Cooperative Localization Based on Visually Shared Objects

Pedro U. Lima[1,2], Pedro Santos[1], Ricardo Oliveira[1], Aamir Ahmad[1], and João Santos[1]

[1] Institute for Systems and Robotics, Instituto Superior Técnico, 1049-001 Lisboa, Portugal
[2] Universidad Carlos III de Madrid, Avda. Universidad, 30, 28911 Leganés, Spain
{pal,psantos,aahmad,jsantos}@isr.ist.utl.pt,
ricardo.oliveira@ist.utl.pt

Abstract. In this paper we describe a cooperative localization algorithm based on a modification of the Monte Carlo Localization algorithm where, when a robot detects it is lost, particles are spread not uniformly in the state space, but rather according to the information on the location of an object whose distance and bearing is measured by the lost robot. The object location is provided by other robots of the same team using explicit (wireless) communication. Results of application of the method to a team of real robots are presented.

1 Introduction and Related Work

Self-localization is one of the most relevant topics of current research in Robotics. Estimation-theoretic approaches to self-localization, as well as to self-localization and mapping (SLAM) have produced significant results in recent years, mainly for single robots, providing effective practical results for different applications. One of the research frontiers in this topic concerns now cooperative localization (and possibly mapping) using a team of multiple robots.

One of the earlier works on cooperative localization [Sanderson, 1996] addresses cooperative localization within a Kalman filter framework, where the relative positions of the robots are the observations of the filtering part of the algorithm, and the state includes the positions of all the robots. Fox et al introduced an extended version of the general Markov Localization algorithm [Fox et al., 2000], where two robots use measurements of their relative distance and bearing to insert an extra step in the belief update algorithm based on the Bayes filter. They used the Monte Carlo Localization (MCL) sampled version of Markov Localization algorithm to influence the weights of the particles of the observed robot from the particles sampling the inter-robot distance and bearing measurement model of the observing robot. Other authors address multi-robot localization using similar approaches, so as to provide relative localization of the team members in one of the team robots local frame from inter-robot distance measurement [Roumeliotis and Bekey, 2002, Zhou and Roumeliotis, 2008]. All these works do not use environment information commonly observed by the team robots to improve their localization.

Other works attempt to take advantage of environment features and landmarks to help a multirobot team to improve the pose estimates of its own team members, while

J. Ruiz-del-Solar, E. Chown, and P.G. Ploeger (Eds.): RoboCup 2010, LNAI 6556, pp. 350–361, 2011.

simultaneously mapping the landmark locations. Fenwick et al [Fenwick et al., 2002] focus on convergence properties and performance gain resulting from the collaboration of the team members on concurrent localization and mapping operations. Jennings et al [Jennings et al., 1999] describe a stereo-vision-based method that uses landmarks whose location is determined by one of the robots to help the other robot determining its location. The first approach addresses a general model that does not take advantage of particular features of the estimation-theoretic methods used (e.g., particle filters) to improve the robustness and to speed up cooperative localization, while the second is focused on a particular application.

In this paper, we introduce a modification of MCL that changes the particle spreading step (used when a robot detects it is lost), using information provided by other robot(s) of the team on the location of an object commonly observed by the lost robot. This modification speeds up the recovery of the lost robot and is robust to perceptual aliases, namely when environments have symmetries, due to the extra information provided by the teammates. The introduced method enables cooperative localization in a multirobot team, using visually shared objects, taking advantage of the specific features of particle filter algorithms. Each robot is assumed to run MCL for its self-localization, and to able to detect when the uncertainty about its localization drops below some threshold. An observation model that enables determining the level of confidence on the ball position estimate is also assumed to be available at each robot of the team. Though these assumptions are, to some extent, stronger than those assumed by cooperative simultaneous localization and mapping methods, they allow global robot and object localization. Though other authors have explored the use of observations to initialize and/or reset particle filters adequately [Lenser and Veloso, 2000, Thrun et al., 2001], the use of shared observations of common objects to cooperatively improve multirobot MCL is novel, to the best of our knowledge.

