# Multi-Robot Coordination in the Robot Soccer Environment

**Ashley Tews and Gordon Wyeth**
Computer Science and Electrical Engineering
University of Queensland
Australia
{tews, wyeth}@csee.uq.edu.au

## Abstract

MAPS (Multi-Agent Planning system) is a system developed for multi-agent coordination developed for use in the robot soccer domain. The system operates without explicit communication of strategy between agents, relying upon observation of team members to produce meaningful coordinated behaviour. Each robot is coordinated with the others by MAPS choosing goal actions that most benefit the team. The choice of goal action is based on the robot's observations of the rest of the team and their behaviours. The various observations are modelled by superposition of potential fields, where the fields represent the influence of the team, other robots, and components of the environment. This paper outlines the system as it functioned in the UQ RoboRoos entry in the real robot small-size league of RoboCup '98. Testing and competition results show that this coordination model keeps the robots in good field position in both attack and defence. The improved field position is shown to provide better opportunities for improving ball position and shooting at goal, as well as preventing collisions between robots.

## 1 Introduction

Coordinating robots in a dynamic environment is a difficult task. They must be able to carry out their contributions to the overall goal of the system efficiently and effectively while not impeding each other. The focus of multi-robot coordination should therefore be twofold: each robot should consider the objectives of the team while maintaining its own functional integrity. As such, the team plan should exist at a level where it provides strategies for each robot to contribute to the teams success. Each robot must consider the strategy it has been allocated and execute it as best it can without compromising its ability to maintain functional operation.

MAPS (Multi-Agent Planning System) is a method developed for multi-robot control. MAPS performs high level multi-robot planning by generating an abstract representation of the robot's environment at a particular point in time. This representation is built from the robot's perception of the world. In particular, the representation accounts the position and behaviour of other robots in the environment, either on the same team or those in opposition. This abstraction is carried out using potential fields. By modelling features of the game from each robot's perspective, strategic commands and coordinates are extracted and proposed to the robots. The robot's low-level system considers MAPS requests and attempts to fulfil them as closely as possible, given further constraints of dynamic motion and obstacle avoidance. These constraints are particularly evident in the MAPS test environment: robot soccer.

### RoboCup as a Test Environment

RoboCup is an international event where teams of robots play soccer. The event consists of several categories with different requirements for each, in terms of embodiment, size and sensing ability. This paper deals with robots developed for the small size category. Small size teams consist of five robots that play with a golf ball on a walled field the same size as a table tennis table. External features of the teams generally consist of an overhead camera for visual feedback, a computer for processing the visual information and determining gameplay, and some form of communication system between the computer and the robots. There are no restrictions on the system design except for the size of the robots.

While reliable electronic and mechanical hardware form part of the engineering solution, the challenge is centred on the design of suitable software. The software problem can be separated into two main areas of concern:

1. the control loop for the robots, and
2. the design for multi-robot cooperation.

The control loop consists of components on-board and off-board the robot to provide an information loop to monitor and control the robots. Designing an efficient control loop is difficult since it contains time-intensive and quality trade-off aspects such as communication to the robots and image interpretation from the camera information. The time delays inherent in these systems make effective control of the robots difficult. This problem is exacerbated by the highly dynamic nature of robot soccer.

This paper concentrates on the second problem, multi-robot cooperation. It describes the implementation of

MAPS, and evaluates its performance in the robot soccer domain. The system was used on the UQ RoboRoos robot soccer team that came second in the small size league of RoboCup '98 in Paris and third at the Pacific Rim Series of RoboCup in 1998.

The multi-robot cooperation strategy is analysed in Section 2 and shows examples of system performance as well as the algorithms used. Section 3 describes the methods used to determine the integrity of the multi-agent approach in the RoboRoos system and presents the results. Section 4 presents a summary of the concepts in this paper.

# 2 Multi-robot Cooperation

Any multi-agent cooperation system requires complex coordination strategies to gain the full benefit of applying more than one agent to a problem. With mobile robotic environments, this has added difficulties due to the idiosyncrasies of both the hardware and software running the hardware. In the robot soccer environment, it includes communication delays, vision system integrity, inconsistent field conditions, and other robots. Since it must be accepted that many of these idiosyncrasies are beyond control, a robust multi-robot control system needs to be developed.

