Implicit Coordination in Robotic Teams using Learned Prediction Models

Freek Stulp, Michael Isik and Michael Beetz
Intelligent Autonomous Systems Group, Technische Universität München
Boltzmannstrasse 3, D-85747 Munich, Germany
{stulp,isik,beetz}@in.tum.de

***Abstract*— Many application tasks require the cooperation of two or more robots. Humans are good at cooperation in shared workspaces, because they anticipate and adapt to the intentions and actions of others. In contrast, multi-agent and multi-robot systems rely on communication to exchange their intentions. This causes problems in domains where perfect communication is not guaranteed, such as rescue robotics, autonomous vehicles participating in traffic, or robotic soccer.**

In this paper, we introduce a computational model for implicit coordination, and apply it to a typical coordination task from robotic soccer: regaining ball possession. The computational model specifies that performance prediction models are necessary for coordination, so we learn them off-line from observed experience. By taking the perspective of the team mates, these models are then used to predict utilities of others, and optimize a shared performance model for joint actions. In several experiments conducted with our robotic soccer team, we evaluate the performance of implicit coordination.

I. Introduction

As robotic systems are becoming more dextrous and sophisticated, they are capable of executing more complex tasks. Many of these more complex application tasks require two or more robots to cooperate in order to solve the task. A key aspect of these systems is that multiple robots share the same workspace, and can therefore not abstract away from the actions of other robots.

Humans are very good at performing joint actions in shared workspaces. Consider two people assembling a bookcase. With apparent ease, actions are *anticipated* and coordinated: one person holds a shelf while the other screws it in place, and so forth. A key aspect of this cooperation is that it is executed with little or no communication. Humans achieve this by inferring the intentions of others. Once the beliefs and desires of the cooperating party are known, we simply imagine what we would do in that situation. This is called the Intentional Stance [5].

If I see you grab a screw-driver, I can assume you intend to screw the shelf in place; there is no need for you to tell me. By integrating your intentions into my own beliefs, I can also anticipate that my holding the shelf will ease our task, thereby coming closer to our joint desire of assembling the bookcase. This is called *implicit* coordination, and is visualized in Figure 1.

In contrast, coordination in multi-agent and multi-robot systems is usually achieved by extensive communication of intentions or utilities. This is called *explicit* coordination, and is visualized in Figure 2. Previous work on cooperation seems to have focussed almost exclusively on this form of coordination [1], [4], [6], [10], [15]. It has also been used in the RoboCup mid-size league to allocate roles to the different players [3], [12].

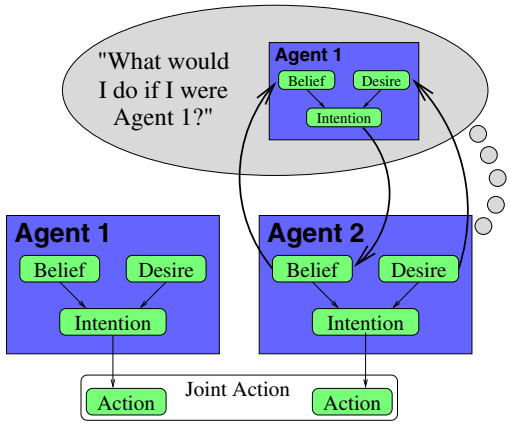

Fig. 1. Implicit coordination with the Intentional Stance

There are many domains in which implicit coordination is used by humans: almost all team sports, construction of bookcases and others, and also in traffic. Because implicit coordination dispenses of the need for communication, there are also many multi-robot domains that could benefit from this approach. Rescue robotics and autonomous vehicles operating in traffic are examples of domains in which robust communication is not guaranteed, but where correct coordination and action anticipation is a matter of life and death. When it comes to saving humans or avoiding accidents, it is better to trust what you perceive, than what others tell you: seeing is believing.

