A New Omnidirectional Vision Sensor for Monte-Carlo Localization

E. Menegatti[1], A. Pretto[1], and E. Pagello[1 2]

[1] Intelligent Autonomous Systems Laboratory
Department of Information Engineering
The University of Padua, Italy
emg@dei.unipd.it
[2] also with Institute ISIB of CNR Padua, Italy

Abstract. In this paper, we present a new approach for omnidirectional vision-based self-localization in the RoboCup Middle-Size League. The omnidirectional vision sensor is used as a range finder (like a laser or a sonar) sensitive to colors transitions instead of nearest obstacles. This makes it possible to have a more reach information about the environment, because it is possible to discriminate between different objects painted in different colors. We implemented a Monte-Carlo localization system slightly adapted to this new type of range sensor. The system runs in real time on a low-cost pc. Experiments demonstrated the robustness of the approach. Event if the system was implemented and tested in the RoboCup Middle-Size field, the system could be used in other environments.

1 Introduction

Localization is the fundamental problem of estimating the pose of the robot inside the environment. Several techniques based on the Monte-Carlo localization (MCL) approach was developed. Two kinds of sensors have been used: range finder devices (i.e. lasers and sonars) and vision sensors (i.e. perspective and omnidirectional cameras). The range finders are used to perform scans of the fix obstacles around the robot and the localization is calculates matching those scans with a metric map of the environment [1, 11]. The vision sensors are used to recognize characteristic landmarks subsequently matched within a map [3, 9] or to find the reference image most similar to the image currently grabbed by the robot (without a map) [13, 7, 8]. In our approach we use an omnidirectional vision system as sensor to emulate and enhance the behaviour of range-finder devices. In this work MCL (Monte-Carlo localization) was implemented based on the approach proposed in [1, 11]. We adapted that approach to take into account the new information given by this sensor. Experiments are made in a typical RoboCup environment (a 8x4 m soccer field), characterized by the lack of fix obstacles that could act as reference for a range finder sensor (as it was with the walls surrounding the field until RoboCup 2001). In this situation it is extremely hard to perform robust localization using conventional range finder devices.

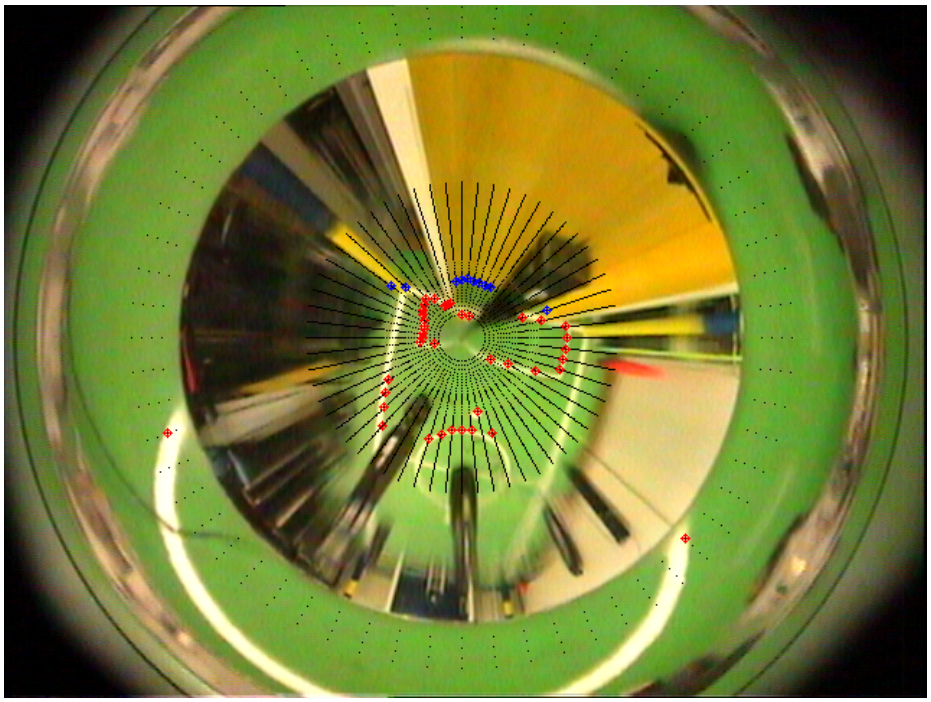


Fig. 1. The scanning algorithm at work: green-white chromatic transitions are highlighted with red crosses, green-yellow transitions with blue crosses, black pixels represent the sample points used for the scan that is performed in a discrete set of distances. No blue-green transitions are detected: robot is far away from the blue goal. Notice the crosses in the outer part of the mirror: this part is used for low distance measures.

