1 2

Omnidirectional vision scan matching for robot localization in dynamic environments

E. Menegatti, *Member, IEEE,* , A. Pretto, A. Scarpa and E. Pagello, *Member, IEEE,*

***Abstract*— The localization problem for an autonomous robot moving in a known environment is a well studied problem which has seen many elegant solutions. Robot localization in a dynamic environment populated by several moving obstacles, however, is still a challenge for research. In this paper, we use an omnidirectional camera mounted on a mobile robot to perform a sort of scan matching. The omnidirectional vision system finds the distances of the closest color transitions in the environment, mimicking the way laser rangefinders detect the closest obstacles. The similarity of our sensor with classical rangefinders allows the use of practically unmodified Monte Carlo algorithms, with the additional advantage of being able to easily detect occlusions caused by moving obstacles. The proposed system was initially implemented in the RoboCup Middle-Size domain, but the experiments we present in this paper prove it to be valid in a general indoor environment with natural color transitions. We present localization experiments both in the RoboCup environment and in an unmodified office environment. In addition, we assessed the robustness of the system to sensor occlusions caused by other moving robots. The localization system runs in real-time on low-cost hardware.**

***Index Terms*— mobile robot localization, omnidirectional vision, scan matching, Monte-Carlo Localization**

I. Introduction

LOCALIZATION is the fundamental problem of estimating the pose of the robot inside the environment. Some of the most successful implementations of robust localization systems are based on the Monte Carlo Localization approach [5], [23]. The Monte Carlo Localization approach has been implemented on robots fitted either with rangefinder sensors or with vision sensors. Lately, vision sensors have been preferred over rangefinders, because they are cheaper and provide richer information about the environment. Moreover, they are passive sensors, so they do not interfere with other sensors and do not pose safety concerns in populated environments.

In this work, we consider the problem of Monte Carlo Localization using an omnidirectional camera. The vision system has been designed to extract the distances of the closest color transitions of interest existing in the environment. Our system uses an omnidirectional camera to emulate and enhance the behavior of rangefinder sensors. This results in a scan of the current location similar to the one obtained with a laser rangefinder, enabling the use of Monte Carlo algorithms only slightly modified to account for this type of sensors. The most significant advantages with respect to classical rangefinders are: (i) a conventional rangefinder device senses the vertical obstacles in the environment, while our system is sensitive to the chromatic transitions in the environment, thus gathering richer information, and (ii) our system can reject measurements if an occlusion is detected. Combining the omnidirectional vision scans with the Monte Carlo algorithms provides a localization system robust to occlusions and to localization failures, capable of exploiting as localization clues the natural color transitions existing in the environment. The Middle-Size RoboCup field is a highly suitable testbed for studying the localization problem in a highly dynamic and densely populated environment. In a dynamic multi-agent world, precise localization is necessary to effectively perform high level coordination behaviors. At the same time, the presence of other robots makes localization 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. Moreover, if, like in RoboCup Middle-Size, collisions among robots are frequent, the localization system must be able to recover from errors after collisions.

In this paper, we explicitly discuss the robustness of our system with respect to the classical problems of global localization, position tracking, and robot kidnapping [17]. We also provide a detailed discussion of the robustness against sensor occlusion when the robot moves in a densely populated environment [18]. In addition, we present experimental evidence that the system developed is not limited to the RoboCup domain, but works in a generic unmodified office-like environment. The only requirements for our system are: (i) an environment with meaningful color transitions, (ii) a geometric map of that environment, and (iii) inclusion of the color transitions in the map.

The following section discusses previous research related to this topic. Section III describes how we process the omnidirectional image to obtain range information. Section IV summarizes the well-known Monte Carlo Localization algorithm and discusses the motion model and sensor model used in the experiments, as well as the modifications to the classical Monte Carlo Localization to adapt it to our sensor. In Section V, we present the experiments performed with the robot in a RoboCup Middle-Size field and in the corridors of our department. A detailed analysis of the performance and robustness of the localization system is presented, paying particular attention to occlusions caused by other robots. Finally, in Section VI, conclusions are drawn.

II. Related Work

The seminal works on Monte Carlo Localization for mobile robots used rangefinders as main sensors. The rangefinders

The authors are with the Intelligent Autonomous Systems Laboratory Department of Information Engineering, The University of Padua, Italy e-mail: emg@dei.unipd.it

E. Pagello is also with Institute ISIB of CNR Padua, Italy

were used to perform scans of static obstacles around the robot and the localization is calculated by matching those scans with a metric map of the environment [5], [24]. However, in dynamic environments, the static features that are detectable are often not enough for a robust localization (as illustrated in Fig. 1), or they may be occluded by moving obstacles. One possibility is to design algorithms able to filter out the moving obstacles in the range scans, leaving only the static obstacles that can be used as landmarks. This is exemplified by *distance filters* [9], but it should be noted that usually these algorithms are computationally intensive. Another possibility is to use a sensor, such as a color camera, that provides richer information and consequently a more reliable *"scan matching"*. In the present work we adopted the latter approach.

Usually, Monte Carlo Localization systems with vision sensors use cameras to recognize characteristic landmarks subsequently matched within a map [7], [21], or to find the reference image most similar to the image currently grabbed by the robot [11], [19], [20], [28], [29]. However, when the robot has to match the current reading with a previous reading, moving obstacles like people or other robots can impair the localization process. Several solutions have been adopted. One possibility is to look at features that cannot be occluded by moving obstacles, like the ceiling in a museum hall [4]. However these features are not always available or they do not carry enough information.

Our sensor is able to detect occlusions as non-expected color transitions, so it can use only a subset of reliable distances in the scan, obtaining a more precise localization. Color transitions are usually available and yield rich information about the environment structure (e.g., change of carpet color, doors with a color different from the walls, etc).

In the RoboCup Middle-Size competitions, an approach based only on laser rangefinders was used, very effectively, by the CS Freiburg Team [27]. They extracted the lines of the walls from the laser scans and matched them against a model of the field of play. However, when in 2002 the walls surrounding the field were removed, the reliability of this approach was impaired by the lack of static features detectable by a rangefinder sensor.

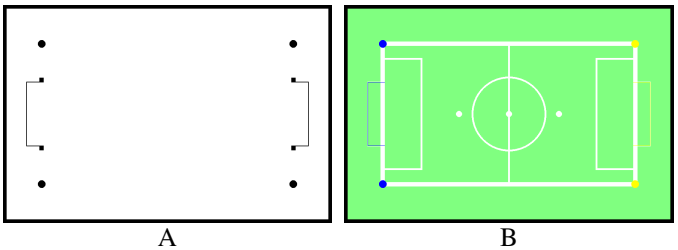


Fig. 1. The metric maps used for the computation of the expected scans: in A are represented the static obstacles (they are too sparse for an effective localization), in B are represented all the chromatic transitions of interest in the environment.

The static objects detectable by a rangefinder in the Middle-Size field (with the layout used since 2003) are presented in Fig. 1A. The only detectable objects are the two goals and the four corner-posts. With a vision system sensitive to color transitions, one can detect not only these static objects, but also all existing color transitions (Fig. 1B). Schulenburg *et al.* combined a laser rangefinder and an omnidirectional camera to extend CS Freiburg's approach by detecting lines using both sensors [22]. However, the integration of laser data with vision data does not significantly improve the localization with respect to vision data alone (due to the shortage of laser detectable features). Moreover, the image processing algorithms used to extract the field lines are computationally demanding. Most of the approaches that use a camera as the only sensor extract geometric features (like lines and corners) from the images, performing template matching against a model of the environment [2]. Geometric features have also been determined by detecting color transitions in visual receptors placed along radial lines (as previously proposed [3]) and the environment is represented by a 3D CAD model [12], [25]. Roefer *et al*. located geometric features and bearings of landmarks by detecting color transitions along vertical lines [21]. In our approach, the chromatic transitions are not used to extract geometric features, but to perform scan matching against a 2D image file that is a geometric map of the environment. In fact, we use the raw range scans and this paper shows that a robust localization is achievable. Lenser and Veloso have developed a system which mimics the working of a sonar sensor using a monocular camera that detects obstacles (as color transitions) along previously determined lines in the image [14]. However, although the basic idea is similar, our aim is much broader: we want to mimic the working of a laser rangefinder with an omnidirectional camera (therefore with a 360° field of view) in order to be able, not only to avoid the obstacles as Lenser, but also to localize the robot with a Monte Carlo Localization software almost unaltered from the one proposed by Thrun *et al.* [24], in which they used laser rangefinders. Another difference with the work of Lenser and Veloso is that we do not color segment the whole image, but we just look for color transitions along the radial lines of Fig. 2, saving a considerable amount of computation.