## 2.1 Defining the Problem

There are many problems to be addressed when designing a multi-robot strategy for teamwork. Having multiple moving objects in the environment adds dimensions of complexity to path planning. The robots can't learn where the obstacles will be and must contend with them reactively while maintaining a longer-term strategic plan. The more moving objects in the environment, the more difficult it is to carry out team strategies. This is contradictory to how a multi-agent system should benefit from added agents.

When designing a multi-agent system, it is difficult to assess which components should be modified for improvement [Mataric, 1997, Murciano *et al.*, 1997]. Each agent has a responsibility to carry out its task to benefit the team. When the team doesn't perform well, it can be difficult analysing where the problem is. The agents may have incorrect strategies or the overlying team strategy could be at fault.

Mataric [1996, 1997], Claus and Boutilier [1997] have suggested communication as a possible solution in certain ideal environments. In environments such as robot soccer which has centralised vision however, communication is not necessarily beneficial. Each agent on the field can obtain enough information about the state of the environment not to require any extra information from other agents. Adding communication can also cause delays in agent reaction as they may wait on incoming information before deciding their next move [Sen *et al.*, 1994].

As a result of these problems, many multiagent designs are environment or problem specific and the technique adopted in one environment may not be transportable to another. However, using controlled environments enables a better focus on the design of the multiagent strategy rather than the effects of the environment. The robotic soccer domain is a good example of a constrained environment for this development. MAPS has been developed within the constraints of the robot soccer environment focusing on multiagent cooperation. The subsequent section describes the MAPS system.

## 2.2 MAPS Overview

MAPS consists of a high-level planning algorithm which analyses gameplay and sends commands and coordinates (directives) to the robots' command interpreters. It is the robots' responsibility to carry out this plan as best as they possibly can. This method of planning is similar to the plan-executor method described by Agre and Chapman [1990] where the planner contains the strategy for the team while the executor exists as the command interpretation mechanism onboard the robots.

Agre and Chapman state that there are two types of planning: hard-wired planning and abstract. The hard-wired planning consists of the agents' actions being predefined with little contingency inclusion. The planner gives the agents directives and if they can't carry them out, the plan fails. Contrasting to this is the abstract planning method defined as 'plan-as-communication'. With this method, the agents are given a high level version of the plan and execute it if they can or decide on a different approach if they deem it more applicable. Hence, the agents have more control in the planning process.

MAPS provides a "plan-as-communication" to the robots. It does not rely upon the robots to complete the directives, and continues to observe them after it has been issued. As the robots attempt to carry out their directives, a new plan may emerge of which MAPS may take advantage.

The three stages of the MAPS algorithm can be summarised as:

1. **Get new information**: Update the world model based on new sensor data or prediction of agent behaviour.
2. **Choose an action**: Based on the influence of the components of the environment (typically agents and obstacles) choose an action type. The action type should designed to reduce interference between team members. Action choice is built from the perspective of the goal of the team.
3. **Find the location for the action**: Based on the action type, the influences of the environment's components can be constructed to show where a robot should perform the action. The choice of location is built from the perspective of the individual robot and its action.

In these steps, there is no mention of an explicit team strategy or a coordinated plan. The choice of action and location for that action emerges from the influences of the various components of the environment. The construction of these influences and the manner in which they interact is the key to successful coordination. MAPS uses potential fields for this mechanism.

**Potential Fields**

Potential fields can represent decisions in action space or physical space. The concept behind their use involves creating a virtual map of the physical environment such that physical features are represented by regions of attraction and repulsion in the virtual map. Different components of the environment or of the agent's action

space of the agent may be represented by components of the field that are superimposed to create a *working field*. The working field is an abstract and more suitable representation of the real world, allowing informed decisions to be made.

Potential fields have been used in a variety of robot applications such as robot motor schemas in navigation [Arkin, 1990], obstacle avoidance [Khatib, 1986; Spence and Hutchinson, 1995] and as action maps for soccer robot movements [Reikki *et al.*, 1998]. Navigation and obstacle avoidance environments are examples of physical space and imply that the agent will traverse the potential field to complete its goal. In action space domains such as soccer robot movements, the potential fields can be used to determine the next appropriate behaviour of the agent.

## 2.3 Constructional Elements for the Potential Fields Generated by MAPS

The three stages of the MAPS algorithm are used to determine the directives for each robot in the environment. The actions for the robots are determined as a function of the task. In the soccer robot example, this means either going for the ball or moving to a good location. The locations for the robots to carry out these actions are determined using potential fields. As a result, the action-location paradigm adopted is closely related to the plan or task of the system. The general algorithm can reflect the policies adopted for choosing agent actions and the potential fields generated from this perspective.