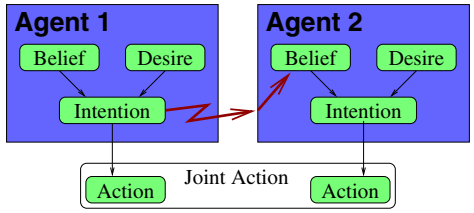

Fig. 2. Explicit coordination with communication

A current research focus in cooperation is human-robot interaction, for instance in space exploration [7] or rescue robotics [9]. Our research group has a long-term project for human-robot interaction in intelligent rooms. The room and robot are equipped with cameras, laser range finders and RFID tags, which provide robots with accurate information about what is going on in the room.

When a robot and a human perform a joint action in their shared workspace, e.g. setting the table in the kitchen, or seam welding in outer space [7], it cannot be expected of humans to continuously communicate their intentions. Instead, the robot must be able to anticipate a human's intentions, based on predictive models of human behavior. We consider implicit coordination to be essential for natural interaction between robots and humans.

A typical coordination task from the robotic soccer domain is to regain ball possession. Acquiring ball possession is a goal for the team as a whole, but only one of the field players is needed to achieve it. The benefit of having only one player approach the ball is obvious: there will be less interference between the robots, and it also allows the other robots to execute other important tasks, such as strategic repositioning or man marking. Of course, the robots must agree upon which robot will approach the ball. The intuitive underlying rule is that only the robot who is quickest to the ball should approach it. This rule is also used in [8], in which each robot determines the distance of each team mate to the ball. Based on this, each agent decides if it will approach the ball or not. Coordination is still explicit, because the agent who decides to approach the ball first must 'lock' a shared resource, which prevents other robots from chasing after it. The use of this global resource requires communication.

In this paper, we present a computational model of implicit coordination and apply this model to the ball approach task. To infer the intentions of others, the agents first learn utility prediction models from observed experience. For the ball approach task, the utility measure is time, so the robots learn to predict how long it will take to approach the ball. During task execution, the robots locally predict the utilities for all robots, and globally coordinate accordingly.

The contributions of this paper are:

- presenting a computational model for implicit coordination
- learning utility prediction models on real robots, and comparing two learning methods previously used to learn such models,
- demonstrating how these learned models can be used for implicit coordination on real robots.

The rest of this paper is organized as follows. In the next section, we present the computational model of implicit coordination. In Section III, we show how prediction models are learned on the robots. In Section IV, these models are used to implicitly coordinate the robots, after which we conclude with and present future work in Section V.

II. Computational Model

A graphical representation of the computational model of implicit coordination can be seen in Figure 3. Green (light) items are fixed, and the same on all agents. Blue (dark) items change during task execution. To the left are modules that can be found in almost any agent. The state estimation modules takes a percept, and derives a belief state from them. Given its beliefs and desires, the agents decides its current intention. This intention is executed in the real world as an action.

The modules that are necessary for implicit coordination are to the right of the dashed line. The main module is the *Teammate Intention Inference*, similar to the Computational Cognitive Module described in [7]. It contains the following submodules:

- *Predictive models*. These models map the intention and belief of an agent to a performance measure. In the ball approach task, the performance measure is time. The models are learned from observed experience, as will be described in Section III
- *Shared performance models for joint actions*. These determine the performance of joint actions. For the ball approach task, the shared performance model states that the performance of the joint action 'approach ball' is higher if only one robot approaches the ball.
- *Perspective Taker*. This module lets the agent imagine what it would do if it were the other agents. To do this, it swaps its own state with that of another agent in the belief state, and determines its own intention based on the joint desires and this 'inverted' belief state. Since the agents have a joint goal, the desires of other agents are the same as my own.
  The agents can only take the perspective of others, if they know each other's state. Therefore, it is essential that the state estimation provides each agent with estimates of the states of others.

