Cooperative Visual Tracking in a Team of Autonomous Mobile Robots

Walter Nisticò, Matthias Hebbel, Thorsten Kerkhof, and Christine Zarges

Institute for Robot Research
Universität Dortmund
Otto-Hahn-Str.8, 44221 Dortmund
forename.surname@uni-dortmund.de

Abstract. Robot soccer is a challenging domain for sensor fusion and object tracking techniques, due to its team oriented, fast-paced, dynamic and competitive nature. Since each robot has a limited view about the world surrounding it, the sharing of information with its teammates is often crucial in order to be ready to react to situations which might involve it in the near future. In this paper we propose a Particle Filter based approach that addresses the problem of cooperative global sensor fusion by explicitly modeling the uncertainty concerning the robots' positions, the data association about the tracked object, and the loss of information over the network.

1 Introduction

The tracking of fast moving objects has received constant attention in the context of autonomous robots interacting with highly dynamic and potentially hostile environments, and the robot soccer domain is particularly suitable to this purpose, as a robot has to interact cooperatively and competitively with a set of moving objects such as teammates, opponents and the ball.

1.1 The Platform

This work has been developed on the robot Sony Aibo ERS-7 [1], which is the only allowed hardware platform on the 4-Legged RoboCup League [2]. This limitation poses several interesting challenges for the task of object tracking, due to the limited computational resources available (576MHz MIPS CPU, 64MB of RAM) and the presence of a single exteroceptive sensor, a low-power CMOS camera with a maximum resolution of 208×160 pixel. Even the localization and tracking algorithms have to consider the running time as a serious issue, and it is highly desirable to be able to process the sensory information at the maximum rate provided by the camera, 30Hz, since this device has a very limited field of view (57° horizontal, 45° vertical) and consequently it has to be moved at high angular velocities to scan the environment. This year, the problem has been further complicated by the new rules of the league, which nearly doubled the

G. Lakemeyer et al. (Eds.): RoboCup 2006, LNAI 4434, pp. 146–157, 2007.

field size (now is 6m×4m) and removed any border or fence which would prevent the robots from observing unknown objects outside of the field. At last, legged locomotion makes it impossible to accurately know the height of the camera relative to the field, as this changes continuously as the robot walks, and the camera is mounted on the snout of the robot, which can rotate with 3 degrees of freedom; the uncertainty in the camera pose relative to the ground adds severe noise levels to the measurements.

1.2 Related Work

The most significant achievement in the field of individual tracking has been presented in [3], where the authors have used a sophisticated Rao-Blackwellised Particle Filter to efficiently model strong non linearities in the ball motion due to the interactions with the environment, such as bouncing on borders or being kicked by a robot. Cooperative object tracking is instead still in its infancy in this context, as the additional problems of the uncertainty on the robots' own positions, and the sharing of information over an unreliable network further complicate the problem. In [4] the authors have compared several sensor fusion techniques applied in the context of the RoboCup Middle-Size League, including Bayesian Filtering techniques [5], simple techniques such as arithmetic or weighted mean of percepts, and an anchoring approach. Not surprisingly, the Kalman Filter and the Particle Filter resulted to be the top performers, with the former as the solution of election due to its limited computational requirements, but it has to be noted that on this platform the uncertainty over the robot location is very small, due to the availability of omni-directional cameras and range sensors such as laser scanners. At last, in [6] has been proposed an Extended Kalman Filter based approach for global sensor fusion in the Four-Legged League, which takes into account the localization problem. However, this paper does not consider the data association problem, assuming the ball to be unique on the field, an assumption which is not true anymore since the rule changes in the league which have removed the protective fence from around the field.