The soccer robot example builds potential fields (with the implicit team plan encoded) out of the elements described below. Each element is designed to provide low values at attractive regions, and high values at unattractive regions. The elements are combined on a grid array that represents the physical environment such that coordinates are easily extracted. Some elements cover the whole area, while others influence only part.

**Base Field**

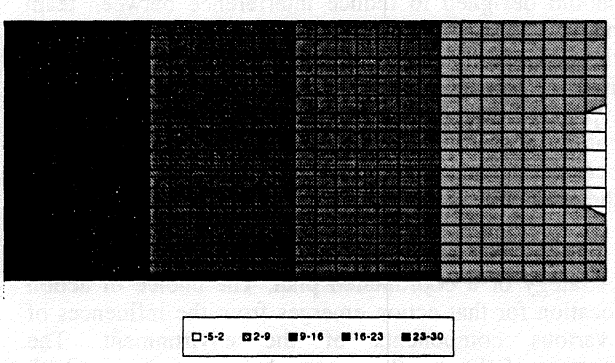


Figure 1. Base field representation

The base field mask is a representation of the physical environment and is biased towards the goal of the system. When looking for low valued coordinates, this has the effect of encouraging the agents to move towards areas of interest. Figure 1 shows the base field used for the RoboRoos system. It is a representation of the physical soccer field used in the small sized league. The opposition goal is at the base of the ramp and this representation encodes the desire for the agents to carry out their directives towards it.

**Object Regions**

This is a field mask that represents an object's presence in the working field. This mask is relatively small in area and is placed wherever the objects are on the field. The masks represent the objects' locations, and regions around them considered to be their influence zone. This can be used for robot locations or obstacles.

**Robot's Position**

Each robot can have an area of responsibility. Once again, this can be an effect of the goal of the system. It may be desired to have the robots maintain responsibility in specific areas. In the soccer robot system, this is used to keep players in their specific field positions (eg, left wing).

**Distance From Current Position**

This function is added to prevent the planner from selecting coordinates on the boundaries of the potential field. It creates a field-wide virtual 'dish' encouraging the selected coordinates to be close to the current position of the object in question. In other words, the potential fields generated can produce desirable coordinates in positions conceivably too far from the robot to be appropriate since the nature of the environment is intended to be highly dynamic, and these locations will be overridden in subsequent planner evaluations.

**High Value Continuation**

In some cases, the planner evaluates the field in a line-of-sight manner similar to the *clear path to object* function described below. This is carried out by adding the highest valued coordinate to all other coordinate values from the object to the boundary of the field which has the effect of building a 'shadow' of high values where it is undesirable for the agent to go.

**Clear Path To Object**

It is important for a robot navigating to positions in the environment to have a clear view of certain objects otherwise their location can't serve any purpose. This function is represented in the potential field by making all occluded coordinates as high values. Figure 2 demonstrates an example of this. A desired object is located in the centre of the field as shown by the black circle. An obstacle is shown on the right by the white cross. The image shows the area behind the obstacle's region occluding the object as a plateau of high values where these locations would obscure the object from the robot.

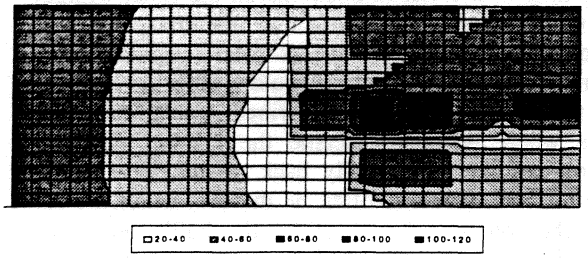


Figure 2. Clear Path To Object function representation

## 2.4 Real Time Implementation of MAPS in the Robotic Soccer Environment

The potential fields generated in the RoboRoos system are used to determine the actions and destinations of the robots. The more specific version of the general algorithm for robotic soccer is shown below.

```
Update robot and ball coordinates from vision system
Build a field to determine robot actions [PF]
for each robot
        if the robot is best to kick the ball
                Build a "kick to" working field [PF]
                Send "kick to" command to robot
                Send "kick to" coordinates to robot
        else
                if we are in control of the ball
                        Build an attacking "go to" field [PF]
                else
                        Build a defensive "go to" field [PF]
                Send "go to" command to robot
                Send "go to" coordinates to robot
```

Algorithm 1. The main loop of the MAPS module for the RoboRoos robot soccer system.

