///VSP///

Advanced Robotics 24 (2010) 1017–1036

ADVANCED ROBOTICS
brill.nl/ar

Full paper

Implicit Coordination with Shared Belief: A Heterogeneous Robot Soccer Team Case Study

Freek Stulp [a,*], Hans Utz [b,], Michael Isik [a] and Gerd Mayer [b]**

[a] Intelligent Autonomous Systems Group, Technische Universität München, 85748 Garching, Germany

[b] Department of Neural Information Processing, University of Ulm, 89069 Ulm, Germany

Received 9 June 2009; accepted 15 July 2009

Abstract

A striking aspect of human coordination is that we achieve it with little or no communication. We achieve this implicit coordination by taking the perspective of others and inferring their intentions. In contrast, robots usually coordinate explicitly through the extensive communication of utilities or intentions. In this paper we present a method that combines both approaches: implicit coordination with shared belief. In this approach, robots first communicate their beliefs about the world state to each other, using a CORBA-based communication module. They then use learned utility prediction models to predict the utility of each robot locally. Based on these utilities, an action is chosen. Within a heterogeneous soccer team, with robots from both the Munich and Ulm research groups, we apply implicit coordination with shared belief to a typical task from robotic soccer: regaining ball possession. An empirical evaluation demonstrates that the redundancy of implicit coordination with shared belief leads to robustness against communication failure and state estimation inaccuracy.

Keywords

Multi-robot coordination, robot communication, heterogeneous robot teams, robotic soccer

1. Introduction

A striking aspect of human coordination is that we achieve it with little or no communication. Consider two people assembling a bookcase. With apparent ease, actions are anticipated and coordinated: if I see you grab a screwdriver, I will take

* Present address: Computational Learning and Motor Control Lab, University of Southern California, Los Angeles, CA 90089, USA

** To whom correspondence should be addressed. Present address: USRA/RIACS, NASA Ames Research Center, Moffett Field, CA 94035, USA. E-mail: hans.utz@nasa.gov

DOI:10.1163/016918610X496964

one of the shelves and hold it in place, and so forth. Instead of communicating, humans achieve this by inferring the intentions of others. Once the beliefs and desires of the cooperating party are known, we imagine what we would do in that situation. This is called the 'Intentional Stance' [1].

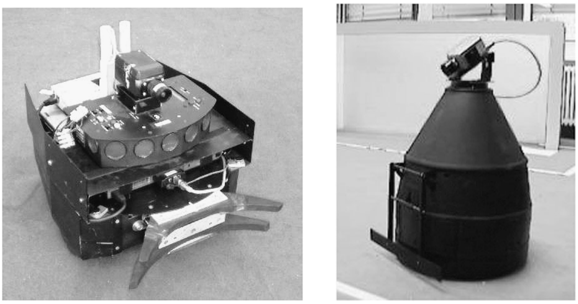
Many complex application tasks in robotics also require the cooperation of two or more robots. A key aspect of these systems is that multiple robots share the same workspace and can therefore not abstract away from the actions of other robots. Therefore, coordination of actions is also essential to solve multi-robot tasks effectively. The standard solution in robotic teams is not to anticipate the actions of others as humans do, but rather to extensively communicate utilities or intentions in a negotiation scheme. Previous work on robot coordination seems to have focussed almost exclusively on explicit coordination, as an overview paper on key architectures shows [2].

Let us illustrate the difference between these coordination methods with a typical task from the robotic soccer domain: regaining 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. 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. A robot should, therefore, only approach the ball if it thinks it has the lowest approach time to the ball. In implicit coordination, each robot would estimate its own approach time, as well as those of other robots, based upon their perceived states. In explicit coordination, on the other hand, each robot only computes its own approach time and communicates this to the other robots or a central arbitrator.

In this paper, we present a third form of coordination that combines these two approaches: implicit coordination with shared belief. It is essentially implicit coordination, in which states of others needed to infer their utilities are not only acquired through state estimation, but also through communication. In this approach, the benefits of implicit and explicit coordination are combined, making it robust against both failure and inaccuracies in both communication and state estimation.

The benefits of implicit coordination were an important motivation for this research. Our goal to form a mixed soccer team with robots from different research groups that demonstrates coordinated behavior was another incentive. The mid-size league soccer teams of our two research groups are described in Refs [3, 4]. One robot from each group is depicted in Fig. 1. In previous work [5], a CORBA-based communication system, also described in this paper, was developed to share beliefs between robots of the two groups. Therefore, implicit coordination with shared belief was the most appropriate coordination method, as this approach alleviates the need to implement utility communication or arbitration mechanisms, and enables each team to implement the 'only the fastest should go for the ball' rule in a way most appropriate for the action selection scheme they use.

In this paper, we apply implicit coordination with shared belief to the ball interception task in a team of robots, formed by robots from both our groups. This



(a) Agilo RoboCupper. (b) Ulm Sparrow.

Figure 1. Robots from the two research groups.

work extends the work presented in Ref. [6] by formalizing the concept of implicit coordination with shared belief, providing results for both a homogeneous and a heterogeneous team, and analyzing the effects of communication and state estimation errors. The main original contributions of this paper are:

- Presenting a computational model for implicit coordination with shared belief.
- The design and implementation of a communication framework for sharing belief within a team of extremely heterogeneous autonomous robots.
- Learning utility prediction models that take technical differences between the robot platforms into account.
- Using shared belief and learned prediction models to enable implicit coordination within a heterogeneous team of robots.
- Conducting a thorough empirical analysis in both dynamic and static scenarios, in simulation and on the real robots.
- Demonstrating that coordination based on shared belief is more robust than explicit coordination.
- Quantatively measuring the effect of state estimation and communication quality on different types of coordination.

