Omnidirectional Distributed Vision System for a Team of Heterogeneous Robots

Emanuele Menegatti*, Alberto Scarpa, Dario Massarin, Enrico Ros and Enrico Pagello†

Intelligent Autonomous Systems Laboratory
Department of Information Engineering
The University of Padua, Italy

†*also with* Institute of Biomedical Engineering
of the National Research Council (ISIB-CNR) Padua , Italy

*emg@dei.unipd.it

Abstract

This paper presents a system designed to cooperatively track and share the information about moving objects using a multi-robot team. Every robot of the team is fitted with a different omnidirectional vision system running at different frame rates. The information gathered from every robot is broadcast to all the other robots and every robot fuses its own measurements with the information received from the teammates, building its own "vision of the world". The cooperation of the vision sensors enhances the capabilities of the single vision sensor. This work was implemented in the RoboCup domain, using our team of heterogeneous robot, but the approach is very general and can be used in any application where a team of robot has to track multiple objects. The system is designed to work with heterogeneous vision systems both in the camera design and in computational resources. Experiments in real game scenarios are presented.

Figure 1: A picture taken during the laboratory experiments to test the Omnidirectional Distributed Vision System.

1 Introduction

To perform a task, an agent needs as much information as possible about the environment. Unfortunately, every robotic agent or human agent, has a limited sensorial horizon, i.e. it can perceive the environment up to a certain limit and beyond that limit it can only make hypothesis. If the environment is static the agent can analyse the environment by parts and store in a sort of memory the acquired information [6]. If the environment is dynamic, this approach does not work anymore, because the information that can be retrieved in the memory of the agent is not up-to-date. This is one of the reasons why in dynamic environments mobile robots are more and more fitted with omnidirectional vision systems [15]. These systems provide in one shot a complete view of the surrounding of the robot. Nevertheless, in highly dynamic environment, as the RoboCup soccer fields, this is not enough. In fact, omnidirectional vision sensors solve the problem of sensing at once the surrounding of the robot, but does not solve the problem of perceiving occluded objects or very distant objects[1]. If the robot is part of a multi-robot team, the sensorial horizon of the single robot can be extended by using the information perceived by the teammates.

The work presented in this paper consists in the implementation of an **Omnidirectional Distributed Vision System** (ODVS). The purpose of this ODVS is to track moving objects in a highly dynamic environment by sharing the

[1]Note that usually, the effective range for omnidirectional sensors is shorter then for perspective cameras due to the lower resolution.

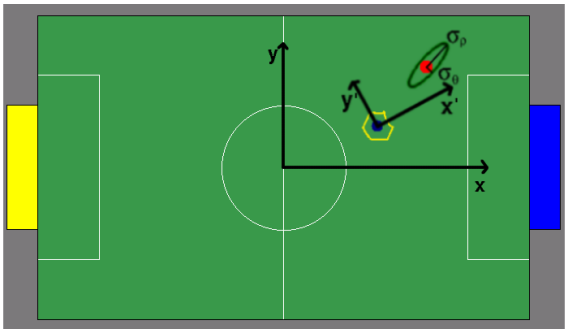

Figure 2: The reference frame of the single robot $x'y'$ and the common reference frame of the robots XY , i.e. the reference frame of the field of play. The ellipse represents the distribution of probability of the single measurement.

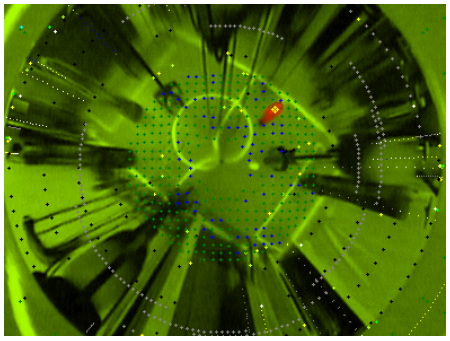

Figure 3: An omnidirectional image processed by the Vision Agent of the perception module. Note the ball has been detected.

