Dealing with Perception Errors in Multi-Robot System Coordination

Alessandro Farinelli and **Daniele Nardi**
Dip. di Informatica e Sistemistica,
University of Rome, "La Sapienza",
lastname@dis.uniroma1.it

Paul Scerri
Robotics Institute,
Carnegie Mellon University,
pscerri@cs.cmu.edu

Alberto Ingenito
Dataspazio S.p.A.,
Alberto.Ingenito@dataspazio.it

Abstract

A prerequisite to efficient behavior by a multi-robot team is the ability to accurately perceive the environment. In this paper, we present an approach to deal with sensing uncertainty at the coordination level. Specifically, robots attach information regarding features that caused the initiation of a course of action, to any coordination message for that activity. Further information regarding such features, acquired by the team, are then combined and the expected utility of the started action is re-evaluated accordingly. Experiments show that the approach allows to coordinate a large group of robots, addressing sensing uncertainty in a tractable way.

1 Introduction

Emerging, large multi-robot teams hold great promise for revolutionizing the way some important, complex and dangerous tasks, such as disaster response[1] and space exploration[2] are performed. Such teams, consist of multiple, heterogeneous robots with imperfect sensors, which must act efficiently in the face of considerable uncertainty and time pressure. Specifically, in many complex environments, robots have systematic sensor noise that leads individual robots to incorrect perception of the objects in the environment. This paper presents an approach that leverages multi-robot cooperation to overcome individual sensing limitations.

Dealing with sensing uncertainty is a key area of robot and agent research. A variety of techniques attempt to either reduce the uncertainty before deciding how to act, e.g., Bayesian filters [Fox *et al.*, 1999], or explicitly deal with the uncertainty when choosing a course of action, e.g., POMDPs [Theocharous *et al.*, 2001]. Such approaches, require that each robot deals with its sensor noise on its own, thus failing to leverage any assistance the rest of the team might be able to supply. Other work explicitly uses input from other members of the team to allow a more accurate picture of the environment to be created. For example, DEC-POMDPs have been used to devise a coordinated course of action [Pynadath and Tambe, 2002]. Several approaches use cooperative perception to deal with perception limitation of the single robot [Rosencratz *et al.*, 2003; Dietl *et al.*, 2001; Stroupe *et al.*, 2001]. The general idea of these approaches is to exchange sensor readings and aggregate them using different filtering techniques (e.g. Kalman filters [Dietl *et al.*, 2001; Stroupe *et al.*, 2001] or particle filters [Rosencratz *et al.*, 2003]).

Coordination in Multi-Robot Systems has been successfully addressed using task assignment approaches [Parker, 1998; Werger and Mataric, 2000; Zlot *et al.*, 2002]. While task assignment has been deeply investigated in several scenarios and many different techniques have been proposed, little attention has been devoted to the impact that limited and noisy perception capabilities have on the task allocation process. In many applications, tasks to be allocated are created by the robots when particular features are extracted from the environment. Such feature extraction process is subject to errors, which can greatly impact the overall task assignment process. Cooperative perception techniques can be used to address this problem, however previous approaches to cooperative perception often require to exchange too many data among robots. A key reason for this is that, typically, each robot attempts to maintain an accurate model of the complete state when, in practice, only a small part of the overall state might be relevant to its activities.

This paper presents an approach to deal with perception errors by exchanging information about observed *features* in a task assignment coordination framework. This approach comprises two key ideas, first, when a robot senses an event in the environment that might lead it to initiate a team activity, it *immediately* computes expected utility of action and inaction, given its confidence in its sensors. If the expected utility of acting is greater than that of not acting, it initiates the team activity. For example, if a team of robots is searching a sparsely occupied burning office building, any sensors suggesting the presence of a trapped civilian should immediately trigger a team response. Over time, new information acquired by the team can impact the confidence in the occurrence of the event. At any point in time, if robots' confidence in the event drops sufficiently low, the team activity can be suspended. The idea of the approach is to balance the need for a rapid response in a time critical domain and the high costs of acting on incorrect sensor information.