2 Omnidirectional Vision as an Enhanced Range Finder

RoboCup is a strongly color coded environment: every object has an unique color associated to it. Usually, the image is color segmented before any image processing. In our system only the pixels along the rays depicted in Fig.1 are segmented into the 8 RoboCup colors [3] plus a further class that include all colors not included in the former classes (called *unknown color*). A look-up table is built to obtain a real time color segmentation. The image processing software scan the image for what we called *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. In fact, lines are white, goals and corner posts are blue or yellow and the play-ground is a green carpet. To detect a colour

[3] In RoboCup environment the ball is red, the lines are white, one goal is blue and the other is yellow, the robots are black, the robots' marker are cyan and magenta

transition is more robust with respect to colour calibration than to identify the colour of every single pixel, as reported in [10].

We measure the distance of the *nearest chromatic transitions of interest* along 60 rays as shown in Fig.1. This enable our "range finder" to scan a 360 degree field of view. Our omnidirectional vision sensor is composed by a camera pointed to a multi-part mirror with a custom profile [6]. The inner part of the mirror is used to measure objects farther than 1 m away for the robot, while the outer part is used to measure objects closer than 1 m from the robot. We first scan for color transition close to the robot body in the outer mirror part, and then we scan the inner part of the image up to some maximum distance.

The distances to the nearest color transition are stored in three vectors (in the following called "*scans*"), one for each color transition. 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 that mean 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 (we called this *ray discrimination*). The scanning is not performed in a continuous way along the ray but sampling the image on a discrete subsets of image pixels corresponding to a sampling step of 4 cm in the real world.

3 Monte-Carlo Localization

The Monte-Carlo localization (MCL) is a well-known probabilistic method, in which the current location of the robot is modelled as a posterior distribution (Eq.1) conditioned on the sensor data and represented by a set of weighted particles. Each particle is an hypothesis of the robot pose, and it is weighted according to the posteriors. The posterior probability distribution of the robot pose is called also the robot belief. The belief about the robot position is updated every time the robot makes a new measurement (i.e. it grabs a new image or a new odometry measure is available). It can be described by:

$$Bel(l_t) = \alpha p(o_t|l_t) \int p(l_t|l_{t-1}, a_{t-1}) Bel(l_{t-1}) dl_{t-1} \tag{1}$$

where $l_t = (x_t, y_t, \theta_t)$ is the robot pose at time t and a_t and o_t are respectively the sensor and the odometry readings at the time t. To calculate Eq. 1, it is necessary the knowledge of two conditional densities, called *motion model* (Sec. 3.1) and *sensor model* (Sec. 3.2). The motion model expresses the probability the robot moved to a certain position given the odometry measures (*kinematics*). The sensor model describes the probability of having a sensor measurement in a certain pose. The motion model and the sensor model depend respectively on the particular robot platform and on the particular sensor. The localization method is performed in 3 steps: (1) All particles are moved according to the

motion model of the last kinematics measure; (2) The weights of the particles are determined according to the sensor model for the current sensor reading; (3) A re-sampling step is performed: high probability particles are replicated, low probability ones are discarded. The process repeats from the beginning. For more details please refer to [1, 11].