III. An omnidirectional camera as a rangefinder

The main sensor of our robot is an omnidirectional camera. The camera is calibrated in order to be able to relate the distances measured in the image with the distances in the real world. Our "rangefinder" has a 360° field of view, much larger than that of conventional rangefinders. The omnidirectional camera is composed by a perspective camera pointed upward to a multi-part mirror with a custom profile, depicted in Fig. 3 [16]. This profile was designed to have good accuracy both for short and long range 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 [16]. The inner part of the mirror is used to measure objects farther

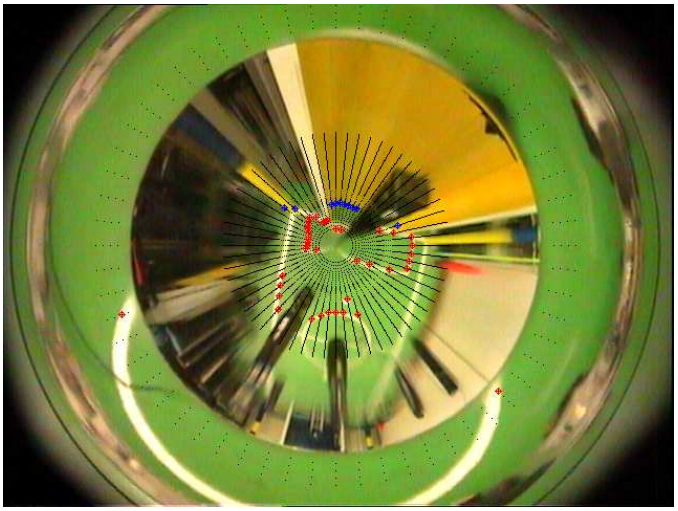


Fig. 2. Omnidirectional image with the detected chromatic transitions. Green-white chromatic transitions are highlighted with red crosses, green-yellow transitions with blue crosses, black pixels represent the receptor pixels used for the scan that is performed in a discrete set of distances. Notice the crosses in the outer part of the mirror: this part is used for low distance measures. If an unexpected color transition is detected (e.g., another robot is occluding the sensor, like the three black robots in the image) the scan is stopped and the value (*FAKE_RAY*) is stored in the distances vector.

than 1 m away from the robot, while the outer part is used to measure objects closer than 1 m from the robot, see Fig.2.

The omnidirectional image is scanned for what we called *chromatic transitions of interest*. In the RoboCup domain, we are interested in *green-white*, *green-blue* and *green-yellow* transitions. These transitions are related to the structure of the RoboCup field, where the playground is green, lines are white, and goals and corner posts are blue or yellow. In the office scenario, we are interested in the red-white and red-gray transitions due to the colors available in the environment. The image is scanned along radial lines 6° apart and with a sampling step corresponding to 4 cm in the world coordinate system, as shown in Fig. 2. We first scan for chromatic transitions of interest close to the robot's body (i.e., in the outer mirror part), we then scan the inner part of the image for transitions up to 4 m away from the robot's body.

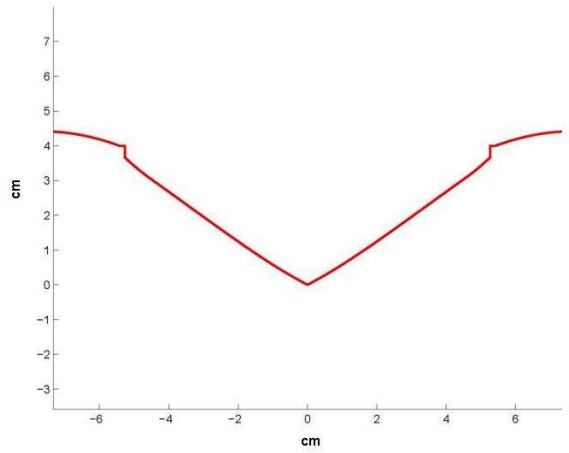


Fig. 3. The plot of the profile of the omnidirectional mirror mounted on the robot.

In RoboCup, a color quantization is usually performed on the image before any further image processing. Our system looks for the chromatic transitions of interest only along the receptors of the 60 rays depicted in Fig. 2. Therefore, we do not need to color quantize the whole image, but only some of the pixels lying along the 60 rays need to be classified into one of the 8 RoboCup colors [1] plus a further class that includes all colors not included in the former classes (called *unknown color*). At the setup stage, the RGB color space is quantized into the nine color classes. To achieve a real-time color quantization, a look-up table is stored in the main memory of the robot. The look-up table associates every possible RGB triple to one of the 9 color classes.

The distances to the nearest *chromatic transition of interest* are stored in three vectors[2], one for each color transition of interest. During the radial scan, we can distinguish three situations:

1) a chromatic transition of interest is found, then the real distance of that point is stored in the corresponding vector;
2) no transition of interest is detected, then a characteristic value called *INFINITY* is stored in the vector (this means no transition can be found along this ray);
3) a non-expected transition is found, then a characteristic value called *FAKE_RAY* is stored in the vector (this means something is occluding the vision sensor).

Moreover, we use the information about the static obstacles extracted from the map of Fig. 1A 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 we stop the scanning process along this ray).

The algorithm to find the nearest chromatic transitions of interest is presented in pseudo-code in Algorithm 1 (to simplify the comprehension, only the scan in the inner section of the multi-part omnidirectional mirror is presented).

The scan obtained from the image is compared with the scans extracted from the chromatic map of the environment, called *expected scans*. The map in Fig. 1B shows the chromatic characteristics of the environment. We use this map to compute the expected scan by ray-tracing, as will be explained in Section IV-B.

In summary, the advantages with respect to conventional rangefinders are: we have three scans for every pose of the robot (one for every chromatic transition of interest: green-white, green-blue and green-yellow) and we immediately know which rays of the scan should be discarded because of occluding objects (detected by non-expected chromatic transitions). The limitations of our sensor are: a smaller accuracy than laser rangefinders and the sensitivity to changes in the illumination, strong enough to alter the appearance of the colors in the environment.

[1] 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' markers are cyan and magenta

[2] The three vectors are called "*scans*" in the remainder of the paper

Algorithm 1 Omnidirectional vision as an enhanced rangefinder.
Function QUANT(x,y) returns quantized color of pixel x,y.
Function REAL_DIST(x,y) returns distance in real world of pixel x,y.
Pixel$(0,0)$ is located in the image center.

Ensure: $dist\_white[N\_RAY], dist\_blue[N\_RAY], dist\_yellow[N\_RAY]$
  for $i = 1 : N\_RAYS$ **do**
    $dist\_white[i] = dist\_blue[i] = dist\_yellow[i] = INFINITY$
    $search\_for\_white = true$
    $x = y = ray = 0$
    $lastColor = QUANT(x, y)$
    for $ray = 1 : MAX\_RAY$ **do**
      $x = ray * cos(\alpha_i)$, $y = ray * sin(\alpha_i)$
      $color = QUANT(x, y)$
      if $color$ isn't unknown or green **then**
        if $color$ is blue and $lastColor$ is green **then**
          $dist\_blue[i] = REAL\_DIST(x, y)$
          break
        else if $color$ is yellow and $lastColor$ is green **then**
          $dist\_yellow[i] = REAL\_DIST(x, y)$
          break
        else if $color$ is white **then**
          if $search\_for\_white$ **then**
            $search\_for\_white = false$
            if $lastColor$ is green **then**
              $dist\_white[i] = REAL\_DIST(x, y)$
            else
              $dist\_white[i] = FAKE\_RAY$
            end if
          end if
        else
          $dist\_blue[i] = dist\_yellow[i] = FAKE\_RAY$
          if $search\_for\_white$ **then**
            $dist\_white[i] = FAKE\_RAY$
          end if
          break
        end if
      end if
      $lastColor = color$
    end for
  end for

To manage the uncertainty in the measurements, we slightly modified the classical Monte Carlo Localization algorithm.

IV. Monte Carlo Localization

Monte Carlo Localization is a well-known probabilistic method, in which the current pose of the robot is modelled as a posterior distribution conditioned by the sensors' data (Eq. 1). The posterior probability distribution of the robot pose is also called the robot's **belief**. The belief about the robot's position is represented with a set of discrete points in the configuration space of the robot. These points are called **particles**. To update the belief over time, the particles are updated. Each particle is an hypothesis of the robot's pose, and it is weighted according to the posteriors. The belief about the robot's 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). This belief can be described by:

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

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, two conditional densities, called *motion model* and *sensor model* are needed. The *motion model* expresses the probability the robot moved to a certain position given the odometry measures (kinematics); see Section IV-A. The *sensor model* describes the probability of having a certain sensor measurement in a certain pose, see Section IV-B. The motion model and the sensor model depend, respectively, on the particular robot platform and on the particular sensor. The localization algorithm is composed by 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 resampling step is performed: high probability particles are replicated, low probability ones are discarded. The process repeats from the beginning.