[1] see http://www.ai.sri.com/project/Centibots
[2] see http://www.cs.cmu.edu/∼fire/

Once a team activity has been initiated, messages are typically passed around the team to coordinate the activity. The second key idea to the proposed approach is to require that these communication messages include the event that led to the team activity being initiated. Such information is considered as a justification for the team activity to be carried out. Robots receiving such messages check whether they have any information that can impact confidence in the occurrence of the event. If so, the information are communicated to the robot initiating the action. This technique focuses the information gathering process on events that are important for team activities and, thus, prevents communication about irrelevant aspects of the state. Notice that, in order to refute justifications, robots need to maintain a history of the areas they have observed so that they can report *not* seeing something at a particular location. In this way, uncertainty in robot perception is addressed at coordination level, focusing only on information which are relevant to robot mission and thus dramatically reducing the required communication overhead.

To evaluate the effectiveness of the proposed approach, experiments were performed both in an abstract simulation environment and in a simulation framework based on Player/Stage[3]. The proposed approach was shown to perform almost as good as an approach that broadcasts every detected features to every other robots, while using orders of magnitude less communication bandwidth. Unsurprisingly, the proposed approach did not perform as well when the environment was very sparse, because fewer robots had helpful information, or when the environment was very dynamic because information from other robots were soon out-of-date.

2 Problem

This section formally describes the problem addressed by this paper. Mobile robots $R = \{r_1, \ldots, r_m\}$, with imperfect sensors, act in an environment with events $E = \{e_1^t, \ldots, e_n^t\}$. The events are spread spatially across the environment. $e_i^t = true$ iff event i is observable at time t. The robots make observations $O_e^{[+|-]}$, where O_e^+ corresponds to a positive reading for event e and O_e^- corresponds to a negative reading. The probability of a false positive reading is $P(O_e^+|\neg e)$. Notice that, false positive (or false negative) should be intended not as direct sensor readings, but as errors in the feature extraction process. The proposed binary model for robot observations is therefore well suited and general enough to represent such issues. Moreover, such model can consider both errors due to systematic issues with sensors, and random errors of the feature extraction process.

The prior probability of $e_i^t = true$ is written $P(e_i^t) \rightarrow [0, 1]$. This is the probability an event occurs before robots acquire any observation. Robot r estimate of $e_i^t = true$, given acquired observation, is $Bel_r(e_i^t) \rightarrow [0, 1]$. The world changes dynamically according to $P(e_i^{t+1}|e_i^t)$ and $P(\neg e_i^{t+1}|\neg e_i^t)$. The *start time* of an event $t_{e_i}^s$ to be t such that $\neg e_i^{t-1} \wedge e_i^t$ and the *end time* of the event $t_{e_i}^f$ to be t such that $e_i^t \wedge \neg e_i^{t+1}$.

[3]see playerstage.sourceforge.net/

When events occur, the robot team should take actions in response. The function $A_e(t) \rightarrow \{0, 1\}$ returns 1 if the team is acting in response to event e at time t and returns 0 otherwise. Informally, the team receives reward (α), for acting when an event is real, and costs for either not acting when the event is real (β_2), or acting when it is not (β_1). Formally, the problem for the team is to maximize:

$$\begin{array}{c} \sum_{e \in E} \alpha \int_{t_e^s}^{t_e^f} A_e(t) + \\ \sum_{e \in E} -\beta_2 \int_{t_e^s}^{t_e^f} (1 - A_e(t)) + \\ \sum_{e \in E} -\beta_1 \left(\int_{-\infty}^{t_e^s} A_e(t) + \int_{t_e^f}^{\infty} A_e(t) \right) \end{array} \quad (1)$$