The algorithm determines which player is the kicker and gives them coordinates to kick the ball. The other players are given coordinates to move to. The kicking player keeps track of the ball and intelligently navigates to a position where it can kick the ball to the given coordinates.

### Construction of the Action Selection Working Field

The action selection field is based on proximity to the ball. The field consists of a single construction element: the *distance from current position* element. This field is established at the ball. The robot with the lowest value is selected as the kicker, and all other robots are given commands to navigate to.

### Construction of the Kicking Coordinates Working Field

To score more goals than the opposition, the active play should be encouraged towards their goal. A potential field is constructed to determine the best location for the kicker to kick the ball to. Below is the algorithm used.

```
Set basefield
for each opposition robot
        Add opposition robot's Object Regions
        Perform High Value Continuation
        for each grid coordinate on the field
        Add the Distance From Object (ball)
Select lowest valued coordinate on field
```

Algorithm 2. The algorithm determining the kicking coordinates

The potential field created from this algorithm represents bad locations for the ball (eg, near the opposition) as high values while the lower valued locations represent less dangerous areas. The base field is used as the foundation. Since it is impossible to kick the ball behind the robots on the field, the occluded coordinates are given higher values via the *high value continuation* element to prevent them from being selected.

The potential field at this point is still strongly influenced by the base field's bias towards the opposition goal. This has the effect of making the goal and areas close to it appear as good locations, which is not always true since it could place the ball too far from any home team robots. A compensation function balances this by adding the *distance from current position* element to each location in the potential field. From the resulting potential field, the lowest valued coordinate is selected and used as the location to kick the ball to. An example is shown in Figure 3 with the opposition players in the configuration shown in the diagram on the right.

In Figure 3, the opposition goal is on the right of the field. Darker areas represent high values as can be seen from the overlapping region between two of the players

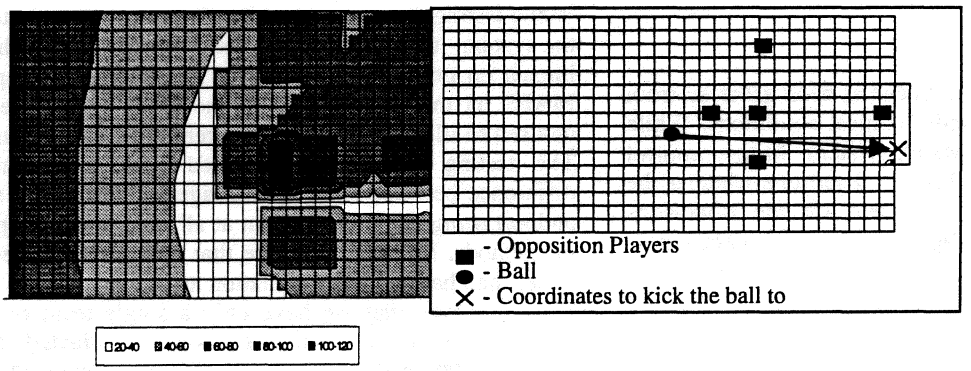


Figure 3. Example of the kicking potential field

creating the largest peak in the figure. The white area shows the best area to kick the ball. From this potential field, the lowest value coordinate is at the bottom of the opposition goal. This is not obvious from the diagram since it is only slightly lower than other coordinate values in this region.

**Construction of the Attack/Defend Potential Field**

When attacking, it is important to position the robots in good locations for them to take a shot at goal or to receive the ball from a pass. The algorithm for determining the attack coordinates is shown below.

```
Set basefield
for each robot
        Add Object Region for that robot
Add Robot's Position mask
Apply Clear Path To Object (ball)
Add the Distance From Current Position of the robot
Select lowest valued coordinate on field
```

Algorithm 3. The algorithm for determining each robot's destination coordinates during attack

Each robot's presence is added to the base field to discourage locations close to other robots. To prevent clustering of the home team robots, each robot's position is added to encourage them to remain in their own area of responsibility. Since it is beneficial to be able to see the ball to receive it, any locations not in sight of the ball are strongly discouraged. Finally, to prevent robots trying to go to extreme locations due to the base field's influence, the distance from the robot to each grid position is added. From the resulting potential field, the lowest valued coordinate chosen as the destination for the focal robot.