information gathered by every single robot. The testbed we choose is the Middle-Size competition of RoboCup[2]. The amount of information shared between the robots of our team increased year after year. Step by step, we introduced information sharing for cooperation, planning and dynamic role assignment [11]. With this work, we introduce information-sharing also in the sensing process. This work was inspired by previous works on cooperative sensing, mainly by the works of Stroupe et al. [14] (that used perspective cameras) and Gutmann et al. [2] (that used perspective cameras and laser range finders). Beside the fact of using omnidirectional vision systems, the major extensions of our work with respect to the work of Stroupe are that: we modified her approach in order to account for observations made at different instants in time and we took into account the speed of the objects to be tracked. With respect to the work presented in [2], the biggest difference is that they use an external computer running what they called a *Global Sensor Integrator* (GSI). This GSI receives the observations made by the different robots, integrates them elaborating a merged vision of the world and send it back to every single robot. Every single robot uses the information received by the Global Sensor Integrator only to locate objects out of its field of view. On the contrary in our system, every robot is fusing, without the need of an external computer, the information coming from the teammates. At the same time, these measures are used to improve the measures made by the robot itself.

The Omnidirectional Distributed Vision Systems, proposed in this paper, could be applied in more general situations than soccer robotics, for instance in systems of surveillance or intelligent space applications. Every time the application requires the monitoring of a large area that cannot be framed in the field of view of a single sensor, the cooperation of different sensors becomes extremely useful. Some examples are: the work of Ishiguro in which a Distributed Vision System composed of perspective cameras is able to drive a robot through a toy-scale model of a town [3], or the work of Nakazawa where multiple perspective cameras can track people moving from one room to another [10], or the work of Lee and Hashimoto in which the Distributed Vision System is able to support the activity of robot and humans [5]. Another interesting work on Distributed Vision System is the work of Ishiguro [13]. Differently from our approach, the problem there is to solve the N-stereo correspondence problem with syncronised homogeneous sensors at precisely known locations and orientations. We are experimenting with Omnidirectional Distributed Vision Systems also within other projects. For instance, we are building an ODVS composed of a network of omnidirectional vision systems able to autonomously learn how to control a mobile service robot and drive it from the field of view of one sensor to the field of view of next sensor [9].

This paper is organised as follows: in Section 2 we explain how the omnidirectional images are processed by the perception module on board of the single robot and we illustrate how we calculated the variance associated to every measure. In Section 3, we explain how different measures are fused to obtain a more accurate estimation for the object position and we detail the Kalman filtering technique adopted. In Section 3.1, we highlight the problems that need to be solved when fusing informations coming from different robots. In Section 4, the realistic experiments performed on the field of play with the real robots are presented.

2 Single Sensor Observation

Every robot of the team is fitted with a catadioptric omnidirectional vision system [4]. Every omnidirectional sensor mounts a mirror with a different profile especially tailored

[2]URL: http://www.robocup.org

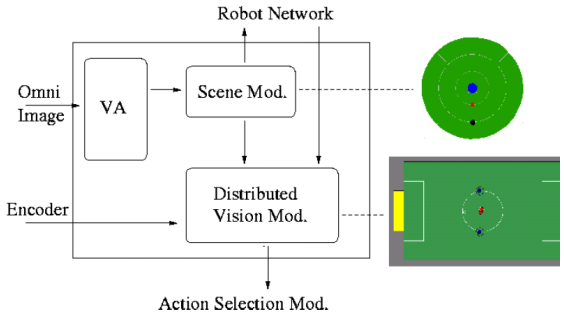

Figure 4: A sketch of the *Perception Module* inside every single robot. On the right the two graphical representation of the information stored in the *Scene Module* (top) and in the *Distributed Vision Module* (bottom).

