

# **MAPS: A System for Multi-Agent Coordination**

Ashley Tews & Gordon Wyeth

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

Revised Paper Submission for Advanced Robotics Journal Special Issue on RoboCup

**Abstract.** MAPS (Multi-Agent Planning System) is a system for multi-agent coordination used 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 coordinates with the others by choosing a goal action that most benefits 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, the influence of other agents, and the influence of the components of the environment. In the robot soccer domain, the robots coordinate with their own team members against the agents on the other team, while negotiating the environment that contains the ball, the field and the goals. 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 purpose 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, and ensuring others', functional integrity. As such, the team plan should exist at a level where it provides strategies for each robot to contribute to the team's 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 of multi-robot control. MAPS performs high level multi-agent 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 of important objects and the position and behaviour of other robots in the environment. This abstraction is carried out using potential fields. By modelling features of the environment from each robot's perspective, strategic commands and coordinates are extracted and proposed to the robots. The robots consider MAPS requests and attempt 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 [ROBOCUP REFERENCE]. The event consists of several categories with different requirements in terms of embodiment, size and sensing ability - this paper deals with robots developed for the small size league 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. Off the field, the teams generally consist of an overhead camera for visual feedback, a computer for processing the visual information and determining game play, 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, namely:

1. the navigation system for the robots, and
2. the design for multi-robot cooperation.

The navigation system consists of components on-board and off-board the robot to provide an information loop to monitor and control the robots. Designing an efficient navigation system is difficult since it is based in part on noisy and infrequent, but highly useful, data from the external camera, and in part on the readily accessible but less useful data available from local sensors. This problem is exacerbated by the highly dynamic nature of robot soccer. The RoboRoos approach to navigation is described in [Browning, Wyeth and Tews, 1999].

This paper concentrates on the second problem, multi-robot cooperation. The paper 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. Section 2 describes the architecture of the RoboRoos system to better understand the design issues in a robotic soccer environment. The multi-robot coordination strategy is analysed in Section 3 and shows examples of system performance as well as the algorithms used. Section 4 describes case study data from system performance under contest conditions, as well as quantitative data describing system performance with and without coordination. A comparison to other researchers'

methods of coordination in these environments is discussed in Section 5. Section 6 presents a summary of the concepts discussed in this paper.

## 2 RoboRoos Architecture

The architecture used in the RoboRoos system is shown in Figure 1. The main hardware of the system consists of a camera, computer, transmitter and the robots. Software controlling the robots and game strategy exists on the robots and computer. The software on the robots controls their movements and interprets commands sent to them. The software on the computer monitors the robots and determines game strategies. The sections below describe each component of the system in some detail.

### 2.1 System Hardware

The overhead camera is a colour CCD camera with variable zoom lens. The computer is a 233 MHz Pentium II based PC running Windows 95. It contains a PCI colour frame grabber that grabs frames directly into system memory. The PC can transmit information via 418 MHz or 433 MHz radio signals at 9600 bps to the robots. Each robot receives the broadcast signal and interprets the information and commands from the PC. The robots have custom designed, 68332 based processor boards which contain all of the required computational equipment and memory, sensor interfaces and power electronics. The radio receivers are mounted on a separate shielded board. The drive system consists of a pair of DC motors and optical encoders in a servo loop arrangement.

### 2.2 Colour Segmentation

The RoboCup rules define the colours for the playing field and equipment with some attempt at precision. Segmentation of the image is meant to be relatively straight forward using colour-based

techniques. Often, the field has non-uniform lighting making the colour segmentation somewhat more challenging than might be expected. This segmentation process is by far the most computationally expensive routine for the RoboRoos system. During RoboCup '98 the system processed between 4-8 frames per second (fps). Subsequent refinements have improved the speed to 25 fps. This area is seen as a key area of investigation for improving the response of the robots to rapidly changing conditions.