The algorithm to produce the potential field for extracting defend coordinates is similar to Algorithm 3 except that it searches for high values instead of low since these will represent locations of the opposition biased toward the home goal. In effect, the robots will be given coordinates to defend that are between opposition robots and the home goal.

**Logistics**

Since the operating environment for MAPS is to be highly dynamic, the directives generated are constantly changing to account for the newest state of the environment to allow the robots to react quickly. In the RoboRoos system, decisions are made every frame processed by the vision system (25 fps) and uses less than 1% of the CPU bandwidth.

## 3 Results

Case study evidence of performance has been gathered from observations of game play under contest conditions. While this evidence supports the MAPS system, further control studies have also been carried out to determine whether the implementation of the MAPS system for the RoboRoos soccer team has any significant affect on the team's goal scoring performance.

### 3.1 RoboCup '98

The most obvious impact of the MAPS system in RoboCup was the field positions maintained by the robots. The field position component of the working field kept the robots operating in their prescribed positions on the field, without limiting the opportunity for a robot to take advantage of a good opportunity that was outside its usual designated area. For example, the defender robot would usually hover back towards the goalkeeper in its prescribed activity area. On occasions where defender became the kicking robot, it would progress the ball rapidly back up the field. As it followed the ball, it would often be the best candidate for taking a shot on goal by virtue of the ball's region of influence. However, if the shot was unsuccessful and play continued with the defender now out of the influence region of the ball, the defender would retire back into its defensive role.

In general play, the field positions kept the robots from interfering with one another, which is the first requirement of the system. There was also clear evidence of a higher level of cooperation with robots passing the ball to one another. While the MAPS implementation does not explicitly contain a passing strategy, passing emerges from the intereaction between the robot using the "kick to" working field, and another robot using the "go to" working field. In both fields, there is an attraction towards free space near the goal mouth. If the kicking robot found the shot on goal to be blocked (by the influence of the high value continuation element), the working field would often contain a region of attraction in open space to either side of the goal mouth. Similarly, the robot using the "go to" field would also be attracted to this space. The effect would be a pass to the free space that a robot was moving towards.

The defensive strategy was also highly effective, and often prevented the opposition from making effective attacks. The robots were able to quickly assume a position between the attackers and the home goal preventing progression of the ball. The play was similar to a traditional human soccer "marking up" procedure.

The pitfalls of the system were mostly due to unexpected failures of other parts of the system. For instance, if a robot crashed, or was unable to move, the system does not reallocate its task to the remaining players. This was particularly noticeable with the kicker selection field, which would always choose the closest robot. To circumvent this type of problem requires identification of lack of performance of individual robots before re-allocation can be performed.

### 3.2 System Evaluation With and Without MAPS

While testing during competition illustrates some features of the system, it does not clearly show whether MAPS has any impact on performance. In order to quantify the impact of MAPS on the performance of the robot team, a test was devised to illustrate the effect of removing all coordination. The testing procedures presented here show only the difference between *using* a multiagent coordination system and *not using* any coordination at all. It is not a comparison of methods, but only evidence that the system presented is better than having no coordination at all.

Given that we have only a single team of robots with which to test, all tests were conducted against stationary opposition in the configuration shown in Figure 4. The opposition robots are depicted as grey boxes.

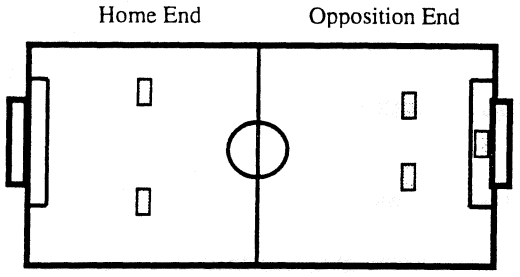


Figure 4. Opposition setup for testing.