3.1 Motion model

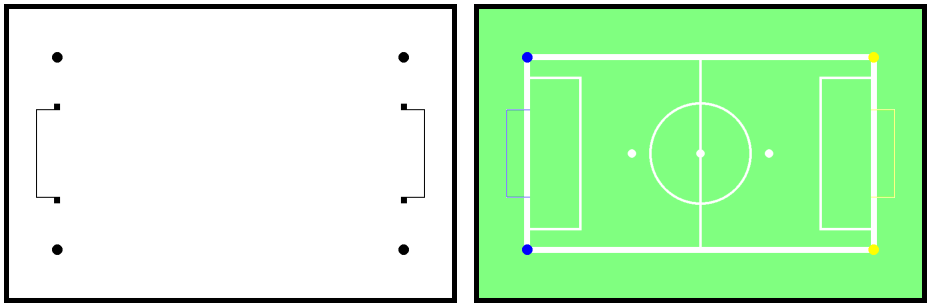
The motion model $p(l_t|l_{t-1}, a_{t-1})$ is a probabilistic representation of the robot kinematics, which describes a posterior density over possible successive robot poses. We implemented the MCL system on an holonomic robot, called Barney. The peculiarity of this robot is that it can move in any direction without the need of a previous rotation. Movement between two poses $l_{t-1} = (x_{t-1}, y_{t-1}, \theta_{t-1})$ and $l_t = (x_t, y_t, \theta_t)$ can so be described with (α_u, T, θ_f), where α_u is the difference of heading between the two poses, T is the translation and θ_f is the motion direction. Updating the robot position according only to the kinematics does not take into account errors given by odometry inaccuracy and possible collisions of the robot with other obstacles. Therefore, a random noise term is added to the values given by the last odometry reading. Noise is modelled with Gaussian zero centered random variables $(\Delta_\alpha, \Delta_{\mathrm{T}}, \Delta_{rr}, \Delta_{r\mathrm{T}})$. They depend on both the amount of translation and of rotation. So, the motion model can be written as:

$$\begin{aligned} \alpha'_u &= \alpha_u + \Delta_\alpha(\alpha_u) \quad ; \\ T' &= T + \Delta_{\mathrm{T}}(T) \quad ; \\ \theta' &= \theta + \Delta_{rr}(\theta) + \Delta_{r\mathrm{T}}(T) \quad . \end{aligned}$$

3.2 Sensor model

The sensor model $p(o_t|l_t)$ describes the likelihood to obtain a certain sensor reading given a robot pose. As introduced in Sec. 3, the sensor model is used to compute the weights of the particles. For each particle j, located in the pose l_t^j, the associated weight is proportional to $p(o_t|l_t^j)$ (i.e. to the likelihood of obtaining the sensor reading o_t when the robot has pose l_t^j). To calculate $p(o_t|l_t^j)$, we need to know the "*expected scan*" $o(l_t)$. The expected scan is the scan an ideal noise-free sensor would measure in that pose, if in the environment there are no obstacles. Given l the robot pose, the expected scan $o(l)$ for some color transition is composed by a set of expected distances, one for each α_i, that are the angles relative to the robot of an individual sensor ray (Fig. 3): $o(l) = \{g(l, i)|0 \leq i < N\_RAYS\}$. We can compute the expected distances $g(l, i)$ for an ideal noise-free sensor using ray tracing technique considering both metric maps in Fig. 2. The likelihood $p(o_t|l_t)$ can be calculated as $p(o_t|l_t) = p(o_t|o(l_t))$. In other words, the probability $p(o_t|o(l_t))$ models the noise in the scan by the expected scan [1, 11].

When using a sonar or a laser, like in [1, 11], the expected scan is computed from a metric map of the environment. The expected scan is obtained simulating



(a) (b)

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

the reflections of the sonar or laser beams against the walls and the fix obstacles. In the RoboCup Middle-Size field, a similar approach was used, very effectively, by the CS Freiburgh Team [12], until RoboCup 2001. However, when in 2002 the walls surrounding the field were removed, the reliability of this approach was impaired by the lack of fix features detectable by a range-finder sensor. In Fig. 2(a), are presented the fix obstacles that a range-finder sensor could detect. In the Middle-Size field with the 2003 layout, the only detectable objects are the two goals and the four corner-posts. With the new sensor we propose, we can detect not only the fix objects in the field shown in Fig. 2(a), but also all color transitions 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 compute the expected scan finding with a ray-tracing approach the *nearest chromatic transition of interest* for every pose, as depicted in Fig. 3. Moreover, we use the information about the fix obstacles extracted from the map of Fig. 2(a) to improve the scanning process, e.g. if we find a yellow pixel, this is a goal or a corner-post, so it is not worth looking farther for a white line and so we stop the scanning process along this ray.