### **2.3 Identification and Tracking**

From the segmented image, the robots and ball are found by template matching and their locations mapped on to a pixel based grid representation of the field. Classifying the robots as “us” and “them” is relatively straightforward, as each robot is required by RoboCup rules to have two distinctly coloured pingpong balls on top. Identifying and tracking the individual robots in our team is a more difficult problem for the vision software, as they all appear identical. Correct identification is important so that commands are sent to the correct robot. The individual robots are tracked by matching the identified robots to the closest last known location of each robot.

### **2.4 MAPS – Multi-Agent Planning System**

Once the robots have been located and identified in the image, the information is passed to MAPS, which controls the game and the robot cooperation. MAPS determines the best locations and actions for the robots. When the planning is complete, the ball location, all robot locations and planner actions are broadcast as a packet with the corresponding robot identification numbers from the transmitter. The algorithms and techniques used for planning form the bulk of this paper.

## 2.5 Robot Software

The software on the robots consists of algorithms for navigation and interpreting MAPS actions and motor control for driving and steering the robots. The navigation software determines the path for the robot to reach its MAPS-determined destination without colliding with other robots, the walls, or restricted areas of the field such as the home goal square. There are three commands from MAPS that need to be interpreted: go to a destination, kick or halt. Each of these invokes different robot behaviours. Going to a destination invokes the navigation path planning procedures while kicking requires a fast drive towards the ball. Halt causes the robot to come to rest smoothly.

When a robot is required to move to a location, its on-board software must determine how fast to move and the paths to be followed by the two wheels. The servo loops formed by the motor/encoder pairs are controlled to ensure that the required path is followed while wheel slip is kept to a minimum, giving greater performance. By integrating the commands sent to the servo loop, the robot can maintain an approximate position in between the information received from the overhead camera.

As can be seen from the above sections, the problems associated with designing a complete robot soccer system are many and varied. The following section addresses the focus of this paper: multi-robot coordination.

## 3 Multi-robot Coordination

Multi-robot cooperation requires *coordination* strategies to gain the full benefit of applying more than one robot to a problem. It is certainly possible to have many robots cooperate on a task in an uncoordinated fashion. As an example, consider a robot soccer team that has every robot chasing the ball at the same time. The robots are likely to impede each other rather than help each other, and

will not be able to cover other areas of the field should the ball be knocked away from the pursuers. A coordinated approach would have, for example, a single robot in pursuit of the ball, with the others offering support by being in position to accept a pass, to catch rebounds or to act defensively should the ball be lost to the opposition.

A typical multi-agent approach would achieve such coordination of play by communication between agents with negotiations over who should do what and where. The system presented in this paper takes a different approach inspired by the problem domain – soccer. Consider a team of human soccer players. In general play, each individual knows what action to take to best benefit the team without any communication with other members of the team. Individuals derive this action from their perceived model of the playing field, including the current positions of their team members, the opposition and the ball. Applying this approach to the multi-agent domain suggests that a team of agents that share a similar perceived model of the world can derive their individual coordinated actions without extensive negotiations over actions types.

An approach based on a shared world model presents other problems in mobile robotic environments. In the robot soccer environment, it includes communication delays, vision system integrity, inconsistent field conditions, and the unpredictable nature of 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 that cope with an incomplete or out-of-date model.

### **3.1 Defining the Problem**

Coordinating robots in dynamic environments requires careful design consideration at both the individual robot and multi-robot level. At the individual level, each robot should be able to operate in the absence of global information relying on information from local sensors. However, to achieve coordination at the multiagent level, the robots must use global information to cooperate with other team members. As a result of this combination, problems with single agent design are compounded

at the multiagent level. If the system doesn't perform well, it can be difficult to determine where the problem may lie ([1], [2]).

### **Environment Representation**

An important element of any mobile robot design is how the environment is represented to them. Individual robots can typically sense features of the environment such as walls, objects, and other robots. However, coordination requires a shared representation of the environment either from a global sensor or pooled information gathered by other robots with local sensors. The degree of coordination that is limited by the extent to which the robots can share their representation of the world. This has trade-off characteristics that are highlighted in subsequent sections.