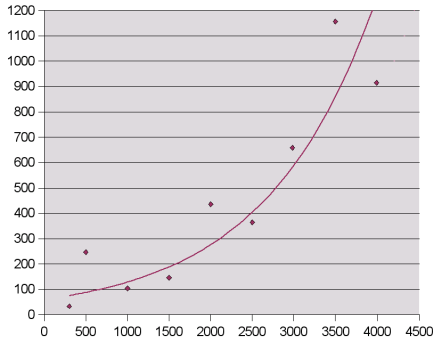

Figure 5: The plot of the measured variances for Robot 1 in the measurements of the distance object-robot vs. the different distances (in mm).

for the task of the robot [7, 8]. The omnidirectional vision sensor is calibrated in order to be able to transform positions in the image to positions in the real world. The instantaneous positions of the objects to be tracked, relative to the robot's frame of reference, is determined assuming the objects lie on the floor. In Fig. 4, we sketched the **Perception Module** implemented inside our robots. The omnidirectional image is the input, on the left, of the image processing module, called **VA Module** (Vision Agent Module). The result of the image processing is sent to the so called **Scene Module**. Here is stored a description of the scene (i.e. the position of the objects of interest) as seen from the sensor of the robot (see radar-like image on the right). The measurements on the positions of the objects are then passed to the **Distributed Vision Module** (DV), discussed in detail in Section 3, and broadcast to the other robots.

A two dimensional Gaussian distribution is associated to every measurement, see Fig. 2. The centroid of the Gaussian is located at the estimated object position. The widths of the Gaussian along the principal axes (σ_r, σ_θ) correspond to the uncertainty of the observation along those axes. The height of the Gaussian at a certain point represents the probability that the object is actually located at that point given that measurement. Every measure is made in the reference frame of the robot and is then transformed in the frame of reference of the field of play. This assumes that the robot perfectly knows its pose in the environment while it moves (i.e. in the field of play). The self-localisation algorithm used was developed within our team of RoboCup [12]. Due to the robustness of the self-localisation algorithm, the assumption of an error free localisation is sensible, even when the robots move in the field of play. The remaining localisation error is taken into account by overestimating the error associated to the single measurements. We determined experimentally the width of the Gaussian along the two major axes, i.e. along the radial direction robot-object (i.e. σ_r) and along the line orthogonal to this direction (i.e. σ_θ). We placed one of the object to the tracked[3] at different known distances from the robot and, for every position, we calculated 1000 times from 1000 omnidirectional images the distance robot-ball and the azimuth of the ball. We extracted the variance σ_r for every set of 1000 measures. The plot representing the measured variance σ_r for Robot 1 is displayed in Fig. 5. The data of the variance have been interpolated with the exponential curve of Eq. 1 and we got a function to associate to every measurement the correct variance along the radial axis[4]. This procedure was repeated for every different robot, because they mount heterogeneous vision systems and they provide measurements with different accuracies.

$$y(x) = kx^a + q \tag{1}$$

Only the plot of the data about the distance object-robot is displayed, because the variance on the azimuth resulted to be so small that one could assume a zero error on azimuth. The zero value is not a valid one, because this would result in a degenerated Gaussian distribution, so we assumed a certain non-zero variance increasing with the distance from the robot. This will also take into account the errors introduced by a non-perfect localisation of the robot.

One has to remember the objects observed by the robots are moving, they are not static. We need to associate at every measurement a precise time stamp indicating when the measurement was made. One cannot make any assumption on the time interval between two measurements. In fact, we are working with robots with very different computational power, working at different frame rates. So, we associate to

[3] In the first experiments, this was the orange soccer ball of Fig. 1.

[4] For Robot 1 the constants of Eq. 1 are $k = 0.0000009$, $a = 2.52$, $q = 90mm$.

every measurement the time stamp of the omnidirectional image processed to extract that measurement. One cannot make any assumption also for measurements made by the same robot. In fact the measurements are not granted to arrive at regular interval. This is because, to fully exploit the low computational resources of our robots, we use a thread-scheduling system which allows a certain flexibility to the execution time of the threads [1], so the measurements are made available at different time intervals. The measurement made on the position and speed of one particular object are not considered independent, they are fused in a track that is used to increase the robustness of the observations.