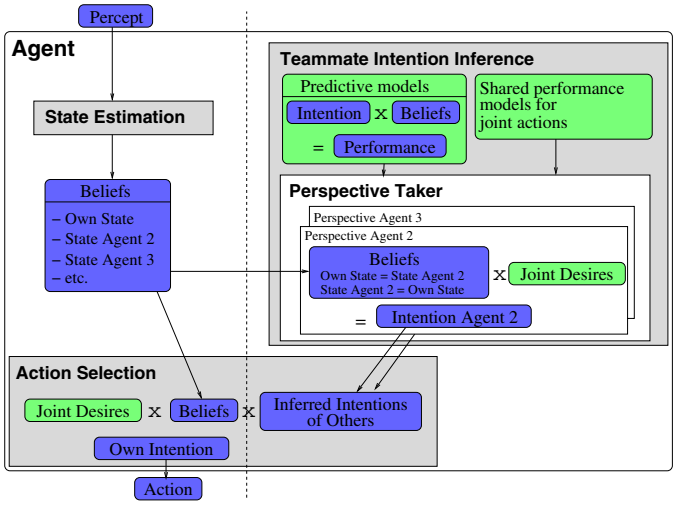

Fig. 3. Computational model of implicit coordination

III. Learning Performance Models

In the ball approach scenario, the utility describing how well a robot can execute this task is time: the faster you can approach the ball, the higher the utility. Therefore, each robot will need to be able to predict how long it will take each robot to approach the ball. These temporal prediction models are learned from observed experience, using model trees and neural networks.

A. Hardware

In this section, we will briefly present the hardware used to conduct the experiments. One of the three field player of our RoboCup mid-size team the 'Agilo RoboCuppers' is shown on the left in Figure 4. The robots are customized Pioneer I robots that use differential drive for locomotion. Two of them use the original Pioneer I controller, and one uses a newer faster Roboteq controller. The robots learn temporal prediction models for both controllers.

A single forward facing CCD camera is used for state estimation. It runs locally on each robot, and yields estimates of the robot's own position, as well as the positions of its team mates, opponent players, and the ball [11]. For the experiments we will present later, it is important that the variables are controllable and reproducible. Therefore, we have used our ground truth system to determine the positions of the robots with even more accuracy. This ground truth system uses three ceiling cameras to detect colored markers on top of the robot. An image of one of the cameras can be seen to the right in Figure 4.

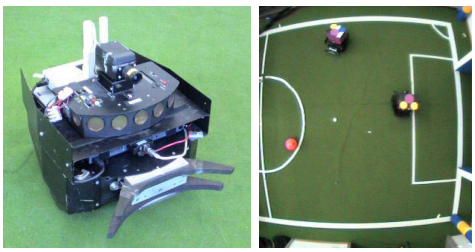

Fig. 4. One of the three customized Pioneer I robots, and an image from the ground truth system.

B. Acquisition of training examples

Examples are gathered by randomly choosing goal destination on the field, approaching them, thereby measuring the time it took for the approach. To acquire sufficient data, we not only record the time from the initial to the goal point, but also from each intermediate point, at a rate of 10Hz. We store the relevant variables from the belief state in a log file. These four variables are: 1) translational velocity, 2) distance to the goal, 3) angle to the goal, and 4) the difference between the robot and goal orientation.

C. Learning methods

Model trees are functions that map continuous or nominal features to a continuous value. The function is learned from examples, by a piecewise partitioning of the feature space. A linear function is fitted to the data in each partition. Model trees are a generalization of decision trees, in which the nominal values at the leaf nodes are replaced by line segments. A benefit of model trees is that they can be transformed into sets of rules that are suited for human inspection and interpretation. For more information about model trees, and how they can be used to learn action models of navigation tasks, we refer to see [13].

In [2], neural networks have been successfully used to learn temporal prediction models of navigation tasks. The neural network we use has four input nodes, one for each of the features described in Section III-B. It has two hidden layers with five nodes each, and one output node: predicted time.

Since it is not clear which learning method is better suited for learning temporal prediction models, we apply both techniques and compare them quantitatively.

D. Evaluation

Both model trees and neural networks were trained with the data acquired on the robots. To analyze how many navigation tasks (or *runs*) are needed to acquire an accurate prediction model we determined the error of the learned models with varying numbers of training runs. In Figure 5 it can be seen how the error decreases as we use more training runs for the Roboteq controller, for both learning methods. The error measure is the mean absolute error between the predicted and the actual time, based on 80 test navigation tasks. This test set is not in the training set. After approximately 300 navigation tasks, adding more examples hardly improves accuracy, and we stopped gathering examples.