2 Tracking the Ball with a Particle Filter

Even if the RoboCup rules allow only a single ball to be present on the field, there are still several situations where ambiguities may arise. The camera is the only exteroceptive sensor of this robot, and the ball is mainly recognized for its color (orange), its spherical shape (although it is frequently incomplete due to occlusions), and the fact that it lays on a green surface (the soccer field), however there are situations where objects around the field, such as clothing or shoes in the audience, can appear as valid ball candidates. Furthermore, the official red jerseys used to distinguish one team of robots have also a rounded shape, and due to limitations in the camera hardware, lighting, and blur, they can appear as potential balls. For all these reasons, we feel that a Kalman Filter based approach as described in [6] is not robust enough for our needs, as it

does not deal very well with sensor ambiguity since it cannot deal with multimodal probability distributions; consequently, we have decided to use a Particle Filter approach similar to what is described in [3], which can track multiple ball hypotheses. To avoid adding the robot localization uncertainty to the ball tracking own uncertainty, the ball position and velocity will be represented in a robot-centric reference system.

2.1 Particle Filters

The *Particle Filter* [5] is a non-parametric implementation of the general *Bayes Filter*, which is a recursive algorithm that calculates the belief or *posterior* $bel(x_t)$ at time t of the state x_t of a certain process, by integrating measurement observations z_t and control actions u_t over the belief of the state at time $t-1$. The Bayes Filter is based on the Markov assumption or *complete state* assumption which postulates the conditional independence of past and future data given the current state x_t. A Particle Filter represents an approximation of the posterior $bel(x_t)$ in the form of a set of samples randomly drawn from the posterior itself; such a representation has the advantage, compared to closed form solutions of the Bayes Filter such as the Kalman Filter [7], of being able to represent a broad range of distributions and model non-linear processes, whereas parametric representations are usually constrained to simple functions such as Gaussians. Given a set of N samples or *particles* $\Pi_t := x_t^1, x_t^2, \ldots, x_t^N$, at time t each particle represents an hypothesis of the state of the observed system; obviously, the higher the number of samples N, the better the approximation, however [8] has shown how to dynamically adjust N. An estimate of $p(x_t|u_t, x_{t-1}^i)$ is called

Algorithm 1. Particle Filter

Require: particle distribution Π_{t-1}, control action u_t, measurement observation z_t
  for $i = 1$ to N **do**
    Process update: update the particle state as the result of control action u_t: $x_t^i \propto p(x_t|u_t, x_{t-1}^i)$
    Measurement update: calculate the particle importance factors $w_t^i = p(z_t|x_t^i)$ from the latest observation
    Add $\langle x_t^i, w_t^i \rangle$ to the temporary set $\overline{\Pi}_t$
  end for
  Resampling: create Π_t from $\overline{\Pi}_t$ by drawing the particles x_t^i in a number proportional to their importance w_t^i.

Process Model, while $p(z_t|x_t^i)$ is known as *Sensor Model*. In the context of robot localization and object tracking, particle filters are often referred to as *Monte Carlo Localization* [9].

2.2 Sensor Model

Unlike other robot platforms, an important source of noise in the camera measurements is the uncertainty about the camera pose relative to the robot-centered reference system: the camera can rotate with 3 degrees of freedom, and its height relative to the ground changes dynamically as the robot walks. Furthermore, the on-board camera has only 3 shutter speed settings, with a minimum exposure time of $\frac{1}{200}s$; as a result, images are affected by blur, which gets more noticeable as the robot and camera speed increase. Lastly, the camera captures an image sequentially from the top scanline to the bottom, with a frequency of 30fps, so that a time delay exists between the top of the image and the bottom of $\approx \frac{1}{30}s$, which distorts the image and the percepts especially in case of a fast camera panning motion (for a description of the problem and a possible solution see [10]). The uncertainty about the measurement is modeled as a 2-dimensional gaussian, with one axis oriented as the distance between the robot and the ball (σ_ρ) and the other perpendicular to it ($\sigma_\perp$); the variances of such gaussian are dependent on the following factors:

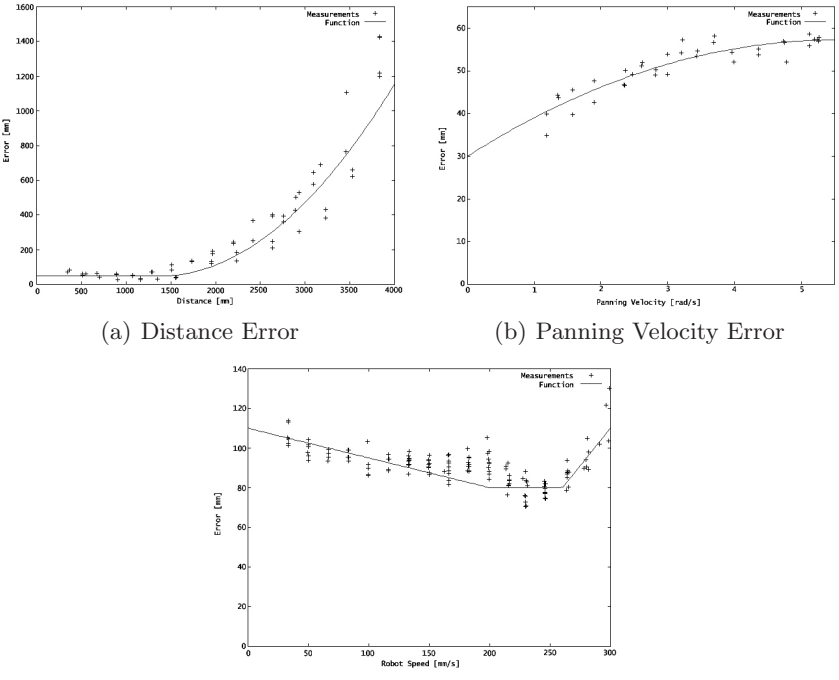
1. Percept confidence $p_c(z) \in [\frac{1}{5}, 1]$ calculated from the image processor based on several criteria used to identify the ball, such as color, shape, sharpness of the contour; its reciprocal is multiplied by both axes
2. Distance to the percept f_ρ; is proportional to σ_ρ
3. Camera panning velocity $f_{\dot{\alpha}}$; is proportional to $\sigma_\perp$
4. Robot speed f_{v_R}, which affects the amount of "head bobbing", causing motion blur and inaccuracy in the camera pose; is multiplied by both axes

In order to estimate f_ρ, $f_{\dot{\alpha}}$ and f_{v_R}, we used an external camera[1] mounted on the ceiling above the soccer field to compare the robot's own measurements with the true ball position. Based on the observed data, we have modeled f_ρ and $f_{\dot{\alpha}}$ as second order polynomials, while for f_{v_R} we have chosen a piecewise linear approximation; the results have been represented in Figure 1. In case of several objects in the image which might look like a ball, the vision system provides a list of candidate ball hypotheses, each one with a certain percept confidence $p_c(z)$, and they are all used to perform the measurement update of the particle filter.

2.3 Process Model

Since the ball is tracked in a robot-centric reference system, the robot's own motion is an apparent speed relative to the ball. At each time instant, an estimate of the robot motion is represented by the odometry vector: $o_t = \left[o^t_{vx}\ o^t_{vy}\ \omega_t \right]$ where (o^t_{vx}, o^t_{vy}) is the robot translation speed and ω_t is its angular velocity at time t. In addition, we use a *constant speed model*, which propagates the position of the ball $\overrightarrow{s}(t)$ at time t based on the speed $\overrightarrow{v}(t)$ at time $t-1$; this because

[1] The average measurement error of such a vision system is below 0.5cm.