Another difference with respect to the usual range-finders is that we do not have just one scan of the environment. We have three scans for every pose of the robot: one for every chromatic transition of interest (green-white, green-blue and green-yellow, see Sec. 2). Moreover, we can filter out some rays when a fake transition is detected (i.e. a chromatic transition that we are not looking for, see Sec. 2). In Fig. 3, two examples in which are compared the expected scans (top) and the real sensor scans (bottom) is presented. In the middle is the image grabbed by the robot. On the left is depicted the scan looking for the *green-white* chromatic transition of interest, on the right the scan looking for the *green-yellow* chromatic transition of interest. Due to the image noise, it might happen that a

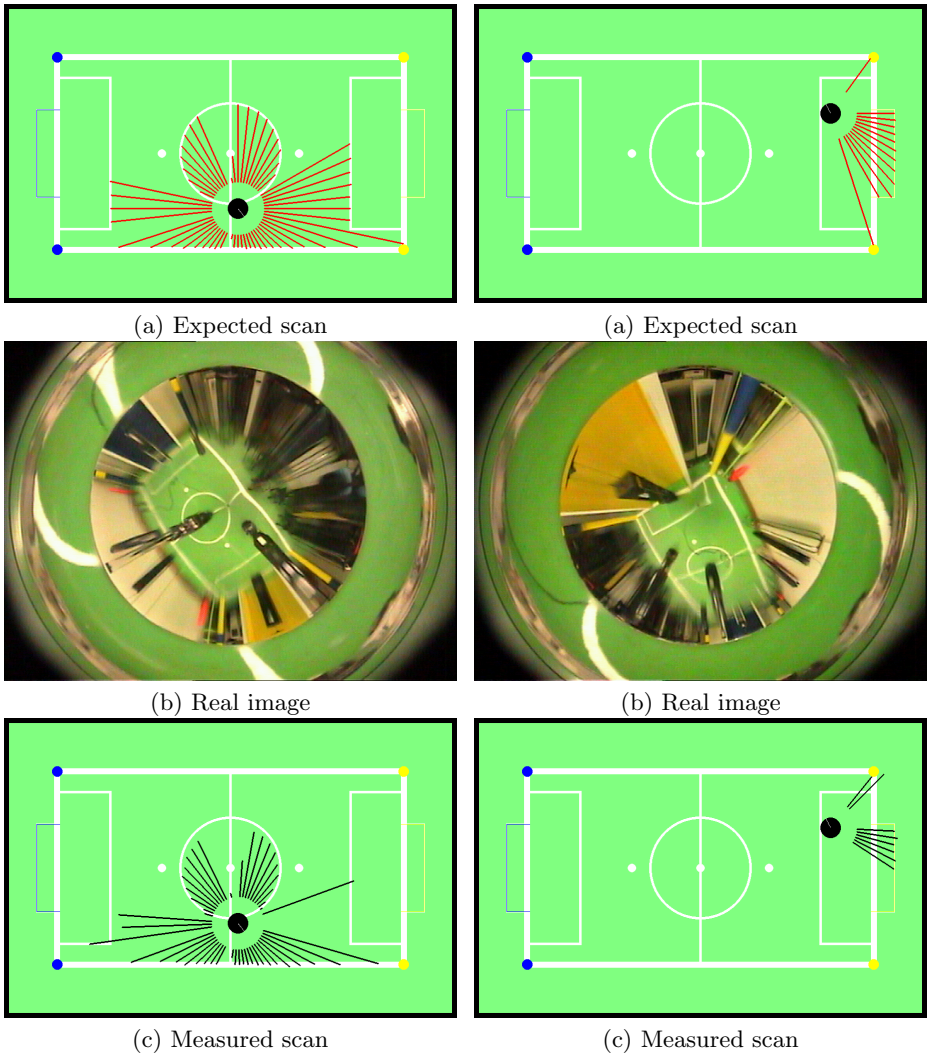


Fig. 3. Two examples of expected and measured scans. The one on the left for the green-white transition, the other on the right for the green-yellow transition. Given a pose, in (a) is represented the expected scan for an ideal noise-free sensor in a free environment. In (b) is shown the frame grabbed by the robot in that pose, in (c) is represented the corresponding measured scan.

color transition is not detected or is detected at the wrong distance or is falsely detected (as shown in Fig. 3). So, we need to create a model of the sensor's noise.