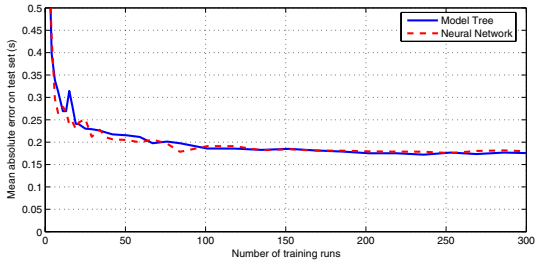

Fig. 5. Dependence of prediction accuracy on number of navigation tasks

Note that the navigation tasks have, on average, a duration of approximately 6 seconds, so 300 training and 80 test runs take about 2300 seconds, or 40 minutes. This is well within the continuous operational range of the robots (battery life-time, etc.). The examples are logged at 10Hz, so 300 navigation tasks yields 18000 (300run$\times$6s/run$\times$10example/s) examples for training the models. The mean absolute error of the models used for the Roboteq controller was 0.176s for the model tree, and 0.180s for the neural network. For the older Pioneer I controller these values are 0.207s and 0.217s respectively.

IV. Implicit Coordination Experiments

To evaluate if the learned prediction models are sufficient for implicit coordination, we have conducted two experiments, one in a dynamic, and one in a static environment. For each experiment, we used three robots, two with the slow controller, and one with the fast controller. Each robot has a temporal prediction model for both controllers, and knows which robot has which controller.

The questions we will answer with these experiments are: 1) Do the robots have accurate estimates of each other's position... 2) and of the time it will take them to reach the goal? 3) Do the robots agree upon who should approach the ball... 4) and did they actually choose the quickest? 5) Are temporal prediction models necessary, or would a more simple value such as distance suffice? 6) How robust is implicit coordination against errors in state estimation? 7) When does implicit coordination fail?

A. Dynamic environment experiment

In this experiment, the robots continuously navigated to randomly generated positions on the field. Once a robot reached its destination, the next random position was generated. These poses were generated such that interference between the robots was excluded. For about half an hour (18 000 examples), the robots perform their random navigation routines. Each robot records the state estimation results locally every 100ms. Figure 6 displays which information was gathered in each log file. Temporal prediction is recorded for both the model trees (`mt`) and neural networks (`nn`). Each robot also records who they think should approach the ball at that time, without ever actually approaching the ball. Before the experiment, the robots synchronize their clocks. The times stamps can therefore be used to merge the three distributed files for further evaluation after the experiment.

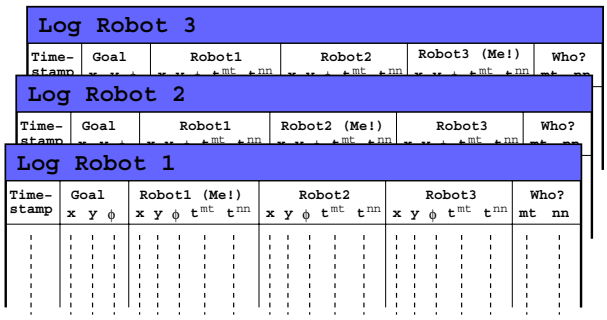

Fig. 6. Visualization of the log-files acquired in the dynamic experiment.

B. Static environment experiment

In the previous experiment, it is impossible to measure if the temporal predictions were actually correct, and if potential inaccuracies caused the robots' estimate of who is quickest to be incorrect. Therefore a second experiment was conducted.

First, the robots navigate to three random positions and wait there. They are then synchronously requested to record the same data as in the first experiment, but only for the current static state. Then, one after the other, the robots are requested to drive to the goal position, and the actual approach duration was recorded. This static environment is less realistic, but allows us to compare the predicted time with the actually measured time for each robot. The log-files are almost identical to the ones in the dynamic experiment. The only difference is that they also contain the actually measured time for the robot, and contain only 200 examples, as we record one example for each episode, and not every 100ms.

C. Results

Q1) *Do the robots have accurate estimates of each other's positions?* Since ground truth provides all the robots with the same information, there are hardly any errors in the robot's estimation of their own and other's state. No system is perfect however, and in Table I we list the small errors that arose. As an example, the value 4.3 in this table means that the distance between the belief of robot 2 (the estimator) about the position of robot 1 (the estimatee), differs 4.3cm from robot 1's own belief of its position. This question, as well as the next two, were answered by using the data of the dynamic experiment.