To maximize expected utility, the team should act on e_i only when:

$$\alpha Bel(e_i^t) - \beta_1(1 - Bel(e_i^t)) > \beta_2 Bel(e_i^t). \quad (2)$$

2.1 Example

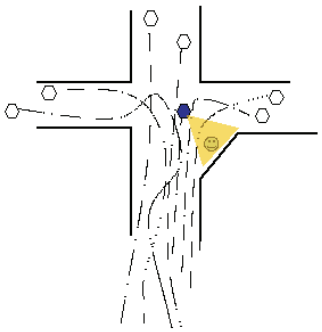

Figure 1: Example of scenario where cooperative perception can enhance system performance

Figure 1 shows an example of the type of situation that cooperative teams of robots should avoid at low cost. Unfilled hexagons show mobile robots performing search and rescue in an office-like environment. Dotted lines indicate the paths the robots have taken through the environment. The filled hexagon is a robot that believes it sees an injured victim, indicated by the emoticon. It should consider initiating a joint activity to rescue the victim. However, there is not actually a victim at that location. Since six other robots have recently passed that same location without sensing a victim the team should be able to quickly establish that the filled hexagon robot is experiencing a perception error and avoid the costs of the joint activity.

3 Approach

The key idea to this approach is to use input from team mates to update confidence in reasons for acting, while starting actions on initial observations. The technical elements of the approach are: i) attach to coordination messages assumptions that justify a coordinated action; ii) add observations to coordination messages when relevant; iii) revise decisions to act

based on observations attached from teammates to coordination messages.

When a robot detects a new event, it performs an expected utility calculation to decide whether to act in response to that event (see Equation 2). If it decides to start a coordinated action, it attaches the justification for that coordinated action to the coordination messages. When a robot receives a coordination message, it evaluates whether it has relevant observations about that event. If it is the case, it will attach to the coordination message the relevant observations and it will send a *cooperative perception* message back to the robot that initiated the action. Finally, when the robot that instantiated a coordinated action receives a cooperative perception message, it integrates all observations attached to the message and re-considers the expected utility, while continuing to perform the coordinated action. In particular, if robot r receives a cooperative perception message at time t related to the event $e_i^{t'}$ (where $t' < t$), it will update $Bel_r(e_i^t)$ using the observations attached to the message. With the update $Bel_r(e_i^t)$ robot r will then re-evaluate the expected utility of continuing the activity.

World representation

To support or refute observations performed by other team members, robots have to store their previous observations. In particular, their representation should include not only observed relevant features, but also in which parts of the environment they did not observe any relevant features. Thus each robot maintains three main elements: i) a Set of *Interesting Points* (IPS); ii) a representation of the *Observable Space* (OS) iii) a set of *positive observations* (PO). An interesting point is a portion of space where an event was detected at a given time step. Each entry of the (IPS) comprises: i) information characterizing the location to which the event is related (e.g., x and y for a bi-dimensional representation); ii) a numerical value (belief) representing the robot's estimates that the observation is correct.

The OS is dependent on the robot's sensors, but in general it is a representation of the portion of the environment that was observed by the robot's sensors at a given time step. The PO is the set of relevant features that were detected at each time step. Given an event e_i detected by a robot r, robot r' will perform the following operations to support or refute the event.

For each entry of the OS history:

- If the portion of space related to e_i is inside the current OS and the corresponding PO set contains an observation for that feature, attach the observation to the *cooperative perception* message (supporting the event detection).
- If the portion of space related to e_i is inside the current OS but the current PO set does not contain any observation for that feature, create a negative observation and attach the observation to the *cooperative perception* message (refuting the event detection).
- If the portion of space related to e_i is not inside the current OS do nothing.

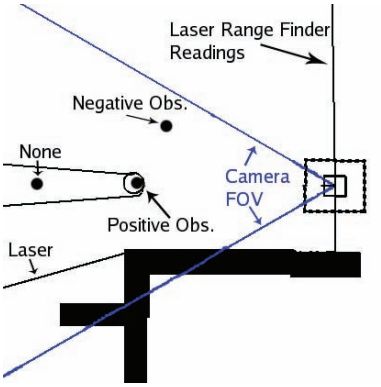

Figure 2: Supporting and refuting observations

Notice that, to relate observations to detected features (or events), the data association problem has to be solved (i.e. which feature is related to which observation). Data association is a well known problem for robotic and tracking applications (see for example [Hall and Llinas, 2001]). Several approaches have been used to address this problem, however, since our method is based on a feature-level representation, and the data association issue is not the main focus of our work, a distance threshold is adopted. Other approaches to address the data association problem can be used; in fact, the decision process we described is general with respect to the chosen data association method.