(a) Distance Error

(b) Panning Velocity Error

(c) Robot Speed Error

Fig. 1. Ball measurement error functions learned from experimental data

accelerations are very difficult to measure with highly noisy sensors like this camera. Consequently, the time-update is performed as follows[2]:

$$\begin{bmatrix} s_x^t \\ s_y^t \\ v_x^t \\ v_y^t \end{bmatrix} = \begin{bmatrix} \cos(\omega_t \Delta t) & -\sin(\omega_t \Delta t) & \Delta t & 0 \\ \sin(\omega_t \Delta t) & \cos(\omega_t \Delta t) & 0 & \Delta t \\ 0 & 0 & \cos(\omega_t \Delta t) & -\sin(\omega_t \Delta t) \\ 0 & 0 & \sin(\omega_t \Delta t) & \cos(\omega_t \Delta t) \end{bmatrix} \cdot \begin{bmatrix} s_x^{t-1} \\ s_y^{t-1} \\ v_x^{t-1} \\ v_y^{t-1} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ o_{vx}^t \\ o_{vy}^t \end{bmatrix} \tag{1}$$

Since the constant speed model is only an approximation and the odometry vector is noisy itself, in the time update we add to the probability distribution a constant amount of gaussian noise ($\sigma = 0.25$ empirically derived).

3 Multi-robot Tracking

As can be seen in Figure 1(a), the measurement error grows very quickly with the distance to the ball (error $\approx$ 1m at a distance of 3.5m), making it very difficult to track a ball which rolls in the opposite side of the field. Further, it

[2] In this equation the sign of the odometry vector has to be reversed, here the negative sign is omitted to keep compact the graphical representation.

happens very frequently that one or more robots in a team have the line of sight to the ball completely occluded by teammates or opponents, but still they need to know where the ball is in order to position themselves strategically on the field. At last, it is desirable that the robots share a common estimate about the ball position, so that they can coherently take strategical decisions such as who should go to the ball and try to get in control of it. Thereto, merging the sensor information of all the robots in a team can often result in a better estimate than what would be possible for each robot alone. However, the cooperative tracking of an object must be performed in a global coordinate system, and this severely complicates the problem compared to local sensor fusion techniques, since the uncertainty on the robots' positions adds to the ball measurement errors. For example, it happens that a robot following the ball focuses its attention for too long on the ground, without looking at global landmarks: since field features such as the field lines, the kick-off circle and the line crossings are symmetrical on the field, the robot location probability distribution tends to concentrate around symmetric modes, and consequently the localization state can jump frequently from a peak to another. When the ball probability distribution of that robot is transformed in global coordinates using the current robot pose, it will jump along with it providing contradictory information to the whole team estimate of the ball position and this is not desirable, especially if such robot is very close to the ball and has a good view of it. For the robot localization, we have implemented a *Monte Carlo* algorithm similar to what is described in [11], with a sample set of 100 particles.

3.1 Multi-robot Belief Merging

Since each particle in the self locator represents a candidate pose, ideally each robot should calculate the global position of its ball probability distribution **for each particle of the self locator**: with 40 particles to represent the (robot centric) ball probability $\pi^l_B(i), i \in [0, 40)$, and 100 particles for the localization probability $\pi_{SL}(i), i \in [0, 100)$, the final set of ball particles in the global reference system would be 4000 particles per robot $\pi^g_B(i), i \in [0, 4000)$. The probability of each ball particle should be also multiplied by the probability of the corresponding localization particle:

$$p(\pi^g_B(i \cdot j)) = p(\pi^l_B(i)) \cdot p(\pi_{SL}(j)) \tag{2}$$

Then such particles have to be sent to the teammates, merged with their particles, and clustered to find the expected "team ball" position. Algorithm 2 can deal with situations where the localization distribution of some robot presents strong ambiguities, because such ambiguities can be resolved by the information provided by the teammates. The main problem with this approach is the huge amount of particles that have to be computed and sent over the network: if each particle is represented by the values $\left[s_x \; s_y \; p(\pi^g_B(i)) \right]$ and each value is stored in 32 bit precision, even when using broadcasts we would still use 1.5Mbit of traffic per iteration of the algorithm just for particle data. An alternative to

Algorithm 2. "Naive" Merging

for all i, j such that $i \in [0, 40), j \in [0, 100)$ **do**
  calculate the position of $\pi_B^g(i)$
  calculate $p(\pi_B^g(i))$
end for
send $\pi_B^g(i)$ to teammates
receive $\pi_B^g(i)$ from teammates
cluster the joint particle set and compute the expected ball position

save on network traffic would be to send $\pi_B^l(i)$ and $\pi_{SL}(j)$ and multiply the two sets at the destination; however this would result in even greater computational costs, and our platform is already not suitable to process particle sets of such dimensions.