		Position (cm) Estimator		
		R1	R2	R3
Estimatee	R1	0	4.3	3.9
	R2	2.0	0	1.8
	R3	1.8	1.5	0

TABLE I
Position estimation errors.

Q2) *Do the robots have accurate estimates of the time it will take each other to reach the goal?* Table II is very similar to Table I. However, here we do not list the difference in position estimate, but the difference in temporal prediction, for both models trees and neural networks. The errors are listed in ms, so the robots do indeed have accurate estimates of each other's approach times.

		MT (ms) Estimator			NN (ms) Estimator		
		R1	R2	R3	R1	R2	R3
Estimatee	R1	0	15	22	0	18	26
	R2	10	0	12	16	0	16
	R3	13	12	0	17	15	0

TABLE II
Time prediction errors for Model Trees and Neural Networks

Q3) *Do the robots agree upon who should approach the ball?* To answer this question, we simply determined how often all three robots agreed on which robot should approach the ball. The results are listed in III, in the row labeled "Chose the same robot?". Given the accurate estimates the robots have of each other's states, and the accurate predicted times that arise from this, it should not be surprising that the robots have

almost perfect agreement (>98%) on who should approach the ball.

	Temporal Predictor		
	Model Tree	Neural Network	Distance
Chose the same robot?	99%	98%	99%
Chose the quickest robot?	96%	95%	81%

TABLE III
AGREEMENT AND CORRECTNESS IN IMPLICIT COORDINATION

Q4) *Do the robots choose the quickest one?* Agreeing about who should go to the ball is of little use if the chosen robot isn't actually the quickest. Therefore, we would also like to know if the chosen robot is actually the quickest one to approach the ball. Of course, this could only be determined in the static experiment, in which the actual times it took each robot to approach the ball were recorded. A robot's decision to coordinate is deemed correct, if the robot that was the quickest was indeed predicted to be the quickest. For model trees, the robots were correct 96% of the time, and for neural networks 95%, as can be seen in Table III.

Q5) *Are temporal prediction models necessary, or would a more simple value such as distance not suffice?* Using only distance as a rough estimate of the approach time, as done in [8], would save us the trouble of learning models. Although time is certainly strongly correlated with distance, using distance alone leads to significantly more incorrect coordinations. The last column in Table III shows this. Agreement is still very good (99%), but the robot that is really the quickest is chosen only 81% of the time. So, when using distance, the robots are still very sure about who should approach it, but they are also wrong about it much more often.

Q6) *How robust is implicit coordination against errors in state estimation?* As we saw, almost perfect coordination was achieved in the dynamic experiment. This is not so surprising, as the robots have very accurate estimates of each other's states. To analyze how noise in the estimates of the other robot's states influences coordination, we took the original log files, and added Gaussian noise of varying degrees to the estimates that robots have of each other's pose ($[x_t,y_t,\phi_t]$). The predicted times were then computed off-line, based on these simulated log files.

The results are shown in Figure 7. The x-axis shows the standard deviation of the Gaussian noise added to the data. So the first column, in which there is no added noise, represents the results of the dynamic experiment, which had been listed in Table III. The y-axis shows the percentage of examples in which 0,1,2 or 3 robots intended to approach the ball. Of course, '1' means that coordination succeeded.

We can clearly see that coordination deteriorates when robots do not know each other's states so well. If you have a robotic (soccer) team, and know the standard deviation between the robot estimations of each other's positions, the graph tells you how well implicit coordination would work in this team.

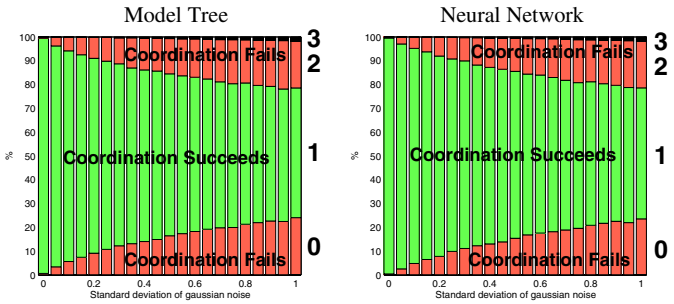

Fig. 7. Influence of simulated state estimation errors on implicit coordination.