The rest of this paper is structured as follows. After discussing related work in the next section, we introduce the computational models of the three types of coordination in Section 3. In Section 4, we describe the design and implementation of the CORBA-based belief communication model. Then, in Section 5, we describe how utility models can be learned and used to predict utilities. An extensive empirical evaluation is provided in Section 6 and we conclude with Section 7.

2. Related Work

Previous research on cooperation has focussed almost exclusively on explicit coordination [2]. On the other hand, work on implicit coordination usually assumes that all agents have access to a central and global representation of the world, which is enabled by simulation, as in Ref. [7], or global perception, as in the RoboCup small-size league [8, 9]. In all these papers, teammates are not reasoned about explicitly, but are considered to be mere environment entities, which influence behavior in similar ways to obstacles or opponents.

Fuji *et al.* [10] use utility communication to coordinate their robots in the mid-size league, using a mixture of system and agent objectives, and priority roles. As there is no quantative analysis of the coordination module or comparison to other coordination methods, it is not clear in which ways this form of coordination contributes to the impressive playing skills of this team. Noma *et al.* [11] propose a similar approach, where the utility being communicated is defined as the value of a Reinforcement Learning (RL) policy. Interestingly, not only the policies are learned with RL, but also the cooperative module that switches between them. States of others are deduced by reconstructing the three-dimensional (3-D) view of other robots by using omni-directional vision. Near-perfect vision is assumed to acquire the global state, which this approach requires.

Most similar to our work is Ref. [12], in which robots in the legged league also coordinate through implicit coordination, which is based on representations that are completed through the communication of belief states. Communication is essential and assumed to be flawless. It is not investigated how communication loss influences coordination. The utility measure is a sum of heuristic functions, which are represented as potential fields. Whereas our utility models are grounded in observed experience and have a well-defined meaning (e.g., execution duration in seconds), these heuristic functions have no clear semantics. Therefore, customizing these functions to individual robots is difficult, as the semantics of and interactions between them are not fully understood. However, this customization is essential for achieving efficient coordination in a heterogeneous team with robots with different dynamics and capabilities.

In the RoboCup simulation league, agents have been coordinated with coordination graphs [13]. These graphs specify in which situations agents need to coordinate their behavior. On-line, agents must have knowledge of each others' states to apply these coordination graphs, so agents can only coordinate if they are in each others' field of view. In Ref. [14], simulated soccer players use local information to estimate the global state of the team, i.e., attacking, recovering, etc. This global state is used to set global playmode parameters. The arbitration uses these playmode parameters to initialize coordinated behaviors. These approaches have not been evaluated on real robots.

In Ref. [15], human communication is divided into explicit (spoken word) and implicit (gestures, facial expression or any non-vocal expression of intention). Here, robots use explicit communication (consisting of direct commands to other robots) to coordinate high-level tasks and implicit communication (being position information) to avoid collision (called modest coordination). In our terminology, modest coordination with implicit communication is not implicit coordination, as the position information about other robots is acquired through communication, not state estimation.

The most similar mixed team cooperation effort was the Azzurra Robot Team, a mid-size team from various Italian universities. They also used a (proprietary) publisher/subscriber communication protocol, utilizing UDP. This team used explicit coordination (i.e., with utility communication) to assign roles among the field players [16]. Unfortunately, the Italian national team was dissolved after the RoboCup tournaments in 2000.

3. Computational Model

In this section, we introduce the computational models of three types of coordination. The ball interception task will be used as an illustrative example.

3.1. Explicit Coordination

The computational model of explicit coordination is depicted in Fig. 2. Previous work on cooperation seems to have focussed almost exclusively on this form of coordination as an overview paper on the key architectures shows [2]. It has also been used in the RoboCup mid-size league to allocate roles and coordinate behavior [16–18].

In explicit coordination, robots compute only their own utility locally. Each robot then sends its utility to the other robots, and receives the utilities of the other robots, over a communication channel. In auction-based approaches, the utilities are sent to a single arbitrator, which communicates roles or actions back to the robots. The utilities in the ball interception task are the ball approach times of the robots. The faster a robot can approach the ball, the higher the utility.

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. The joint utility model formalizes the intuitive rule that only one robot should approach the ball. It computes the best action a robot can execute, given its own utility for this action, as well as the utilities of other robots. So, to avoid interference of robots in the ball interception task, the joint utility model returns the action approachBall if a robot predicted to be the fastest to approach the ball, and another action otherwise.

Of course, all soccer teams will have implemented this strategy in some way, to avoid all robots continuously pursuing the ball. The contribution of the approach

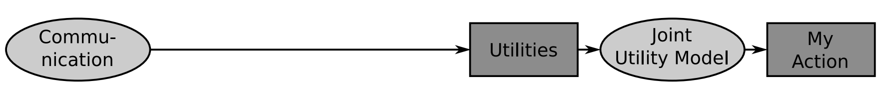


Figure 2. Explicit coordination, in which the utilities of other robots are communicated. This is the standard approach in robotics.

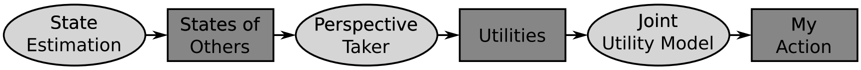


Figure 3. Implicit coordination without communication, in which utilities are computed from states using utility prediction models. States are determined through state estimation. Humans usually use this approach to coordinate.

presented here is not to implement the concept of having only one robot going there. It rather demonstrates how exploiting utility models to reason about the outcome of the actions of others enables robots to become more independent of communication for coordination.