### **Planning and Reactive Systems**

Once an agent has information about its environment, it needs to be able to carry out actions to achieve the system goal. The actions and the method for deciding the appropriate actions are part of the robot's architecture. In purely reactive systems, the chosen robot actions will correspond closely to features in the environment. While this is necessary to handle situations that require immediate attention (such as a collision), and can be adequate as the sole action selection mechanism in some environments [REFERENCE FOR HERBERT], it can lack a forward approach to set up the environment more favourably. Some environments require the robots to be able to carry out their actions according to a plan that provides more efficient solutions to problems than a reactive approach can solve. Team sports are good examples of this. To provide better chances of scoring a goal, the team may have formation plays or have players move to unusual positions in order to benefit the team at a future stage. This requires the team to know about the plan and attempt to position themselves accordingly in space and time.

There are also disadvantages in planning systems, especially in dynamic environments. Plans may fail if unforeseen contingencies arise. It is also difficult to predict the future in dynamic

systems so the further ahead the planning, the less likely it will be successful. Agre and Chapman [6] state that there are two types of planning: hard-wired planning and abstract planning. The hard-wired planning consists of the agents' actions being predefined with little contingency inclusion. The planner gives the agents directions 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.

### **Level of Coordination**

For the reasons outlined above, and from the experience of other robot soccer teams [LOTS OF ROBOCUP REFS], coordination requires the relevant agents to have common information to base their decisions on, such as a similar model of the environment. The integrity of the information common between the agents is an important factor to the effectiveness of the coordinated activity. As the amount of information shared between agents increases, higher levels of precision in coordination may be achieved.

On the other hand, communication in robotic systems should be kept to a minimum to improve robustness [CONF RPT REFS Garland and Alterman, 96; Sen, 94] and efficiency [5]. If each agent communicates with each other agent, the communication paths can increase exponentially [CONF RPT REFS Arkin, 92; Kube and Zhang, 94] with the number of agents. The bandwidth of communication not only impacts the amount of information that may be shared, but also the timeliness of that information.

It becomes important then to trade off the need for a high degree of precision in the world model to achieve high levels of coordination against the need for robust, high throughput communications to achieve the high degree of precision in the world model in the first place. In seeking to balance this trade off, the system designer must address the level of coordination that is

required for the application, and hence the degree of precision required in the world model and the corresponding bandwidth needs.

### **Requirements for Good Multiagent Control in Dynamic Environments**

In dynamic environments, methods must be chosen to give the agents as complete and timely information about the environment as possible to enable them to be able to accurately plan and coordinate their actions. This is even more important in constantly changing environments where the goals of the system can depend heavily on the actions of other moving objects. Agents that have local sensing in these types of environments can suffer from impoverished information when making their action decisions. To improve their world model, they can either send what they know to each other frequently [3,1,4], or to an external data fusion system that reports back more complete information. Another method is to have global sensors, which provide information to all agents, or to the agent decision-making mechanism. The important point is to reduce the amount of communications, and therefore latency in information, for the decision-making mechanism.

In multi-robot environments, the adequateness of the world model affects the performance of the system. Planning is carried out on the information known about the environment and since it involves determining agent states in the future, it heavily depends on the accuracy of the information. Coordination also depends on the world model representation. While robots can operate in some environments without a shared, common or complete world model, and without coordination, it is beneficial in robot soccer. This is evident from the above discussion and from the team descriptions provided in [REF BOTH PARIS AND PRICAI]. Most of the teams had a method of modelling the world either from the global vision system or by combining the local data sent from robots with on-board vision. From the world model, strategies, coordination techniques, and robot movements are determined.

This paper presents a system that provides robot coordination and planning in a dynamic environment. The concepts discussed are intended as a solution in constrained spatial environments where the robots must coordinate their activities to achieve their tasks.

