The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

ISROBOTNET: A Testbed for Sensor and Robot Network Systems

Marco Barbosa, Alexandre Bernardino, Dario Figueira,
José Gaspar, Nelson Gonçalves, Pedro U. Lima,
Plinio Moreno, Abdolkarim Pahliani, José Santos-Victor, Matthijs T. J. Spaan, João Sequeira

Abstract— **This paper introduces a testbed for sensor and robot network systems, currently composed of 10 cameras and 5 mobile wheeled robots equipped with several sensors for self-localization, obstacle avoidance and vision cameras, and wireless communications. The testbed includes a service-oriented middleware to enable fast prototyping and implementation of algorithms previously tested in simulation, as well as to simplify integration of subsystems developed by different partners. We survey an integrated approach to human-robot interaction that has been developed supported by the testbed under an European research project. The application integrates innovative methods and algorithms for people tracking and waving detection, cooperative perception among static and mobile cameras to improve people tracking accuracy, as well as decision-theoretical approaches to sensor selection and task allocation within the sensor network.**

I. INTRODUCTION

Since the mid-90's, the dissemination of low cost networking media is extending the domains of Robotics applications. The ubiquity of the internet and the huge amount of work on intelligent systems and robotics has further pushed the research on networked robot systems (NRS) to embrace also the topic of human-robot interaction. This induces the development of test tools where different techniques can be assessed in real, and controlled, conditions.

NRS call for the integration of several research and development topics: perception, ubiquitous computing, network communications, decision-making under uncertainty, as well as robot localization, navigation, and world modeling. Furthermore, cooperative techniques that take advantage of the distributed nature of NRS can also be used for perception, navigation and decision-making.

The European URUS project, started December 2007, aims at deploying a NRS in real urban scenarios in the city of Barcelona, Spain. The system is composed by a set of cameras, connected through internet, and multiple heterogeneous robots with onboard computational power, odometry, sonars, GPS, and laser range finder sensors [1]. Experiments on urban surveillance, and transportation and guidance of people and goods are scheduled within the project. The aim is to demonstrate that networked robots can interact naturally with people in urban environments, going beyond usability aspects traditionally considered in human-computer interaction.

Within URUS, the Institute for Systems and Robotics (ISR) at Instituto Superior Técnico (IST), in Lisbon, has developed a testbed for NRS, the *Intelligent Sensor and Robot Network* (ISROBOTNET), that enables testing a wide range of (possibly cooperative) perception and robot navigation techniques (e.g., cooperative localization and navigation, cooperative environment perception, cooperative map building), as well as human robot-interaction, distributed decision making, and task and resource allocation. Such a testbed enables fast integration of the heterogeneous subsystems involved in a NRS, in a fairly controlled environment. This way, later stages of testing in real urban scenarios benefit of the fact that integration is already fully functional, and the focus can then be turned to the decision-making, task allocation, human-robot interaction, perception and navigation subsystems.

Deploying a NRS requires an additional component to integrate all the subsystems involved. This integration component interfaces the mobile and static sensor and actuator components with the physical devices, and has similarities to a standard information system, namely concerning the involved requirements, e.g., flexibility, scalability, platform independence, development process simplification, real-time performance, integration with existing infrastructure, promoting software reuse, programming language independence, and robustness. ISROBOTNET is built around the kernel of a service-oriented architecture (`MeRMaID`) originally developed for ISR's RoboCup Middle Size League (MSL) robot team [2], and extended to use the YARP networking software [3]. The resulting research environment is being used to test a myriad of heterogeneous subsystems independently developed by the different groups in the project team.

Design options for the testbed build upon the similarity between architectures for the integration of different components in multiple domains. Among the most representative examples of NRS architectures, we have selected just a few that identify interesting concepts.

The DAMN architecture [4] is a collection of independently operating modules implementing a group of distributed behaviors. These communicate with a centralized arbiter which is responsible for combining the behaviors such that the resulting action reflects their objectives and priorities. A networking concept for robots based on the LACOS context manager was proposed in [5]. Each node in the network is composed of a robot and a LACOS interface to the

This work is supported by the European Project FP6-2005-IST-6-045062-URUS and by Fundação para a Ciência e a Tecnologia (ISR/IST pluriannual funding) through the POS_Conhecimento Program that includes FEDER funds, and by grants PTDC/EEA-ACR/73266/2006,SFRH/BD/23804/2005 and SFRH/BD/30957/2006

All the authors are with the Institute for Systems and Robotics at Instituto Superior Técnico, Technical University of Lisbon, Portugal. {mafb, alex, wurmd, jag, ngoncalves, pal, plinio, apahliani, jasv, mtjspaan, jseq}@isr.ist.utl.pt

network bus. The LACOS interface contains an application interface, a query issue engine, a result collector and a context database and corresponding manager. A three tier architecture with application, infrastructure and middleware layers was proposed in [6]. The application layer contains the functional blocks related to single robot activities, e.g., path planning. The infrastructure layer provides the network services. The middleware layer handles the communication among services. The Distributed Field Robot Architecture (DFRA) has been applied to a simulated demining task [7]. It is a behavioral architecture that implements a standard perception-to-actuation scheme with additional blocks for map building and sensor and actuator management.