The paper is organized as follows: in Section 2, we describe our cooperative localization method. Results of experiments with real soccer robots in the RoboCup Middle-Size League (MSL), that use the ball as the visually shared object, are presented in Section 3. Conclusions and prospects for future work are discussed in Section 4.

2 Cooperative Localization Using a Visually Shared Object

Let us consider a team of N robots, $r_1, \ldots, r_n$. Robot r_i has pose (position + orientation) coordinates $\mathbf{l}_{r_i} = (x_{r_i}, y_{r_i}, \theta_{r_i})$ in a global world frame, and estimates them using a MCL algorithm.

Each robot can determine the position of an object o in its local frame, therefore being able to determine its distance and bearing to that object as well. Robots can also determine if they are lost or kidnapped, i.e., if their confidence in the pose estimate drops below some threshold. If a robot is not lost, it can also determine the object position in the global world frame using the transformation between its local frame and the global world frame that results from the knowledge of its pose. The estimate of the object position in any frame is determined based on a probabilistic measurement model that includes the uncertainty about the actual object position. When the global world frame is used, additional uncertainty is caused by the uncertain pose of the observing robot.

The position of the object as determined by robot r_i in the global world frame is denoted by $\mathbf{p}_o^i = (x_o^i, y_o^i)$, while the distance and bearing of the object with respect to the robot, as measured by the robot, are given by d_o^i and ψ_o^i, respectively.

2.1 Overall Description

The original MCL algorithm used by each of the team robots to estimate its pose is as follows:

Algorithm MCL$(\mathbf{L}(t-1), u(t), z(t), map)$
static w_{slow}, w_{fast}
$\bar{\mathbf{L}}(t) = \mathbf{L}(t) = \emptyset$
$w_{avg} = 0$
for $m = 1$ to M **do**
  $\mathbf{l}^{[m]}(t) =$ ***sample_motion_model***$(u(t), \mathbf{l}^{[m]}(t-1))$
  $w^{[m]}(t) =$ ***measurement_model***$(z(t), \mathbf{l}^{[m]}(t), map)$
  $\bar{\mathbf{L}}(t) = \bar{\mathbf{L}}(t) + \langle \mathbf{l}^{[m]}(t), w^{[m]}(t) \rangle$
  $w_{avg} = w_{avg} + \frac{1}{M} w(t)^{[m]}$
endfor
$w_{slow} = w_{slow} + \alpha_{slow}(w_{avg} - w_{slow})$
$w_{fast} = w_{fast} + \alpha_{fast}(w_{avg} - w_{fast})$
for $m = 1$ to M **do**
  with probability $\max\{0.0, 1 - w_{fast}/w_{slow}\}$ **do**
    add random pose to $\mathbf{L}(t)$
  else
    draw $i \in \{1, ..., M\}$ with probability $\propto w^{[i]}(t)$
    add $\mathbf{l}^{[i]}(t)$ to $\mathbf{L}(t)$
  endwith
endfor
return $\mathbf{L}(t)$

where $\bar{\mathbf{L}}(t)$ is a set of Mparticles and their weights at step t of the iteration process, $\langle \mathbf{l}^{[m]}(t), w^{[m]}(t) \rangle,\ m = 1, \ldots, M$, $\mathbf{L}(t)$ is a set of M unweighted particles $\mathbf{l}^{[m]}(t),\ m = 1, \ldots, M$, $u(t)$ are odometry readings at time t, $z(t)$ are robot observations at time t, concerning its self-localization, *map* is a map of the robot world (e.g., a set of landmarks or other), and w_{fast}, w_{slow} are auxiliary particle weight averages, with $0 \leqslant \alpha_{slow} \ll \alpha_{fast}$, such that w_{slow} provides long-term averages and w_{fast} provides short-term averages. The algorithm uses a ***sample_motion_model*** and a ***measurement_model*** to update, at each step, the robot pose, from the odometry $u(t)$ and measurements $z(t)$ information. It keeps adding random particles to those obtained in the re-sampling step (where the probability of cloning an existing particle is proportional to its weight), in a number which increases with the deviation of the w_{fast} average from the long-term w_{short} average. In the limit case, when all particle weights tend to zero in the short-term, all particles are reset according to an uniform distribution.