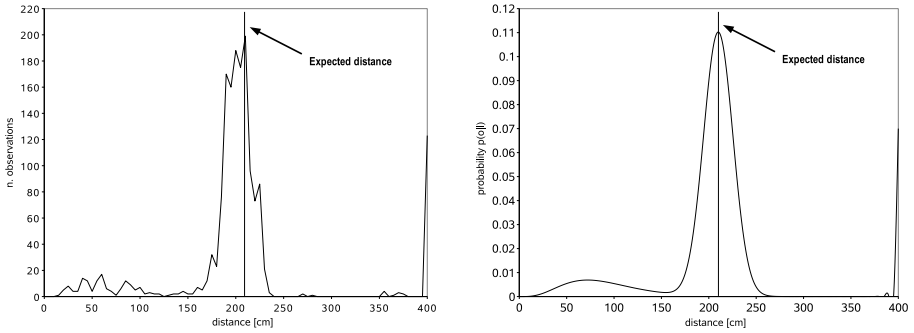
Sensor noise To compute $p(o|o(l))$, the first step is to model the sensor noise. We implemented a three steps process. First, we modelled the probability a single ray of the scan correctly detects the chromatic transition of interest. Second, we take into account all rays and we calculate a single probability value that the measured scan match the expected scan to the corresponding chromatic transitions of interest. Third, the three probability values of the three measured scans are combined to obtain a single value.

Let us describe these steps in more details. The scan performed by the sensor is composed by a set of distances, one for each α_i: $o = \{o_i | 0 \leq i < N\_RAYS\}$. To compute $p(o_i|l)$, i.e. the probability to obtain for a single ray a distance o_i given the pose l, we can consider directly the single expected distance $g(l, i)$, so we can write $p(o_i|l) = p(o_i|g(l, i))$. To calculate $p(o_i|l)$, we collected a large number of omnidirectional images in different known poses in the field (in total about 2.000 images). Then, with the scan algorithm we measure 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 the measured probability density with the mixture of three probability density of Eq. 2. The numerical values of the parameters in Eq. 2 are calculated with a modified EM algorithm iteratively run on the 2000 images [2]. The resulting mixture, for the green-white transition, is plotted in Fig. 4(b). The three terms in Eq. 2 are respectively: an Erlang probability density, a Gaussian probability density and a 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. The Gaussian density 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. 2.

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

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.

Once the $p(o_i|l)$ is computed, it is possible to compute the probability of the whole scan given a pose l multiplying all the $p(o_i|l)$, Eq. 3. 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). This is the process called *ray discrimination*, see Sec. 2. The detection of occluding obstacles along the rays of a scan is very frequent in the densely crowded environment of the Middle-Size RoboCup field. This rays discrimination allow us to avoid to use other techniques, e.g. *distance filters* [5], that can affect negatively the computational performance of the system.



(a) (b)

Fig. 4. In (a) the distribution of the measured distances for an expected known distance. There is a peak for the expected distance. The measures before the expected one are due to image noise. The high number of maximum distance measures means no chromatic transition was detected. In (b) the density $p(o|l)$ that represent our sensor model computed using EM-algorithm, matematically described by Eq.2.

$$p(o|l) = \prod_{\{i|o_i \neq \phi\}} p(o_i|l) = \prod_{\{i|o_i \neq \phi\}} p(o_i|g(l,i)) \tag{3}$$