3.2. Implicit Coordination

Implicit coordination, depicted in Fig. 3, is a variation of explicit coordination, in which the utilities of others are not communicated, but computed by the robot itself. There are many domains in which implicit coordination is used by humans: team sports, construction (e.g., of bookcases) and also in traffic.

The robot does achieve implicit coordination by first acquiring the states of others through state estimation. In the belief state of the soccer robots, robot states are represented by a pose: the position and orientation of the robot. The perspective taker enables the robot to make utility predictions for other robots. To do this, the robot swaps its own state with that of another robot in the belief state and computes the utility. This 'perspective taking' [19] is performed for all other robots, until the utilities for all robots are known.

There are several important benefits that implicit coordination without communication has over explicit coordination:

- Protocols and arbitration mechanisms must be adopted between communicating entities to enable utility communication and arbitration, which adds complexities and can degrade the system. It is generally argued that communication can add unacceptable delays in information gathering and should be kept to a minimum [8].
- 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.
- A current research focus in cooperation is human–robot interaction, e.g., in space exploration [20] or rescue robotics [21]. 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, it cannot be expected of humans to continuously communicate their intentions or utilities. Instead, the robot must be able to anticipate a human's intentions, based on predictive models of human

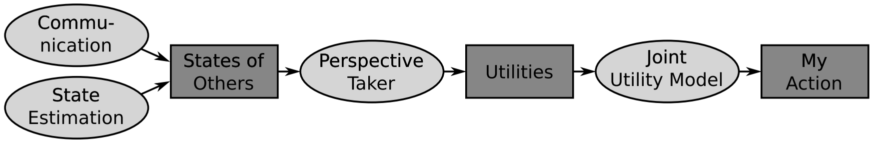


Figure 4. Implicit coordination with belief communication.

behavior. We consider implicit coordination to be essential for natural interaction between robots and humans, so adhering to explicit coordination will prevent robots from making a breakthrough into these application domains.

3.3. *Implicit Coordination with Shared Belief*

The most difficult aspect of implicit coordination is estimating the states of others. Especially due to the limited sensory capabilities of most of todays robot platforms, this is problematic. Therefore, we use communication of beliefs as an alternative to estimating the states of others based on perception. The computational model of this implicit coordination with shared belief (IC-SB) is depicted in Fig. 4.

3.3.1. *Belief* versus *Utility Communication*

This computational model might seem contrary to our communication-free paradigm, but there is an important difference between communicating utilities and communicating beliefs, which we shall explain in this section. Of course, implicit coordination without communication is the ideal situation, which we cannot achieve due to limitations in sensors and state estimation. Still, implicit coordination with state communication is preferable over explicit coordination for the following reasons:

- Since explicit coordination is only possible if you know the utilities of others, delays or failures in utility communications will often cause complete coordination failure. With implicit coordination, the robot can still rely on its own sensors and state estimation to deduce the utilities of others. Coordination might then not be perfect, due to sensor limitations, but at least it does not collapse completely. One of the experiments in the experimental evaluation will verify this (Q6 in Section 6.2). In a sense, combining the two methods exploits the best of both worlds.

- Improvements in sensor technology and state estimation methods will allow robots to autonomously acquire a increasingly complete and accurate estimation of the states of others. In RoboCup, for instance, almost all mid-size teams have resorted to omni-directional vision to achieve exactly that. Thus, beliefs needed to infer the utilities of others are becoming more complete and accurate, independent of communication. More accurate state estimation essentially reduces communication needs. Teams that have omni-directional vision could probably abandon communication altogether when using implicit coordination. This is certainly not the case for explicit coordination, which will always fully rely on communication.

- To enable human–robot cooperation, robots will at some point have to rely on state estimation only, as humans cannot be expected to communicate their state. Implicit coordination with shared belief is an intermediate step to this ideal situation.

A disadvantage is that belief communication requires more information to be communicated. The pose, velocity and covariance matrix of each observed object is significantly more data than than a single utility value. For the application domain this is not a big handicap though. Assuming a 2-D pose/velocity estimate is sufficient for a soccer robot, the dynamic pose can be formulated in 24 single precision floating point numbers. The belief state information of a five against five players match amounts to about 1 kB of belief state data that needs to be communicated by each player. This amount of data fits well into a single UDP packet and, published at a moderate frequency (i.e., 1 Hz), does not excessively stress the available network bandwidth. Furthermore, the robots do not only exchange significantly more data. They exchange significantly more information, which in turn is used to improve the individual robots utility estimate.

Summarizing, the robots use communication as a backup system for incomplete state estimation, rather than as the backbone of their coordination. With utility communication, robots are completely dependent on continuous and robust communication. In contrast, when sharing belief, improvements in sensor and state estimation will allow implicit coordination to depend less and less on belief communication. This is necessary to simplify communication schemes, increase coordination robustness and enable human–robot cooperation. This work proposes a step in this direction.

For objects that are observed by multiple robots, various information fusion approaches can be applied. The model of the Ulm Sparrows robots for fusion of the ball position was, for instance, to always trust their own observations above shared observations, but use averaging and threshold-based outlier rejection, to fuse multiple observations of the same object from teammates.

So far, we have not specified how robots communicate belief states or compute utilities. Sections 4 and 5 will elaborate on their implementations.

4. Sharing Belief

To share beliefs, the teams must agree upon structures that encapsulate the information they want to exchange and the communication framework over which this information will be sent. The next two sections are dedicated to these topics.

4.1. Design of Belief Exchange

The three main elements of the exchanged belief state are a time-stamp, the probabilistic dynamic pose of the robot itself and a list of observed objects. The main design principle is that we want an expressive belief exchange structure, but not simply a superset of all the information that the teams have used so far.

