 1. A snapshot of the algorithm looking for chromatic transitions. Green-white chromatic transitions are highlighted with red crosses, green-yellow transitions with blue crosses, black pixels represent the sample points used for the scan that is performed in a discrete set of distances. Notice the crosses in the outer part of the mirror: this part is used for low distance measures. If a not expected transition is detected (i.e. another robot detected along some ray, look in the image at the dark shapes inside the green field) the scan is stopped and a typical value (*FAKE RAY*) is stored in the distances vector. In this paper we focus our attention in the localization problem in presence of sensor's occlusions caused by other agents.

II. A Selective Range Finder using Omnidirectional Vision

As we said, we search the omnidirectional image for *chromatic transitions of interest*: we are interested in *green-*

white, *green-blue* and *green-yellow* transitions. These transitions are related to the structure of the RoboCup fields, where the play-ground is green, lines are white, and goals and corner posts are blue or yellow. We measure the distance of the *nearest chromatic transitions of interest* along rays every 6 degrees and with a sample step corresponding to 4 centimeters in the world coordinate system, as shown in Fig.1. The omnidirectional vision sensor is composed by a camera pointed upward to a multi-part mirror with a custom profile [11]. This gives our "range finder" a 360 degrees field of view, much larger than the usual field of view of conventional range finders. Moreover, the custom profile of the omnidirectional mirror was designed to have good accuracy both for short and long distance measurements. In fact, conic omnidirectional mirrors fail to obtain good accuracy for short distance measurements (because the area close to the robot is mapped in a very small image area), while hyperbolic mirrors fail to obtain good accuracy for long distance measurements (because of the low radial resolution far away from the sensor). With our mirror, the area surrounding the robot is imaged in the wide external ring of the mirror and the area far away from the robot is imaged in the inner part of the mirror where the radial resolution is high, see Fig.1. The inner part of the mirror is used to measure objects farther than 1 m away from the robot, while the outer part is used to measure objects closer than 1 m from the robot. We first scan for chromatic transitions of interest close to the robot's body, in the outer mirror part, and then we scan the inner part of the image from the center of the image up to 4 meters away from the robot's body.

The system looks for the chromatic transitions of interest along 60 rays by mapping the color of the sampled pixels into one of the class of the 8 RoboCup colors[1] plus a further class that include all colors not included in the former classes (called *unknown color*). At the setup stage, we perform a color calibration depending on the illumination of the environment and the RGB color space is quantized into a look-up table to obtain a real-time color quantization.

Once a chromatic transition of interest is detected, its distance is stored in a vector. We have three different vectors, one for every chromatic transition of interest. During the radial scan, we can distinguish three situations:

1) a chromatic transition of interest is found; the real distance of that point is stored in the corresponding vector;
2) there are no transitions of interest, a characteristic value called *INFINITY* is stored in the vector (this means no transition can be founded along this ray);
3) a not expected transition is found: a *FAKE RAY* value is stored in the vector (this means something is occluding the vision sensor). All rays with *FAKE RAY* value are discarded in the matching process (*ray discrimination*).

[1]In RoboCup environment the ball is red, the lines are white, the goals are blue and yellow, the robots are black, the robots' marker are cyan and magenta

III. Monte-Carlo Localization

Monte-Carlo localization (MCL) has been successfully used by many researchers [12], [2], [13]. This is a probabilistic methods based on Batesian Filtering (Marked Localization in robotics) [1], [4]. It calculates the probability density of robot position (the **belief**) and recursively propagates this probability density using motion and perception information. In our implementation, motion data come from the odometric sensors and perception data come from the omnidirectional vision system. The belief about the robot's position is updated every time the robot grabs a new image (i.e. a new observation of the world). The belief about the robot position is represented with a set of discrete points in the configuration space of the robot. These points are called **samples**. To update the belief over time, these samples are updated. To every sample is associated a weight indicating the probability that the robot is occupying that position. The samples with the lowest weights are deleted and the samples with highest weights survive and generate new samples.