Despite the numerous NRS proposals, implemented flexible NRS testbeds are not common. After a section describing the ISROBOTNET testbed in detail, this paper describes multi-disciplinary research work, where innovative methods have been ported from typical research simulation tools to a real NRS environment with minimal effort, using the ISROBOTNET facilities.

II. THE SENSOR AND ROBOT NETWORK TESTBED

Currently, the ISROBOTNET testbed is composed of an indoor area of around 160 m^2 with 10 webcams placed at the ceiling such that some of the fields of view do not overlap. The cameras are distributed in 4 groups, each of which is managed its by its own computer, namely for image acquisition. The managing computers are connected to the ISR/IST network and can be accessed by duly authorized external parties. Ongoing work will extend the number of cameras and the usable indoor space to include multiple floors. Robots will use the same elevators as ordinary people to move between floors. Besides the camera sensors, four Pioneer AT and one ATRV-Jr robots are available. Each of the robots is equipped with sonars, onboard cameras, laser range finder and is Wi-Fi connected to the network. Figure 1 shows one of the floors at ISR/IST where the testbed is implemented and a map with the cameras field-of-view.

The testbed can be used to demonstrate both the individual and integrated operation of NRS subsystems. The use of the YARP networking software (extending `MeRMaID`) creates the basis network transparent layer over which a wide variety of integration architectures can be deployed. Arbitrary computational resources can be distributed over the network. The integration architecture (see Section III) allows a fully decentralized use of these resources.

In addition to the specific integration infrastructure, basic services such as robot teleoperation and direct image acquisition and recording from the camera network, as well as event logging, are also available.

III. MIDDLEWARE AND SUBSYSTEMS INTEGRATION

A. Service-Oriented Architecture

The URUS project includes 11 partners, from different institutions (universities, research centers and companies) and countries, working on different aspects of what should be seen by the user as an integrated system. Therefore, special attention was focused on the integration approach. The initial requirements for the integration platform were:

- minimize the effort required from each individual developer that is concerned with only a very limited portion of the system, by providing a simple "light-weight" integration setup both in terms of technology used and effort required to integrate existing code;
- avoid constraining the ability to build a system that is able to cope with the project's requirements while constraining the developer's options in integration-critical aspects of the system;
- enforce a non-fixed initial structure for the system since specific capabilities to be developed by the partners were not completely defined in the beginning.

To fulfill the previous general requirements, the service-oriented architectural style was chosen as the general framework through which the problem of connecting needs to capabilities is viewed. The OASIS group's "Reference Model for Service Oriented Architecture" [8] provided all the basic concepts and terminology that are used project-wide by the several partners. Within this model, a *Service* is viewed as a software artifact that has some kind of internal dynamics. Therefore, a *Service*'s internal state may change through time. This artifact exists with the purpose of providing a certain capability which it controls. This capability may be, for instance, some kind of robot navigation algorithm, a high-level coordination mechanism or even some non-functional capability like data logging and display. Each *Service* exists and runs by itself, having well-defined mechanisms through which interaction with other *Service*s may be made.

B. Service Interaction Mechanisms

A fundamental aspects of *Service*s is that they need to interact with each other in order to, cooperatively, enable the whole system to behave as expected. Integration-wise it is important to have standard mechanisms through which *Service*s interact. This constrains the developer's options but allows more control over how *Service*s are used. In order to establish well-defined mechanisms through which *Service*s may interact, two types of interactions where defined:

a) **Service Request:** a polling mechanism through which a request from a *Requester Service* is sent to a *Target Service*'s *Service Interface*. Upon reception, the *Target Service* is supposed to process that request according to the rules stipulated for requests sent to one of its *Service Interface*s. After processing is finished, the *Target Service* is expected to send a reply back to the *Requester Service*. This is the preferred mechanism for single-shot interactions.

b) **Data Feed:** a data push mechanism through which data generated by a *Writer Service* is made available to the rest of the system. Any other *Service* interested in the data may simply read it (named a *Reader Service* in the context of this interaction). This is the preferred mechanism for continued interactions.