(i) *Time-stamped message-based communication.* The beliefs of the robots are exchanged within the team by a message-based protocol. Messages that are corrupted by packet loss in the wireless network do not influence each other and can safely be ignored. As a basic requirement for sharing information in a highly dynamic environment each message is accurately time-stamped. This allows for interpolation of the beliefs to minimize the estimation error between messages.

(ii) *The own dynamic pose.* Many sensor data processing algorithms assume that the probability of measuring a certain quantity is distributed according to a Gaussian normal distribution. We also use this concept, and represent the robot's pose by a 3-D vector (x, y, α) and covariance matrix. The vector represents the mean of the Gaussian distribution, with $(0, 0, 0)$ being the center of the field, facing the opponent goal. The covariance matrix holds the associated uncertainty as a 3-D Gaussian distribution around this mean. How these values can be computed is discussed more elaborately in Ref. [22]. This probabilistic representation allows robots to communicate not only where they belief they are, but also how certain they are about it.

Apart from its pose, the robot also communicates its velocity as a tuple (v_x, v_y, ω), without uncertainty measures. The combination of pose, covariance matrix and speed is called 'probabilistic dynamic pose'.

(iii) *Observed objects.* Apart from their own pose, the shared belief state also contains a representation of all the observations of objects, such as the ball. Each object is represented by the same dynamic pose and covariance matrix, which have been discussed in the previous section. Although this representation is somewhat redundant (goal posts will never have a velocity), it is very general. The observed objects are projected in an egocentric frame of reference, in which the observer always has pose $(0, 0, 0)$.

Design issue: allocentric versus *egocentric.* The most obvious way to fuse observations of different robots is by the use of a shared frame of reference. Therefore, the robots use the allocentric frame of reference defined in the previous section to communicate their poses. However, this approach requires knowing where you are with sufficient accuracy, something that cannot always be guaranteed. For many tasks (shooting a goal, passing to a teammate) it is sufficient to know the relative location of certain objects to the robot. For these reasons we have decided to use an egocentric frame of reference for the observations.

4.2. Implementation of the Communication Framework

The team communication uses a message-based, type safe high-level communications protocol that is transfered by IP-multicast, as such a protocol keeps the communicated data easily accessible and prevents subtle programming errors that are hard to trace through different teams. As the communication in a team of autonomous mobile robots has to use some kind of wireless LAN, which is notoriously unstable especially in RoboCup tournaments, a connection-less message-based protocol is mandatory. This way, network breakdowns and latencies do not block the sending robot. To save bandwidth, IP-multicast is used, since in this way each message has only to be broadcasted once, instead of n times for n clients.

The implementation uses the notify multicast module (NMC) of the Middleware for Robots (Miro) [23]. Miro provides a generalized CORBA-based sensor and actuator interfaces for various robot platforms as well as higher-level frameworks for robotic applications. Additionally to the method-call-oriented interfaces, Miro also uses the event driven, message-based communications paradigm utilizing the CORBA Notification Service. This standardized specification of a publisher/subscriber protocol is part of various CORBA implementations [24]. Publishers (suppliers in their terminology) offer events. The so-called consumers subscribe to those events. They then receive the events supplied by the publisher through an event channel (EC). The data exchanged is specified in the CORBA interface definition language (IDL). Standardized mappings from IDL to most modern programming languages (C, C++, Java) exist.

CORBA uses a connection-oriented (TCP/IP-based) communication layer by default, the NMC module therefore plugs into the Notification Service architecture and exchanges events between the robots of a team transparently, using IP-multicast. For this purpose a service federation quite similar to the one described in Ref. [25] is used. A local instance of the EC runs on each robot. A 'NMC event consumer' subscribes for all events that are offered only locally but subscribed by other teammates and sends them to the multicast group. A 'NMC event supplier' in turn listens to all events published *via* IP-multicast and pushes those into the local event channel, which are subscribed but not offered locally. To keep track of the offered and subscribed message types, NMC utilizes two fields of the standard event message format: the domain name and the type name. By convention, the domain name contains the name of the robot producing the event. The type name describes its payload. As these fields are also part of the native offer/subscription management and filtering protocol of the notification service, robots can easily determine whether events they offer are currently subscribed in the team and skip their production entirely if there are no subscribers.

Figure 5 illustrates a sample configuration of the notification channel setup. Two robots (A, B) produce two types of events $(1, 2)$; the resulting events are $\{$A1, A2, B1, B2$\}$. The events in the supplier and consumer boxes denote the offered and subscribed events. The events labeling the arrows denote the actual flow of events. Note that suppliers and consumers can offer/subscribe for multiple events.

Communicating the IDL-specified belief state discussed in Section 4.1 at 10 Hz with all teammates uses, on average, less than 10% of the available bandwidth of a standard 802.11b WLAN (11 MBit/s). This should be available, even on heavily loaded networks, such as those in RoboCup tournaments.

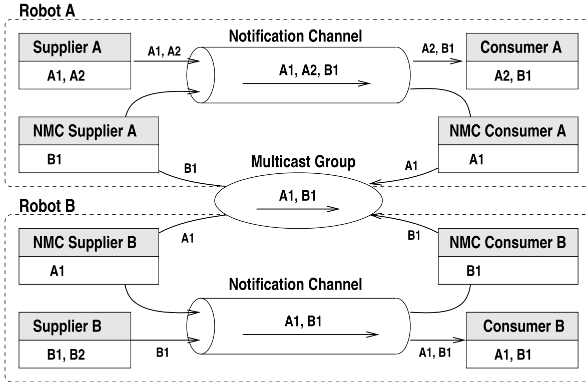


Figure 5. A federated notification channel setup.