To update the belief, the knowledge of two conditional densities, called *motion model* and *sensor model* is needed. The *motion model* $p(l_t|l_{t-1}, a_{t-1})$ is a probabilistic representation of the robot's kinematics which describes a posterior density over possible following robot's poses. We implemented the MCL system on an holonomic robot, called Barney. The peculiarity of this robot is that it can move in any direction without the need of a previous rotation. Updating the robot's position according only to the kinematics does not take into account errors given by odometric inaccuracy and possible collisions of the robot with other obstacles. Therefore a random noise term is added to the values given by the odometry. Noise is modelled with Gaussian zero centered random variables, they depend on both the amount of translation and of rotation. In our implementation a good value for the error term is a normal distribution with σ equal to 200 mm/meter for translation and 30 deg/meter for rotation. These values are quite large to take into account the frequent slipping of the three driving wheels of the holonomic platform. The *sensor model* describes the probability for taking certain sensor measurements at certain poses: it strongly depends on the particular sensor used and it is described in detail in the next section. The localization algorithm basically consists of three steps:

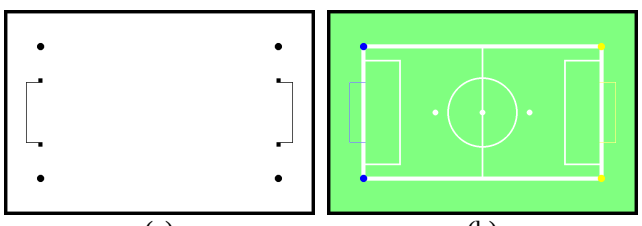


Fig. 2. The metric maps used for expected distances computation: in (a) are represented the fix obstacles, in (b) are represented all the chromatic transitions of interest of the environment

- all particles are moved according to the motion model of the last kinematics measure;
- the weights of the particles are determined according to the observation model for the current sensor reading;
- a re-sampling step is performed: high probability particles are multiplied, low probability ones are discarded;

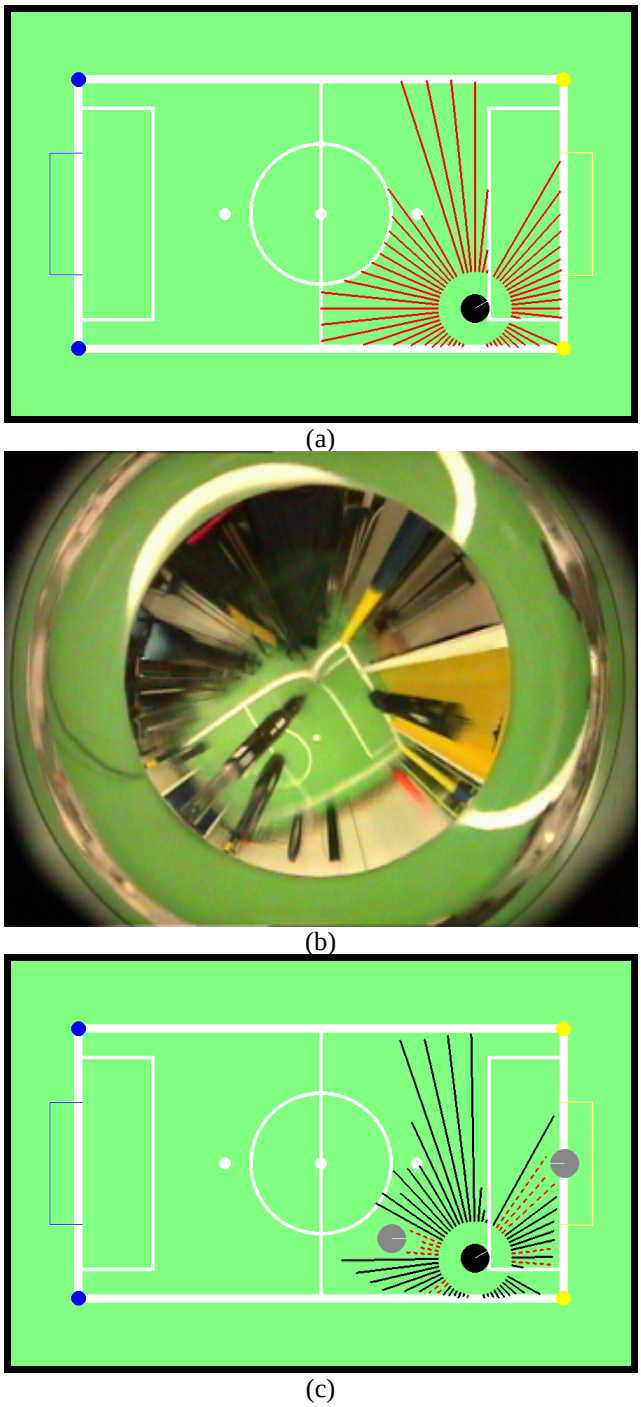
Finally, the process repeats from the beginning. For more details please refer to [2], [13].