3.2 Reducing the Joint Particle Set Size

In most game situations, the particle distribution is not spread uniformly accross the field (this normally happens only when the robot is placed on the field for the first time) but it is concentrated in a very limited number of clusters. This is because low probability particles are replaced with new samples in the positions calculated from the latest observation, following the sensor-resetting [12] / Mixture Monte Carlo idea [13], so even when the robot is teleported or "kidnapped" by the referee, a new cluster forms very quickly at the new position of the robot. In our experiments, in a typical match over 90% of the total probability is concentrated in at most 3 clusters, so we calculate the 3 robot pose hypotheses with the highest probabilities to generate the ball distribution in global coordinates. Further, to keep the running time and network traffic low, we subsample the ball global probability distribution to obtain up to 12 *"representative particles"* out of the set of 120. Since each robot provides at most 12 particles to his teammates, the global ball estimate is calculated out of 48 particles as the cluster with the highest probability. To efficiently calculate the representative particles, the soccer field is recursively split into cells to form a quad-tree, with a maximum depth $\delta = 7$:

- *Basic Cell*: a cell which contains one particle or none
- *Composite Cell*: a cell which contains 4 Basic Cells

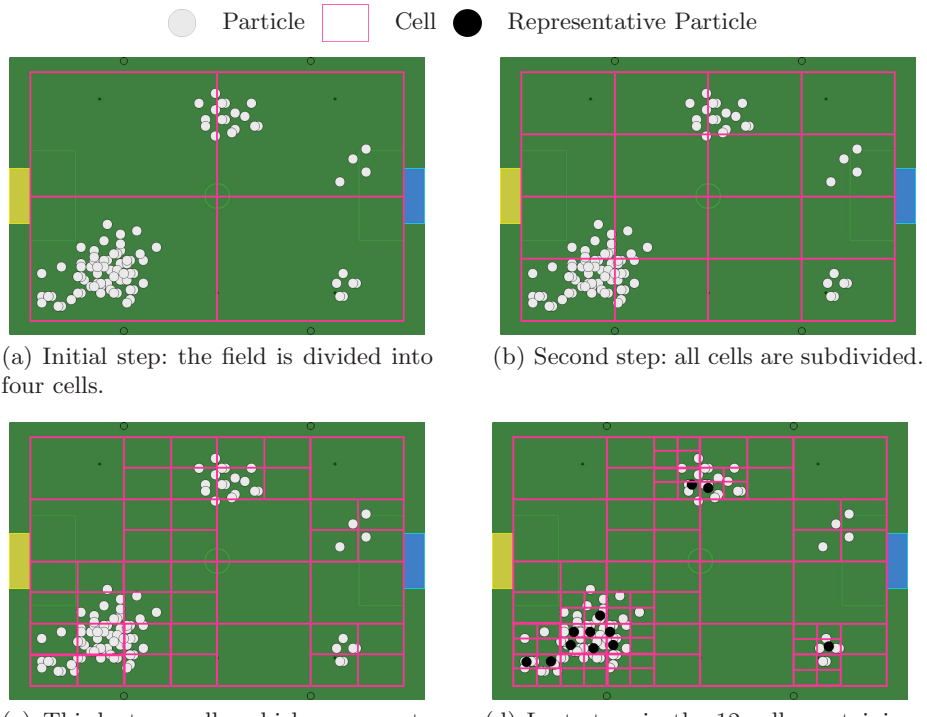
While it might appear that 12 particles are too few to represent the belief of a single robot about the ball position, it has to be noted that Algorithm 3 is applied to the particle set *before* its normalization / resampling. As such, a small number of particles can carry the same amount of information of a much larger set after the normalization, because such a process replaces high probability particles with several copies having all the same importance factor.