Figure 2 provides a graphical representation of supporting and refuting observations, showing a robot with the current laser readings and the field of view of the camera. The OS, in this case, is the intersection between the laser reading and the camera field of view. The point labeled as *none* in the figure is a point for which the robot cannot provide any information, because it can not observe that part of the environment.

Bayesian update of robot knowledge

In this work, the belief update of each robot is done using a Bayes filter. Specifically, a Bayes filter is instantiated for each relevant detected event in the environment (Equations 3 and 4).

$$Bel(e_i^t) = \eta \prod_j Pr(O^j_{e_i^{t'}} | e_i^t) Bel'(e_i^t) \quad (3)$$

$$Bel'(e_i^t) = \sum_{e_i^{t-1}} Pr(e_i^t | e_i^{t-1}) Bel(e_i^{t-1}) \quad (4)$$

Since readings $O^j_{e_i^{t'}}$ obtained from team mates may be older than present time $t' < t$, they cannot be directly integrated using the filter equation, because, referring to a time step in the past, they should have influenced the robot's belief up to the present time. When a reading referring to a past time t' is obtained the filter is reinitialized with a state $Bel(e_i^{\bar{t}})$ where $\bar{t} = Max\{t \mid e_i^t \in IPS \wedge \bar{t} < t'\}$. If the event location is not inside the IPS a new interesting point is added to the IPS. Observations for the locations are generated using the PO and the OS. Maintaining the history for OS, PO

and IPS for all the process has a cost in terms of memory that grows with time. To limit such cost, a valid time window T is used that goes from the current time t_c back to $t_c - T$. An appropriate time window can be chosen by considering the evolution model of the environment.

4 Experiments and Results

To evaluate the approach, we have tested our method both in an abstract simulation environment and in a simulation framework based on Player/Stage. The former simulator has been used to test our method with a very large team of robots (100 team members). Moreover, the behavior of our method has been tested under varying environmental conditions such as the world dynamism and the world size. The latter simulation environment allowed us to consider important issues such as sensor occlusion or message delay among team members. Overall, the first experimental setting allows to test the general behavior of the method, while the second set of experiments provides more accurate indications on specific robotic issues. Two metrics were used to measure performance: the percentage of stopped actions out of the total number of actions incorrectly instantiated (*percentage found*, pFound on graphs) and the percentage of correctly stopped actions out of all the actions that were stopped (*percentage good*, pGood on graphs). In both cases, higher is better. Notice that, the *percentage good* measure is related to the *correctness* of the approach while the *percentage found* measure is related to the *completeness*. These metrics are the key values underlying Equation 1 and hence, with knowledge of the relative costs of action and inaction, allow for the computation of the team performance. In the experiments presented below, each robot initiates a coordinated action according to Equation 2.

The communication overhead is evaluated using two measures: i) number of messages exchanged at each time step by each robot; ii) size of the messages (in bytes) exchanged at each time step by each robot. We assume that the overhead of a broadcast message is higher than the overhead of a point to point message. In particular, we count a broadcast message as point to point message times the number of robots. While for a more precise analysis of the overhead one should consider the specific network used, this provides a general cost model for communication which is suitable for our level of analysis.

In our experiments we used a task assignment algorithm based on token passing [Scerri *et al.*, 2005]. *Tokens*, representing tasks to be accomplished, get propagated through the team until a robot accepts the task. The algorithm has been chosen, because it requires a very low communication overhead and is thus well suited for our interest domain.

The proposed approach (referred as *ShareRelInfo*[4] in the following) was compared to a benchmark strategy, called *Share All*, where each robot shares all its observations with all other robots at each time step. Clearly, this type of approach is infeasible for large teams, but it provides an upper bound on the performance that can be achieved by the cooperative perception process.