A. Model of the new range finder

The sensor model $p(o|l)$ describes the likelihood to obtain a certain sensor reading o given a robot pose l. As introduced in Sec. III, the sensor model is used to compute the weights of the particles. For each particle j, located in the pose l^j, the associated weight is proportional to $p(o|l^j)$ (i.e. to the likelihood of obtaining the sensor reading o when the robot has pose l^j). To calculate $p(o|l^j)$, we need to know the *"expected scan"* $o(l)$. The expected scan is the scan an ideal noise-free sensor would measure in that pose, if in the environment there are no obstacles. The likelihood $p(o|l)$ can be calculated as $p(o|l) = p(o|o(l))$. In other words, the probability $p(o|o(l))$ models the noise in the scan by the expected scan [2], [13].

When using a sonar or a laser, like in [2], [13], the expected scan is computed from a metric map of the environment. The expected scan is obtained calculating the reflections of the sonar or laser beams against the walls and the fix obstacles. If used in the current RoboCup Middle-Size field this would lead to a very sparse information because the only detectable features are the goals and the corner posts, as depicted in Fig. 2(a). Moreover, if you have several robots in the field they will likely occlude most of the static obstacles and so the available features will be even less. For this reason we proposed this new sensor that can detect all color transition existing in Fig. 2(b). This enable us to detect much more fix features performing a more reliable *"scan matching"*. The map in Fig. 2(b) shows the chromatic characteristics of the environment. We use this map to pre-compute the expected scan finding with a ray-tracing approach the *nearest chromatic transition of interest* for every pose. Moreover, we use the information about the fix obstacles extracted from the map of Fig. 2(a) to improve the scanning process; for instance if we find a yellow pixel, this can be only a goal or a corner-post, i.e. an object that is outside of the boundary of the field, so it is not worth looking farther for a white line, thus we stop the scanning process for this ray.

From every image grabbed by the robot, we obtain three scans: one for every chromatic transition of interest (green-white, green-blue and green-yellow, see Sec. II). Thanks to the ability to distinguish between different colors, we are able to filter out some rays when a "wrong" chromatic transition is detected (i.e. a chromatic transitions that we aren't looking for, see Sec. II). This is the case, for example, when another robot partially occlude the omnidirectional sensor: black pixels are detected along the occluded rays and a green-black transition is found, but we aren't looking for any transitions with black color. So, we can discard these



(a) (b) (c)

Fig. 3. An examples of expected and measured scan for the green-white color transition. Given a pose, in (a) is represented the expected scan for an ideal noise-free sensor in a free environment. In (b) is shown the frame grabbed by the robot in that pose, in (c) is represented the corresponding measured scan. In (c) the black line represents the measured distances while the dotted red line represents the rays in which a not expected transitions is detected (*FAKE RAYS*). This can be caused by image noise or other robots (represented with gray circles). Look for example in the omnidirectional image (b) at the yellow goal: inside there is a robot (the goalkeeper) and three rays of the scan detect it (c) (along these rays is detected the black color that we are not searching for). For all these rays a *FAKE RAYS* value is stored instead of the proper distance.

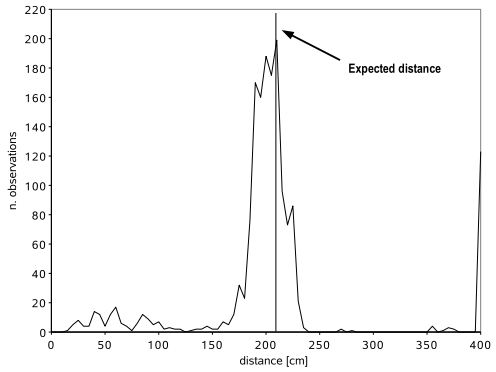
rays that could negatively affect the sensor reading (*ray discrimination*). This process is not possible with classical range finder sensors and one needs to apply quite complex algorithms to filter out in the range scan the moving obstacles from the reference obstacles.

In Fig. 3 a comparison between an expected scan (a) and a real sensor scan (c) for the green-white transition is presented. In the middle (b) is the image grabbed by the robot. What makes the difference between the expected scans and the real scans is the presence of occlusion in the field of play and the presence of noise in the image. If occlusion is present, a non-expected transition is detected, and the ray is labelled as *FAKE_RAY* (the red dotted lines in 3 (c)). Due to the image noise, it might happen that a color transition is not detected or is detected at a wrong distance.