When cooperative localization in a multirobot team, using visually shared objects, is intended, MCL running in a given robot r_i must be modified so as to use information from other robot(s) in the team, when w_{fast}/w_{slow} drops below a given confidence threshold $C_{threshold}$, meaning that r_i is lost or was kidnapped. That information comes in the form of the object position determined by the other robot(s). Assuming r_i (the lost/kidnapped robot) can observe the same object, the re-sampling is then based on a spatial probability distribution which depends on the distance and bearing of the lost robot to the object and on the uncertainty associated to the object position measurement provided by the other robot(s). This way, while a uniform distribution is still used to keep a certain level of exploration of the pose space to make the algorithm robust to measurement and motion errors, if those errors influence becomes too high, the particles are completely reset according to the cooperative information from teammates about a visually shared object.

The new MCL algorithm used by robot r_i from the team to estimate its pose becomes (the subindex r_i is used for local estimates, odometry readings, observations and particle weights):

Algorithm Cooperative_Shared_Object_MCL$(\mathbf{L}_{r_i}(t-1), u_{r_i}(t), z_{r_i}(t), map)$
static w_{slow}, w_{fast}
$\bar{\mathbf{L}}_{r_i}(t) = \mathbf{L}_{r_i}(t) = \emptyset$
$w_{avg} = 0$
for $m = 1$ to M **do**
  $\mathbf{l}_{r_i}^{[m]}(t) =$ ***sample_motion_model***$(u_{r_i}(t), \mathbf{l}_{r_i}^{[m]}(t-1))$
  $w_{r_i}^{[m]}(t) =$ ***measurement_model***$(z_{r_i}(t), \mathbf{l}_{r_i}^{[m]}(t), map)$
  $\bar{\mathbf{L}}_{r_i}(t) = \bar{\mathbf{L}}_{r_i}(t) + \langle \mathbf{l}_{r_i}^{[m]}(t), w_{r_i}^{[m]}(t) \rangle$
  $w_{avg} = w_{avg} + \frac{1}{M} w_{r_i}^{[m]}(t)$
endfor
$w_{slow} = w_{slow} + \alpha_{slow}(w_{avg} - w_{slow})$
$w_{fast} = w_{fast} + \alpha_{fast}(w_{avg} - w_{fast})$
if $w_{fast}/w_{slow} < C_{threshold}$ **and** info about object position in global world frame available from teammate(s) $r_j \neq r_i$ **and** object visible to r_i **then**
  draw $\mathbf{L}_{r_i}(t)$ according to object pose spatial probability distribution determined from r_i and r_j information
else
  for $m = 1$ to M **do**
    with probability $\max\{0.0, 1 - w_{fast}/w_{slow}\}$ **do**
      add random pose to $\mathbf{L}_{r_i}(t)$
    else
      draw $k \in \{1, ..., M\}$ with probability $\propto w_{r_i}^{[k]}(t)$
      add $\mathbf{l}_{r_i}^{[k]}(t)$ to $\mathbf{L}_{r_i}(t)$
    endwith
  endfor
endif
return $\mathbf{L}_{r_i}(t)$