3 Fusing Multiple Observations

The measures of the position and speed of the tracked objects can come from two sources: the repeated observations of the single robot or the observations of other robots. In our system, all these information are processed in the same way. In the work of Stroupe et al. [14], they do not take into account the dynamics of the objects to be tracked and they assume that the objects are instantaneously steady. They use a minimal variance estimation approach to fuse the measurements of the different robots assuming the measurements are made at the same instant. In the real world, measurements of different robots are never made at the same time and because the objects are moving, every robot will measure the object when it is in a different position. Stroupe solved the problem by throwing away measurements to distant in time to be compatible. On the contrary, the solution we adopted is similar to the one proposed in [2], with the main difference that they used the external *Global Sensor Integrator* to fuse the information, while in our implementation, every robot is fusing the received measures. Every time a new measurement is received, independently if it comes from the robot it-self or from another robot, this is compared with the existing tracks of the objects. If this is compatible with an existing track, the measurement is added to the track, otherwise a new track is initialised. In order to test the compatibility of a new measurement with a track, because the objects are moving, we need to predict the position of the track at the instant in which was taken the new measurement. The prediction is made using the last position registered for the track and the speed associated to it. The objects are assumed to move at constant speed from the last observed position. The state of an object at the time t_k is described by a Gaussian distribution with mean $\vec{x}$ and variance $\Sigma_{\vec{x}}$.

$$\vec{x}(t_k|t_k) = \begin{pmatrix} x \\ y \end{pmatrix} \qquad \Sigma_{\vec{x}(t_k|t_k)} = \begin{pmatrix} \sigma_r & 0 \\ 0 & \sigma_\vartheta \end{pmatrix} \quad (2)$$

So, given the Gaussian variable $\vec{x}(t_k|t_k)$ representing the estimate of the object's position at the instant t_k given observations up to the instant t_k, when we receive a new observation $\vec{y}(t_{k+1})$ at the instant t_{k+1}, we can predict the state of the system at the instant t_{k+1} with Eq. 3 and Eq. 4:

$$\vec{x}(t_{k+1}|t_k) = \vec{x}(t_k|t_k) + \vec{v}(t_{k+1} - t_k) \quad (3)$$

$$\Sigma_{\vec{x}(t_{k+1}|t_k)} = F(t_{k+1} - t_k)\Sigma_{\vec{x}(t_k|t_k)}F(t_{k+1} - t_k)^T \quad (4)$$

where F is a matrix whose coefficients depend on the time interval between the two measurements. We consider that the new measure is compatible with the track if the measure is within one standard deviation from the mean $\vec{x}$ of the track. As we said, if the measurement is not compatible, a new track is initialised, otherwise we fuse it with the track using the formulas of Eq. 5 for the mean and of Eq. 6 for the covariance (Fig. 6).

$$\begin{aligned} \vec{x}(t_{k+1}|t_{k+1}) = \vec{x}(t_{k+1}|t_k) + \Sigma_{\vec{x}(t_{k+1}|t_k)} \\ [\Sigma_{\vec{x}(t_{k+1}|t_k)} + \Sigma_{\vec{y}(t_{k+1})}]^{-1}(\vec{y}(t_{k+1}) - \vec{x}(t_{k+1}|t_k)) \end{aligned} \quad (5)$$

$$\begin{aligned} \Sigma_{\vec{x}_{k+1}|t_{k+1})} = \Sigma_{\vec{x}(t_{k+1}|t_k)} - \Sigma_{\vec{x}(t_{k+1}|t_k)} \\ [\Sigma_{\vec{x}(t_{k+1}|t_k)} + \Sigma_{\vec{y}(t_{k+1})}]^{-1}\Sigma_{\vec{x}(t_{k+1}|t_k)} \end{aligned} \quad (6)$$

This is the classical Kalman filtering in which every new data that is coming can reduce the variance of the Gaussian distribution, reducing the uncertainty on the position of the object, as detailed in [14]. In the classical Kalman filtering approach is implicit that measurements arrive one after the other. In Section 3.1, we will see how we modified this classical approach to take into account the fact that measurements come from different robots and that we want to be able to accept measurements older than the last one (usually, only newer measurements are taken in account).

When a track is not updated, the associated variance increases in time. When the variance of a track is too high, the track is considered no longer valid and it is deleted. This means that, if none of the robot can see the tracked object, the system presumes the object is still in the last position where it was seen, but the uncertainty on its position increases. When the uncertainty on the position grows over a certain threshold the information carried by the track is considered useless and the old track is deleted.