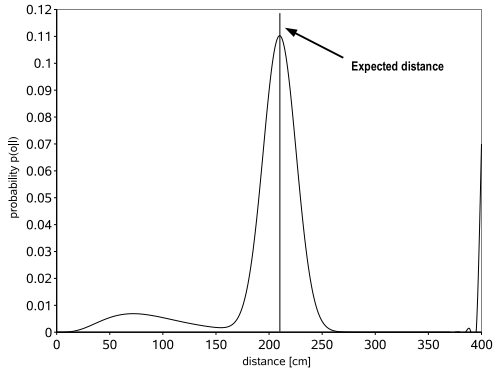
1) Sensor noise: The probability $p(o|o(l))$ models the noise in the measured scan conditioned on the expected scan. For every frame grabbed by the sensor we obtain three scans, one for each chromatic transitions of interest, so we have to calculate three probability values. Since every scan is composed by a set of distances, one for each ray, we first model the probability that a single ray correctly detects the chromatic transition and then we take into account all rays. Eventually, we need to combine the three probability values given by the three chromatic transitions of interest. To compute the probability to obtain for a single ray a distance o_i given the pose l ($p(o_i|l) = p(o_i|o(l))$), we collected a large number of omnidirectional images (about 2.000) in different known poses in the field. Then, we measured the distance of the chromatic transitions of interest. As an example, the probability density of the measured distance $p(o_i|l)$ for the green-white color transition is plotted in Fig. 4(a). We described this density with the mixture of three probability density of Eq. 1. The numerical values of the parameters in Eq. 1 are calculated with a modified EM algorithm [10]. The resulting mixture, for the green-white transition, is plotted in Fig. 4(b). The three terms in Eq. 1 are respectively: an Erlang probability density, a Gaussian probability density and a large discrete density. The Erlang variable models wrong readings in the scan caused by image noise and non-perfect color segmentation. The index n depends on the profile of the omnidirectional mirror used in the sensor. Our mirror (Sec. II) maps the area around the robot in the outer image ring where we have good accuracy and almost no noise, while in the inner part a certain amount of noise is present. We set the value of n, the Erlang variable, equal to the index of the first pixel scanned in the inner part of the image. So, the Erlang density will have a peak at the distance corresponding to the transition between the two mirror parts. The Gaussian distribution models the density around the maximum likelihood region, i.e. the region around the true value of the expected distance. The discrete density represents the probability of obtaining an *INFINITY* value for the distance, as described in Sec. II.

$$p(o_i|l) = \zeta_e(\frac{\beta^n o_i^{n-1} e^{-\beta o_i} 1(o_i)}{(n-1)!}) + \zeta_g \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-(o_i - g(l,\alpha_i))^2}{2\sigma^2}} + \zeta_d \delta(o_i - \infty) \quad (1)$$

where $\zeta_e, \zeta_g, \zeta_d$ are the mixture coefficients, with $\zeta_e + \zeta_g + \zeta_d = 1$. We computed a different mixture for every different chromatic transition.



(a)



(b)

Fig. 4. In (a) the distribution of measured distances for an expected known distance. Notice the peak at the expected distance, the measures before the expected one due to the image noise and the high number of max distance measures that represents that no chromatic transition has been detected. In (b) the density $p(o|l)$ that represent our sensor model computed using EM-algorithm. The plotted curve is the result of three contributions: (i) an Erlang variable with index n which depending on the geometry of the mirror (low noise in the close-distance mirror part), (ii)a Gaussian distribution centered at the expected distance and (iii) a discrete distribution representing the measurements resulting in the $INFINITY$ value.