The resampling step is performed with the Sampling Importance Resampling (SIR) algorithm [10] with the resampling technique of [13]. The final estimation on the pose of the robot is obtained simply averaging the poses of all particles. For more details, refer to [5], [24].

A. 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 Monte Carlo Localization system on a holonomic robot, called Barney. The peculiarity of this robot is that it can move in any direction without the need of a previous rotation. A 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 thus 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:

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

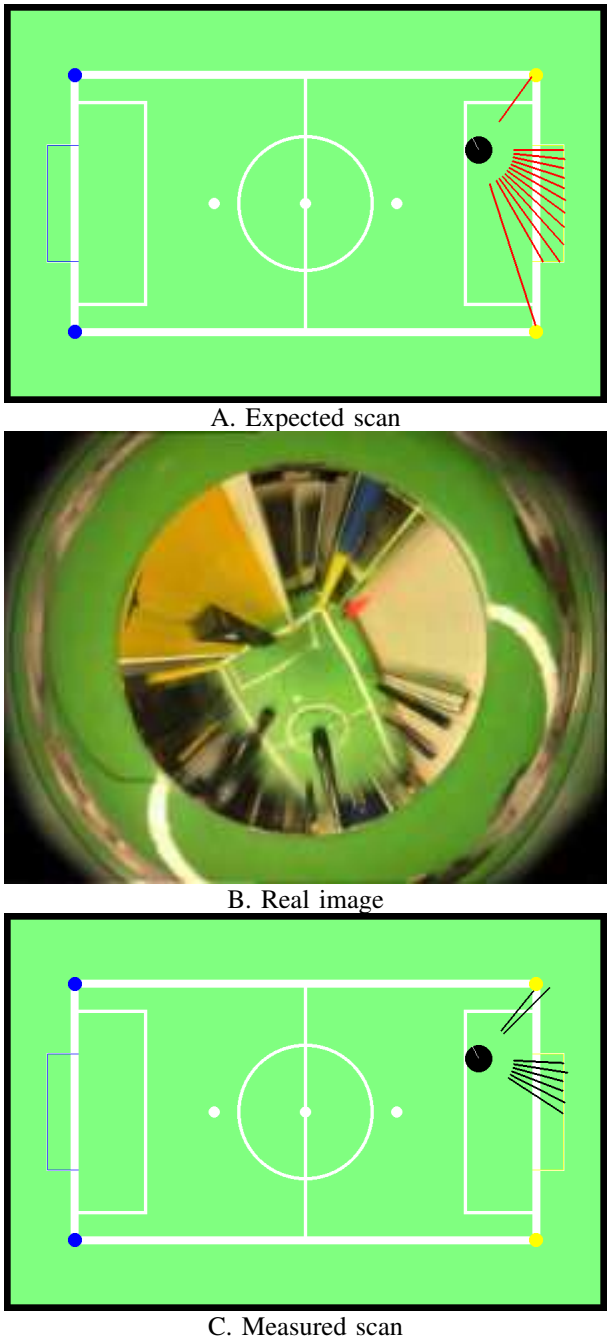
For our holonomic platform, we found that good values for the standard deviations of the added noise contributions are $\sigma_\alpha = 30°/360°$, $\sigma_T = 200mm/m$, $\sigma_{rr} = 30°/360°$, $\sigma_{rT} = 30°/m$. We experimentally verified that these values overestimate the actual errors and so provide good performance.

B. Sensor model

The sensor model $p(o_t|l_t)$ describes the likelihood to obtain a certain sensor reading given a robot pose. 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 a scan that an ideal noise-free sensor would measure in that pose, if there were no obstacles in the environment, Fig. 4A. Given l the robot pose, the expected scan $o(l)$ for one of the three chromatic transitions of interest is composed by a set of expected distances, one for each α_i, the rays of the scan (the black radial lines in Fig. 2): $o(l) = \{g(l,i)|0 \leq i < N\_RAYS\}$. We can compute the expected distances $g(l,i)$ for an ideal noise-free sensor in an empty environment, using a ray tracing technique. The basic idea is: (i) to reproduce the pose of the robot in the metric maps of Fig. 1; (ii) to trace the rays exiting from the robot until they encounter the first chromatic transition of interest; (iii) to store the length of these rays in the expected scan. 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 [5], [24].

Fig. 4 compares the expected scan A and the real sensor scan C; in B is the image grabbed by the robot. The scan is looking for the *green-yellow* chromatic transition of interest. As such only rays with a correct value in the distance vector are depicted in Fig. 4A, (i.e., the rays intercepting the yellow goal and the corner posts). Due to the noise in the image, it might happen that a color transition is not detected. Examples are the rays striking the lower part of the yellow goal and the ray striking the lower corner post (compare Fig. 4A and Fig. 4C); the color transition can also be detected at the wrong distance (like the fourth ray starting from the top in Fig. 4C), or be falsely detected (like the second ray in Fig. 4C). It might also happen that a color transition is not detected because of occlusion (e.g., in Fig. 4B the goalkeeper occludes part of the yellow goal), but we will discuss this in detail in Section V-B. As such, we need to create a model of the sensor's noise.

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 transition 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 combine the measurements of all rays. Ultimately, we need to combine the three probability values given by the three chromatic transitions of interest.



A. Expected scan

B. Real image

C. Measured scan

Fig. 4. An example of expected and measured scans 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 B is represented the corresponding measured scan. Only the rays with a correct measurement are shown. Rays with $INFINITY$ or $FAKE\_RAY$ values are not displayed.

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 create a statistical model of the distance measurement along a single ray of the scan, we collected a large number of omnidirectional images (about 2,000) in

different known poses in the field of play. For every image, we calculated the estimated distance of the chromatic transition of interest. The resulting measures are distributed along a different probability density, one for each chromatic transition 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. 5A. We described this density with the mixture of three probability densities of Eq. 2. The three terms in Eq. 2 are respectively: an Erlang probability density, a Gaussian probability density and a discrete density. The numerical values of the parameters in Eq. 2 are calculated with a modified EM algorithm iteratively run on the 2,000 images [6]. The resulting mixture, for the green-white transition, is plotted in Fig. 5B. 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 (Section III) 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. As such, the Erlang density will have a peak at the distance corresponding to the transition between the two mirror parts. 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 to miss the detection of the chromatic transition, obtaining an *INFINITY* value in the scan vector, as described in Section III.

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

The mixture coefficients are $\zeta_e, \zeta_g, \zeta_d$, and normalization implies $\zeta_e + \zeta_g + \zeta_d = 1$. A different density mixture was computed for each one of the three chromatic transitions.

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.

$$p(o|l) = \prod_i p(o_i|l) = \prod_i p(o_i|g(l,i)) \quad (3)$$

2) Sensor occlusion: To cope with unexpected measures due to occlusion of the sensor by moving objects in the environment (i.e., the other robots in the field or the ball), we filtered out all the rays in which the distance o_i equals the *FAKE_RAY* value, see Section III (the $FAKE\_RAY$ value is represented by ϕ in Eq. 4). We called this process **ray discrimination**. The detection of occluding obstacles along the rays of a scan is very frequent in a densely crowded environment like the Middle-Size RoboCup field. In conventional rangefinders there is no ray discrimination system, so all measured distances contribute to the computation of $p(o|l)$. If a large number of distances are affected by the presence of other agents around the robot, the localization process might fail. Our ray discrimination technique enables to compute the sensor model only with a subset of reliable distances, obtaining a faster and more reliable localization.