5. Learning Utility 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.

In the first step, training examples are gathered by randomly choosing goal destination on the field, approaching them and measuring the time taken 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 10 Hz. The first 30 examples (3 s) of each episode are used as training data. We store the relevant variables from the belief state in a log file. These four variables are: (i) translational velocity, (ii) distance to the goal, (iii) angle to the goal, and (iv) difference between robot and goal orientation.

Then, model trees are trained with this data to acquire a model that generalizes over unseen cases. 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 utility prediction models of navigation tasks, we refer to Ref. [26].

Table 1 lists the number of episodes n needed to acquire an accurate prediction model. The mean absolute error (MAE) is computed over a separate test set, which contains $n/3$ episodes. Training data was gathered until the error stabilized.

The advantage of this learning approach over analytical methods is that it is based on real experience, and therefore takes all factors relevant to execution duration into account. If one robot is slower than the other or has a slight bias to the right due to

Table 1.
Accuracy of the learned models

Robot	n	Duration	MAE
Agilo (real)	290	0:31	0.31 s
Ulm Sparrow (real)	517	0:40	0.32 s
Agilo (simulated)	750	1:18	0.21 s

slight damage to the left motor, this will all be reflected in the observed data and, thus, in the model learned from the data. The model is therefore tailored specifically to each robot, using the same general experience-based learning approach. For instance, when new motor control boards were installed on the Agilo RoboCuppers, they were twice as fast as the old ones and had very different dynamics. We then simply gathered new data, trained the learning algorithm and acquired a model tailored to the new boards. We have also learned temporal prediction models for very different robots altogether, e.g., a B21 service robot, or Powercube arms [26].

Another advantage is that many hand-coded actions are difficult to formalize analytically, as is well summarized by the following quote on navigation actions: "Navigation behavior is the result of the subtle interplay of many complex factors. These factors include the robot's dynamics, sensing capabilities, surroundings, parameterizations of the control program, etc. It is impossible to provide the robot with a deep model for diagnosing navigation behavior" [27]. Our approach does not rely on accurate analytical models of the robot; the model is learned from observed data. In addition, analysis can be impossible because the inner workings of the action are simply unknown, e.g., when observing the behavior of other robots (or humans). In principle, learning models can also be done on-line, so that action models can adapt to changing environments.

6. Empirical Evaluation

We will now present the experimental design and the results of conducting these experiments.

6.1. Experimental Design

To evaluate if the learned prediction models and shared representations are sufficiently accurate for implicit coordination, we have conducted three experiments — one in a dynamic environment, one in a static environment and one in simulation.

6.1.1. Dynamic Experiment

In this experiment, the robots continuously navigated to random targets on the field, for about 30 min, as depicted in Fig. 6. The paths were generated such that interference between the robots was excluded. At 10 Hz, each robot records its own position and orientation, as well as that of its teammate and the ball. Each robot

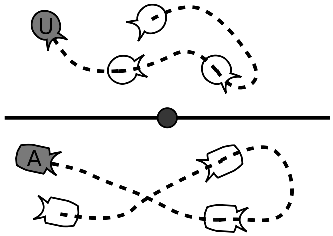


Figure 6. Dynamic experiment.

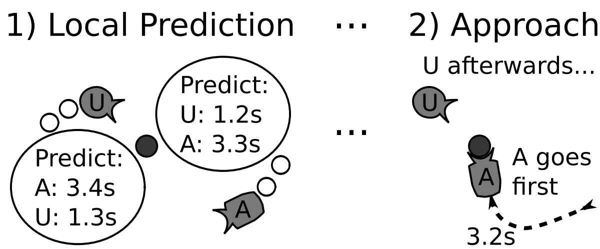


Figure 7. Static experiment.

also logs the locally predicted approach time for both robots and, based on these times, which robot it thinks should approach the ball. Note that the robots never actually approach the ball.

This experiment, as well as the next, was conducted with three Agilo robots, and also with one Agilo robot and an Ulm Sparrow.

6.1.2. Static 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 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 recorded, as in Fig. 7. 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, not every 100 ms.

6.1.3. Simulated Experiment

Here, the experimental setup is identical to the dynamic experiment. It is conducted with two simulated Agilo robots. The simulator allows us to vary two variables that most strongly influence the success of implicit coordination, as depicted in Fig. 8. The first is communication quality. At random times, and for random durations,

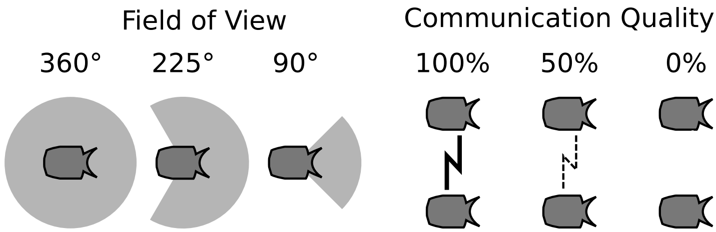


Figure 8. Simulated experiment.

Table 2.
Agreement and correctness in implicit coordination

	Learned model (%)		Distance (%)	
	Three Agilo	Mixed	Three Agilo	Mixed
Chose the same robot?	99	96	99	95
Chose the quickest robot?	96	92	81	68

communication is switched off in both directions. By controlling the length of the intervals, we can vary between perfect (100%) and no (0%) communication. The second is the field of view of the robot. We can set the view angle of the robot's forward facing camera between 0 (blind) and 360° (omni-directional vision). The other robot and the ball are only perceived when in the field of view. Gaussian noise with a standard deviation of 9, 25 and 22 cm is added to the robot's estimates of the position of itself, the teammate and the ball, respectively. These correspond to the errors we have observed on the real robots.