2) Sensor occlusion and ray discrimination technique: Once the $p(o_i|l)$ is computed, it's possible to compute the probability of the whole scan given a pose l (2). To cope with unexpected measures due to occlusion of the scans by the moving objects in the environment (i.e. the other robots and the ball), we filtered out all rays which distance o_i equal the *FAKE RAY* value (ϕ in the formulas, see Sec. II). The detection of occluding obstacles along the rays of a scan is very frequent in the densely crowded environment like the Middle-Size RoboCup field. In conventional range finders there isn't any ray discrimination system, so all measured distances contribute to sensor model computation: if a large number of distances are affected by the

presence of other agents around the robot, localization process might fail. Our ray discrimination technique allow us to compute sensor model only with a subset of reliable distances: this enable to obtain a more reliable localization without using other techniques, e.g. *distance filters* [4], that can affect negatively the computational performance of the system.

$$p(o|l) = \prod_{\{i|o_i \neq \phi\}} p(o_i|l) = \prod_{\{i|o_i \neq \phi\}} p(o_i|g(l,i)) \quad (2)$$

Returning to Monte Carlo Localization, we are now able to compute the weight $w^{(j)}$ associated to each particles j. Since we get three scans for every frame, using (2) we obtain three weights for each particle. To obtain a single weight value, after a normalization step, we simply compute the product of the three weights

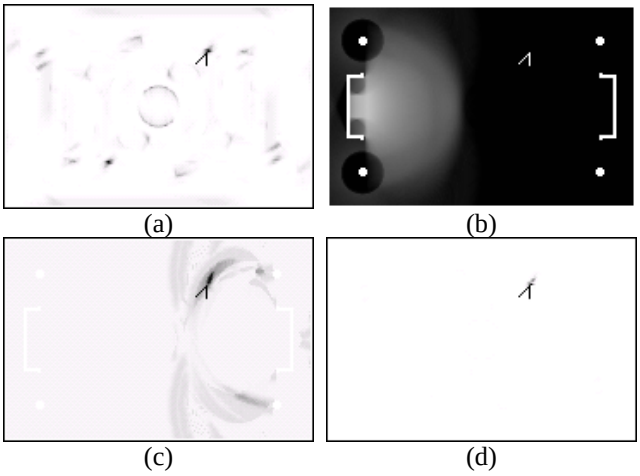


Fig. 5. Probability distributions $p(o|l)$ for all possible positions l of the robot in the field given the scans of a single image. Darker points corresponds to high likelihood. The arrow represents the actual robot's pose. In (a) is represented the probability given the scan for green-white transitions, in (b) for green-blue transitions, in (c) for green-yellow transitions, in (d) the three probability densities are combined.

In Fig. 5, we give a pictorial visualization of the weights calculated by the three different scans of the three chromatic transition of interest. The real pose of the robot is marked by the arrow. Higher weight values are depicted as darker points, lower weight values are depicted as lighter points. In Fig. 5 (a), are represented the weight contributions calculated from the scan looking for the green-white transition. One can notice that, due to the symmetry of the white lines in the field two symmetric positions resulted to have high likelihood. In Fig. 5 (b), are depicted the weight contributions calculated from the scan looking for the green-blue transition. One can notice that all positions far away from the blue goal have a high likelihood, because no green-blue transition was found in the image scan. In Fig. 5 (c), are represented the weight contributions calculated from the scan looking for the green-yellow transition. One can notice there is an approximate symmetry around the yellow goal. All these contributions are combined together to calculate the overall weights represented in Fig. 5 (d).

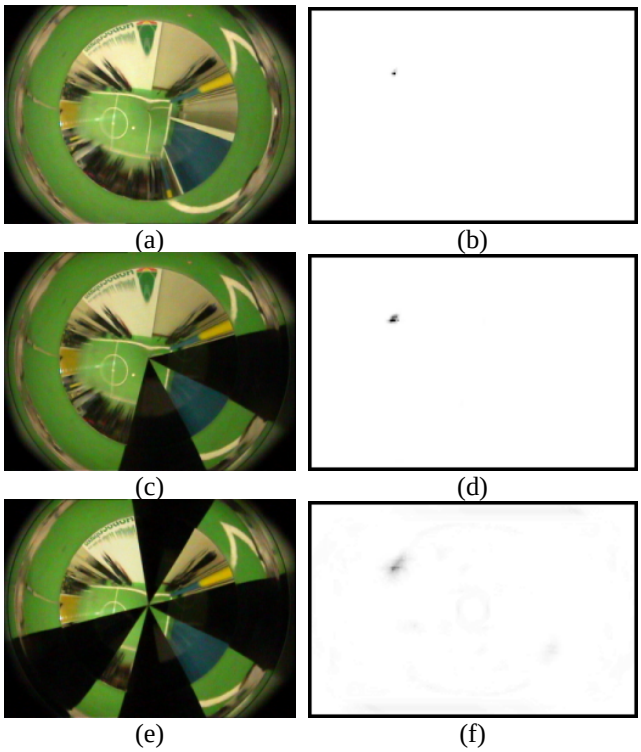
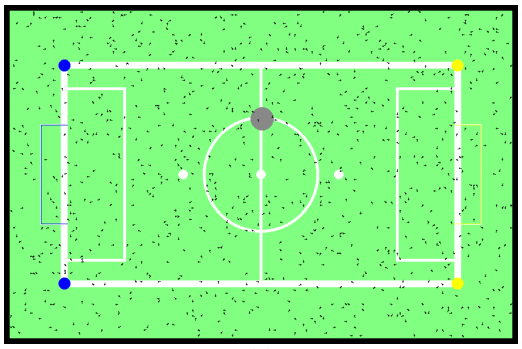