### 3.2 MAPS Overview

MAPS consists of a high-level planning algorithm which analyses the environment and sends commands and coordinates to the robots. It is the robots' responsibility to carry out this plan as best as they possibly can. This method of planning can be compared to the plan-executor method described by Agre and Chapman [6]. The planner contains the strategy for the team while the executor exists as the command interpretation mechanism on-board the robots. By their classification, MAPS provides a "plan-as-communication" to the robot navigation system, by providing each robot with a command and a set of coordinates. It does not rely upon the robots to complete the commands, and continues to observe them after the command has been issued. As the robot carries out the plan and new information about the environment is given to MAPS, a new plan may emerge from which MAPS will take advantage. If the robots cannot carry out their part of the plan, perhaps due to some dynamic obstacle or the current state of motion of the robot, it will carry out the plan as best as possible.

The physical environment is represented in MAPS by a coarse grid, which acts as a single operating surface for all operations MAPS carries out to determine directives for the robots, simplifying computations and removing the need for cross calibration between system components. The system operates rapidly, working with incomplete data when necessary since delays in planning lead to outdated plans. The three stages of the MAPS algorithm can be summarised as:

1. **Get new information:** Update the grid-based world model from new sensor data or prediction of robot behaviour.

2. **Choose an action:** Based on the influence of the components of the environment (typically robots and obstacles) choose an action type for each agent. The action type should be designed to reduce interference between the agents. 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 each robot should perform its selected action. The choice of location is built from the perspective of the individual robot and its chosen action.

In these steps, there is no mention of an explicit team strategy or a coordinated plan. Coordination emerges from the decisions made by MAPS as the same world model is used for determining each robot's directives. 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 the successful coordination. MAPS uses potential fields for this mechanism.

Potential fields can represent decisions in action space or physical space. The concept behind their use revolves around creating a virtual map of the physical environment such that features of the physical map are represented by regions of attraction and repulsion in the virtual map. Different components of the environment or of the action space of the agent may be represented by components of the field that are then 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 [7], obstacle avoidance [8], [9] and as action maps for soccer robot movements [10]. Navigation and obstacle avoidance environments are examples of physical space and imply that the agent will traverse the potential field to obtain 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. In the approach discussed in this paper, the potential field is used to provide MAPS with both a choice of action and a set of coordinates for that action.

### 3.3 MAPS applied to Robot Soccer

MAPS embodies a predetermined game strategy through a variety of potential field evaluations. Based on incoming data from the vision system, MAPS builds a field that determines the actions of the players. The field is used to decide which player has to control the ball. This player is issued a “kick to” command that means it has to try to kick the ball to a set of MAPS coordinates. The coordinates to “kick to” are determined by evaluation of the “kick to” field. All other players are given “go to” coordinates to navigate to. The goalie is excluded from this process since its job is specialised. The algorithm that runs the three step process of MAPS is outlined 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 coordinates to “kick to”
```