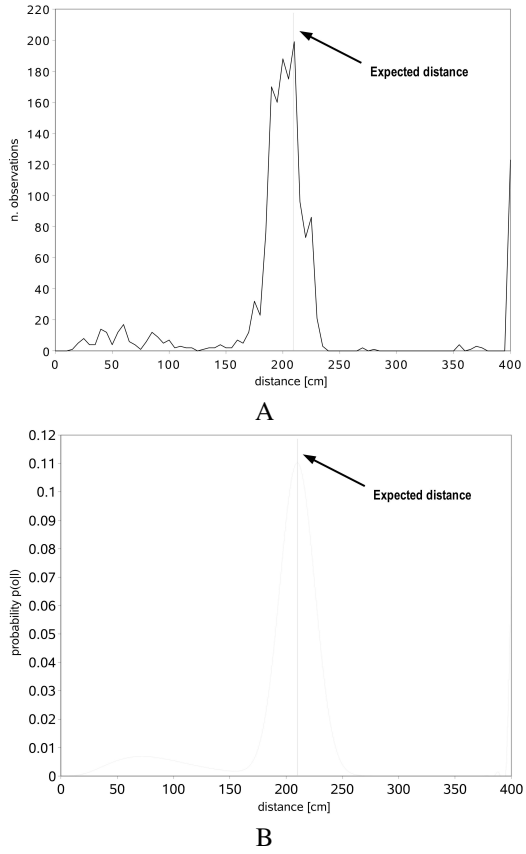


Fig. 5. In A the experimental distribution of measured distances for an expected known distance. The peak is at the expected distance. The measures before the expected one are due to the image noise. The last peak on the right of the plot means that due to image noise several time the chromatic transition has not been detected. In B the density $p(o|l)$ that represent our sensor model computed using EM-algorithm. The curve is the result of three contributions: (i) an Erlang variable with index n which depending on the geometry of the mirror, (ii)a Gaussian distribution centered at the expected distance and (iii) a discrete distribution representing the measurements resulting in the $INFINITY$ value.

$$p(o|l) = \prod_{\{i|o_i \neq \phi\}} p(o_i|l) = \prod_{\{i|o_i \neq \phi\}} p(o_i|g(l,i)) \quad (4)$$

From this equation, it follows that if the occlusion of the sensor increases, more and more rays will be discriminated and less information will be available for localization. Nevertheless, in our system all reliable information is exploited. As will be shown in Section V, the ray discrimination technique enables to correctly localize the robot even in situations of severe occlusion.

C. Weights Calculation

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

$$\tilde{w} = \frac{\bar{w}^{(j)}}{\sum_j \bar{w}^{(j)}} \quad (5)$$

Since our system scans the acquired image for the three chromatic transitions of interest, this provides three scans for every frame, so three weight values are associated to every particle. To obtain a single weight value, we compute the product of the three weights (Eq. 6), and re-normalize all weights with Eq. 5 again.

$$w^{(j)} = \prod_{k=1}^{N} \tilde{w}_k^{(j)} \tag{6}$$

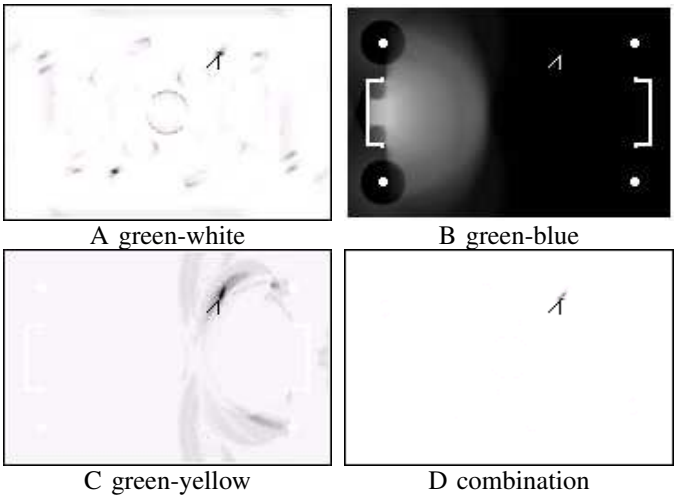


Fig. 6. Probability distributions $p(o_t|l_t)$ for all possible poses $l = (x, y, \theta)$ of the robot in the field, given the scans of a single image (the heading is not represented). Darker points correspond to a higher likelihood. The arrowhead represents the actual robot pose. IN A, B, and C are represented the probabilities given the scan for the green-white, green-blue , and green-yellow transition, respectively, in D is shown the combination of the three.

In Fig. 6, we give a pictorial visualization of the weights calculated by the three different scans of the three chromatic transitions of interest. The real pose of the robot is marked by the arrowhead. Higher weight values are depicted as darker points, lower weight values are depicted as lighter points. The weight contributions calculated by the scan looking for the green-white transition are represented in Fig. 6 A; due to the symmetry of the white lines in the field, two symmetric positions resulted to have high likelihood. The weight contributions calculated by the scan looking for the green-blue transition are depicted in Fig. 6 B; all positions far away from the blue goal have a high likelihood because no green-blue transition was found in the image scan. The weight contributions calculated by the scan looking for the green-yellow transition are represented in Fig. 6 C; there is an approximate symmetry around the yellow goal. All these contributions are combined with Eq. 6 to calculate the overall weights as depicted in Fig. 6 D, where indeed, the weights with higher values are clustered only around the actual position of the robot.

V. Experiments

The robot we used in the experiments is a holonomic custom-built platform, equipped with the omnidirectional sensor described in Section III. This section is divided in three parts. In the first, we evaluate the performance of the localization system depending on the number of particles used. In the second, the robustness of the system to sensor occlusion is evaluated. In the third, we present experiments in the corridors of our department to show that the proposed system can be applied in any environment in which stable color transitions can be identified.

In order to improve the time performance of the system, the distances in the RoboCup and in the office environment are divided in a grid of 5x5 cm cells; similar approaches have been successfully used previously [9]. 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 two look-up tables for every chromatic transition. Each look-up table takes about 13 Mb. In the RoboCup field we have 6 look-up tables (three chromatic transitions of interest) and in the office environment we have 4 look-up tables (two chromatic transitions of interest). In this way, the probability $p(o_i|l)$ can be quickly computed with two look-up operations, which 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. Localization in the RoboCup field of play

We tested the system on five different paths (an example path is shown in Fig. 7). 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. During the experiments, in order to take into account the odometric errors, the robot was moved using its own motors between the reference locations. This was done by sending position commands to the robot controller and not by moving the robot by hand. We tested our algorithms using different amounts of particles calculating the mean localization error for the three fundamental localization problems: (1) the global localization problem (the robot must be localized without any a priori knowledge on its actual position), (2) the position tracking problem (a well localized robot must maintain the localization) and (3) the kidnapped robot problem (a well-localized robot is moved to a different pose without any odometry information). In our experiments the kidnap was done by lifting the robot by hand and moving it about 3 m away. In every trial, the robot was moved to a different location; this is intended to simulate situations in which, for whatever reason, the robot is lost and must be able to recalculate its correct localization starting from a wrong belief. Moving the robot by 3 m takes into account also possible situations of incorrect localization generated by collisions with other robots (a problem that can frequently occur in a highly populated environment like the RoboCup games, where robots often push each other while trying to win the ball). We performed specific experiments about collisions, in which we pushed or blocked a moving robot, but the results are very similar to those of the kidnapping experiment; the only difference is that the localization error is usually smaller for a collision than for a kidnapping situation. To address the kidnapped robot problem, we adopted the classical technique to reserve a certain percentage of the particles to this scope and

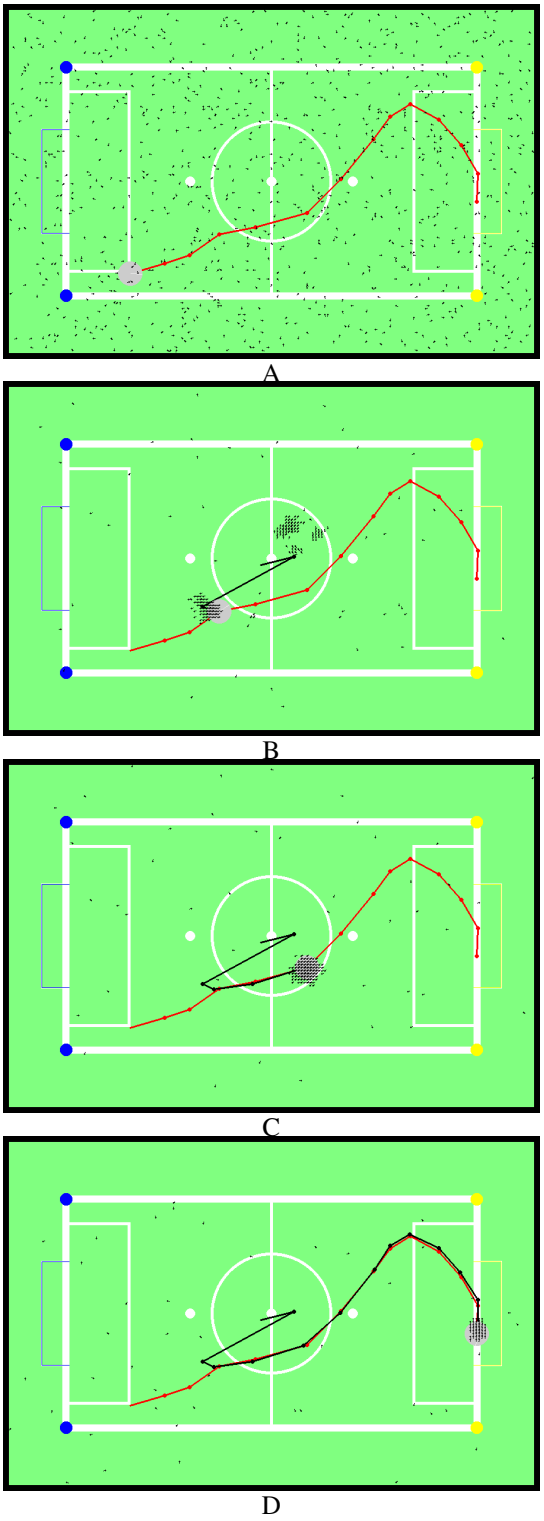


Fig. 7. A sequence of global localization and position tracking. The gray spot represents the actual robot pose, the red line represents the ground-truth path, the black line represents the estimated path of the robot, the black dots represent the particles (1000 particles are used). Note the heading of the particle is not displayed.