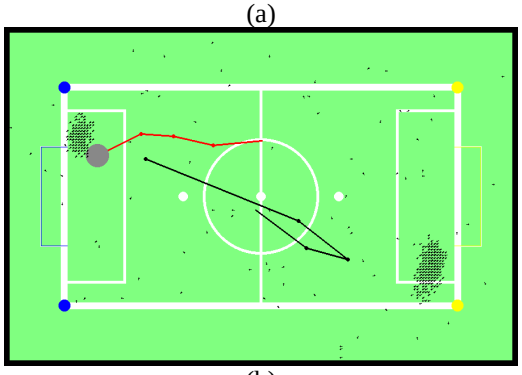
Fig. 6. For the same position, the probability distributions for different amount of sensor's occlusion: in (b) for 0% of occlusion, in (d) for 25% and in (f) for 50%. In (a),(c),(d) the images grabbed in these 3 situations respectively. Notice that in a more densely crowded environment (f) the high likelihood region is wider than an empty one (b), but most of the probability is still condensed around the right position. Occlusion is obtained with black stripes that simulate the presence of other robots closed to the sensor.

The weights with higher values are clustered only around the actual position of the robot. The Fig. 6 shows the probability distributions for the same pose with different amount of sensor's occlusion: with high rate of occlusion the uncertainty increases, but most of the probability is still condensed around the right position. Occlusion is obtained covering the sensor with black strips. Every strip cover 12.5% of the sensor and well simulate the presence of one robot close to the sensor. In real situations is extremely hard to have more than two robot close to the sensor, while other robots are usually quite far and occlude a small fraction of the sensor.

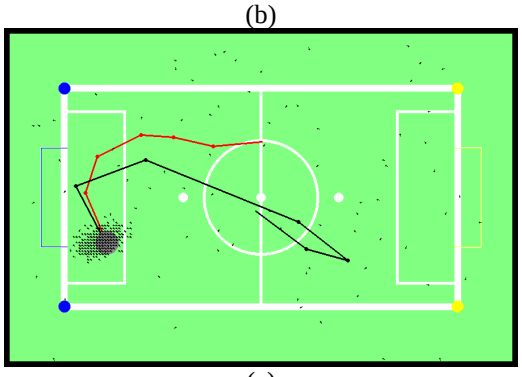
In order to improve the performance of the system, the distances (see Sec. II) in the environment are discretized in a grid of 5x5 cm cells, in a way similar to [4]. The expected distances for all poses and the probabilities $p(o_i|g(l,i))$ for all $g(l,i)$ can be pre-computed and stored in six (two *for each* chromatic transition) look-up tables. Each look-up tables takes about 13Mb. In this way the probability $p(o_i|l)$ can be quickly computed with two look-up operations, this enables our system to work in real-time at 10 Hz on a PC-104 Pentium III 700 MHz fitted with 128 Mb of RAM using 1000 particles.



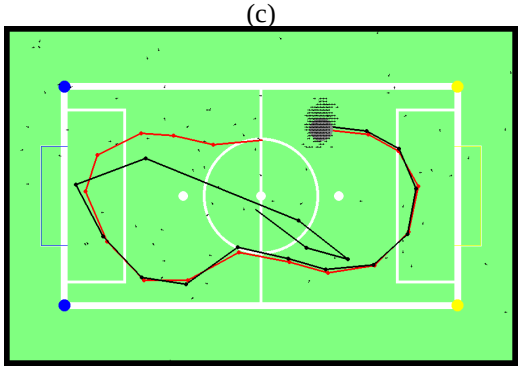
(a)



(b)



(c)



(d)

Fig. 7. A sequence of global localization: the gray circle represents actual robot pose, the red line represents ground-truth path, the black line represents the estimated path of the robot, the black points represent the particles. In (a) particles are uniformly distributed (no knowledge is available on robot position), in (b), after moving 4 meters away and grabbing 5 frames and getting 5 odometric readings, the particles are condensed around two possible poses (due to environment symmetry). In (c), after 6 meters, 7 frames and 7 odometric readings, uncertainty is solved and particles are condensed around the actual pose of the robot. In (d) after 18 steps: the position of the robot is well tracked. The particles scattered in the environment are the 10% of the total number of particles uniformly distributed to solve the kidnapping problem.