```
    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 coordinates to "go to"  
end
```

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

The players use the information from MAPS for their on-board navigation to moves them as close as possible to their destinations if they can't reach it. The kicking player's role is different. It must continually assess the position of the ball with respect to the coordinates sent from the planner, to determine where to move to, in order to kick the ball. This aspect of its navigation requires some intelligence so it can recognise the need to move faster than the ball to set itself up for the kick in much the same way that human players run slightly past the ball before bringing it under control for a more accurate kick. Once in position, it moves towards the ball displacing it towards the goal coordinates.

MAPS analyses the game every frame processed by the vision system. It uses less than 1% of the CPU bandwidth, with most of the time devoted to the expensive segmentation techniques in the vision system. Since the soccer environment is highly dynamic, the commands from the planner can be constantly changing to account for the newest state of the environment that allows the robots to quickly react. This is analogous to how humans play soccer: constantly updating and moving to increase the team's chances of scoring a goal, or defending successfully.

### **3.4 Constructional Elements for Potential Fields in Robot Soccer**

Decisions about the type of action and the location for the action are determined by analysis of the various working fields. The working fields are constructed from elements that represent actions in the game, and components of the environment. Each element is designed to provide low values at attractive locations, and high values at regions that are not attractive. The elements are combined by addition and are located in a 30 x 16 grid array that represents the field. Some elements cover the entire field, while others influence only a small portion.

The values for each of the elements is established by modelling the effect that element should have on the overall decision making process. For example, the positions of the opposition players are very important so they are modelled as high peaks to discourage any play near them. The combination of elements is decided by determining which elements are actually present in the environment, and their importance. Both the values for the elements and the combination are chosen empirically. The fields that are presented in this paper have been derived by trial, error and intuition and are the actual fields used in competition and the quantitative experiments presented later in this paper.

### **Base Field**

The base field mask is a representation of the physical field and is biased towards the opponent's goal. When looking for low valued coordinates, this has the effect of encouraging the game-play towards the opposition's goal. Figure 2 shows the opposition goal at the base of the ramp. Note that the goal line corners are also elevated to encourage play towards the goal mouth.

### **Robot's Personal Regions**

This is a field mask that represents a robot's presence in the potential field. This mask is relatively small in area and is placed wherever the robots are on the field. It represents the robot's location, and a region around them considered as their influence zone. These masks may be either attractive or repulsive depending on the action type.

### **Robot's Position**

Each robot on the team plays in a specific field position. The field mask used for this position encourages the robot to remain in that position to provide better robot dispersion around the field. Figure 3 shows a representation of the right winger's region of influence.

### **Clear Path To The Ball**

To enable a home side robot to move to a strategically good position to receive the ball, its view must not be occluded by another field object otherwise their location can't serve any purpose. This function is represented in the potential field by making all occluded coordinates as peaks when MAPS is searching for low values. Figure 4 demonstrates an example of this. The ball is located in the centre of the field as shown by the black circle. The opposition robot is shown on the right by the white cross. The image shows the area behind the robot's personal region occluding the ball as a plateau of high values.

### **Distance From Current Position**

This function is added to prevent MAPS from selecting coordinates on extreme boundaries of similarly valued regions in the potential field. It creates a field-wide virtual 'dish' that encourages the selected coordinates to be close to the focal location as shown in figure 5. In this figure, the current position of the object is in the centre of the lowest valued region.

### **High Value Continuation**

In some cases, MAPS needs to evaluate the field in a line-of-sight manner similar to the *clear path to the ball* function. It extends the clear path to ball function to the focal object. This is carried out by adding the highest valued coordinate to all other coordinate values from the object to the boundary of the field. It has the effect of discouraging locations *towards* and *beyond* an object whereas clear path to ball discourages locations *at* and *beyond* the impinging object.

## **3.5 Potential Field Construction in the RoboRoos System**

As illustrated in Algorithm 1, there are four potential fields generated in the RoboRoos system. These are used to determine the actions of the players, the coordinates to kick the ball to for the kicking player, and the defend or attack coordinates for the others.

### 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, all other robots are given “go to” commands.

### Construction of the Kicking Coordinates Working Field

To win the game, the home team must score more goals than their opponents. To achieve this, the active play should be encouraged towards the opposition goal. A potential field is constructed to determine the best location for the kicker to kick the ball to. The algorithm for constructing this potential field is shown below.

```

Set basefield
for each opposition robot
    Add opposition Robot Personal Region
Perform High Value Continuation
for each grid coordinate on the field
    Add the Distance From Current Position of the ball
Select lowest valued coordinate on field

```

### Algorithm 2. The algorithm for determining the kicking coordinates

The algorithm creates a potential field by representing 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. Opposition players are represented in this potential field as high valued areas. Since it is currently 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 as good locations, which is not always true since it could place the ball too far away 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 (working field), the lowest valued coordinate is selected and used as the location to kick the ball to. An example of the resultant field is shown in Figure 6 with the opposition players in the configuration shown in the diagram on the right. The opposition goal is on the right hand side. Darker areas represent high values as can be seen from the overlapping region between two of the players creating the largest peak in the figure. The white area shows the most favourable 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 falls in the region 20-40 and is only slightly lower than many other coordinate values in this region.

### **Construction of the Attack Coordinates Potential Field**

Perhaps the most widely used potential field is the one determining where to send robots that are not going for the ball during attacking play. It is hoped that the home team will be attacking more frequently than defending and 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 these coordinates is shown below.