to randomly scatter them in the environment to act as seeds for a re-localization process in case of localization failure [8].

One of the five test paths is shown in Fig. 7. Initially, the particles are uniformly distributed (no knowledge is available on robot position), Fig. 7A. After the robot moved 2 m, having grabbed 4 images and obtained 4 odometry readings, the particles are condensed around three possible poses, Fig. 7B. After 4 m, 6 images and 6 odometry readings, uncertainty is solved and particles are condensed around the actual pose of the robot, Fig. 7C. After 14 steps, one can see that the position of the robot is well tracked along the ground-truth path (position tracking), as shown in Fig. 7D. The particles that are still dispersed in the environment are the particles scattered to solve the kidnapped robot problem.

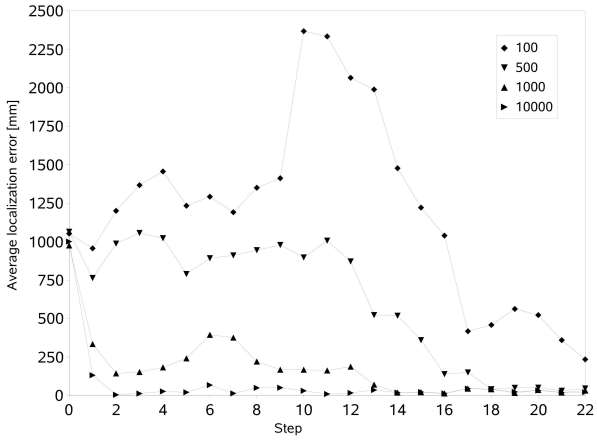


Fig. 8. The average error in the global localization problem for a specific path with different amounts of particles.

Both reactivity and the accuracy of the localization system increase with the number of particles, but the computational load is also increased. We tested the performance of the system with different numbers of particles. In Fig. 8 is shown the average localization error for global localization using 100, 500, 1,000, and 10,000 particles when the same path is repeated 100 times. A thousand particles is compatible with real-time requirements and assures a robust and accurate localization. This number is also a good value for the position tracking problem. In Fig. 9 is shown the average and the maximum localization error in the position tracking phase for all test paths using different amounts of particles. With 1,000

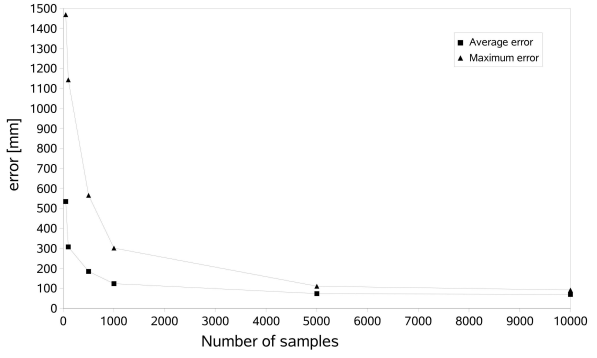


Fig. 9. The average error and maximum error in the position tracking problem over the five reference paths, calculated with different amounts of particles.

particles it is already possible to achieve a good accuracy, an acceptable average error (about 10 cm) and an acceptable maximum error (about 30 cm), without burdening the CPU of the robot.

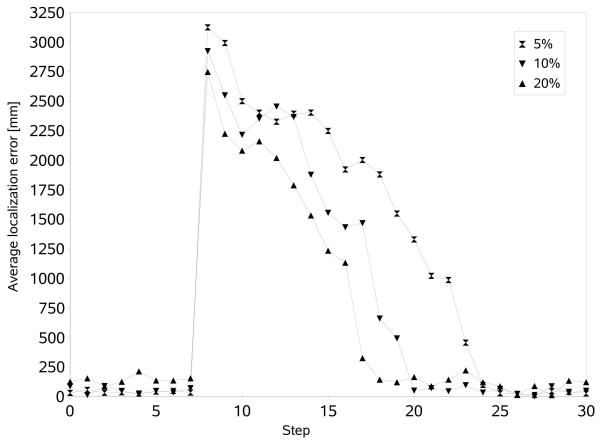


Fig. 10. The average error in the re-localization phase after kidnapping the robot, using 1,000 particles, and varying the rate of uniformly distributed particles.

In Fig. 10 is shown the error for a kidnapped robot episode using 1,000 particles and different rate of particles uniformly distributed in the environment. With a higher rate of particles scattered in the environment the re-localization is faster (there is a higher likelihood that a particle is close to the actual position of the robot), but the average error is higher due to the lower number of particles clustered around the correct robot's pose. Notice that with 20% the re-localization is faster, but once recovered the correct localization, the average position error during position tracking is higher. This is because the number of the randomly distributed particles is so high that their contribution in the calculation of the center of gravity of the particles spoils the correct estimation of the robot pose. We therefore chose to distribute uniformly 10% of the 1,000 particles. This ensures low contribution in the calculation of the center of gravity and acceptably fast recovery from the kidnap situation.

B. Robustness to sensor occlusion

In order to show the robustness of our approach in crowded environments, we tested the system on six different paths; an example path is shown in Fig. 13. To understand how occlusion of the omnidirectional camera affects the localization process, consider the images in Fig. 12. The corresponding plots (to the right of each image) show the probability distributions of the robot's pose. As occlusion increases (0%, 25%, and 50%, respectively) the particles become more dispersed around the true position of the robot. Uncertainty increases, but most of the probability is still condensed around the correct position. This is the result of the ray discrimination technique presented in Section IV-B.2. To obtain a measurable amount of occlusion, the sensor was covered with black strips; every strip covers 12.5% of the sensor and adequately simulates the presence of one robot close to the sensor. In real situations,