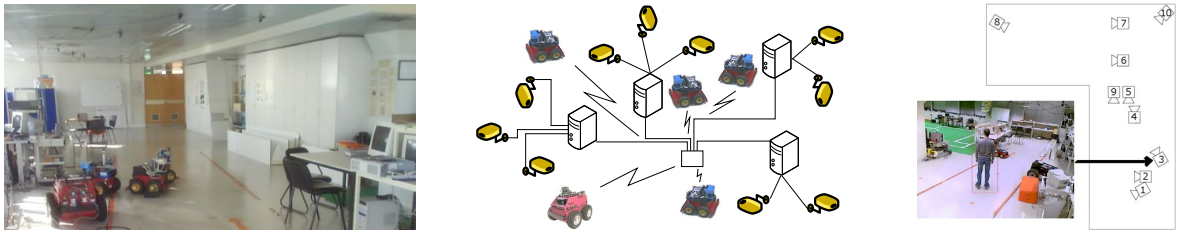

Fig. 1. Left: Partial view of the ISROBOTNET testbed, showing most of the cameras in the ceiling and the mobile robots equipped with several sensors. Center: Diagram of the connectivity links between cameras and robots, which have to share the limited network bandwidth. Right: The map of the environment showing the cameras' field-of-view and a picture taken from camera 3.

The implementation of these interaction mechanisms was defined to use YARP [3] as a communication library. YARP was chosen given its capability to work in several platforms, its simple *Port* concept for communication endpoints and usage experience within the URUS consortium.

C. System Description Files

Besides knowing how a *Service* may interact with another *Service* the developer also needs to know which other *Service*s exist, what capabilities do they offer, under which interaction mechanism are they offered and what is the structure of all the data exchanged between *Service*s. It was decided that a full specification of all these issues was needed in order to successfully integrate software, in the form of description files written in XML.

These description files are used to describe the structure of the whole system and also to validate syntactically, during run-time, all the interactions between *Service*s. This way it is possible to validate that all the data exchanged between *Service*s is correctly formatted and that the interaction mechanisms being used are valid. It is expected that each *Service* should validate semantically all the interactions it is part of, and act accordingly.

D. `MeRMaID`

Although one of the objectives while defining the integration framework was to keep it as simple as possible, it still would require a fairly strong effort from each individual developer if he were to design, implement, test and deploy his/her own instance of all of issues covered by the integration framework. Therefore we chose to adapt and extend our available middleware `MeRMaID` (Multiple Robot Middleware for Intelligent Decision-making) to comply with all the integration rules that have been specified. For the URUS project, only `MeRMaID`'s low-level layer, designated as `MeRMaID::support`, was used.

Unlike other middleware projects, for instance, MIRO [9] and OROCOS [10], the emphasis was not put in the development of the software middleware itself. The main issues that were addressed were the definition of system-wide concepts (such as those of *Service* and the interaction mechanisms) and their low-level implementation (e.g., usage of *YARP* as a communications library). After this was all defined, a natural next step was to develop a software package that forces the developed solution to take into account

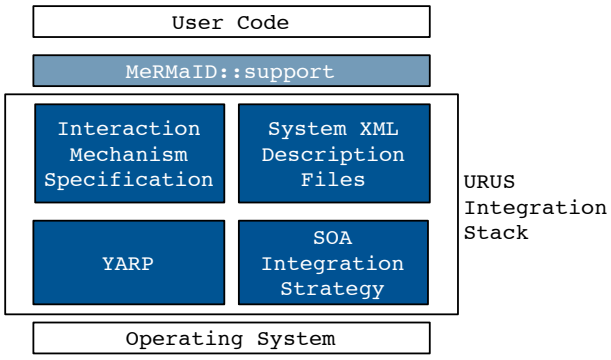

Fig. 2. URUS integration stack in dark blue with optional `MeRMaID::support` middleware in light blue. Usage of `MeRMaID::support` forces the user to follow all integration specifications.

all of the requirements of the overall integration framework. As a result, integration can both be accomplished with or without the usage of `MeRMaID::support`, as long as all of the integration requirements are followed. A diagram of the URUS integration stack is shown in Figure 2.

All the work described in the remainder of the paper was implemented using the above service-oriented architecture and, in some cases, `MeRMaID::support` as well.