IV. Experiments

We evaluated our approach on an holonomic custom-built robotic platform, in a 8x4 m RoboCup soccer field. The robot was equipped with the omnidirectional sensor described in Sec. II. In order to shown the robustness of our approach in densely crowded environments, we tested the system on six different paths (like the one shown Fig. 7). For each path we collected five sequences of omnidirectional images respectively with 0%, 12.5%, 25%, 37.5%, and 50% occlusion. In addition to the image we recorded for every position the ground truth pose of the robot and the odometric readings between two consecutive positions. An example of frames grabbed in the same position, but with different amount of occlusion is shown in Fig. 6(a),(c),(e). In order to take into account the odometric errors, robot movements were performed by remote control. We tested our algorithms for the three fundamental localization problems: global localization (the robot must be localized without any a priori knowledge on the actual position of the robot, i.e. Fig. 7 (a)(b)), position tracking (a well localized robot must maintain the localization, i.e. Fig. 7 (c)(d)) and kidnapped robot (a well-localized robot is moved to some other pose without any odometric information).

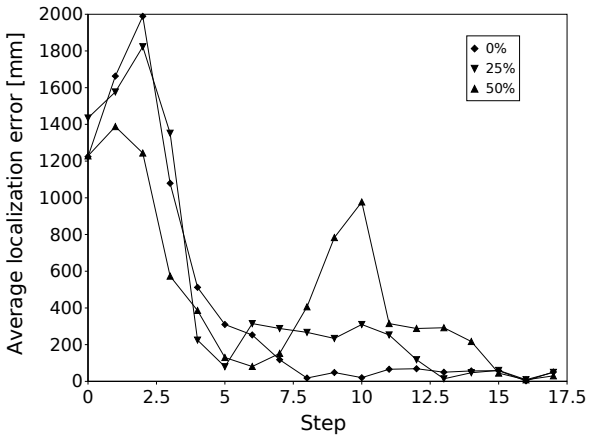


Fig. 8. The plots compares the global localization errors for a fixed path with different amount of sensor's occlusion.

In Fig. 8 is shown the average error for a global localization experiment along the same reference path for three different amounts of sensor occlusion. Obviously without occlusion, localization is quickly and accurate, but also in a densely crowded environment (sensor always covered for a rate of 50%) the robot is able to localize itself and to maintain localization with good accuracy. We obtained very good results also in the kidnapped robot problem. Recovery from a localisation failure is obtained thanks to a small amount of samples (10% of the total number of samples) uniformly distributed in the environment. A few steps after a kidnapped robot episode most of the samples are again concentrated around the correct position and the situation is the same of global localization (due to lack of space we did not reported these experiment in this paper).

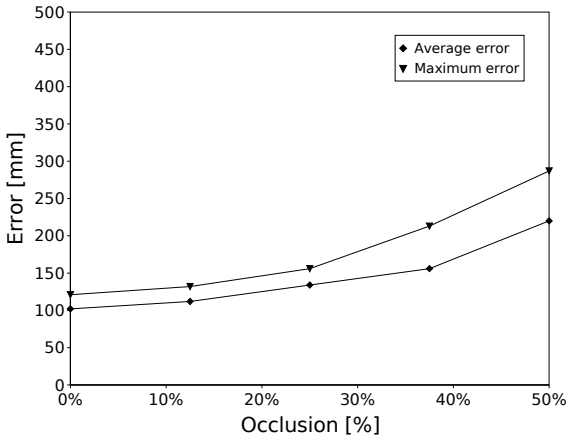


Fig. 9. Statistical evaluation of our system in the position tracking problem for all our reference paths. Accuracy (average error end maximum error) is represented for different amount of sensor's occlusion (0%, 12.5%, 25%, 37.5%, 50%).