6.2. Results

The experiments presented in the previous section allow us to answer several questions relevant to the robustness and accuracy of implicit coordination with shared belief.

(Q1) *Do the robots agree upon who should approach the ball?* To answer this question, we determined how often all robots agreed on which robot should approach the ball. The results are listed in Table 2, in the row labeled ‘Chose the same robot?’. The result for both the homogeneous (three Agilo robots) and heterogeneous (one Agilo and one Ulm Sparrow) teams are listed. 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 (>99 and >96%) on who should approach the ball.

(Q2) *Do the robots choose the quickest one?* Agreeing about who should go to the ball is of little use if the chosen robot is not 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. The robots' choice was correct 92% of the time, as can be seen in Table 2.

(Q3) *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 Ref. [28], 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 2 shows this. Agreement is still very good (99 and 95%), but the robot that is really the quickest is chosen only 81 and 68% 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.

(Q4) *How robust is implicit coordination against errors in state estimation?* As we saw, almost perfect coordination was achieved in the dynamic experiment. To analyze how noise in the estimates of the other robot's states influences coordination, we took the original log files of the homogeneous team and added Gaussian noise of varying degrees to the estimates that robots have of each other's pose ([x_t, y_t, ϕ_t]). The predicted times were then computed off-line, based on these simulated log files.

The results are shown in Fig. 9. 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 2. The y-axis shows the percentage of examples in which zero, one, two or three Agilo robots intended to approach the ball. Of course, '1' means that coordination succeeded.

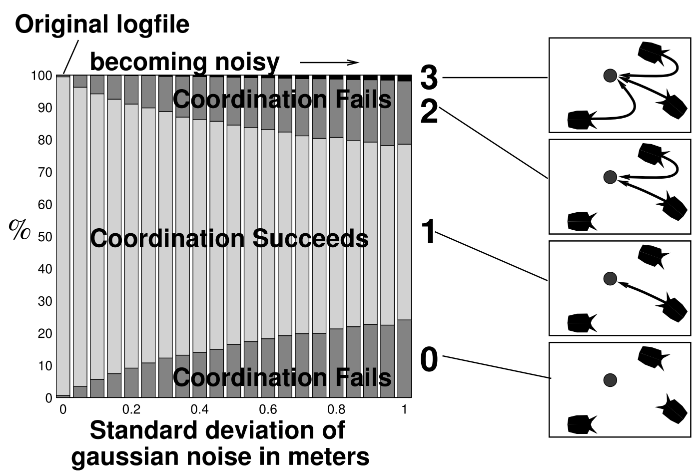


Figure 9. Influence of simulated state estimation errors on implicit coordination.

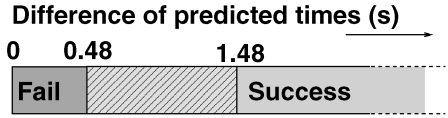


Figure 10. Visualization of the decision tree that predicts coordination failure.

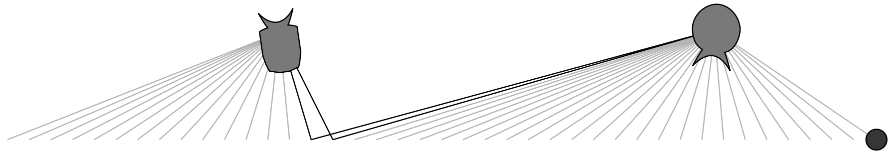


Figure 11. Example of implicit coordination. Bright lines represent that only one robot would approach the ball at this position. Black lines show when coordination is predicted to likely fail. The robots must all approach the ball from the right.

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.

(Q5) *When does implicit coordination fail?* In the dynamic experiment with the heterogeneous team, coordination succeeds 96% of the time. In the 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 Fig. 10. The main rule is that if the difference in predicted times between two robots is small, coordination is likely to fail; 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 estimation errors 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 examples, so for clarity, we will not discuss them here.

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!'. Thus, 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).

In Fig. 11 we present an illustration of how the robots could use this model in practice. It is easiest to understand this image if one imagines that the robots

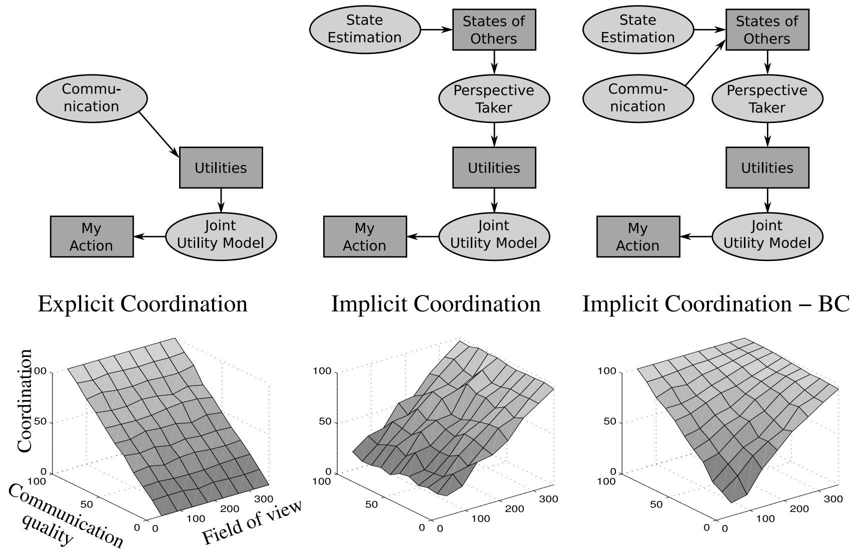


Figure 12. Results of the simulation experiment, which show how the performance of coordination strategies depends on the quality of communication and the field of view.

are standing still at the drawn positions and the ball is rolling slowly from left to right. At every 5 cm 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 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.