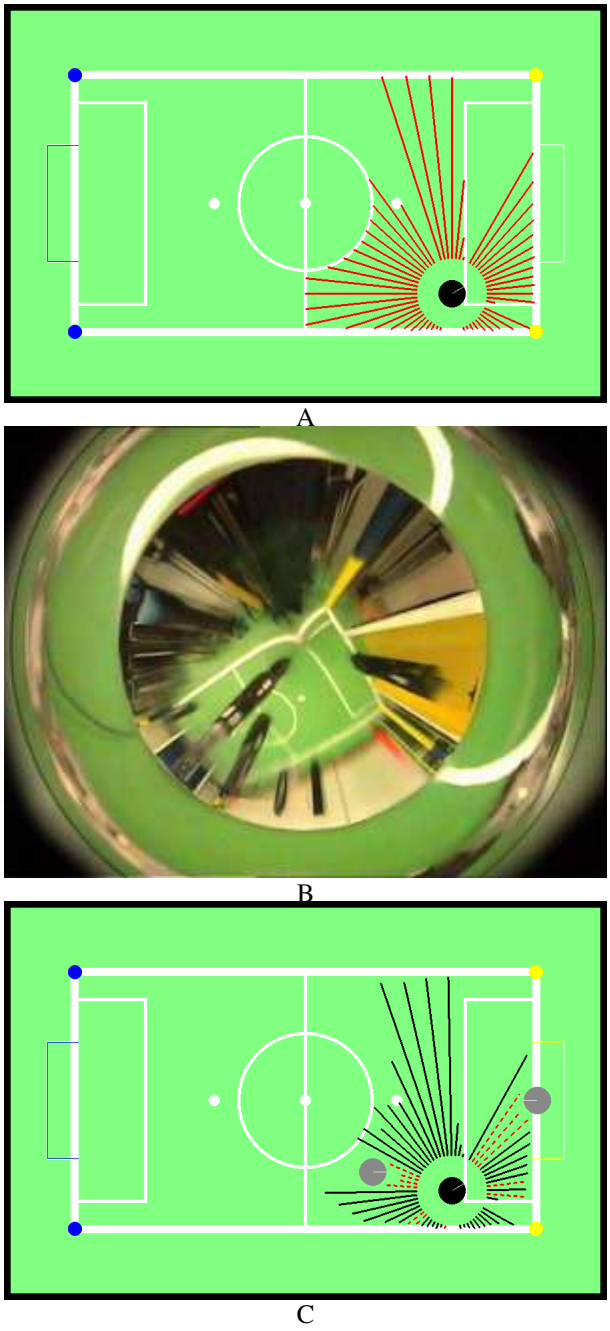


Fig. 11. An example of expected and measured scans for the green-white color transition in presence of occlusion. The robot's pose is represented by the black spot. A represents the expected scan for an ideal noise-free sensor in an empty environment. B shows the frame grabbed by the robot in that pose. C represents the corresponding measured scan. In C, the solid lines represent the measured distances, while the dotted lines represents the rays in which an unexpected transitions was detected (*FAKE_RAYS*). This can be caused by image noise or other robots (represented with gray spots). As can be seen in B, there is a robot (the goalkeeper) at the yellow goal; three rays of the scan detected it, as shown in C. Along these rays, a black, unexpected color was detected and *FAKE_RAYS* values were stored instead of the proper distance.

like the one depicted in Fig. 11, it is extremely hard to have more than two robots close to the sensor, while other robots are usually quite far and occlude only a small fraction of the sensor. The actual amount of occlusion during a real game strongly depends on the shape of the opponent robots. There

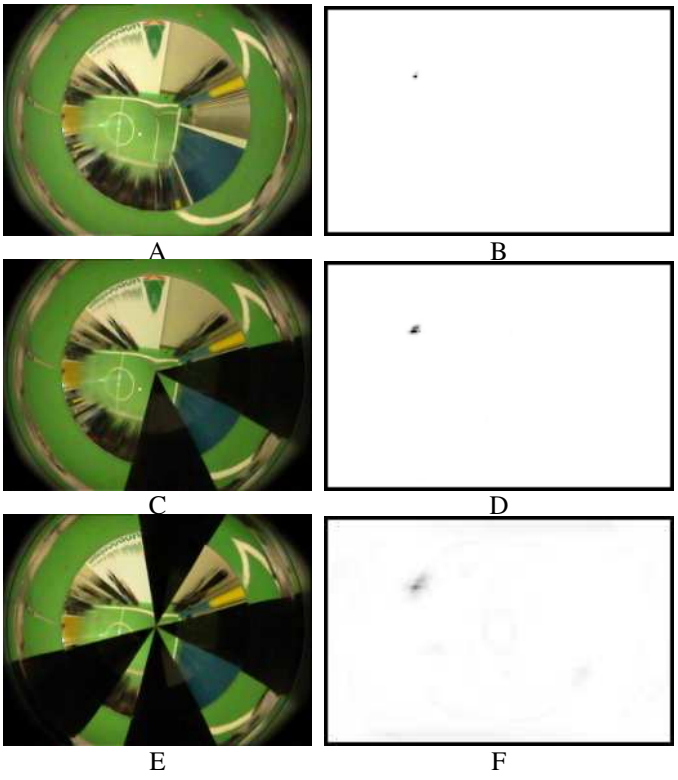


Fig. 12. (On the left) The occlusion of the sensor is obtained with black stripes simulating the presence of other robots close to the sensor,. This was done in order to have a measurable amount of sensor's occlusion (A 0% of occlusion, C for 25% and E for 50%). (On the right) The probability distributions calculated for the corresponding amount of sensor's occlusion. Notice that in the situations of higher occlusion D F, the particles are more scattered than in B, but most of the probability is still condensed around the correct position.

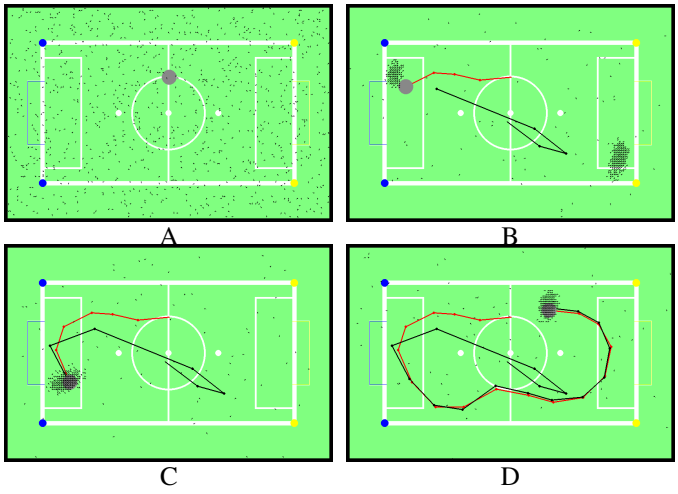


Fig. 13. A sequence of global localization and position tracking in presence of 12.5% of sensor occlusion. Note that, with respect to Fig. 7, the particles are more scattered around the true position of the robot.

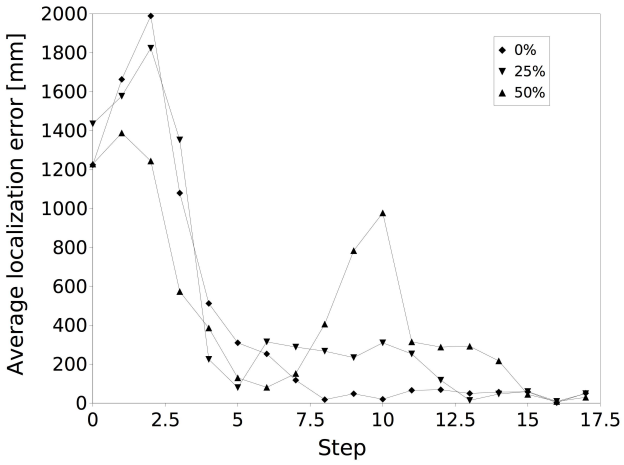


Fig. 14. The plots compares the global localization errors for a specific path with different amount of sensor's occlusion.

are robots like the ones of the Philips team [1] or IsocRob team [15] that are quite large and tall, while other robots like the ones of Fu-Fighter team [26] are rather small and very short. Our experiment shows that the performance of our system degrades slowly when occlusion increases. We estimated that a 50% continuous occlusion is well above the maximum occlusion that one robot can experience, but even in this case, our system functions correctly.

For each path, we collected five sequences of omnidirectional images with 0%, 12.5%, 25%, 37.5%, and 50% occlusion, respectively. For every image, we recorded the ground truth pose of the robot and the odometric readings between two consecutive positions. In order to take into account the odometry errors, robot movements were performed by sending position commands to the robot. We tested our algorithms for the three fundamental localization problems: global localization (Fig. 13 A and B); position tracking (Fig. 13 C and D); and kidnapped robot (not shown).

In Fig. 14 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 fast and accurate. Also in a "densely crowded" environment (sensor always 50% covered) 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 localization failure is obtained thanks to a small amount of particles (10% of the total number of particles) uniformly distributed in the environment. A few steps after a kidnapping episode, most of the particles are again concentrated around the correct position and the situation is the same as that of the global localization experiment.

Finally, we performed a statistical evaluation of our approach in the conventional situation of position tracking, repeating 100 times all reference paths with different amounts of occlusion. In Fig. 15 are reported the average error and the maximum error over all reference paths. Notice that both remain small also in a densely and constantly crowded environment.

C. *Localization in an office environment*

Even though our system was developed for the RoboCup domain, it was designed from the beginning having in mind

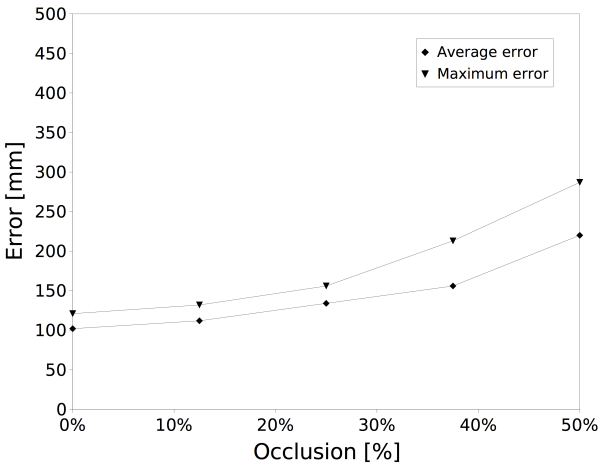


Fig. 15. Statistical evaluation of our system in the position tracking problem for all our reference paths. Accuracy (average error end maximum error) is represented for different amount of sensor's occlusion (0%, 12.5%, 25%, 37.5%, 50%).

its applicability to every-day environments. As a result, the localization system is not dependent on specific chromatic transitions. Chromatic transitions of interest can be of any number and color combination. As stated before, the only requirements for our system are: (i) an environment with meaningful color transitions, (ii) a geometric map of that environment, and (iii) inclusion of the color transitions in the map.