IV. (COOPERATIVE) PERCEPTION

A. Human-Robot Detection and Localization

A key element in NRS is the ability to detect and locate items of interest in the environment. In ISROBOTNET, this ability is provided by the fixed camera network, through the use of real-time algorithms for the detection and localization of persons and robots – the main entities of interest in the considered scenarios. We assume a calibrated camera network (homography transformations relate the cameras' image planes to the ground plane), allowing us to describe the location of objects in a unified world coordinate system, representing the area under surveillance. We compute the uncertainty of the measurements in world coordinates, which constitutes an essential input for data fusion in the higher level temporal tracking and cooperative localization algorithms. Despite the numerous works on fixed camera methods for detection, object identification and uncertainty modeling, ours contributes to the state-of-the-art in the addition of 3D

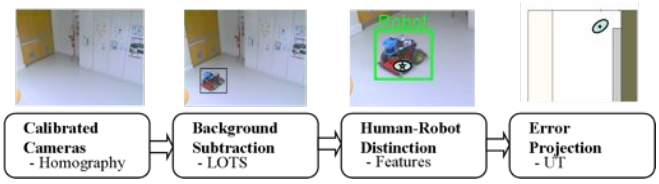

Fig. 3. Diagram of the person-robot detection, classification and localization algorithm.

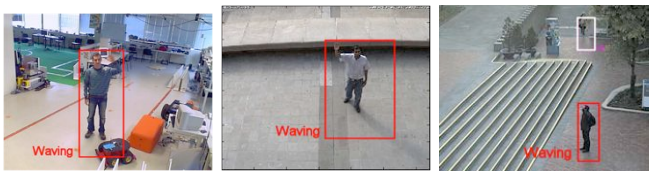

Fig. 4. Waving detection, considering very different scenarios, image resolutions and human postures.

features to the person-robot discrimination methods and a context-dependent characterization of image measurement noise arising in the detection phase.

A diagram of the detection, categorization and localization algorithms is provided in Figure 3. The first step consists of detecting moving objects in the images using the LOTS [11] adaptive background differencing algorithm. This process provides a list of image blobs that potentially correspond to objects of interest in the scenario. The next step consists of extracting features from the image blobs to support their discrimination into persons, robots or other objects. 2D features are extracted directly from the images: blob's area, bounding box aspect ratio, occupancy ratio (blob's area over bounding box area). 3D features include the blob's location and height in world coordinates and requires the use of the camera calibration homographies. The blob's location is used to discard objects that are outside the valid ground plane region. The remaining features are used to classify the blobs in persons, robots or other. The classifier is currently a simple decision tree with fixed structure.

After targets have been categorized, we compute their location and associated uncertainty in world coordinates. To localize the target's position in world coordinates we first select an image point corresponding the contact point on the ground floor. This point is selected as a fraction of the bounding box height. Depending on the target's class (persons or robots), the mean and covariance of this computation was assessed with manually defined ground truth data. The mean corrected point is then projected to the ground plane coordinates via the camera calibration homography. Also the image error covariance ellipse is projected on the ground plane through the Unscented Transform [12], in order to compute the world plane uncertainty. Further details about the described algorithms can be checked in [13].

B. Waving Recognition

As a part of the EU project CAVIAR [14], ISR has addressed the recognition of activities in a surveillance scenario, detecting events such as left luggageand people fighting, using qualitative representations of body parts' movements. We apply this type of representations to model the waving gesture, a universal mechanism whereby people call for attention and signal emergency situations. We develop a real-time waving detector that can be applied to indoor and outdoor scenarios. Our model relies on a qualitative representation of body parts' movements. Waving patterns are represented by simple motion statistics information, not requiring the (time-consuming) pose reconstruction of parts of the human body. We use focus of attention (FOA) features [15] which compute optical flow statistics with respect to the target's centroid. In order to detect waving activities at every frame, a boosting algorithm uses labeled samples of FOA features in a binary classification problem: waving *vs* not waving. We use the Temporal Gentleboost algorithm [16] which improves boosting performance by adding a (short-term) temporal support to the features. Noise robustness is further improved by defining a waving event, which imposes the occurrence of a minimum number of single-frame waving classifications in a suitably defined temporal window.

From the person-robot detection algorithm, each bounding box classified as a person is tested by the waving detector. A rudimentary form of tracking is provided by data association based on the distance between bounding box positions in consecutive time frames (nearest neighbor or Hungarian assignment [17]), which allow us to cope with moving targets. In the image regions corresponding to the detected targets, we compute FOA features based on a dense optical flow algorithm [18]. The optical flow algorithm is based on a new metric for intensity matching, which removes noisy flow vectors with a low computational load. It allows high frame rate performance (up to 20fps) and low false positive rate. In addition, the method is able to detect waving patterns in low resolution targets, which is frequently the case in cameras with wide field of view in surveillance infrastructures. In [19], the robustness of the waving model (FOA and GentleBoost) is tested on the KTH action database and compared to the state-of-the-art results in both indoors and outdoors datasets. In Figure 4 we show both indoor and outdoor examples of waving detection. In one case, despite the person being in a lateral posture, the method is able to successfully detect the gesture.