This approach also allows to store multiple tracks for a single object, in other words to have multi-modal distributions for every object. The real position of the object is decided to be the one with smallest variance, i.e. the one with smaller uncertainty. Let us see how this strategy works in a real game scenario. During a match, it may happen that the ball is kidnapped by the referee, e.g. in ball-stuck situations. The referee lifts the ball from the field of play, shielding it to the view of the robots and put the ball down

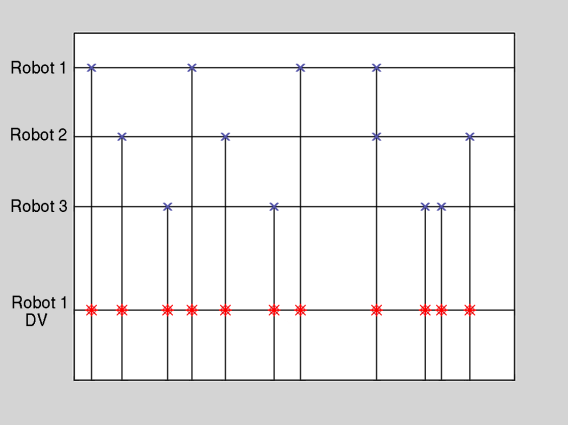

Figure 6: A conceptual sketch of the fusion of measures coming from different robot in the Distributed Vision Module (DV) of Robot1. Note that Robot 1 is measuring the object s position only 4 times in this time interval and the object's position is updated 11 times.

again in a different point. In this case the track of the ball behaves as follow: when the ball is lifted from ground, the ball track is no longer updated, its variance starts to grow, but this is the only track of the ball available to the robot, so the robot will move toward it. When the ball is again put on the floor, a second track is initialised. The variance of this new track is smaller than the previous one. The new track having smaller variance is chosen as the true position of the ball, see Fig. 10. Once the variance of the old track grows over the set threshold, the track is deleted[5].

This system is designed to be totally independent from the number of robot active on the field. Every robot uses all the measurement available, independently from the number of the teammates. This is intended in order to be robust to failures of the single robots. Again, this is the reason why every robot is equal to the others. If we had a master or an external computer governing the fusion of the information a failure on this master will cause the collapse of all the system, in this case we implemented a real distributed vision system.

3.1 Fusing Observations from Different Robots

Fig. 6 shows the process of fusing measurements coming from different robots. Every time that the DV Module (Distributed Vision Module) of Robot 1 receives a new measure from Robot 2 or Robot 3, it fuses this measure with its own

[5]If there were two balls in the field of play, every robot will consider only the ball with the smallest variance, so probably every robot will move toward the closest ball. This never happens in real game situations.

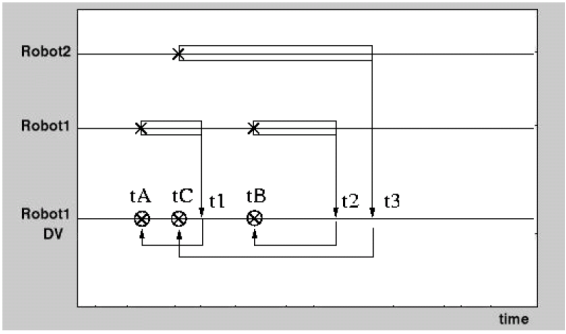

Figure 7: A conceptual time diagram showing the process of fusion of measures older than the last one received.

measure. In Fig. 6, Robot 1 is measuring the position of the ball only 4 times, but the ball position estimate is updated 11 times in its DV Module. This means the robot has a more reliable world model, but when fusing measurement from different robots, there are several problems to solve.

The first problem is that, in order to combine the different observations, all the robot must share the same spatio-temporal frame of reference. In fact, it is not enough to be able to refer all the measurement made in the frame of reference of the single robot to the common frame of reference of the field, the robots need also to be able to refer the time stamp associated to every measurement to a common frame of reference in time. In other words, the internal clock of the robots need to be precisely synchronised in order to know the time relation between the different measurements. The problem of the synchronisation of the internal clocks of the single robots was solved using the Network Time Protocol, developed by the Network Time Protocol Project[6]. Anyone of the robot can act as a server for synchronising the clocks of the other.