Algorithm 3. Representative Particles Computation

1. The whole field is initialized as a Composite Cell
2. **if** a Basic Cell contains more than one particle
   - transform it in a Composite Cell by subdividing it into 4 Basic Cells
   - particles are inserted into each new Basic Cell depending on their position on the field
3. apply recursively step 2 until a maximum depth $\delta = 7$ is reached
4. representative particles are generated out of the 12 cells which contain the highest probability; if a chosen cell contains more than one particle, the representative particle position is calculated as a weighted average of the particles there contained

3.3 Loss of Information over the Network

To keep network utilization and latency to low levels, in our system a robot exchanges data with his teammates through UDP broadcasts [14]. However, UDP does not guarantee that the packets will reach their destination, and it is quite common to have network performance problems in crowded places or at the competition sites, since 802.11 networks are now so widely popular. Since our ball tracking runs at a fairly high rate (the same as the vision system, 30Hz), it is not so unlikely that for a frame or two no particle is received from a certain teammate. In such unfortunate case, it is wiser to use older information from the corresponding robot instead of immediately discarding all the particles of the previous iteration, since in such a short interval of time (up to 100ms) the ball state cannot change too much. Therefore the current implementation of our ball tracker stores the particles received from all teammates. If in the new frame, at least one particle is received from a certain robot, all its old particles are discarded and substituted by the new ones. Otherwise, the old particles can be used, but random noise has to be added to reflect the increased uncertainty due to the unmodeled ball motion, and their reliability has to be lowered. That is achieved by replacing all old particles with two new ones, each new particle carrying half of the original probability. If this results in more than 12 particles for that particular robot, only the 12 particles with the highest validities are retained. Afterwards all particles are spread by the addition to their position of gaussian noise, to represent a probabilistic search around their original position, since the direction of movement of the ball is unknown. The standard deviation of such noise is a function of the number of frames where no particles were received from the teammate. Finally, the validity of these particles is decreased by a factor which is also a function of their age. In our tests, this approach has worked better than propagating the old state by using the speed, because the speed estimation is very noisy in itself, the ball motion is often non-linear, and the constant-speed model is a valid approximation only for very short periods of time.



(a) Initial step: the field is divided into four cells.

(b) Second step: all cells are subdivided.

(c) Third step: cells which are empty or contain only a single particle are not further subdivided. The other are again subdivided.

(d) Last step: in the 12 cells containing the highest probability, a representative particle is calculated.

Fig. 2. Visualization of Algorithm 3 with an example

4 Experiments and Results

Finding a suitable reference system to compare against our proposed approach has not been an easy task, as global sensor fusion is still in its infancy on our development platform, and results from other leagues such as [4] cannot be directly compared, due to the vast difference of computational resources and sensor capabilities. A good candidate for our comparison has been found in [15], since this approach is adopted by 5 different teams on this hardware platform and its source code is publicly available.

4.1 Reference System

The approach described makes use of a Kalman Filter [7] to track the ball position and velocity in a robot-centric reference system. The robots in the team exchange their localization and local ball estimates, and the global team ball is calculated from the information provided by the robot with the highest confidence in its own localization.

4.2 Experimental Results

Our experiments have been performed by running in parallel on the same robots the reference system and our new solution, processing exactly the same data. The results of both systems are compared with the ground truth obtained from a ceiling camera global vision system as described in Section 2.2. In all of the following scenarios, the environment surrounding the soccer field is unstructured and unknown, and the robots might incorrectly identify false landmarks and false balls in it.