(Q6) *How do communication quality and state estimation accuracy influence coordination?* The results of the simulation experiment, which show how the performance of different coordination strategies depends on the quality of communication and the field of view, are depicted in Fig. 12. Communication quality is the percentage of packets that arrive and field of view is in degrees. The z-axis depicts coordination success, which is the percentage that only one robot intended to approach the ball.

Since explicit coordination is based completely on communication, it is not surprising that it perfectly correlates with the quality of the communication, but is independent of the size of the field of view. No communication means no coordination and perfect communication means perfect coordination. For implicit coordination without communication, the relation is converse. If a robot is able to estimate the states of others better, it is able to coordinate better. The third graph shows implicit coordination with belief state exchange, as used on the robots. If the robot has a teammate in its field of view, it determines the other's state through state estimation, otherwise it uses communication (if possible) to exchange beliefs. These states are then used to predict the utilities of others, independent if they were perceived or communicated. The graph clearly shows that this approach combines the benefits of both.

The third graph in Fig. 12 clearly shows that implicit coordination with belief exchange achieves better performance with communication loss than explicit coordination alone, as discussed in Section 3.3. Instead of complete coordination failure in the case of communication loss, there is a graceful decay, because a second system based on state estimation can still be used to estimate the intentions of others.

7. Conclusions

In this paper we discussed belief sharing — a communication principle that combines implicit and explicit communication models for teams of robots in cooperative application domains. Instead of voting protocols, the exchange of observations allows robots to fuse their beliefs in a non-centralized manner and individually select actions that result in implicit team cooperation. The implementation of the communication infrastructure was discussed as well as learning utility models for actions in a heterogeneous robot soccer robot team. Our approach is general and versatile, as both the experience-based learning of utility models and the CORBA-based communication do not depend on the specific robot hardware and software. This has enabled us to apply our approach to two very different mid-size league robots and create a heterogeneous team.

Performance and reliability were evaluated empirically. The conducted experiments demonstrated that this coordination scheme is more robust in the presence of communications errors than explicit communication.

Acknowledgements

The research described in this paper is partly funded by the German Research Foundation (DFG) in the SPP-1125, ‘Cooperating Teams of Mobile Robots in Dynamic Environments’, and also by the CoTeSys cluster of excellence (Cognition for Technical Systems, http://www.cotesys.org), part of the Excellence Initiative of the DFG.

References

1. D. Dennett, *The Intentional Stance*. MIT Press, Cambridge, MA (1987).
2. B. P. Gerkey and M. J. Matarić, Multi-robot task allocation: analyzing the complexity and optimality of key architectures, in: *Proc. IEEE Int. Conf. on Robotics and Automation*, Taipei, pp. 3862–3868 (2003).
3. M. Beetz, T. Schmitt, R. Hanek, S. Buck, F. Stulp, D. Schröter and B. Radig, The AGILO robot soccer team — experience-based learning and probabilistic reasoning in autonomous robot control, *Autonomous Robots* **17**, 55–77 (2004).
4. G. Kraetzschmar, G. Mayer, H. Utz, P. Baer, M. Clans, U. Kaufmann, M. Lauer, S. Natterer, S. Przewoznik, R. Reichle, A. Reisser, A. Roth, M. Schmidt, C. Sitter, F. Sterk and G. Palm, The Ulm Sparrows (2003), in: *Proc. RoboCup Symp.*, Padova, pp. 238–249 (2004).