Experiments have been performed in a 2D office-like environment. To exchange information about features present in the environment, robots need to share a common reference framework. To simplify the experimental setting, we do not explicitly consider localization errors. As a matter of fact, standard localization techniques [Fox *et al.*, 1999] can be used for our experimental scenario, and localization errors can be taken into account in the error model of the feature extraction process. Each graph reports values averaged over 10 trials of the same experiment.

[4]Because it shares only information relevant to the tasks.

4.1 Abstract Simulator Results

The abstract simulator captures key features of the environment, while being sufficiently abstract to test a wide range of parameters and configurations efficiently. The simulated robots have limited knowledge of the overall team state and can communicate only with a subset of the overall team. In each experiment there were 100 simulated robots, each with the same perception model.

Figure 3 shows that *Share All* does perform better and the advantage increases as the world becomes more dynamic. However, for less dynamic environments *ShareRelInfo* performs almost as well. For this team size (100 robots) the communication gain of *ShareRelInfo* is approximately two orders of magnitude.

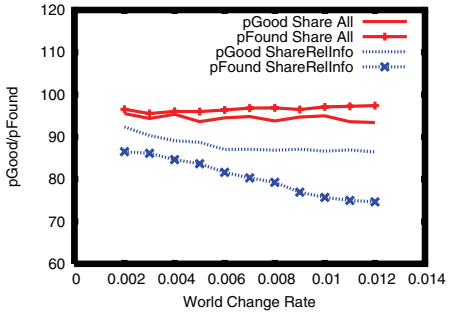

Figure 3: Comparison between *Share All* and *ShareRelInfo*, varying world dynamics

Figure 4 shows performance as the size of the environment is varied, while keeping the number of robots and the sensor range constant. The performance decreases as the robot density becomes lower. This is because, when the density of robots decreases there will be less opportunity for mutual observations of same features, therefore less information for supporting or refuting observations can be provided.

4.2 Player/Stage results

In the Player/Stage experimental framework each robot is equipped with a laser range finder and a color camera. Robot controllers are run as distributed processes over a network of PCs and messages are exchanged using the UDP protocol. In this configuration, we can validate the coordination method with possible message loss and delay. Objects of various colors are distributed over the map. The goal of the team is to detect objects and locate them in the map (Figure 5 represents the reference experimental scenario).

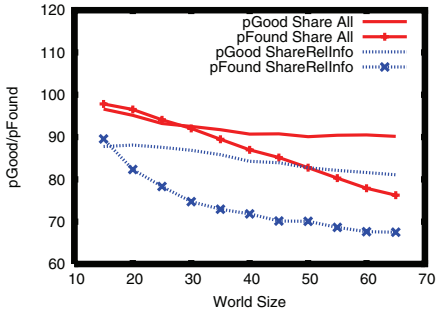

Figure 4: Comparison among *Share All* and *ShareRelInfo*, varying world size

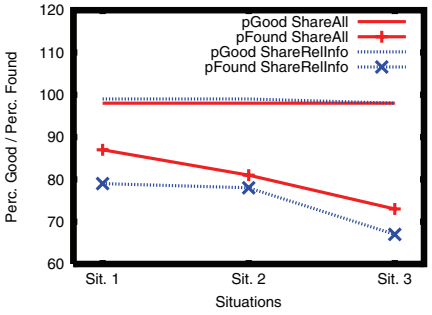

Figure 6: Performance comparison among *Share All* and *ShareRelInfo* in the tree different configurations