A second problem is that when an agent is cooperating with other agents, it needs to trust the other agents. The amount of confidence in the measurements of the others influences the amount of cooperation. In the previous section, we said the robot processes in the same way the measurements coming from the internal Scene Module and the measurements coming from the other robots. Actually, this is not totally true. Every robot weights more its own measurements, i.e. it is more confident on its measurements than on measurements received by the others. This is done by dubbing the variances associated to the measurements received from the teammates. From Eq. 5 follows, that in the combined estimation, the weight of the internal measurement is bigger than the ones of the external measurements.

A third problem emerging when working with heterogeneous vision system running at different speeds is that the

[6]URL: http://www.eecis.udel.edu/∼ntp

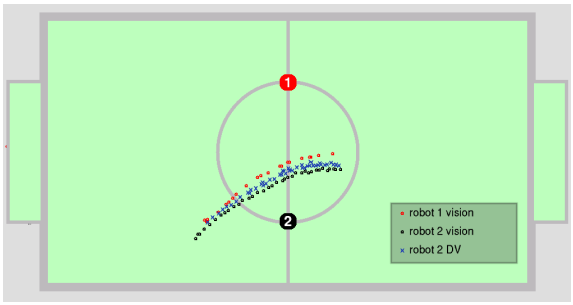

Figure 8: A screenshot of the Omnidirectional Distributed Vision System visualisation software with two robots steady and the ball moving.

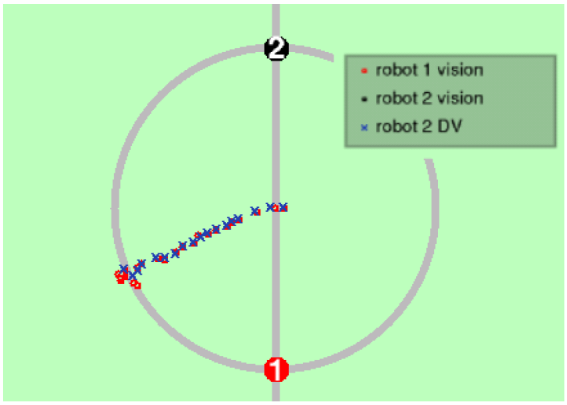

Figure 9: In this experiment Robot 2 cannot see the ball. Nevertheless, it is able to locate it using the measures received by Robot 1.

measurements arrives at different instants in time. We already explained the solution adopted to project in the future the state of the system in order to combine the old state of the system with the new observations, but sometimes we need to project the state of the system in the past. This mights occur when, the robot receives from another robot a measurement older than its own measurement. This older measurement cannot just be thrown away. Many times, it is carrying useful information even if it is old. For a practical example, consider the situation of a robot with a very slow vision system, but close to the object, which reports very accurate measurements. These measures can improve the estimation of a robot with a faster, but imprecise, vision system.

The solution we adopted is conceptually sketched in Fig. 7. Robot 1 makes two measurements at the instants t_A and t_B, these measurements are respectively available at the instants t_1 and t_2 (the boxes in Fig. 7 represent the image processing time required by the robot). So, at the instant t_2 the state of the system is estimated by the DV Module of Robot 1, using the measurements made at t_A and t_B. At the instant t_3 the DV Module of Robot 1 receives a measure from Robot 2 referring to an instant t_C preceding t_B. In order to take into account this new measurement, the DV module retrieves the state of the system at the instant t_A, reorders the available measures and evolves again the system state from t_A fusing, in the correct time sequence, all the received measures.

4 Experiments

We performed many preliminary experiments on this system, but we did not yet test it in an official competition. We carried out extensive experiments on the simulator of our RoboCup team [12]. In this simulator is possible to simulate also the vision process of the robots, adding a custom percentage of random noise. These tests were very valuable in the debugging stage, but were not enough to test the robustness of the system in real game situations. So, the experiments presented in Fig. 8, Fig. 9 and Fig. 10, are performed with the real robots reproducing real games situations in our laboratory.