The first tests that were conducted used the multi-agent coordination method described in the preceding section. The second (control) tests were performed by simply telling each robot to kick the ball to the opposition goal. In this comparison, the first system consists of using team and agent plans that conform more towards the plan-by-communication method while the second system is purely a plan-by-command method. Tables 1 and 2 summarise the results of five sessions of five minutes play under RoboCup rules.

| Time period | Goals Scored | Goals Against | Locked Robots | Free Balls |
|---|---|---|---|---|
| 1 | 8 | 0 | 3 | 1 |
| 2 | 7 | 0 | 3 | 1 |
| 3 | 6 | 0 | 4 | 0 |
| 4 | 6 | 0 | 2 | 2 |
| 5 | 6 | 0 | 4 | 1 |
| **Average** | **6.6** | **0** | **3.2** | **1** |

Table 1. Results of system playing with coordination switched on.

| Time period | Goals Scored | Goals Against | Locked Robots | Free Balls |
|---|---|---|---|---|
| 1 | 3 | 0 | 3 | 2 |
| 2 | 5 | 0 | 4 | 5 |
| 3 | 4 | 0 | 4 | 3 |
| 4 | 6 | 0 | 9 | 0 |
| 5 | 5 | 1 | 5 | 2 |
| **Average** | **4.6** | **0.2** | **5** | **2.4** |

Table 2. Results with coordination off.

The results show an improvement using the multi-robot coordination system. In the games without coordination, there were many problems with robot-robot collisions, often resulting in the robots becoming temporarily deadlocked. The robots tended to cluster near the ball, so that if the ball rebounded to the other end of the field, it took a long time to recover, particularly with the robots impeding one another. Another side-effect of clustering was the number of free balls that resulted from the robots being too close to the ball and being unable to move closer to kick it. The coordinated team however, maintained a more even presence across the field ensuring that there was always one player to attend to the ball. Robot-robot collisions happened less in the coordinated team and the game play was more organised.

## 4 Summary

If a group of agents share a common representation of the world, and a common way of choosing actions, it is possible to produce emergent cooperative strategies without explicit communication. MAPS implements this idea by modelling the world in a coarse grid structure and producing a working field that illustrates the influence of the components of the environment. MAPS forms cooperative strategies by observing team agents at the current point in time and choosing appropriate actions to increase the likelihood of cooperation in the near future. The strategies produced by MAPS are "plans-as-communication" where the execution module uses the plan as a resource for intelligent navigation, rather than as an explicit command.

Use of this type of multi-robot control has proven effective in the robotic soccer environment with the RoboRoos achieving second place in RoboCup '98 in Paris in 1998. The emergent properties and robustness of MAPS overcame many of the problems that exist in the vision and navigation systems of the RoboRoos to produce a cooperative team of robots that were capable in both attack and defence.

## References

[Agre and Chapman, 1990] Agre, P., Chapman, D.: What are Plans For? Robotics and Autonomous Systems, 6, (1990) 17-34.

[Arkin, 1990] Arkin, R.: Integrating Behavioral, Perceptual, and World Knowledge in Reactive Navigation. Robotics and Autonomous Systems, 6. (1990) 105-122.

[Claus and Boutilier, 1997] Claus, C., Boutilier, C.: The Dynamics of Reinforcement Learning in Cooperative Multi-agent Systems. AAAI-97 Workshop on Multi-agent Learning, (1997).

[Khatib, 1986] Khatib, O.: Real-Time Obstacle Avoidance for Manipulators and Mobile Robots. The International Journal of Robotics Research, Vol. 5, No 1, (1986) 90-98.

[Mataric, 1997] Mataric, M.: Reinforcement Learning in the Multi-robot Domain, Autonomous Robots, 4(1), (1997) 73-83.

[Mataric, 1996] Mataric, M.: Using Communication to Reduce Locality in Distributed Multi-agent Learning. *Brandeis University Computer Science Technical Report CS-96-190*, Nov 1996.

[Murciano and Millan, 1997] Murciano, A., Millan, J.: Learning Signalling Behaviors and Specialization in Cooperative Agents. Adaptive Behavior, Vol 5, No 1, (1997) 5-28.

[Reikki *et al.*, 1998] Reikki, J., Pajala, J., Tikanmäki, A., Röning, J.: Executing Primitive Tasks in Parallel. Proc. of the Second RoboCup Workshop, Paris 1998, 339-345.

[Sen *et al.*, 1994] Sen, S., Mahendra, S., Hale, J.: Learning to Coordinate Without Sharing Information. Proceedings of the Twelfth National Conference on Artificial Intelligence, (1994) 426-431.

[Spence and Hutchinson, 1995] Spence, R., Hutchinson, S.: An Integrated Architecture for Robot Motion Planning and Control in the Presence of Obstacles With Unknown Trajectories. IEEE Transactions on Systems, Man, and Cybernetics. Vol. 25, No. 1, (1995) 100-110.