C. Active Cooperative Perception

One of the advantages of adding mobile sensors to static sensor networks is the ability to actively change some of the sensors' vantage points, so as to improve the quality of perception, as obtained by fusing (some of) the sensors' estimates. An active cooperative perception (ACP) network [20] is able to improve the quality of its fused estimates by taking actions (e.g., moving robots with onboard cameras) that change network topology to minimize a given cost function.

In ISROBOTNET, people tracking can be improved by fusing the information from the static cameras with sensor readings provided by cameras onboard mobile robots. Mobile

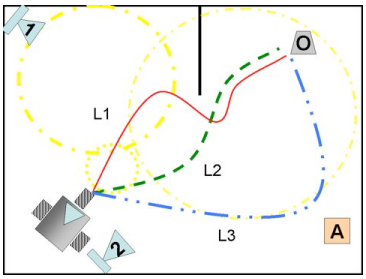

Fig. 5. Scenario where a mobile robot can take several paths.

robots can get closer to the person and improve the accuracy of its localization over time. The vision-based people tracking system described in Section IV provides means and covariances of people locations so that these can be used in the (active) sensor fusion process.

To reach closer to the person location, a robot can initially take different paths. A path can be longer than others but might be faster to take, or require less energy, e.g., due to less changes in robot orientation. Additionally, alternative paths will be covered differently by the camera network. Since sensor fusion requires estimates to be transformed to a common global frame, the uncertainty about the mobile sensor's (robot's) location matters. If one uses the camera network to improve the robot's localization, paths better covered by the camera network will reduce people tracking uncertainty. Therefore, several factors influence the decision of whether to move a robot or not to improve people tracking, and which path should the robot take [21].

Consider the scenario depicted in Figure 5 with 2 static cameras and a mobile robot that can take different paths L_1, L_2 and L_3 to approach object O and estimate its location. The object is within the field of view of camera 2. Camera 1 covers part of the environment but can not see the object. However, camera 1 field of view can be used by the robot to improve its own localization uncertainty. The shortest path to reach the object is L_2. L_1 might be a good alternative candidate, but a wall obstacle on the way to the object requires some maneuvering, increasing the spent energy. On the other hand, the apparently longer-than-necessary path L_1 may have a reduced cost, as the robot will be observed by camera 1 on its way to object O, therefore improving its own localization uncertainty. As for path L_3, it takes a detour to observe closer another object of potential interest, A. If the network goal includes getting additional information on this object, this should be considered as well.

We use a Markov Decision Process (MDP) framework to determine the best path, in the sense of minimizing a path cost which includes the energy spent, and uncertainties about object and mobile sensor localization as penalties, and the relevance of visiting some extra objects, as rewards.

V. DECISION-MAKING UNDER UNCERTAINTY

A. Sensor Selection

Besides the classical surveillance setup in which a human operator monitors video streams, we have been working on automating this task as demonstrated in Section IV. However, given the large bandwidth and computing power demands of imaging sensors, processing the video streams of a large number of cameras simultaneously might not be feasible. For instance, state-of-the-art human activity recognition algorithms such as the one presented in Section IV-B, require high-resolution video streams at a high frame rate, as well as significant computational resources. The bandwidth problem becomes even more prominent given the growing popularity of so-called IP-cameras, which are connected directly to a (local area) network, and need to share this medium.

Given these constraints and a set of sensors, we study the problem of selecting a subset that can be active at any point in time. The goal of the system is to optimize a user-defined objective. We consider several possible objectives, for instance maximizing coverage or minimizing uncertainty when tracking people [22]. Related problems have been studied in the wireless sensor network literature [23], where the resource constraint considered typically is energy, given each sensor's limited battery life. We focus on developing dynamic sensor selection methods, which can change the active subset of sensors over time. In this way, the system can react to the observed state of its environment, significantly improving the system's performance.

In particular, we consider a decision-theoretic approach to dynamic sensor selection [22]. We propose to use Partially Observable Markov Decision Processes (POMDPs) [24], as they form a strong methodological framework for sequential decision-making under uncertainty. We model the problem of tracking a person using n cameras as a POMDP, under the constraint that only k cameras can emit observations at any point in time, with $k \ll n$. This resource constraint forms a general way to model restrictions in available bandwidth or computing power. Now, the POMDP's actions are defined as selecting the k sensors to be active in the next time frame. As the POMDP's belief state forms a sufficient statistic for the decision-making problem, the system incorporates all observations made in the past. We show how, by changing the POMDP's reward function, we can change the system's behavior in a straightforward manner, fulfilling the user's chosen objective. We have demonstrated our techniques on the ISRobotNet network of 10 cameras, illustrating the rich set of behaviors we can achieve [22].