In the first experiment depicted in Fig. 8, two robots are steady at the centre of the field, like in Fig. 1, and the ball is passing between them. Every one of the two robot is measuring the ball position, sending to the other these measures and fusing the measures received by the teammate. In Fig. 8, the red squares are the measurements made by Robot 1, the black squares are the measurements made by Robot 2 and the blue crosses are the positions of the ball calculated for Robot 2 taking into account the measurements made by Robot 1. Note that the frequencies of the measurements made by the two robots are different: Robot 1 is measuring at 16.5 fps (frame per second), Robot 2 is measuring at 25 fps.

In a second experiment, Fig. 9, one robot (Robot 2) cannot see the ball, because the colour threshold of its vision system for ball recognition were expressly set to wrong values. The other colour threshold are set correctly, so the robot is able to self-localise. A second robot (Robot 1) can see the ball and sends its measures to Robot 2 (the red squares in Fig. 9). Robot 2 is able to correctly estimate the position of the ball using the measures of Robot 1 (the blue crosses in Fig. 9).

The third experiment tests the robustness of the proposed system to the *ball kidnapping* described in Section 3. In this experiment, the two robots are at the two sides of the field of play (approximately 6m apart) and the ball is *kidnapped* from the position on the left, where it was steady, to a position on the right where it starts to move. Note that the DV Module considers the new ball position only after

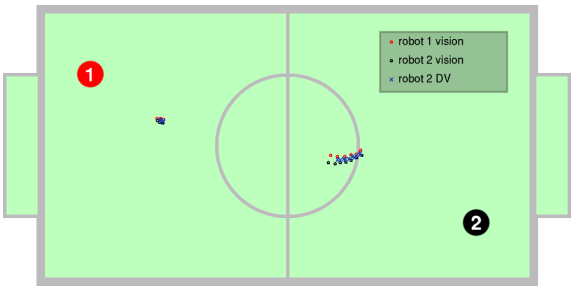

Figure 10: In this experiment the ball is kidnapped from the position on the left, where it was steady to a position on the right where it starts to move.

it received 4 measures (i.e. the blue crosses appears on the right-side only after 4 measurements). This is the time interval needed by the variance of the old track to grow over the variance associated to the new track.

At the moment of writing we are carrying more intensive tests in real games situations in which three and more robots are fusing their measurements to track multiple-moving objects while moving them-self. These first experiments showed there are several parameters that requires to be finely tuned in order to have a good performance of the distributed vision system (e.g. the weights associated to the measures coming from different robots, the rate of grow in time of the variance associated to a track, the time-to-live of non-updated tracks, etc.).

5 Conclusions

In this work, we presented our implementation of an Omnidirectional Distributed Vision System where heterogeneous Omnidirectional Vision Systems share the information extracted from their omnidirectional images. This completes the information sharing in our team, where information are broadcast for cooperation, planning, dynamic role assignment, and now, for sensing. We faced (and solved) many of the problems that a multi-robot team encounters when working in a real environment. For instance: to be able to relate all the measurements to a common spatial frame of reference (we have a robust self-localisation software), to be able to synchronise the observation of the members (we used the Network Time Protocol to synchronise the internal clocks of the robots), to take into account the different computational resources and the heterogenity of the robots (we introduced the possibility to recalculate the state of the system using measures older than the last one), to improve the performances of the system by exploiting the redundancy of the observations and of the observers (we fused the measures coming from different robots shrinking the uncertainty of the single measure), to make accessible the measurements of an high resolution sensor to all the member of the team (we expressly managed the heterogenity of the sensors). Experiments realised in real games situations where the real robots have to track moving objects are presented. In these experiments the robot are static, but we are testing the system also on moving robots.

Despite the fact, that in this work we fused information coming only from omnidirectional vision sensor, this approach is very general and can be applied to fuse measurements coming from different sensors (e.g. perspective cameras or laser range finders) provided those measures can be represented as Gaussian distributions. In the future, we wish to port this system, developed in the RoboCup domain, to surveillance applications. We are realising a surveillance system, where static Vision Agents and mobile Vision Agents (i.e the Vision Agents mounted on mobile robots) cooperate to uncover and collect information on intruders in a off-limit zone.