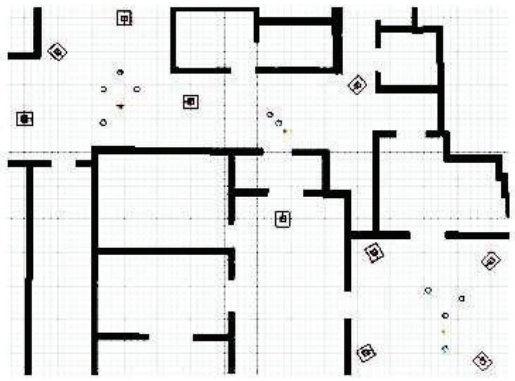

Figure 5: Reference experimental scenario

The approach was tested in three different configurations, where the object and robot positions are different. Specifically, the first configuration comprises one group of eight robots which observe the same group of objects. In this configuration the number of shared observations is very high. However, due to occlusions in the visual field the observations for each robot are not exactly the same. The second configuration includes two groups of robots. The first one is composed of eight robots which observe the same group of objects, the second group is composed of two robots which observe a group of other two objects. In this configuration the two groups of robots do not share any observations. Finally, the third configuration (shown in Figure 5) shows three groups of robots, in this case the number of shared observations among groups is very limited.

Figure 6 reports results obtained for the two methods in the three described configurations. Figure 6 shows that the lower number of shared observation has a negative impact for both strategies and both measures. The *completeness* (pFound in graph) is more disadvantaged by the lower availability of shared observations than the *correctness* (pGood in graph). This can be explained by considering that, when there are fewer shared observations, there will be a lower chance that a robot will obtain enough information to stop an invalid action. On the other hand, once the relevant information has reached a robot, the chances to stop an invalid action remain almost constant.

ShareRelInfo achieves results which are very close to the *Share All* strategy. As for *completeness*, the proposed method attains lower values and is more sensitive to the differences among configurations. This is due to the smaller amount of information that this method uses. In fact, when fewer observations are shared among robots, there is a lower chance that coordination messages reach robots that can provide relevant information to refute or support coordinated actions. However, the performance decrease is minimal while the gain in communication is very high (see below).

As for *correctness*, *ShareRelInfo* attains slightly better results than the *Share All* strategy. This can be explained by considering how actions are created and stopped in the two policies. In the *ShareRelInfo* a coordinated action can be stopped only when a message containing observations of a subset of the teammates is received and the computation of Equation 2 indicates that the action should be stopped. In the *Share All* strategy the policy to stop actions is different: each robot monitors, at each time step, the belief associated with features that originated corresponding actions, and actions are stopped according to Equation 2. In this way, if at some time step a subset of the robots experiences an error in the perception process for the same feature, the corresponding action will be stopped immediately. Since the proposed approach integrates measures over time before stopping a coordinated action, it is less sensitive to this problem.

Figure 7 and 8 show that the proposed approach not only requires a lower number of messages, but ensures also a smaller communication overhead in terms of message size. Since the communication overhead does not change significantly across the different configurations, we report results referred to one particular configuration (configuration 2).

The graphs show the number of messages and communication load for different world change rate. For this team size (10 robots) the communication gain is approximatively one order of magnitude. When the world change rate is lower, less coordinated actions will be instantiated. In such a situation, *ShareRelInfo* requires a lower communication overhead, while the communication overhead for the *Share All* strategy remains almost constant.

Notice that, having a smaller communication overhead can

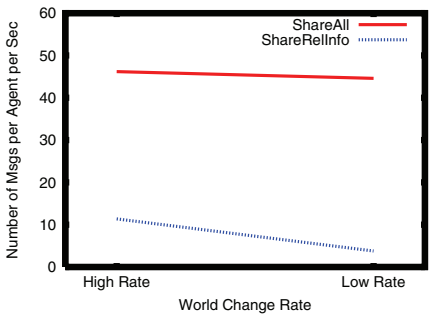

Figure 7: Communication comparison for different world change rate (number of messages)

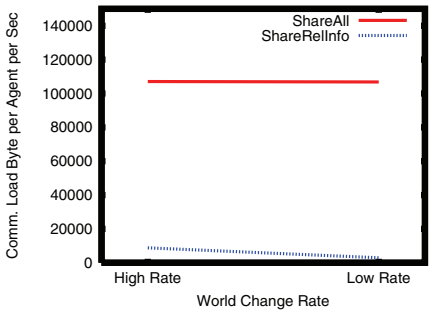

Figure 8: Communication comparison for different world change rate (communication load)

be exploited to enhance the robustness of the system. For example, it could be possible to use a reliable communication protocol (e.g., based on acknowledgment) to avoid or reduce message loss. Moreover, the available bandwidth could be allocated to other processes to have a better coordination among team members (e.g., to exchange the planned path to avoid collision or complex maneuvers).