B. Task Allocation

In ISRobotNet, the tasks that the mobile robots can provide are related to navigation, cooperative perception (with the camera network) and human-robot interaction. The requests to execute tasks originate from multiple sources, such as the camera network or the human system operators In general, the number of tasks to be executed at some time instant is not equal to the number of mobile robots. Furthermore, the mobile robots are heterogeneous and the order in which tasks will be requested is not known in advance. Under these conditions, the problem of computing an optimal allocation solution is NP-Hard [25].

An auction protocol is employed for the allocation of tasks in ISROBOTNET. The mobile robot tasks to are first collected at a central node, denoted the *auctioneer*. The tasks can arrive at any instant, but are announced to the robots only at regular intervals. The robots then reply with their individual estimated fitness (see below) for the execution of each task. This information is used to formulate a linear assignment problem of tasks to robots, which is solved efficiently using the Hungarian algorithm, [17]. In some applications, tasks may have distinct assignment priorities. This is implemented at the auctioneer level, where the optimization cost functional is modified accordingly.

The main advantage of this protocol is that the auctioneer is not required to know the state of each robot. The protocol is also robust to communication failures: tasks are allocated if only some of the robot replies are received.

The synthesis of controllers for the execution of tasks by the mobile robots is accomplished using a Partially Observable Markov Decision Process (POMDP) [26]. For each task, suitable models of the robot's rewards, actions and observations are determined. The resulting POMDPs are solved off-line to obtain an execution policy and the expected discounted reward at each belief state. The latter is identified with the fitness of the mobile robot to execute the task.

Combining POMDPs and auction protocols for task allocation is a natural solution. The performance of auction protocols is dependent on the correct estimation of each robot fitness [27], which is readilly obtained from expected discounted reward value of each task controller. Another advantage is that the fitness for tasks using different action sets can be compared.

A POMDP-based solution for task allocation can only be approximated for problems with modest dimensions. However, since coordination among the system components is implicitly provided in the auction protocol, the POMDPs for each task need not account for other robots. As a results, the dimension of the POMDPs are smaller than if the states and actions of all robots where to be considered in a single assignment problem.

VI. INTEGRATED EXPERIMENTS

In this section we show an experiment integrating several of the existing functionalities of the described setup. The experiment illustrates a *rendezvous* scenario where a service robot meets a person detected by the camera network.

As explained in Section IV, each camera detecting persons and robots associates an uncertainty measure (covariance matrix) to localization information in world coordinates. When targets are observed by more than one camera, a Bayesian data fusion process joins the information from the several cameras and provides the POMDP with the *a posteriori* distribution of the data, assuming Gaussian uncertainties. This process is illustrated in Figure 6. Two cameras observe the same target and each one provides an independent location measure with high uncertainty. After the fusion process, the uncertainty is significantly reduced.

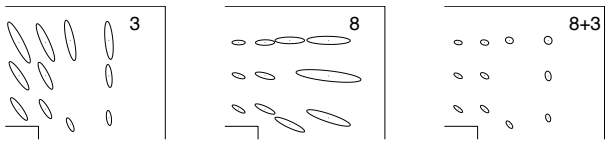

Fig. 6. Detail of the uncertainty models of cameras 3,8 and their fusion. Refer to Figure 1 for the cameras' locations.

Figure 7 shows (i) the detections provided by the camera network; (ii) the person and robot positions displayed in the 2D world map and and (ii) the environment discrete cells of a POMDP. All cameras are working but we display only two of them at each time: one that shows the person's detection and the other to show the robot's detection. In the maps we plot the trajectories of persons and robots obtained by gathering information from all the cameras. There are some outlying detections: sometimes a person is mis-detected as a robot due, mainly, to occlusions in the field of view of the cameras, but the false detection rate is very low and does not influence the probabilistic localization methods. The POMDP map represents the likelihood of person and robot positions (the darker the cell, the higher the likelihood) and the blue circle shows the selected cell for commanding the next robot position, determined by the decision process.

VII. CONCLUSIONS AND FUTURE WORKS

In this paper we introduced a testbed for sensor and robot network systems, describing its hardware and service-oriented middleware, as well as some of the research whose development it has supported so far. The main benefit of the testbed is that it has enabled fast integration, using systematic principles, and fairly simple rules, of considerably diverse subsystems (e.g., vision-based people tracking and waving recognition, cooperative perception among static and mobile cameras, decision-theoretical camera selection and task allocation) in a coherent manner. Furthermore, this is accomplished in a fairly realistic but controlled scenario, thus focusing on the integration aspects, in such a way that the testbed can act as a first step towards more demanding applications, e.g., in outdoor scenarios.