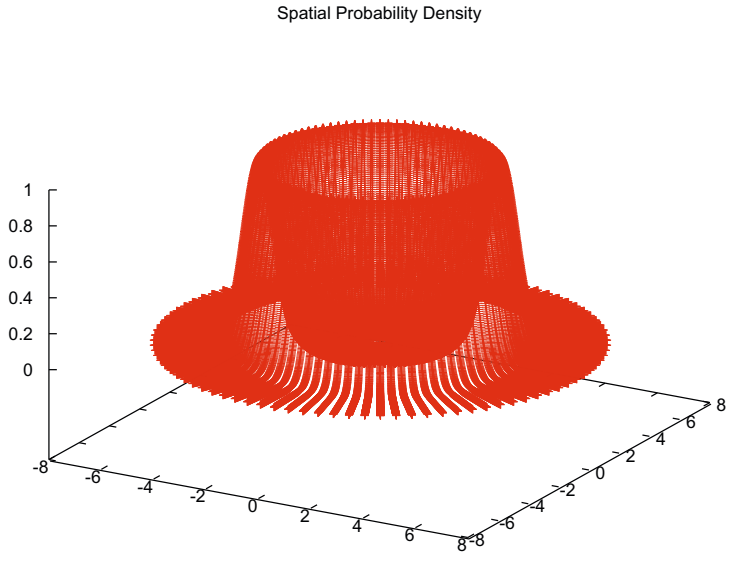

Fig. 1. Typical spatial probability density function from which particles are drawn, after a decision to reset MCL

In the new algorithm one needs to further detail how to handle the following issues:

1. info about object position in global world frame available from teammate(s) $r_j \neq r_i$;
2. draw $\mathbf{L}_{r_i}(t)$ according to object pose spatial probability distribution determined from r_i and r_j information.

Item 1. concerns $\mathbf{p}_o^j$, i.e., the object position in the global world frame, as determined by r_j (in general, r_j may be any robot but r_i, or several such robots, in which case the object position results from the fusion of their information). Furthermore, we assume that, associated with $\mathbf{p}_o^j$, r_j provides a confidence measure regarding that information. That confidence measure depends on r_j's

– object observation model;
– self-localization estimate uncertainty.

Assuming a bivariate Gaussian object observation model for r_j centered on $\mathbf{p}_o^j$ and with a covariance matrix $\Sigma_o^j(d_o^j, \psi_o^j)$ dependent on the distance and bearing to the object, this item contributes to the confidence measure with $|\Sigma_o^j(d_o^j, \psi_o^j)|^{-1}$.

The self-localization estimate confidence factor is not so simple, since one must determine it from the particle filter set. One good approach to this is to consider the *number of effective particles* $n_{eff}^{r_j} = \frac{1}{\sum_{m=1}^{M}(w_{r_j}^{[m]})^2}$ [Thrun et al., 2005].

Considering both factors, the measure of confidence of r_j on its own estimate of object o position $n_{\mathbf{p}_o^j}$ is given by

$$CF_{\mathbf{p}_o^j} = \eta|\Sigma_o^j(d_o^j, \psi_o^j)|^{-1} n_{eff}^{r_j}$$

where η is a normalization factor.

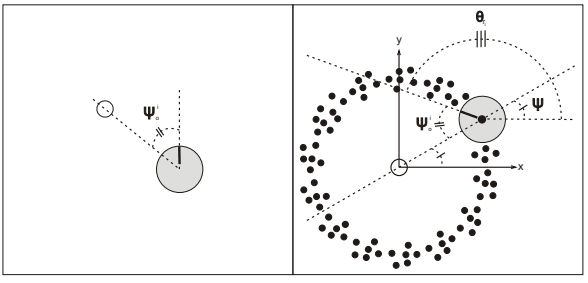

Fig. 2. Computing the orientation of a particle representing a robot pose hypothesis. On the left, bearing of the object with respect to the robot. On the right: relevant angles for the computation of the robot orientation hypothesis for each particle.

Regarding item 2., and assuming that the object is visible to the lost robot r_i, this robot determines the distance and bearing of the object in its local frame, (d_o^i, ψ_o^i). The distance d_o^i is used to parametrize the spatial probability density function (pdf) from which particles representing the robot position in polar coordinates are drawn, after a decision to reset MCL. This bivariate (using polar coordinates d and ψ centered in the object) pdf is:

- Gaussian in the d variable, with mean value d_o^i and variance inversely proportional to the confidence factor $CF_{\mathbf{p}_o^j}$;
- uniform in the ψ variable, in the interval $[0, 2\pi[$ rad.