```
Set basefield
for each robot
    Add Robot Personal Region
Add Robot's Position mask
Apply Clear Path To The 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**

The algorithm creates a potential field of high points being unattractive locations and lows being attractive. 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.

Figure 7 shows the resulting potential field for finding the coordinates for the left winger. The field configuration is shown on the diagram on the right of Figure 7. The left winger's location is in the upper centre. Dark areas in the left diagram represent unattractive locations and the white area indicates the lowest valued region. The cross in the diagram on the right shows the destination chosen for the left winger.

**Construction of the Defend Coordinates Potential Field**

When required, defending the home goal is more important than moving to good attack locations. The team must be able to find positions that will minimise the chances of the opposition scoring which means they must position themselves between the opposition players and the home goal. The coordinates for each robot to move to are found using the algorithm shown below.

```
Set Base Field
for each opposition robot
    Add Robot Personal Region
```

```
for each home robot
    Add large negative Robot Personal Region
for each grid coordinate on the field
    Add the negative Distance From Current Position
Select highest valued coordinate on field
Find the intercept coordinate with the home goal
```

**Algorithm 4. The algorithm for determining each robot's destination coordinates during defence**

Since the aim is to protect the home goal, construction of the potential field revolves around high values rather than low as in the previous algorithms. Locations are encouraged between the opposition robots and the home goal by making the home team's presence as negative regions whilst the opposition's presence are positive. To prevent the robot from trying to move too far, the negative distance from it to all other locations is added. From the constructed potential field, the highest valued coordinate is found. This usually (but not necessarily) represents the location of an opposition robot so an intercept location is found between it and the home goal which is the destination defence coordinates for the robot.

## 4 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 effect on the team's goal scoring performance.

## 4.1 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 the ball came through to the defender so that the defender had time to get to and control the ball, the robot would progress the ball rapidly back up the field. As the defender 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 interaction 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 emergent defensive strategy was 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 goal mouth 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 catastrophic failures of other parts of the system. MAPS was generally tolerant of the erroneous information, noise and delays introduced by the vision system, and the frequent failure of the robots to carry out their assignments due to navigation errors or communications breakdown. However, the system could not recover if a robot fatally crashed, or was unable to move. MAPS is not able to reallocate its task to the remaining players. This was particularly noticeable with the kicker selection field, which would generally choose the closest robot. If that robot was not able to approach the ball (due to being wedged against another robot or hardware failure) the system would not reallocate the task. This problem occurred in a slightly different form in the final match against CMUnited in RoboCup '98 in Paris. In this case, a robot failed during defence, leaving one CMUnited player unmarked from midfield. The unmarked player was then able to score a goal that put CMUnited into the lead. To circumvent this type of problem requires some form of identification of lack of performance of individual robots before re-allocation can be performed.

## 4.2 Comparison of Results 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 coordination system and *not using* any coordination at all. It is not a comparison of methods, but only evidence that some coordination is better than having no coordination at all.

Given that there is only a single team of robots with which to test, all tests were conducted against stationary opposition (blocks of wood that were about robot sized) in the configuration shown in Figure 8. The opposition robots are depicted as grey boxes. Testing against stationary opponents provides a stable reference to evaluate system performance.

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 goal mouth. This second test has no explicit coordination, with the only coordination emerging from the robots' on-board navigation software preventing collisions with their team members as they encroached upon each other. Both tests used the same robots with identical on-board navigation software, and the same "plan-as-communication" type of commands from the planner. Tables 1 and 2 summarise the results of five sessions of five minutes play under RoboCup rules (the clock was stopped for restarts and kick-offs).