3.3 Weights Calculation

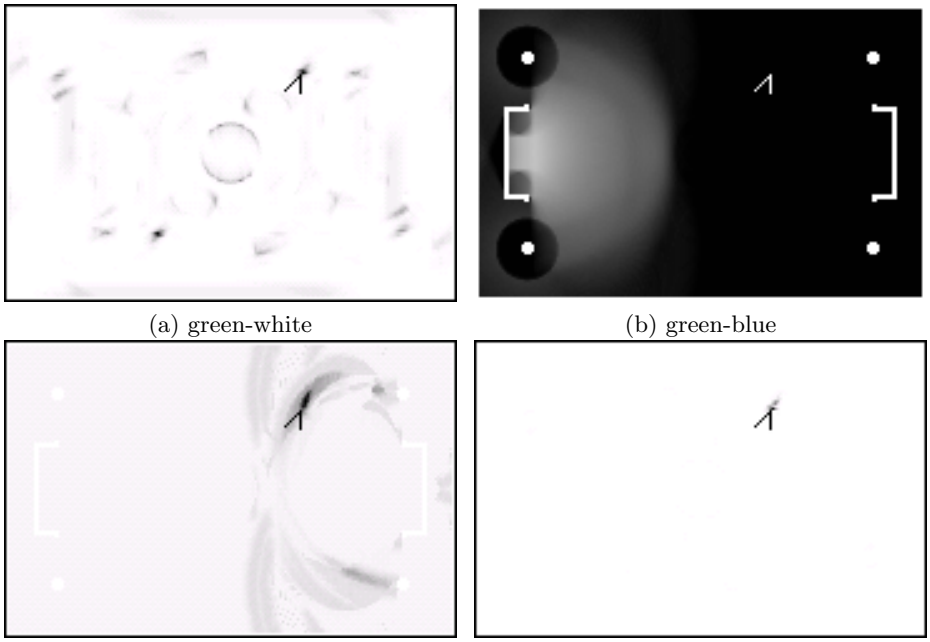
Returning to Monte Carlo Localization, we are now able to compute, the weight $w^{(j)}$ associated to each particles j. We first calculate the quantity $\bar{w}^{(j)} = p(o_|l_j)$ using (3). Subsequently, all $\bar{w}^{(j)}$ are normalized such that $\sum_j \tilde{w}^{(j)} = 1$

$$\tilde{w} = \frac{\bar{w}^{(j)}}{\sum_j \bar{w}^{(j)}} \tag{4}$$

Our system scans the acquired image for the three chromatic transitions of interest. This ensures three scans for every frame, so three weight values are associated to every particles. To obtain a single weight value, we compute the product of the three weights (Eq. 5), and re-normalize all weights with (4) again.

$$w^{(j)} = \prod_{k=1}^{N} \tilde{w}_k^{(j)} \tag{5}$$

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 with 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 by the scan looking for the



(a) green-white (b) green-blue

(c) green-yellow (d) overall

Fig. 5. Probability distributions $p(o_t|l_t)$ for all possible positions $l = (x, y, \theta)$ 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 pose. In (a) is represented the probability given the scan for transition white, in (b) for transition blue, in (c) for transition yellow, in (d) the three are combined.

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 by 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 by the scan looking for the green-yellow transition. One can notice there is an approximate symmetry around at the yellow goal. All these contributions are combined with Eq.5 to calculate the overall weights and depicted in Fig. 5 (d). Here, the weights with higher values are clustered only around the actual position of the robot.

In order to improve the performance of the system, the distances in the environment are discretized in a grid of 5x5 cm cells, in a way similar to [5]. 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. 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.