An example of this pdf is shown in Figure 1.

One can trivially map the polar coordinates onto Cartesian coordinates, thus obtaining the x_{r_i}, y_{r_i} position components of the pose $\mathbf{l}_{r_i}$ for robot r_i.

The orientation component θ_{r_i} of $\mathbf{l}_{r_i}$ is computed from the bearing angle ψ_o^i of the object, i.e., the angle between r_i longitudinal axis (the one pointing towards its "front") and the line connecting its center with the object center (see Figure 2 - left), and the actual angle ψ of this line with respect to the x-axis of a frame centered on the object, i.e., the particle angle in polar coordinates centered on the object (see Figure 2 - right). We add some random noise θ_{rand} with a zero mean Gaussian pdf representing the bearing measurement error model. Hence, from Figure 2:

$$\theta_{r_i} = \psi + \pi - \psi_o^i + \theta_{rand}.$$

In summary, the **Cooperative_Shared_Object_MCL** algorithm modifies the plain MCL algorithm, replacing the "standard" particle reset, using a spatially uniform pdf, by a particle reset based on the information about the position, in the global world frame (determined by one or more teammates), and the distance and bearing, in the local frame of the lost/kidnapped robot, of an object visible to all the intervening robots. Additional input parameters of this algorithm are (assuming r_i as the lost/kidnapped robot and r_j as any of the robots providing information to improve its localization):

- $C_{threshold}$ to determine if the robot is lost or was kidnapped;
- determinant of the object measurement model covariance matrix $|\Sigma_o^j(d_o^j, \psi_o^j)|^{-1}$ — sent by r_j to r_i when r_i detects it is lost and requests support from teammates;
- the *number of effective particles* in $n_{eff}^{r_j}$ MCL algorithm — sent by r_j to r_i when r_i detects it is lost and requests support from teammates;
- distance and bearing of the object in r_i local frame, (d_o^i, ψ_o^i) — measured by r_i when it is lost and receives the above information from teammate(s);
- variance of the zero mean Gaussian pdf representing the bearing measurement error model at r_i (see previous item).

The first two items can be combined first in r_j, that sends to r_i its confidence factor $CF_{\mathbf{p}_o^j}$ on the object position in the global world frame.

The regular procedure for each of the team robots is to run **Cooperative_Shared_Object_MCL**. When $w_{fast}/w_{slow} < C_{threshold}$ at r_i, this robot requests help to teammates. One or more teammates r_j send the object position in global world coordinates and the associated confidence factor. Then, r_i particles are spread uniformly over a circle centered on the object, with nominal radius equal to the distance to the object measured by r_i, added to a Gaussian uncertainty around this value, with variance proportional to r_j confidence factor. The orientation component of the pose results from the bearing ψ_o^i of the object measured by r_i in its local frame, including an uncertainty proportional to the variance of the Gaussian representing this measurement model.

A couple of practical issues to be considered are:

- after a robot detects it is lost and spreads its particles over a circle centered with the visually shared object, it should run the regular MCL algorithm in the next steps (a number dependent on the application), so that it does not keep resetting its particles over a circle around the object, while its pose estimate has no converged to the actual value;
- the decision on which teammates can contribute with useful information may be taken by considering their own confidence factor and only using the information provided by robots with $CF_{\mathbf{p}_o^j}$ above some given threshold. In general, all teammates can contribute, but in some cases their information may be highly uncertain.