5 Conclusions

In this paper we proposed a novel approach to deal with distributed unreliable perception in dynamic environments. The approach enables robots to integrate previously made sensor readings to help refute incorrect sensing. The novelty of the proposed approach lies in dealing with perception errors at coordination level. This is achieved by introducing an abstract representation of the state and by sharing information driven by events.

While obtained results refer to a simulated environment and thus deserve further investigation, this work is a first important step to explicitly consider uncertainty at the coordination level in a tractable way.

Addressing the problem of sensor inaccuracy at the coordination level provides several advantages: First of all the method is general and does not strictly depend on the tasks to be performed and on the types of sensors used. Moreover, it uses an explicit representation of beliefs about the world, which is shared among team members. This could enable team members to reason about their states. For example, a very interesting issue for future work would be to analyze team member perception failures, based on coordination messages, and change the team coordination policy accordingly.

6 Acknowledgment

Alessandro Farinelli is supported by the European Office of Aerospace Research and Development under grant number 053015. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the European Office of Aerospace Research and Development.

References

[Dietl *et al.*, 2001] Dietl, J. S. Gutmann, and B. Nebel. Cooperative sensing in dynamic environments. In *Proc. of Int. Conf. on Intelligent Robots and Systems*, 2001.

[Fox *et al.*, 1999] D. Fox, W. Burgard, and S. Thrun. Markov localization for mobile robots in dynamic environments. *Journal of Artificial Intelligence Research*, 11:391–427, 1999.

[Hall and Llinas, 2001] D.L. Hall and J. Llinas, editors. *Handbook of Multisensor Data Fusion*. CRC Press, 2001.

[Parker, 1998] L. E. Parker. ALLIANCE: An architecture for fault tolerant multirobot cooperation. *IEEE Transactions on Robotics and Automation*, 14(2):220–240, April 1998.

[Pynadath and Tambe, 2002] D. V. Pynadath and M. Tambe. The communicative multiagent team decision problem: Analyzing teamwork theories and models. *Journal of Artificial Intelligence Research*, 16:389–423, 2002.

[Rosencratz *et al.*, 2003] M. Rosencratz, G. Gordon, and Thrun S. Decentralized sensor fusion with distributed particle filters. In *UAI-03*, 2003.

[Scerri *et al.*, 2005] P. Scerri, A. Farinelli, S. Okamoto, and M. Tambe. Token approach for role allocation in extreme teams. In *In Proc. of AAMAS 05*, pages 727–734, 2005.

[Stroupe *et al.*, 2001] A.W. Stroupe, M.C. Martin, and T. Balch. Distributed sensor fusion for object position estimation by multi-robot systems. In *Proc. of Int. Conf. on Robotics and Automation (ICRA2001)*, volume 2, pages 1092–1098, 2001.

[Theocharous *et al.*, 2001] Georgios Theocharous, Khashayar Rohanimanesh, and Sridhar Mahadevan. Learning hierarchical partially observable markov decision processes for robot navigation. In *IEEE Conference on Robotics and Automation , (ICRA)*, Seoul, South Korea, 2001.

[Werger and Mataric, 2000] B. B. Werger and M. J. Mataric. Broadcast of local eligibility for multi-target observation. In *DARS00*, pages 347–356, 2000.

[Zlot *et al.*, 2002] R. Zlot, A Stenz, M. B. Dias, and S. Thayer. Multi robot exploration controlled by a market economy. In *Proc. of the Int. Conf. on Robotics and Automation (ICRA'02)*, pages 3016–3023, 2002.