Scenario 1. In our first test scenario, 4 robots are placed on the field, without opponents or obstacles which might occlude their sight. Their vision systems are perfectly calibrated for the lighting conditions, and all the robots move freely on the field. The "observing robot" can never see the ball; the other 3 can, but they also have to periodically distract their attention from it in order to localize themselves. This test represents a "best case" scenario to evaluate the performance in a condition where the sensor information is relatively accurate and reliable. The results are shown in Figure 3(a).

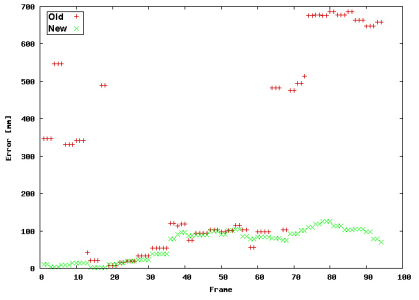
Scenario 2. In this scenario, the conditions are similar to the first test, but one of the robots which can see the ball has a problem in its vision system, so that it consistently detects "ghost balls" inside the yellow goal. Such a problem is not infrequent during the competitions, and can severely penalize the performance of the whole team. The results are shown in Figure 3(b).

Scenario 3. This scenario is based on a real game situation. All robots are free to move and look at the ball, but the presence of opponents can occlude their sight to the ball and to the landmarks. Even worse, the opponents struggle against the observing team for getting control over the ball, compromising the localization state of the robots, as such collisions cannot be detected since the robots do not have any range or contact sensors. The results are shown in Figure 3(c).

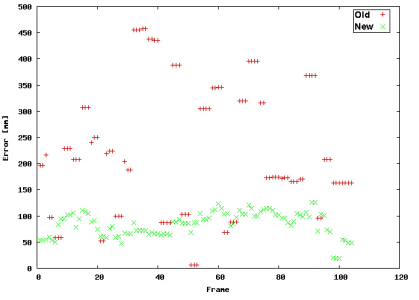
Scenario 4. This test is similar to the previous scenario, but the frequency and entity of the collisions is greater. The results are shown in Figure 3(d).

4.3 Performance

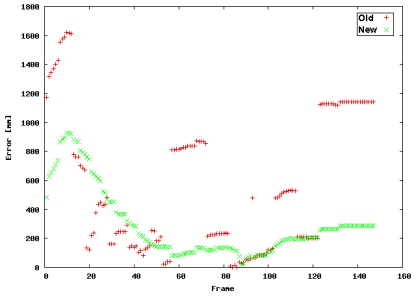
Our goal was to be able to process all the sensory information at the native camera frame rate of 30Hz, this because of the limited field of view of the camera, which forces the robot to look around continuously at high angular velocities. The tracking system composed of the individual robot-centric tracker and the global tracker requires about $\approx$ 1ms to execute, being on par in terms of run-time with Kalman Filter based approaches, and many times faster than other Particle Filter based implementations. On average, each robot broadcasts 5 particles per frame, 12 bytes per particle, 30 times per second, for a total network traffic (team of 4 robots) of $\approx$ 56Kbit/s, while in the worst case, this value reaches



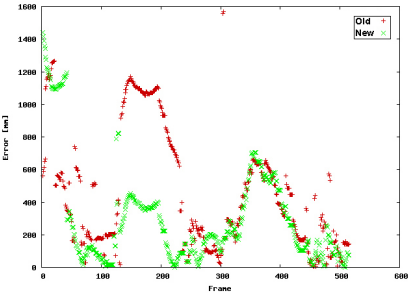
(a) Scenario 1: the x axis of the graph represents the temporal axis, where 30 frames = 1s. The old system performs particularly poorly after frame 70



(b) Scenario 2: the old system cannot cope with the ghost balls, and is consistently outperformed



(c) Scenario 3: in the beginning of the test there is a strong collision. The new system performs constantly better.



(d) Scenario 4: several collisions, around frame 150 the new system is 3 times more accurate than the reference

Fig. 3. Test scenarios

135Kbit/s; this is compatible with the competition constraints, which limit to 512Kbit/s the total bandwidth available to a team that has to be used also for other communication tasks such as role assignments and strategical data exchange.