Finally, we tested our approach in the conventional situation of position tracking, for different amount of occlusions: in Fig. 9 is shown the average and the maximum error over all reference paths. Notice that both remain small also in a densely and constantly crowded environment.

V. Conclusions

In this paper we propose a new vision-based Monte Carlo localization system particularly suitable for densely crowded environments. The omnidirectional vision sensor is used to emulate the behavior of range-finder devices and, thanks to the ability to distinguish different color transitions it can detect and reject data caused by other robots occluding the sensor. We tested our system in the Middle-Size RoboCup domain with different amount of sensor's occlusion to simulate the presence of other agents around the robot: experiments demonstrated that our approach is able to localize the robot with a maximum error of less than 30 cm even in the very unlikely situation of a continued occlusion of 50% of the omnidirectional camera. Even if our system was developed for the RoboCup competition, it can be used to localize the robot in any environments simply building the right metric and chromatic maps that depicts respectively the fix obstacles and the chromatic transitions of interest of the environment. These maps can be as simple as drawings stored in image files, representing the plan of the environment augmented with the information on the color transitions. From these maps, the system will automatically recalculate all look-up tables used in the localization process. At the moment of writing we are testing our system in a indoor office environment without any artificial landmark or modification of the environment.

References

[1] S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for on-line non-linear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, February 2002.

[2] F. Dellaert, D. Fox, W. Burgard, and S. Thrun. Monte carlo localization for mobile robots. In *Proc. of the IEEE International Conference on Robotics & Automation*, 1999.

[3] S. Enderle, M. Ritter, D. Fox, S. Sablatng, G. Kraetzschmar, and G. Palm. Soccer-robot locatization using sporadic visual features. In E. Pagello, F. Groen, T. Arai, R. Dillman, and A. Stentz, editors, *Proceedings of the 6th International Conference on Intelligent Autonomous Systems (IAS-6)*. IOS Press, 2000.

[4] D. Fox, W. Burgard, and S. Thrun. Markov localization for mobile robots in dynamic environments. *Journal of Artificial Intelligence Research*, 11, 1999.

[5] J.-S. Gutmann and D. Fox. An Experimental Comparison of Localization Methods Continued. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2002.

[6] Emanuele Menegatti, Takeshi Maeda, and Hiroshi Ishiguro. Image-based memory for robot navigation using properties of the omnidirectional images. *Robotics and Autonomous Systems, Elsevier*, page (to appear), 2004.

[7] Emanuele Menegatti, Alberto Pretto, and Enrico Pagello. A new omnidirectional vision sensor for Monte-Carlo localisation. In *Proceedings of the 8th RoboCup International Symposium 2004*, page (to appear), 2004.

[8] Emanuele Menegatti, Mauro Zoccarato, Enrico Pagello, and Hiroshi Ishiguro. Image-based Monte-Carlo localisation with omnidirectional images. *Robotics and Autonomous Systems, Elsevier*, page (to appear), 2004.

[9] Thomas Röfer and Matthias Jüngel. Vision-based fast and reactive monte-carlo localization. In *IEEE International Conference on Robotics and Automation*, 2003.

[10] A. P. Dempster N. M. Laird D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. In *Journal of the Royal Statistical Society*, volume 39 of *B*, pages 1–38. 1977.

[11] E. Menegatti F. Nori E. Pagello C. Pellizzari D. Spagnoli. Designing an omnidirectional vision system for a goalkeeper robot. In A. Birk S. Coradeschi and P. Lima, editors, *RoboCup-2001: Robot Soccer World Cup V*, LNAI No. 2377, pages 78–87. Springer, 2001.

[12] S. Thrun, M. Beetz, M. Bennewitz, W. Burgard, A.B. Cremers, F. Dellaert, D. Fox, D. Hähnel, C. Rosenberg, N. Roy, J. Schulte, and D. Schulz. Probabilistic algorithms and the interactive museum tour-guide robot minerva. *International Journal of Robotics Research*, 19(11):972–999, 2000.

[13] S. Thrun, D. Fox, W. Burgard, and F. Dellaert. Robust monte carlo localization for mobile robots. *Artificial Intelligence*, 128(1-2):99–141, 2000.

[14] J. Wolf, W. Burgard, and H. Burkhardt. Using an image retrieval system for vision-based mobile robot localization. In *Proc. of the International Conference on Image and Video Retrieval (CIVR)*, 2002.