Future work will consist of continuous improvements on the middleware (a new version is almost ready, and also ported to the RoboCup MSL team) and current algorithms, as well as the implementation of new ones, including the more ambitious integration of several subsystems on a building evacuation mission, where the network must detect several different events, with different priorities and detection level of confidence, such as robot needing recharge, eruption of fire bursts, people intrusion, people asking for support, or people moving towards the wrong exit, and take cooperative decisions on the mobile robots paths, so as to guide people towards safe exits. This will enable further development of theoretical research on the involved enabling disciplines (computer vision, decision-making under uncertainty, cooperative perception, task allocation), driven by the practical challenges posed by a realistic implementation.

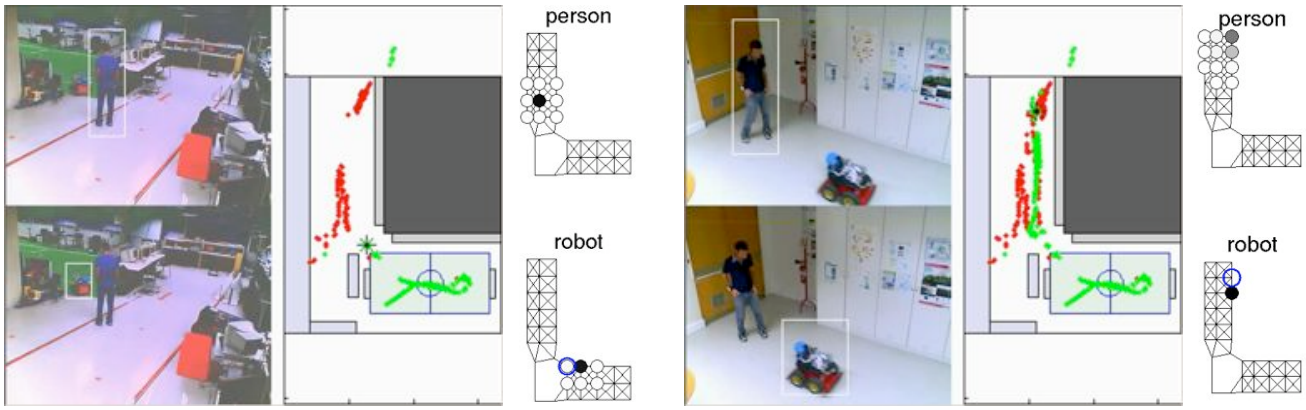

Fig. 7. The *rendezvous* experiment. We show two meeting points: one at the center of the map (left figures) and another at one of the extremes (right figures). At each time step the location of person and robot is determined by the camera network, displayed by red and green markers in the map, respectively. The POMDP spatially fuses the temporally fused detections provided by the camera network and computes the likelihood of person and robot locations in a discrete cell based map. Periodically, the POMDP computes the best action to apply to the robot in order to follow the person (shown as a blue circle in the figures).

REFERENCES

[1] A. Sanfeliu and J. Andrade-Cetto, "Ubiquitous networking robotics in urban settings," in *Workshop on Network Robot Systems. Toward Intelligent Robotic Systems Integrated with Environments. Procs. of 2006 IEEE/RSJ International Conference on Intelligence Robots and Systems (IROS2006)*, October 2006.

[2] M. Barbosa, N. Ramos, and P. Lima, "Mermaid - Multiple-Robot Middleware for Intelligent Decision-Making," in *Procs. of the IAV2007 - 6th IFAC Symposium on Intelligent Autonomous Vehicles*, 2007, Toulouse, France.

[3] G. Metta, P. Fitzpatrick, and L. Natale, "Yarp: Yet another robot platform," *Int. Journal of Advanced Robotic Systems*, vol. 3, no. 1, pp. 43–48, 2006.

[4] J. Rosenblatt, "Damn: A distributed architecture for mobile navigation," *Journal of Experimental and Theoretical Artificial Intelligence*, vol. 9, no. 2-3, pp. 339–360, 1997.

[5] S. Satake, H. Kawashima, and M. Imai, "LACOS: Context management system for a sensor-robot network," in *Procs. 10th IASTED International Conference on Robotics and Applications (RA 2004)*, 2004, pp. 160–165.

[6] E. Woo, B. MacDonald, and F. Trépanier, "Distributed mobile robot application infrastructure," in *Procs. Int. Conf. Intelligent Robots and Systems (IROS'03)*, October 2003, pp. 1475–1480, las Vegas.