2.2 Particle Spreading Validation

There is a specific situation where the proposed algorithm requires some improvement, e.g., when a robot is kidnapped to a pose where it observes the object at approximately the same distance of where the robot was before. In this case, when the robot detects it is lost by checking its w_{fast}/w_{slow} value, it will still spread new particles over a circle centered with the object, including a region around where the robot wrongly estimates its pose. To prevent such situations, the algorithm must include a restriction that requires all the re-spread particles to be out of a region (e.g., a circle) that includes the majority of the particles at the MCL step when the robot detected it was lost. Nonetheless, it is important to note that particles may be correct in the old pose, if they just have similar positions but fairly different orientations. Because of this, the algorithm must also check if the orientation hypothesis associated to each particle is within a small range of values around the orientation at kidnapping detection time. If they are not, the restriction above does not apply.

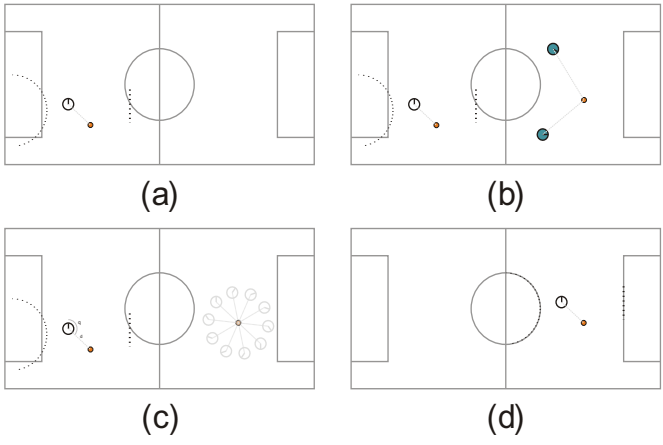

Fig. 3. Cooperative robot localization in RoboCup Soccer MSL: (a) The white robot determines the ball position in its local frame but its estimated pose (based on field line detection - lines observed by robot are dashed and do not coincide at all with the actual solid field lines) is incorrect, because the robot was kidnapped. (b) Teammates (green robots) communicate the ball position in the global world frame, as well as the corresponding confidence factor. (c) Lost robot measures its distance and bearing to the ball and re-spreads the particles according to this and to the ball position team estimate. (d) The previously lost robot regains its correct pose.

3 Results of Implementation in Real Soccer Robots

We have applied the **Cooperative_Shared_Object_MCL** algorithm to real robots in RoboCup Soccer Middle-Size League (MSL), in which the robots use the ball to regain their pose when they are kidnapped and detect to be lost on the field. Figure 3 provides an example that illustrates the algorithm application in this scenario.

3.1 Experimental Setup 1

In this setup tests were made by kidnapping a robot to nine different positions, in one quarter of the whole soccer field, as depicted in Figure 4. The other field regions would not provide extra information, due to the soccer field symmetry. One teammate stopped in a random position always sees the ball and informs the kidnapped robot of the ball's position on the global field frame. Both robots use MCL with 1000 particles, as described in a previous paper [Santos and Lima, 2010]. The standard deviation of the Gaussian used to model the bearing angle measurement error at the kidnapped robot was adjusted experimentally as $\pi/12$ rad.

We kidnapped the robot five times for each of the nine positions. Kidnapping was carried out by picking up the robot and moving it to another location with MCL on and after its convergence to a correct estimate. After kidnapping, the increase of uniformly distributed particles was visible, turning quickly to a re-spread over circle centered with the ball position, as estimated by the teammate.