Acknowledgements

We wish to thank all the members of *Artisti Veneti Team*, the RoboCup team of The University of Padua and the company Garbuio s.p.a. that supported the IAS-Lab.

References

[1] L. Burrelli, S. Carpin, F. Garelli, E. Menegatti, and E. Pagello. Ade: a software suite for multi-threading and networking. Technical report, Intelligent Autonomous Systems Laboratory, Department of Information Engineering, University of Padova, ITALY, 2002.

[2] M. Dietl, J.-S. Gutmann, and B. Nebel. Cooperative sensing in dynamic environments. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'01)*, Maui, Hawaii, 2001.

[3] H. Ishiguro. Distributed vision system: A perceptual information infrastructure for robot navigation. In *Proceedings of the Int. Joint Conf. on Artificial Intelligence (IJCAI97)*, pages 36–43, 1997.

[4] H. Ishiguro. Development of low-cost compact omnidirectional vision sensors. In R. Benosman and S. Kang, editors, *Panoramic Vision*, chapter 3, pages pp. 23–38. Springer, 2001.

[5] J.-H. Lee, N. Ando, T. Yakushi, K. Nakajima, T. Kagoshima, and H. Hashimoto. Applying intelligent space to warehouse - the first step of intelligent

space project. In *In IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, July 2001.

[6] J. J. Leonard and H. J. S. Feder. Decoupled stochastic mapping. *IEEE Journal of Oceanic Engineering*, pages pages 561–571, October 2001.

[7] E. Menegatti, F. Nori, E. Pagello, C. Pellizzari, and D. Spagnoli. Designing an omnidirectional vision system for a goalkeeper robot. In A. Birk, S. Coradeschi, and S. Tadokoro, editors, *RoboCup-2001: Robot Soccer World Cup V.*, pages pp. 78–87. Springer, 2002.

[8] E. Menegatti and E. Pagello. Omnidirectional mirror design based on the vision task. In *Proceeding of 1a Conferenza Nazionale su "Sistemi Autonomi Intelligenti e Robotica Avanzata*, pages pp.151–159, Frascati - Roma, Ottobre 2002.

[9] E. Menegatti, E. Pagello, T. Minato, T. Nakamura, and H. Ishiguro. Toward knowledge propagation in an omnidirectional distributed vision system. In *Proceedings of 1st International Workshop on Advances in Service Robotics (ASDER'03)*, pages pp. 1–6, March 2003.

[10] A. Nakazawa, H. Kato, S. Hiura, and S. Inokuchi. Tracking multiple people using distributed vision systems. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA2002)*, pages pp. 2974–2981, May 2002.

[11] D. Nardi, G. Adorni, A. Bonarini, A. Chella, G. Clemente, E. Pagello, and M. Piaggio. Art99 azzurra robot team. In M. Veloso, E. Pagello, and H. Kitano, editors, *RoboCup99: Robot Soccer World Cup III*, volume 1856 pp. 695-698 of *LNCS*. Springer, 2000.

[12] E. Pagello, M. Bert, M. Barbon, E. Menegatti, C. Moroni, C. Pellizzari, D. Spagnoli, and S. Zaffalon. Artisti veneti 2002: evolving an heterogeneous robot team for the middle-size league. In G. A. Kaminka, P. U. Lima, and R. Rojas, editors, *RoboCup-2002: Robot Soccer World Cup VI.*, L. N. on A. I. Springer, 2002.

[13] T. Sogo, H. Ishiguro, and M. M. Trivedi. Real-time target localization and tracking by n-ocular stereo. In *Proceedings of IEEE Workshop on Omnidirectional Vision (OMNIVIS'00)*, pages pp. 153–160, June 2000.

[14] A. W. Stroupe, M. C. Martin, and T. Balch. Distributed sensor fusion for object position estimate by multi-robot systems. In *Proceedings of International Conference on Robotics and Automation (ICRA2001)*, May 2001.

[15] Y. Yagi. Omni directional sensing and its applications. *IEICE TRANS. INF. & SYST.*, VOL. E82-D(NO. 3):pp. 568–579, MARCH 1999.