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

<b>Time period</b>	<b>Goals Scored</b>	<b>Goals Against</b>	<b>Locked Robots</b>	<b>Free Balls</b>
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
<b>Average</b>	<b>6.6</b>	<b>0</b>	<b>3.2</b>	<b>1</b>

**Table 2. Results with coordination off.**

<b>Time period</b>	<b>Goals Scored</b>	<b>Goals Against</b>	<b>Locked Robots</b>	<b>Free Balls</b>
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
<b>Average</b>	<b>4.6</b>	<b>0.2</b>	<b>5</b>	<b>2.4</b>

The results show a significant improvement with multi-robot coordination switched on. Robots in the games without coordination had many problems with robot-robot collisions, often resulting in the robots becoming temporarily deadlocked. The robots tended to cluster near the ball, so that any rebounds that moved the ball to the other end of the field took a long time to fetch, particularly with the robots frustrating one another. The other side-effect of the robots clustering was the number of free balls that resulted from them all being too close to the ball and not being able to navigate closer to kick it. A free ball results from the ball being stationary for around ten seconds. The coordinated team on the other hand 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.

## **5 Discussion**

This paper argues that effective multi-robot cooperation requires coordination, and that effective coordination requires agreement between robots about the world model. A system that provides coordination based on a shared world model, MAPS, has been presented, and shown to be more capable of scoring goals in a robot soccer domain than a system with no coordination.

The tests run in this paper had MAPS executing on a central server that distributed global information from the overhead vision sensor. Could MAPS run in a system with no central server, and no global sensor? Consider such a team of robots, fully distributed with no central resources. One approach to their coordination may be to attempt to coordinate plans based on each robot's individual view of the world. The arguments in this paper suggest that perhaps the communication bandwidth between robots is better spent coalescing the relevant aspects of the world model, rather than trying to negotiate a coordinated plan. If each robot has the same coalesced global view of the world, and executes the same MAPS algorithm, the resulting "plan-as-communication" will be same on each robot – as if the MAPS algorithm had been run on the global view in the centralised server. Clearly such a coalesced view of the world is likely to be noisy and lagging the system. However the results presented in this paper are based on data with just those characteristics, albeit data in a central repository.

As examples of systems that have individual robots with no global view, it is interesting to look at the medium size robots in RoboCup. From a review of the team descriptions presented, it becomes apparent that global world modelling is considered important for coordination. Many of the teams that competed in the 1998 RoboCup medium sized league allowed the robots to maintain a local model of the world and transmit this information to an external arbiter that generated a global world model. Once generated, environmental information is sent back to the robots to enable them to more accurately model their environment. This two layer method allowed the robots to generate plans locally and still receive global commands as to appropriate strategies. The common factor in these teams was the generation and maintenance of the world model which assisted coordination between the robots. [This section needs refs].

In the small sized league, the need for an arbiter is removed since the global sensor provides a global world model to the robots. Coordination is achieved through either a planning system external to the robots or onboard planning based on this information. In general, both the global and

local sensor approach relies on generating a global world model to improve coordination between the robots.

## **6 Conclusions**

If a group of agents share a common representation of the world, and a common way of choosing action, 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.

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. The concepts embodied in MAPS should be considered in other applications requiring timely control of multiple robots in an ever changing environment. Future work includes its use in the RoboCup soccer simulation league and a predator/prey environment.

## References

1. Mataric, M.: Reinforcement Learning in the Multi-robot Domain, *Autonomous Robots*, 4(1), (1997) 73-83.
2. Murciano, A., Millan, J.: Learning Signaling Behaviors and Specialization in Cooperative Agents. *Adaptive Behavior*, Vol 5, No 1, (1997) 5-28.
3. Mataric, M.: Using Communication to Reduce Locality in Distributed Multi-agent Learning. *Brandeis University Computer Science Technical Report CS-96-190*, Nov 1996.
4. Claus, C., Boutilier, C.: The Dynamics of Reinforcement Learning in Cooperative Multi-agent Systems. *AAAI-97 Workshop on Multi-agent Learning*, (1997).
5. Sen, S., Mahendra, S., Hale, J.: Learning to Coordinate Without Sharing Information. *Proceedings of the Twelfth National Conference on Artificial Intelligence*, (1994) 426-431.
6. Agre, P., Chapman, D.: What are Plans For? *Robotics and Autonomous Systems*, 6, (1990) 17-34.
7. Arkin, R.: Integrating Behavioral, Perceptual, and World Knowledge in Reactive Navigation. *Robotics and Autonomous Systems*, 6, (1990) 105-122.
8. Khatib, O.: Real-Time Obstacle Avoidance for Manipulators and Mobile Robots. *The International Journal of Robotics Research*, Vol. 5, No 1, (1986) 90-98.
9. 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.
10. Riekkki, J., Pajala, J., Tikanmäki, A., Röning, J.: Executing Primitive Tasks in Parallel. *Proceedings of the Second RoboCup Workshop, Paris 1998*, 339-345.