5 Conclusion

It has been presented a Particle Filter based approach that tackles the problem of global sensor fusion in presence of high uncertainty concerning the robot positions, the data association about the tracked object, and the loss of information over the network. The system meets all the performance constraints set by the platform, and is competitive in terms of running time with simpler approaches which do not deal with all the aforementioned problems. In the future, we plan to investigate the possibility to make use of the speed information in the global

tracker, by building a better sensor model with the help of our ceiling camera application and machine-learning techniques.

Acknowledgment

We would like to thank Microsoft MSDNAA for their support which made it possible for us to take part in the fascinating world of RoboCup.

References

1. Sony Aibo ERS-7: (2005), http://www.aibo.com
2. The RoboCup Federation: In: Bredenfeld, A., Jacoff, A., Noda, I., Takahashi, Y. (eds.) RoboCup 2005. LNCS (LNAI), vol. 4020, Springer, Heidelberg (2006), http://www.robocup.org
3. Kwok, C., Fox, D.: Map-based Multiple Model Tracking of a Moving Object. In: Nardi, D., Riedmiller, M., Sammut, C., Santos-Victor, J. (eds.) RoboCup 2004. LNCS (LNAI), vol. 3276, Springer, Heidelberg (2005)
4. Ferrein, A., Hermanns, L., Lakemeyer, G.: Comparing Sensor Fusion Techniques for Ball Position Estimation. In: Bredenfeld, A., Jacoff, A., Noda, I., Takahashi, Y. (eds.) RoboCup 2005. LNCS (LNAI), vol. 4020, Springer, Heidelberg (2006)
5. Fox, D., Hightower, J., Liao, L., Schulz, D., Borriello, G.: Bayesian Filtering for Location Estimation. PERVASIVE computing, 10–19 (2003)
6. Karol, A., Williams, M.A.: Distributed Sensor Fusion for Object Tracking. In: Bredenfeld, A., Jacoff, A., Noda, I., Takahashi, Y. (eds.) RoboCup 2005. LNCS (LNAI), vol. 4020, Springer, Heidelberg (2006)
7. Kalman, R.E.: A new approach to linear filtering and prediction problems. Transactions of the ASME - Journal of Basic Engineering 82, 35–45 (1960)
8. Fox, D.: Adapting the sample size in particle filters through kld-sampling. I. J. Robotic Res. 22(12), 985–1004 (2003)
9. Fox, D., Burgard, W., Dellaert, F., Thrun, S.: Monte carlo localization: Efficient position estimation for mobile robots. In: Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI'99) (1999)
10. Nistico, W., Röfer, T.: Improving percept reliability in the Sony Four-Legged League. In: Bredenfeld, A., Jacoff, A., Noda, I., Takahashi, Y. (eds.) RoboCup 2005. LNCS (LNAI), vol. 4020, Springer, Heidelberg (2006)
11. Röfer, T., Jüngel, M.: Fast and robust edge-based localization in the sony four-legged robot league. In: Polani, D., Browning, B., Bonarini, A., Yoshida, K. (eds.) RoboCup 2003. LNCS (LNAI), vol. 3020, Springer, Heidelberg (2004)
12. Lenser, S., Veloso, M.: Sensor resetting localization for poorly modeled mobile robots. In: Proc. of the IEEE International Conference on Robotics and Automation (ICRA), IEEE Computer Society Press, Los Alamitos (2002)
13. Thrun, S., Fox, D., Burgard, W.: Monte carlo localization with mixture proposal distribution. In: Proc. of the National Conference on Artificial Intelligence, pp. 859–865 (2000)
14. Postel, J.: RFC 768 - User Datagram Protocol (1980), http://www.ietf.org/rfc/rfc768.txt
15. Röfer, T., et al.: German Team RoboCup 2004. Technical report (2004) Available online: http://www.germanteam.org/GT2004.pdf