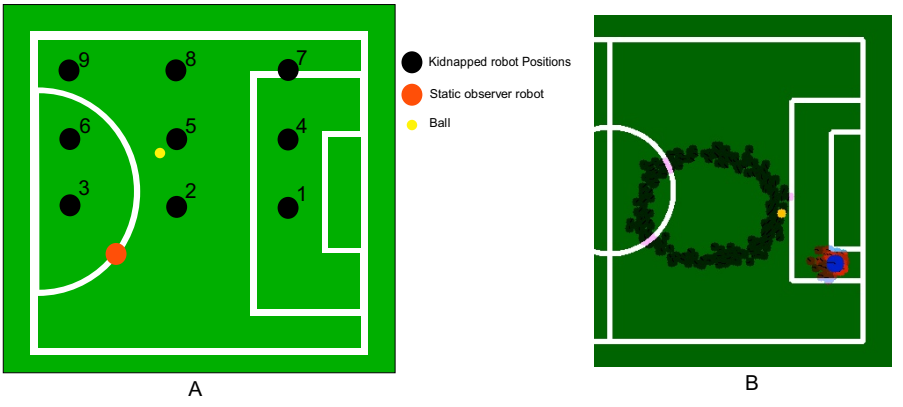

Fig. 4. Layout of experiments. A) The black numbered spots correspond to the positions to where one of the team robots was kidnapped. The red circle represents a static robot, always well localized, and watching the ball (smaller yellow circle). B) The figure in B is an example snapshot of the global frame interface which plots real robot and ball positions as well as particles used by MCL in real-time. Here it shows the kidnapped robot spreading particles after getting lost and using the shared ball.

The algorithm runs on a NEC Versa FS900 laptop with a Centrino 1.6GHz processor and 512Mb of memory, using images provided by a Marlin AVT F033C firewire camera, and in parallel with the other processes used to make the robot play soccer. The camera is mounted as part of an omnidirectional dioptric system, using a fish-eye lens attached to it.

The robots disposed as in position 4 are depicted in Figure 5.

Fig. 5. Real robot example for one of the layout locations: the robot on the left is the robot that informs about the ball position, while the robot on the right is the kidnapped robot, in location 4

3.2 Experimental Setup 2

In this setup, the teammate which observes the ball tracks and follows the ball by maintaining a fixed orientation and distance with respect to ball. We kidnapped the other robot in the following 4 cases during this setup:

- Case 1: Both observer robot and the ball moving when the ball is in the field of view (FOV) of both robots. Robot kidnapped twice in this case.
- Case 2: Observer robot stopped and the ball is moving while the ball is in the FOV of both robots. Robot kidnapped once in this case.
- Case 3: Both observer robot and the ball stopped when the ball is in the FOV of both robots. Robot kidnapped twice in this case.
- Case 4: Both observer robot and the ball moving when the ball moves away from the FOV of the kidnapped robot during the time of kidnapping and then later reappears in its FOV. Robot kidnapped once in this case.

The rest of the details for this setup is similar to experimental setup 1.

3.3 Results and Discussion

Results of experiments in setup 1 are shown in Table 1. In the table, successes correspond to the number of experiments where the robot could regain its correct pose after kidnapping occurred. Iterations to converge refer to the mean value of iterations required by the algorithm to converge after a kidnapping, over 5 experiments for a given kidnapping location. Note that one prediction and one update iteration of MCL take approximately 0.1s each, therefore the mean value of iterations should be multiplied by 0.2 s to have an idea of the time taken by the algorithm in each case. Overall, the algorithm performed quite well in real situations, including cases where we kidnapped the robot to a position where its distance to the ball remained the same. Some field locations are clearly more demanding than others (e.g., 5, 6, 7) causing the robot to fail to regain its posture in one of the tests, possibly due to perceptual aliasing relatively to other field positions.

In the experimental setup 2 the robot successfully recovers and re-localizes itself in 5 situations (3 Cases) and fails in 1. The number of iterations performed by the

Table 1. In experimental setup 1 results of kidnapping a robot to 9 different positions on the field (third column is the average of 5 experiments per location)

Field position	Successes	Iterations to converge
1	5	14.6
2	5	14.8
3	5	7.2
4	5	19.5
5	4	25.5
6	4	10
7	4	15.3
8	5	21
9	5	16

Table 2. Results of kidnapping as explained in experimental setup 2

Case	Situation	Result	Iterations to converge
1	1	Success	17
	2	Success	26
2	1	Failure	18
3	1	Success	15
	2	Success	19
4	1	Success	61

algorithm to converge are presented in Table 2. In case 4 of this setup, the robot performs a very high number of iterations to converge mainly due to the absence of the ball from kidnapped robot's FOV for a while before it comes in the FOV of both robots.