Q7) *When does implicit coordination fail?* In our original dynamic experiment, implicit coordination almost never fails, so this question does not really apply to this data. Therefore, we analyzed the log files to which Gaussian noise with a standard deviation of 0.1 was added. In Figure 7, this is the third column in both bar plots. For this noise level, coordination succeeds 90% of the time. In the simulated log file we labeled all examples in which exactly one robot decided to approach the ball with `Success`, and others with `Fail`. A decision tree was then trained to predict this value.

The learned tree is represented graphically in Figure 8. For both prediction models the main rule is that if the difference in predicted times between two robots is small, coordination is likely to fail, and if it is large, it is likely to succeed. This is intuitive, because if the difference between the times is large, it is less likely that adding errors to them will invert which time is the smallest. Note that in between these two limits, there is a 'gray' area, in which some other rules were learned. They only accounted for a small number of example, so for clarity, we will not discuss them here.

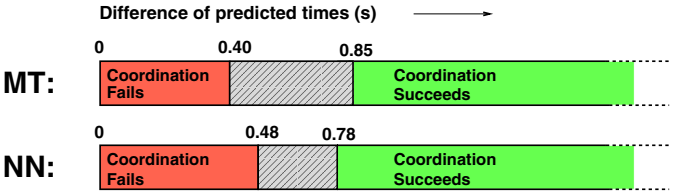

Fig. 8. Representation of the decision tree that predicts coordination success.

Humans also recognize when coordination might fail. For example, in sports like soccer or volleyball, it is sometimes not completely clear who should go for the ball. Humans solve this problem by making a brief exclamation such as "Mine!", or "Leave it!". So in these cases, humans resort to explicit coordination and communicate their intentions. Not only do humans have utility models of each other to coordinate implicitly, they are also aware when confusion might arise. The learned decision tree essentially provides the robots with similar awareness, as they predict when implicit coordination failure is likely. So, they could be used to determine when robots should resort to other methods of coordination. For instance, our robots have a simple locker-room agreement

that when coordination failure is predicted, the robot with the higher number will approach the ball (excluding the goalie).

Finally, in Figure 9, we present an illustration of how the robots coordinate in practice. It is easiest to understand this image if one imagines that the robots are standing still at the drawn positions, and the ball is rolling slowly from left to right. At every 5cm of the ball's trajectory, the robots determine who is quickest to the ball at that time. This robot is connected to the current ball's position by a brighter (green) line. When the decision tree predicts that coordination might fail, the robots between which confusion might arise are both connected to the ball's position by a black line. Note that this image was generated in simulation, not with the real robots.

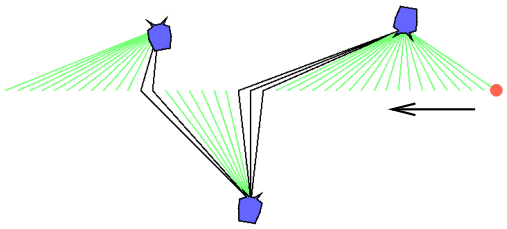

Fig. 9. Example of implicit coordination. Green (bright) lines represent that only one robot would approach the ball at this position. Black lines show when coordination is predicted to likely fail. The robots must all approach the ball from the right.

V. Conclusion and Future Work

In this paper we have presented a computational model of implicit coordination. It requires that agents know each other's state, and that they have performance prediction models of each other. We have learned accurate temporal prediction models using neural networks and model trees, and compared both methods. These temporal prediction models were used to coordinate three robots in a typical robotic soccer task: regaining ball possession. Various experiments were conducted to demonstrate the good performance of implicit coordination.

Up till now the temporal prediction models were learned from experience that was gathered in an environment in which there were no interfering robots. However, in a real game the approach time depends very much on the position of opponent robots. We are currently learning temporal prediction models that include other (opponent) robots.