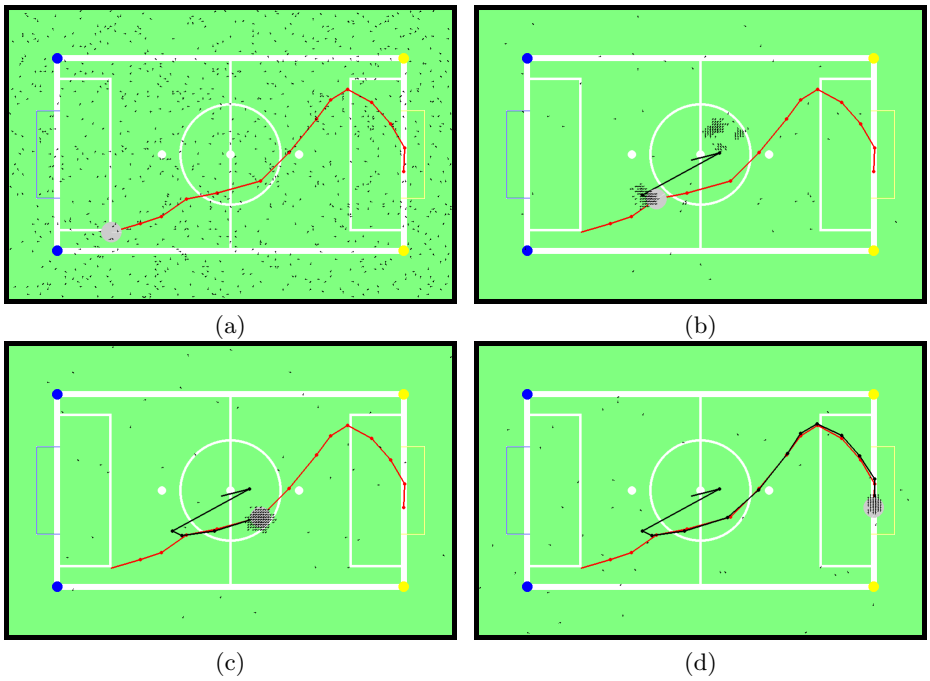
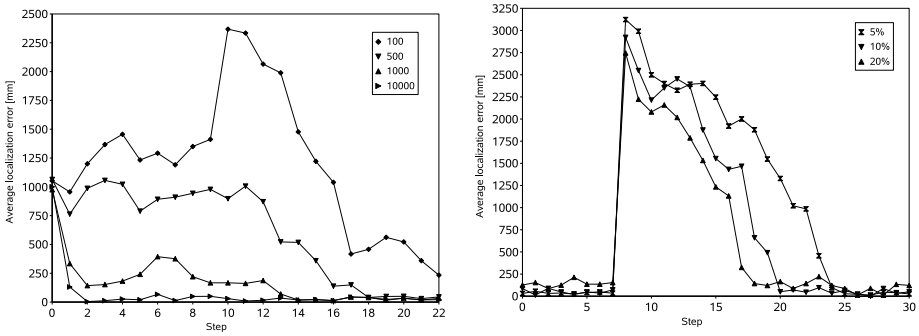


Fig. 6. A sequence of global localization using 1000 particles: the gray circle represents actual robot pose, the red line represents ground-truth path, the black line represents the estimated path of the robot, the black points represent the particles. In (a) particles are uniformly distributed (no knowledge is available on robot position), in (b), after moving 2 meters away and grabbing 4 images and getting 4 odometry readings, the particles are condensed around three possible poses. In (c), after 4 meters, 6 images and 6 odometry readings, uncertainty is solved and particles are condensed around the actual pose of the robot. In (d) after 14 steps: the position of the robot is well tracked. The particles distributed in the environment are the particles scattered to solve the kidnapped robot problem.

4 Experiments

We evaluated our approach on an holonomic custom-built platform, in a 8x4 m soccer field. The robot was equipped with the omnidirectional sensor described in Sec. 2. We tested the system on five different paths (an example path is shown Fig. 6). For each path we collected a sequence of omnidirectional images with the ground truth positions where those images were grabbed and with the odometry

readings between two consecutive positions. In order to take into account the odometry errors, robot movements were performed by remote robot control. We tested our algorithms using different amount of samples calculating the mean localisation error for the three fundamental localization problems: (1) global localization (the robot must be localized without any a priori knowledge on the actual position of the robot, i.e. Fig. 6 (a)(b)), (2) position tracking (a well localized robot must maintain the localization, i.e. Fig. 6 (c)(d)) and (3) kidnapped robot. The kidnapped robot is the problem in which a well-localized robot is moved to some other pose without any odometry information: this problem can frequently occur in an high populated environment like RoboCup, where often robots push each other attempting to win the ball.



(a) (b)

Fig. 7. The plots compares the performance of the system varying the number of the samples used in MCL. In (a) global localization errors for a fixed path with different amount of samples, in (b) re-localization after kidnapped robot problem with different rate of uniformly distributed particles. Notice that with 20% the re-localization is faster but the average position tracking error is higher

In Fig. 7(a) is shown the error for a global localization sequence using 100, 500, 1000, 10000 samples in the same reference path. The reactivity and the accuracy of the localisation system increase with the number of samples, but a large number of samples like 10000 increases dramatically the computational load. A number of 1000 particles is perfectly compatible with real-time requirements and assures a robust and accurate localisation. In Fig. 7(b) is shown the error for a kidnapped robot episode using 1000 samples and different rate of samples uniformly distributed in the environment [4]. With a higher rate of samples scattered in the environment the re-localization is faster (there are more possibility that the the samples are distributed close to the robot position), but the average error is higher due to the lower number of sample clustered closed to the robot pose during position tracking.