[7] M. Long, A. Gage, R. Murphy, and K. Valavanis, "Application of the distributed field robot architecture to a simulated demining task," in *Procs. IEEE International Conference on Robotics and Automation (ICRA'05)*, 2005, barcelona, Spain.

[8] OASIS, "Reference Model for Service Oriented Architecture 1.0," OASIS, Tech. Rep., August 2006.

[9] H. Utz, S. Sablatnog, S. Enderle, and G. Kraetzschmar, "Miro - middleware for mobile robot applications," *Robotics and Automation, IEEE Transactions on*, vol. 18, no. 4, pp. 493–497, August 2002.

[10] H. Bruyninckx, "Open Robot Control Software: the OROCOS project," *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, 2001.

[11] T. E. Boult, R. J. Micheals, X. Gao, and M. Eckmann, "Into the woods: Visual surveillance of noncooperative and camouflaged targets in complex outdoor settings," *Proceedings Of The IEEE*, vol. 89, no. 10, pp. 1382–1402, October 2001.

[12] S. Julier, J. Uhlmann, and H. F. Durrant-Whyte, "A new method for the nonlinear transformation of means and covariances in filters and estimators," *Automatic Control, IEEE Transactions on*, vol. 45, no. 3, pp. 477–482, 2000.

[13] D. Figueira, P. Moreno, A. Bernardino, and J. Gaspar, "Detection and localization of persons and robots in a networked camera system," 2009, submitted. Also as ISR/IST Technical Report. [Online]. Available: http://www.isr.ist.utl.pt/ alex/techrpts/VislabISRTechReport200901.pdf

[14] "CAVIAR: Context aware vision using image-based active recognition," 2002-2005. [Online]. Available: http://homepages.inf.ed.ac.uk/rbf/CAVIAR/

[15] F. Pla, P. C. Ribeiro, J. Santos-Victor, and A. Bernardino, "Extracting motion features for visual human activity representation," in *Proceedings of the IbPRIA'05*, 2005.

[16] P. C. Ribeiro, P. Moreno, and J. Santos-Victor, "Boosting with temporal consistent learners: An application to human activity recognition," in *Proc. of 3rd International Symposium on Visual Computing*, 2007, pp. 464–475.

[17] R. Ahuja, T. Magnanti, and J. Orlin, *Network Flows*. Prentice Hall, 1993.

[18] A. S. Ogale and Y. Aloimonos, "A roadmap to the integration of early visual modules," *International Journal of Computer Vision*, vol. 72, no. 1, pp. 9–25, april 2007.

[19] P. Moreno, A. Bernardino, and J. Santos-Victor, "Waving detection using the local temporal consistency of flow-based features for real-time applications," in *Proc. of ICIAR - 6th International Conference on Image Analysis and Recognition*, 2009.

[20] D. Fox, W. Burgard, and S. Thrun, "Active markov localization for mobile robots," *Robotics and Autonomous Systems*, vol. 25, pp. 195–207, 1998.

[21] A. Pahliani, M. Spaan, and P. Lima, "Decision-theoretic robot guidance for active cooperative perception," in *Proc. of IEEE International Conference on Intelligent Robots and Systems (IROS 2009)*, 2009.

[22] M. T. J. Spaan and P. U. Lima, "A decision-theoretic approach to dynamic sensor selection in camera networks," in *Int. Conf. on Automated Planning and Scheduling*, 2009.

[23] H. Rowaihy, S. Eswaran, P. Johnson, T. Brown, A. Barnoy, D. Verma, and T. F. La Porta, "A survey of sensor selection schemes in wireless sensor networks," in *SPIE Unattended Ground, Sea, and Air Sensor Technologies and Applications IX*, 2007.

[24] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial Intelligence*, vol. 101, pp. 99–134, 1998.

[25] J. Jungwattanakit, M. Reodecha, P. Chaovalitwongse, and F. Werner, "A comparison of scheduling algorithms for flexible flow shop problems with unrelated parallel machines, setup times, and dual criteria," *Computers & Operations Research*, vol. In Press, Corrected Proof, 2007.

[26] M. T. J. Spaan, N. Gonçalves, and J. Sequeira, "Multiagent coordination by auctioning POMDP tasks," in *Multi-agent Sequential Decision Making in Uncertain Domains*, 2009, workshop at AAMAS09.

[27] M. B. Dias, R. Zlot, N. Kalra, and A. Stentz, "Market-based multirobot coordination: a survey and analysis," *Proceedings of the IEEE*, vol. 94, no. 7, pp. 1257–1270, 2006.