5. H. Utz, F. Stulp and A. Mühlenfeld, Sharing belief in teams of heterogeneous robots, in: *Proc. RoboCup Symp.*, Lisbon, pp. 508–515 (2004).
6. M. Isik, F. Stulp, G. Mayer and H. Utz, Coordination without negotiation in teams of heterogeneous robots, in: *Proc. RoboCup Symp.*, Bremen, pp. 355–362 (2006).
7. S. Sen, M. Sekaran and J. Hale, Learning to coordinate without sharing information, in: *Proc. 12th Natl Conf. on Artificial Intelligence*, Seattle, WA, pp. 426–431 (1994).
8. A. Tews and G. F. Wyeth, Thinking as one: coordination of multiple mobile robots by shared representations, in: *Proc. Int. Conf. on Robotics and Systems*, Takamatsu, pp. 1391–1396 (2000).
9. M. Veloso, P. Stone and M. Bowlin, Anticipation as a key for collaboration in a team of agents: A case study in robotic soccer, in: *Proc. of SPIE Sensor Fusion and Decentralized Control in Robotic Systems II*, Boston, MA, pp. 134–141 (1999).
10. H. Fujii, M. Kato and K. Yoshida, Cooperative action control based on evaluating objective achievements, in: *Proc. RoboCup Symp.*, Osaka, pp. 208–218 (2005).
11. K. Noma, Y. Takahashi and M. Asada, Cooperative/competitive behavior acquisition based on state value estimation of others, in: *Proc. RoboCup Symp.*, Atlanta, GA, pp. 101–112 (2007).
12. D. Vail and M. Veloso, Multi-robot dynamic role assignment and coordination through shared potential fields, in: *Multi-Robot Systems: From Swarms to Intelligent Automata II*, A. Schultz, L. Parker and F. Schneider (Eds), pp. 87–98. Kluwer, Dordrecht (2003).
13. J. R. Kok, M. T. J. Spaan and N. Vlassis, Non-communicative multi-robot coordination in dynamic environments, *Robotics Autonomous Syst.* **50**, 99–114 (2005).
14. E. Pagello, A. D'Angelo, F. Montesello, F. Garelli and C. Ferrari, Cooperative behaviors in multi-robot systems through implicit communication, *Robotics Autonomous Syst.* **29**, 65–77 (1999).
15. M. Rude, T. Rupp, K. Matsumoto, S. Sutedjo and S. Yuta, Iron: An inter robot network and three examples on multiple mobile robots' motion coordination, in: *Proc. Int. Conf. on Intelligent Robots and Systems*, Grenoble, pp. 1437–1444 (1997).
16. C. Castelpietra, L. Iocchi, D. Nardi, M. Piaggio, A. Scalzo and A. Sgorbissa, Coordination among heterogenous robotic soccer players, in: *Proc. Int. Conf. on Intelligent Robots and Systems*, Takamatsu, pp. 1385–1390 (2000).
17. M. T. J. Spaan and F. C. A. Groen, Team coordination among robotic soccer players, in: *Proc. RoboCup Symp.*, Fukuoka, pp. 409–416 (2002).
18. K. Yokota, K. Ozaki, N. Watanabe, A. Matsumoto, D. Koyama, T. Ishikawa, K. Kawabata, H. Kaetsu and H. Asama, Uttori united: cooperative team play based on communication, in: *Proc. RoboCup Symp.*, Paris, pp. 479–484 (1999).
19. J. G. Trafton, N. L. Cassimatis, M. D. Bugasjska, D. P. Brock, F. E. Mintz and A. C. Schultz, Enabling effective human–robot interaction using perspective taking in robots, *IEEE Trans. Syst., Man Cybernet.* **35**, 460–470 (2005).
20. T. Fong, I. Nourbakhsh, C. Kunz, L. Flückiger and J. Schreiner, The peer-to-peer human–robot interaction project, in: *Proc AIAA Space 2005*, Long Beach, CA, AIAA-2005-6750 (2005).
21. I. Nourbakhsh, K. Szcara, M. Koes, M. Yong, M. Lewis and S. Burion, Human–robot teaming for search and rescue, *Pervas. Comput.* **4**, 72–78 (2005).
22. T. Schmitt, R. Hanek, M. Beetz, S. Buck and B. Radig, Cooperative probabilistic state estimation for vision-based autonomous mobile robots, *IEEE Trans. Robotics Automat.* **18**, 670–684 (2002).
23. H. Utz, S. Sablatnög, S. Enderle and G. Kraetzschmar, Miro — middleware for mobile robot applications, *IEEE Trans. Robotics Automat.* **18**, 493–497 (2002).
24. D. C. Schmidt, A. Gokhale, T. Harrison and G. Parulkar, A high-performance endsystem architecture for real-time CORBA, *IEEE Commun. Mag.* **14**, 72–77 (1997).

25. T. H. Harrison, D. L. Levine and D. C. Schmidt, The design and performance of a real-time CORBA event service, in: *ACM SIGPLAN Notices* **32**, 184–200 (1997).
26. F. Stulp and M. Beetz, Refining the execution of abstract actions with learned action models, *J. Artif. Intell. Res.* **32**, 487–523 (2008).
27. M. Beetz, *Plan-based Control of Robotic Agents: Improving the Capabilities of Autonomous Robots*. Springer, Berlin (2002).
28. J. Murray and F. Stolzenburg, Hybrid state machines with timed synchronization for multi-robot system specification, in: *Proc. Workshop on Intelligent Robotics*, Covilha, pp. 236–241 (2005).

About the Authors

Freek Stulp is a Postdoctoral Researcher at the Computational Learning and Motor Control Lab at the University of Southern California. His research interests include robotics, motor primitives, robot planning and model fitting for image understanding. He has pursued these interests at the University of Edinburgh, the Instituto Superior Técnico (Lisbon) and the Technische Universität München (where the work described in this paper was performed). He received his Doctorandus degree from the University of Groningen, in 2001, and his Doctorate from the Technische Universität München, in 2007. The topic of his dissertation was tailoring robot actions to task contexts using learned action models.

Hans Utz is a Staff Scientist for the Research Institute for Advanced Computer Science (RIACS) and works as a Robotic Software Architect in the Intelligent Robotics Group at NASA Ames Research Center (ARC). His work is concerned with the development of advanced software architectures for autonomous mobile robotics. Before joining NASA ARC, he was a Research Associate at the Department of Neuroinformatics, University of Ulm, Germany (where the work described in this paper was performed). He designed the robotics middleware Miro and coached the Ulm Sparrows team, a project on ‘autonomous mobile robotics in highly dynamic environment’, which is better known as robot soccer. He received his Doctorate in Computer Science from the University of Ulm, Germany, in 2005.

Michael Isik received his Diploma (Master) degree in Computer Science at the Technische Universität München, in 2007, and subsequently worked on the Pybrain project at the Cobotlab at the same university. He is currently pursuing his PhD studies at the Bio-Inspired Information Processing Department at the Technische Universität München. His research interests include computational neuroscience, machine learning and robotics.

Gerd Mayer received his Doctorate at the University of Ulm, in 2007, for his work in robotic vision with an emphasis on attention control, object detection, neural classification methods and middleware support for image processing tasks. His research interests also include cognitive neuroinformatics and computational vision on which he focused during his time at the Ludwigs-Maximilian University in the Institute of Medical Psychology. Currently, he is working for Harman Becker Automotive Systems as a software developer. His work encompasses embedded systems, in-car network communication and real-time processing in the CoC ‘Speech and Connectivity’.