In all the sets of experiments, communication delay between the robots was consistently monitored during the run-time of the algorithm. Older data (> 2seconds) was discarded. A chunk of iterations performed by the robot to converge to the right posture is attributed to this communication delay.

4 Conclusions and Future Work

In this paper we presented a modified MCL algorithm for cooperative localization of robots from a team, where an object visually observed by all the team members involved in the cooperative localization is used. The algorithm takes advantage of the information on the visually shared object, provided by teammates, to modify the particle reset step when a robot determines it is lost (e.g., because it was kidnapped). The algorithm was applied to real robots in RoboCup Soccer MSL with considerable success.

The major issue with our approach is the confusing situation which can arise due to false positive identification of the shared object. A proper approach to solve it would be to use a fused information of the shared object, where the fusion algorithm can discard false positives detected by teammates. Secondly, a fast moving ball creates larger uncertainty about its position which also affects the robustness of our approach to some extent.

Future work will include testing the algorithm in more demanding situation, such as during actual games, with the robots continuously moving. Furthermore, we plan to improve the algorithm by modifying the original MCL such that a fraction of the particles is always spread over a circle algorithm, depending on the ratio between the short-term average and long-term average of their weights, instead of checking when this ratio drops below a given threshold. Other objects, such as the teammates, can also be shared to improve cooperative localization, as long as one can determine their position and track them.

Acknowledgment

This work was supported by project FCT PTDC/EEA-CRO/100692/2008 (author Aamir Ahmad) and also a Banco de Santander Chair of Excellence in Robotics grant from the U. Carlos III de Madrid (author Pedro Lima).

References

[Fenwick et al., 2002] Fenwick, J.W., Newman, P.M., Leonard, J.J.: Cooperative Concurrent Mapping and Localization. In: Proc. of the IEEE Intl. Conf. on Rob. and Autom. (2002)

[Fox et al., 2000] Fox, D., Burgard, W., Kruppa, H., Thrun, S.: A Probabilistic Approach to Collaborative Multi-Robot Localization. Autonomous Robots 8(3) (2000)

[Jennings et al., 1999] Jennings, C., Murray, D., Little, J.: Cooperative Robot Localization with Vision-Based Mapping. In: Proc. of the IEEE Intl. Conf. on Rob. and Autom., vol. 4, pp. 2659–2665 (1999)

[Lenser and Veloso, 2000] Lenser, S., Veloso, M.: Sensor Resetting Localization for Poorly Modelled Mobile Robots. In: Proc. of the IEEE Intl. Conf. on Rob. and Autom., San Francisco, CA, USA (2000)

[Roumeliotis and Bekey, 2002] Roumeliotis, S.I., Bekey, G.: Distributed Multirobot Localization. IEEE Transactions on Robotics 18(5), 781–795 (2002)

[Sanderson, 1996] Sanderson, A.C.: Cooperative Navigation Among Multiple Mobile Robots, pp. 389–400 (1996)

[Santos and Lima, 2010] Santos, J., Lima, P.: Multi-Robot Cooperative Object Localization — a Decentralized Bayesian Approach. LNCS (LNAI), vol. 2010 (2010)

[Thrun et al., 2005] Thrun, S., Burgard, W., Fox, D.: Probabilistic Robotics. MIT Press, Cambridge (2005)

[Thrun et al., 2001] Thrun, S., Fox, D., Burgard, W., Dellaert, F.: Robust Monte Carlo localization for Mobile Robots. Artificial Intelligence 128(1-2), 99–141 (2001)

[Zhou and Roumeliotis, 2008] Zhou, X.S., Roumeliotis, S.I.: Robot-to-Robot Relative Pose Estimation from Range Measurements. IEEE Transactions on Robotics 24(5), 1168–1185 (2008)