Fig. 16. The office-like environment in which our localization system was tested to prove its portability to real-world environments.

The environment in which we tested the generality of our system are the corridors of our department, as shown in Fig. 16. The floor of the corridor is composed of red tiles, the walls are painted white, while doors and furniture are gray. The corridor is 26 m long and its width ranges from a 2 m to 4 m. The trapezoidal room is about 4×5 m. Along the corridors there are two gray lockers 2 m wide. The plan of the whole environment is depicted in Fig. 20.

This environment is much more challenging than the RoboCup environment due to uneven illumination and to the

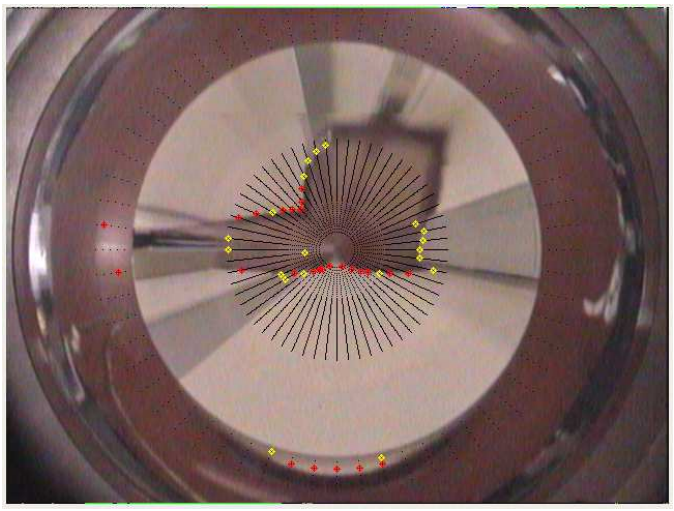


Fig. 17. The scanning algorithm at work on an image grabbed in a corridor of an office-like environment. The colored crosses highlight the color transitions of interest of the environment detected along the dotted radial lines in the omnidirectional image.

low contrast between existing colors. Nevertheless, as our experiments demonstrate, the system is able to locate sucessfully the robot, even though the vision system sometimes mistakes the type or misses the chromatic transitions. A typical input image for the robot is the image shown in Fig. 17. The chromatic transitions of interest we selected in this environment are *red-white* and *red-gray*. The red and yellow crosses mark red-white and red-gray transitions, respectively. In Fig. 18 is shown a comparison between the expected and the real scans of the office-like environment for red-white transitions.

To test the robustness of the system in a general indoor environment without any lighting control, we performed the test on an overcast day. Due to low ambient light, the noise in the image is high and the contrast between white and gray is low. In this situation, some chromatic transitions of interest are not detected (e.g., the one pointed by the arrow in Fig. 18A) or are erroneously detected (e.g., the one pointed by the arrow Fig. 18C). The probability distribution calculated from the red-white transitions in the image shown in Fig. 18 are depicted in Fig. 19. Dark and light regions represent, respectively, high and low probabilities to represent the correct robot pose. As can be seen, the probability distribution is quite sparse. Nevertheless, by combining both the information from the second chromatic transition of interest, as well as the information coming from different measurements using the Monte Carlo Localization algorithm, a robust localization can be achieved, as shown in Fig. 20. Starting without any knowledge about the robot's position (Fig. 20A) a few steps later most of the particles condense around the true position (Fig. 20D). As before, we randomly spread 10% of the particles to address the kidnapped robot problem.

VI. Conclusions

In this paper, we propose a vision-based Monte Carlo Localization system particularly suitable for densely populated environments, using a ray discrimination technique.

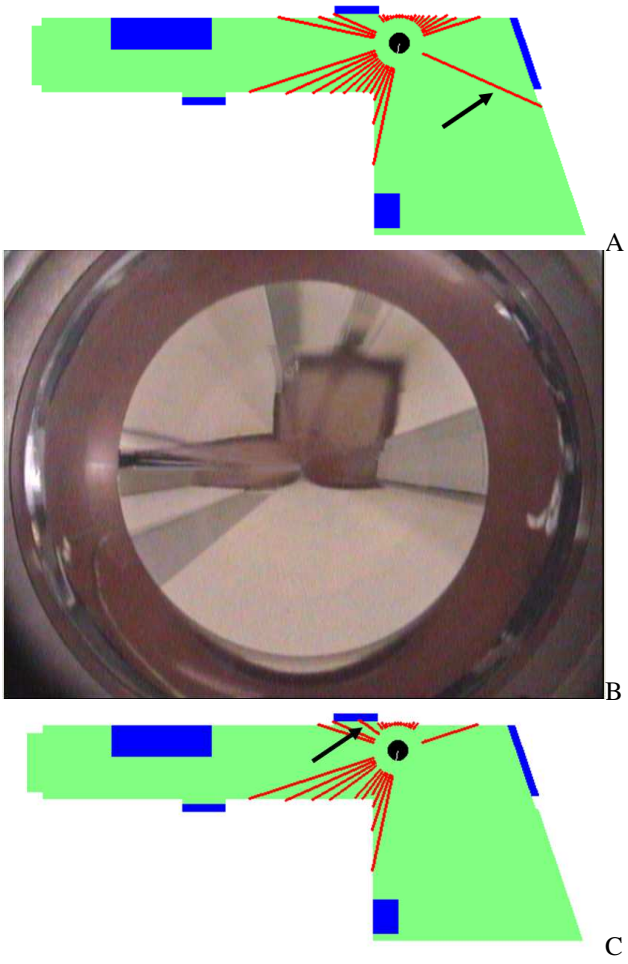


Fig. 18. Detection of red-white transitions in an office environment. The floor is represented in green, the gray objects are drawn in blue, and the rays of the scan are painted in red.The black spot represents the pose of the robot. In A is depicted a detail of the map with the expected scan for the red-white transitions. In B is shown an acquired image, while C shows the extracted scan. Note that C contains several wrong detections due to noise in the image. The arrow in A represents a transition not detected in C; the arrow in C represents an erroneously detected transition.

Fig. 19. The probability distribution calculated from the red-white transition.

The omnidirectional vision system emulates the behavior of rangefinder devices and, due to the ability to distinguish different color transitions, it can detect and reject wrong measurements caused by occlusions. We developed our system in the Middle-Size RoboCup domain, but we showed that it can be used to localize the robot in any environment in which meaningful chromatic transitions exist. Our system requires only a map with the metric and chromatic characteristics of the environment. This map must contain the static obstacles and the chromatic transitions of interest, and can be as simple as a drawing stored in an image file (representing a floor plan plus the information on color transitions). From such a map,

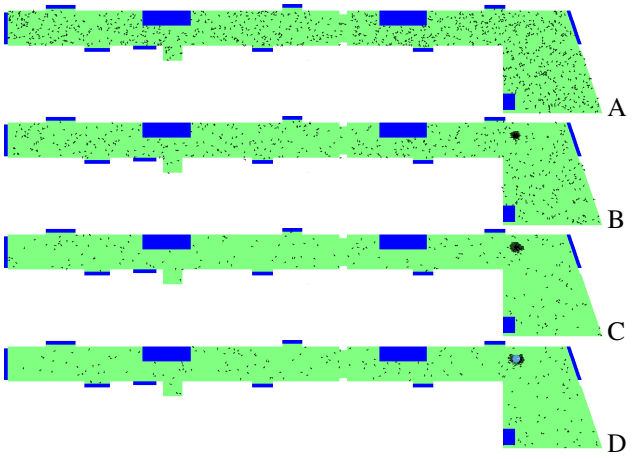


Fig. 20. An example of global localization in the office-like environment in our department. 10% of the particles are randomly distributed in the environment to recover in case of wrong localization (kidnapped robot).

the system will automatically recalculate all look-up tables used in the localization process.