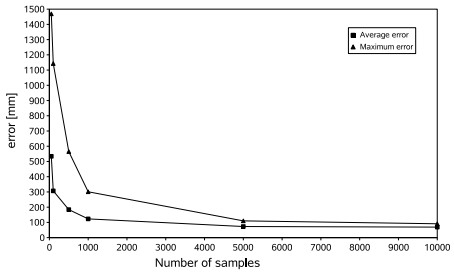


Fig. 8. Statistical evaluation of our system in the position tracking problem for all our reference paths. Accuracy (average error end maximum error) is represented for different amount of samples (50,100,500,1000,5000,10000)

Finally, we tested our approach in the conventional situation of the positions tracking: in Fig. 8 is shown the average and the maximum error for all our reference paths using different amount of samples. Like in the global position problem, with 1000 samples is possible to achieve good accuracy and an acceptable maximum error.

5 Conclusions

This paper presents a novel approach for vision-based Monte Carlo localization. Experiments in the Middle-Size RoboCup domain were presented. An omnidirectional vision sensor mounted on a holonomic robot platform was used in an innovative way to find the distance from the robot of the nearest chromatic transitions of interest. This approach mimics and enhances the way conventional range-finder sensors, like lasers and sonars, find the distance of the nearest objects. The proposed sensor enables to extract more features from the environment thanks to the capability to distinguish different chromatic transitions of interest. The well-known Monte Carlo localization technique was adapted to the characteristics of the sensor. The EM algorithm was used to extract the parameters of the sensor model from experimental data. We presented experiments in a actual Middle-Size RoboCup field to prove the robustness and the accuracy of our technique. We are porting the system to other environment than RoboCup. Depending on the environment, different chromatic transitions of interest can be identified. The system is designed to automatically recalculate the expected scans given the metric and chromatic maps of the new environment.

References

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

2. A. P. Dempster, N. M. Laird, and 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.
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, F. Dellaert, and S. Thrun. Monte carlo localization: Efficient position estimation for mobile robots. In *Proc. of the National Conferenceon Artificial Intelligence*, 1999.
5. D. Fox, W. Burgard, and S. Thrun. Markov localization for mobile robots in dynamic environments. *Journal of Artificial Intelligence Research*, 11, 1999.
6. E. Menegatti, F. Nori, E. Pagello, C. Pellizzari, and D. Spagnoli. Designing an omnidirectional vision system for a goalkeeper robot. In A. Birk, S. Coradeschi, and S. Tadokoro, editors, *RoboCup-2001: Robot Soccer World Cup V.*, pages 78–87. Springer, 2002.
7. E. Menegatti, M. Zoccarato, E. Pagello, and H. Ishiguro. Hierarchical image-based localisation for mobile robots with monte-carlo localisation. In *Proc. of European Conference on Mobile Robots (ECMR'03)*, pages 13–20, September 2003.
8. E. Menegatti, M. Zoccarato, E. Pagello, and H. Ishiguro. Image-based monte-carlo localisation with omnidirectional images. In *Robotics and Autonomous Systems, Elsevier*, page (to appear), 2004.
9. T. Röfer and M. Jüngel. Vision-based fast and reactive monte-carlo localization. In *IEEE International Conference on Robotics and Automation*, 2003.
10. T. Rofer and M. Jungel. Vision-based fast and reactive monte-carlo localization. In *Proc. of International Conference on Robotics and Automation (ICRA-2003)*, 2003.
11. S. Thrun, D. Fox, W. Burgard, and F. Dellaert. Robust monte carlo localization for mobile robots. *Artificial Intelligence*, 128(1-2):99–141, 2000.
12. T. Weigel, J.-S. Gutmann, M. Dietl, A. Kleiner, , and B. Nebel. Cs freiburg: Coordinating robots for successful soccer playing. *IEEE Transactions on Robotics and Automation (T. Arai, E. Pagello, L. Parker, (Eds.))*, 18(5):685–699, 2002.
13. 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.