In soccer, ball approach duration is certainly not the only measure of utility. Who should approach the ball also depends on the different roles the players have, as well as strategic positioning considerations. If I am a defender marking an opponent attacker in my penalty area, it is unwise to approach the ball, just because I happen to be the closest. Including more complex utility models that take strategic considerations into account in our framework is possible, as long as each robot knows the utility models of the others.

In joint research with the University of Ulm and Technical University of Graz, we are forming a heterogeneous soccer team of robots with differing hardware and software systems. The goal is to achieve cooperation without major changes in the software of the individual teams. The first step had been to facilitate belief exchange [14], and currently we are working on realizing implicit coordination, for which we have just acquired the first promising results. Since the robots differ in their dynamics, a prediction model for each type of robot must be learned and distributed before playing as a team.

The inspiration for this research actually arose from the practical implications of this collaboration. By using implicit coordination, each group could implement the abstract idea of the Intentional Stance independently of the software of the others. Therefore, rewriting parts of the different action selection architectures, or implementing negotiation schemes to allow for explicit coordination was not necessary.

Acknowledgments

The work described in this paper was partially funded by the Deutsche Forschungsgemeinschaft in the SPP-1125 "Cooperating Teams of Mobile Robots in Dynamic Environments". We would like to thank Suat Gedikli for helping us with the ground truth system.

References

[1] S. Botelho and R. Alami. M+: A scheme for multi-robot cooperation through negotiated task allocation and achievement. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA99)*, 1999.

[2] S. Buck, M. Beetz, and T. Schmitt. Reliable Multi Robot Coordination Using Minimal Communication and Neural Prediction. In M. Beetz, J. Hertzberg, M. Ghallab, and M. Pollack, editors, *Advances in Plan-based Control of Autonomous Robots.*, Lecture Notes in Artificial Intelligence. Springer, 2002.

[3] C. Castelpietra, L. Iocchi, D. Nardi, M. Piaggio, A. Scalzo, and A. Sgorbissa. Coordination among heterogeneous robotic soccer players. In *Proceedings of the International Conference on Intelligent Robots and Systems (IROS 2000)*, 2000.

[4] L. Chaimowicz, M. Montenegro Campos, and V. Kumar. Dynamic role assignment for cooperative robot. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA02)*, 2002.

[5] D. Dennett. *The Intentional Stance*. MIT-Press, 1987.

[6] M. Dias and A. Stentz. A market approach to multi-robot coordination. Technical Report CMU-RI -TR-01-26, Robotics Institute, Carnegie Mellon University, August 2001.

[7] T. Fong, I. Nourbakhsh, C. Kunz, L. Flückiger, and J. Schreiner. The peer-to-peer human-robot interaction project. *Space*, 2005.

[8] J. Murray and F. Stolzenburg. Hybrid state machines with timedsynchronization for multi-robot systemspecification. In *Proceedings of Workshop on Intelligent Robotics (IROBOT2005)*, 2005.

[9] I. Nourbakhsh, K. Szcara, M. Koes, M. Yong, M. Lewis, and S. Burion. Human-robot teaming for search and rescue. *Pervasive Computing*, 2005.

[10] L. Parker. Alliance: An architecture for fault-tolerant multi-robot cooperation. *IEEE Transactions on Robotics and Automation*, 1998.

[11] T. Schmitt, R. Hanek, M. Beetz, S. Buck, and B. Radig. Cooperative probabilistic state estimation for vision-based autonomous mobile robots. *IEEE Transactions on Robotics and Automation*, 18(5), 2002.

[12] M. Spaan and F. Groen. Team coordination among robotic soccer players. In *Proceedings of RoboCup International Symposium 2002*, 2002.

[13] F. Stulp and M. Beetz. Optimized execution of action chains using learned performance models of abstract actions. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2005.

[14] H. Utz, F. Stulp, and A. Mühlenfeld. Sharing belief in teams of heterogeneous robots. In *Proceedings of RoboCup International Symposium 2004*, 2004.

[15] B. Werger and M. Matarić. Broadcast of local eligibility: behavior-based control for strongly cooperative robot teams. In *AGENTS '00: Proceedings of the Fourth International Conference on Autonomous Agents*, pages 21–22, 2000.