The reliability of the localization system could be further improved if a more robust color transition detection algorithm is used (like the ones proposed in [12], [21]), but such studies are beyond the scope of the present work. We are currently implementing a color transition detector robust to illumination changes to test our system in outdoor environments.

This paper presented successful experiments of global localization, position tracking, and robot kidnapping, both in the RoboCup environment and in the corridors of our department. We experimentally showed the robustness of the localization system to sensor occlusion and to chromatic transitions with poor contrast. The proposed system has characteristics that enable its use in a variety of applications, including navigation in populated environment, outdoor localization, and integration with other localization systems.

REFERENCES

[1] Philips team description. http://www.apptech.philips.com/robocup/.

[2] G. Adorni, S. Cagnoni, S. Enderle, G. Kraetzschmar, M. Mordonini, M. Plagge, M. Ritter, S. Sablatnög, and A. Zell. Vision-based localization for mobile robots. *Robotics and Autonomous Systems*, 36:103–119, 2001.

[3] A. Bonarini. The body, the mind or the eye, first? *Proc. of the Third Int. Workshop on Robocup (Robocup99) IJCAI Press*, pages 40–50, 1999.

[4] F. Dellaert, W. Burgard, D. Fox, and S. Thrun. Using the condensation algorithm for robust, vision-based mobile robot localization. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'99)*, June 1999.

[5] F. Dellaert, D. Fox, W. Burgard, and S. Thrun. Monte Carlo Localization for mobile robots. In *IEEE International Conference on Robotics and Automation (ICRA99)*, May 1999.

[6] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977.

[7] S. Enderle, M. Ritter, D. Fox, S. Sablatnög, 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.

[8] D. Fox, W. Burgard, F. Dellaert, and S. Thrun. Monte Carlo localization: Efficient position estimation for mobile robots. In *Proceedings of National Conference on Artificial intelligence (AAAI'99)*, pages 343–349, July 1999.

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

[10] N. Gordon, D. Salmond, and A. F. M. Smith. Novel approach to nonlinear and non-gaussian bayesian state estimation. *IEEE Proceedings-F*, 140:107–113, 1993.

[11] H.-M. Gross, A. Koenig, C. Schroeter, and H.-J. Boehme. Omnivision-based probabilistic self-localization for a mobile shopping assistant continued. In *IEEE/RSJ Int. Conference on Intelligent Robots and Systems (IROS 2003)*, pages 1505–1511, October 2003, Las Vegas USA.

[12] F. Hundelschausen, S. Behnke, and R. Rojas. An omnidirectional vision system that finds and tracks color edges and blobs. In A. Birk, S. Coradeschi, and S. Tadokoro, editors, *RoboCup-2001: Robot Soccer World Cup V.*, L. N. on A. I. 2377, pages 374–379. Springer, 2002.

[13] G. Kitagawa. Monte Carlo filter and smoother for non-gaussian nonlinear state space models. *Journal Of Computational and Graphical Statistics*, 5(1):1–25, 1996.

[14] S. Lenser and M. Veloso. Visual sonar: Fast obstacle avoidance using monocular vision. In *IEEE/RSJ Int. Conference on Intelligent Robots and Systems (IROS 2003)*, pages 886–891, October 2003, Las Vegas USA.

[15] P. Lima, L. Custódio, P. Marcelino, H. Costelha, G. Neto, V. Pires, M. Arroz, and B. Vecht. Isocrob 2004 team description paper. In *RoboCup-2004 -Proceedings of the International Symposium*, 2004.

[16] 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.*, L. N. on A. I, pages 78–87. Springer, 2002.

[17] E. Menegatti, A. Pretto, and E. Pagello. A new omnidirectional vision sensor for Monte-Carlo localisation. In *RoboCup Symposium 2004*, pages CD–ROM (13 pages), 2004.

[18] E. Menegatti, A. Pretto, and E. Pagello. Testing omnidirectional vision-based Monte-Carlo localization under occlusion. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS04)*, pages 2487–2494, September 2004.

[19] 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.

[20] E. Menegatti, M. Zoccarato, E. Pagello, and H. Ishiguro. Image-based Monte-Carlo localisation with omnidirectional images. *Robotics and Autonomous Systems, Elsevier*, 48(1):17–30, August 2004.

[21] T. Röfer and M. Jüngel. Vision-based fast and reactive Monte-Carlo localization. In *Proc. of International Conference on Robotics and Automation (ICRA-2003)*, 2003.

[22] E. Schulenburg, T. Weigel, and A. Kleiner. Self-localization in dynamic environments based on laser and vision data. In *Proc. of the 2003 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS03)*, pages – , Las Vegas, NV, 2003.

[23] S. Thrun, A. Bücken, W. Burgard, D. Fox, T. Fröhlinghaus, D. Henning, T. Hofmann, M. Krell, and T. Schmidt. Map learning and high-speed navigation in RHINO. In D. Kortenkamp, R. Bonasso, and R. Murphy, editors, *AI-based Mobile Robots: Case Studies of Successful Robot Systems*. MIT Press, 1998.

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

[25] F. von Hundelshausen. An omnidirectional vision system for soccer robots. Master thesis, Institut für Informatik, Freie Universität Berlin, Takustr. 9, 14195 Berlin, April 2001.

[26] F. von Hundelshausen, R. Rojas, F. Wiesel, E. Cuevas, D. Zaldivar, and K. Gunarsson. Fu-fighters team description 2003. In D. Polani, B. Browning, A. Bonarini, and K. Y. (Co-chairs), editors, *RoboCup-2003-Proceedings of the International Symposium*, 2003.

[27] 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*, 18(5):685–699, 2002.

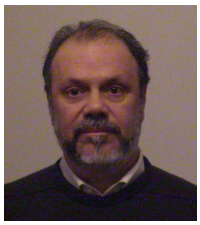
[28] T. Wilhelm, H.-J. Böhme, and H.-M. Gross. A multi-modal system for tracking and analyzing faces on a mobile robot. *Robotics and Autonomous Systems*, 48:31–40, August 2004.

[29] 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.

Emanuele Menegatti Emanuele Menegatti received his "Laurea" in Physics at the University of Padua in 1998. He got an MSc in Artificial Intelligence from the University of Edinburgh (U.K.) in 2000, and a Ph.D. in Computer Science from the University of Padua in 2002. From 2003 to 2004 he was a Post-Doc Researcher at the Intelligent Autonomous Systems Laboratory (IAS-Lab) of the University of Padua where he is now an Assistant Professor of Computer Science. In 2004, he was a visiting researcher at the Laboratory of Prof. H. Ishiguro at Osaka University (Japan), and the Laboratory of Prof. Frank Dellaert at the Georgia Institute of Technology. His research interests are in the field of Robot Vision, particularly omnidirectional vision.

Alberto Pretto Alberto Pretto received his "Laurea" on Electronic Engineering at the University of Padua (Italy) in 2003. After his graduation, he worked for three months as scientific collaborator at the Intelligent Autonomous Systems Laboratory (IAS-Lab) of the University of Padua. In 2004 and 2005, he was a software engineer at Trastec Scpa. Since 2006, he is Ph.D. student at The University of Padua. Pretto's main research interests are in the field of Robot Vision and Robot Navigation.

Alberto Scarpa Alberto Scarpa is a student member of the Intelligent Autonomous Systems Laboratory (IAS-Lab) of the University of Padua since 2001. He has worked on several projects on mobile robot localization and on robot vision, both for the Artisti Veneti (the RoboCup team of the University of Padua) and for other IAS-Lab projects.



Enrico Pagello Enrico Pagello received his "Laurea" on Electronic Engineering at the University of Padua (Italy) in 1973. From 1973 to 1983, he was a Research Associate at the Inst. of Biomedical Engineering of the National Research Council of Italy, where he is now a part-time collaborator. Since 1983 he belongs to the Faculty of Engineering of the University of Padua, where he is Professor of Computer Science at the Dept of Information Engineering. During 1977-78, he was a Visiting Scholar at the Lab. of Artificial Intelligence of Stanford University. Since 1994, he has regularly visited the Dept of Precision Engineering of the Univ. of Tokyo, in the context of a joint scientific agreement between the Padua and Tokyo Universities and of the JSPS Senior Fellow Program. He was the General Chair of the Sixth Int. Conf. on Intelligent Autonomous Systems in July 2000, and a General Chairman of RoboCup-2003, in July 2003. He has been a member of the Editorial Board of the IEEE/Trans. on Rob. and Aut., and he is currently a member of the Editorial Board of RAS International Journal. He is a President of the Intelligent Autonomous Systems International Society. His current research interests concern the application of Artificial Intelligence to Robotics with particular regards to the Multi-robot Systems domain.