Figures

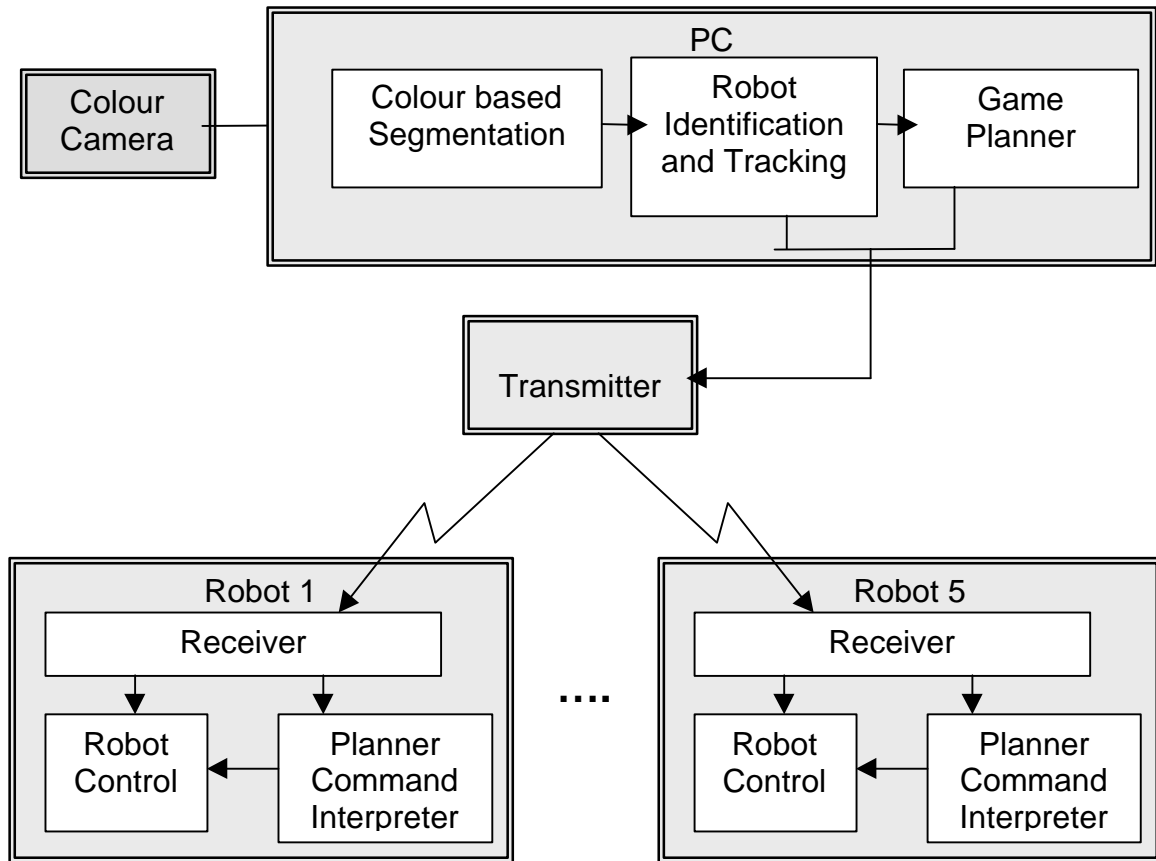


Figure 1: Block diagram illustrating the architecture of the RoboRoos soccer playing system.

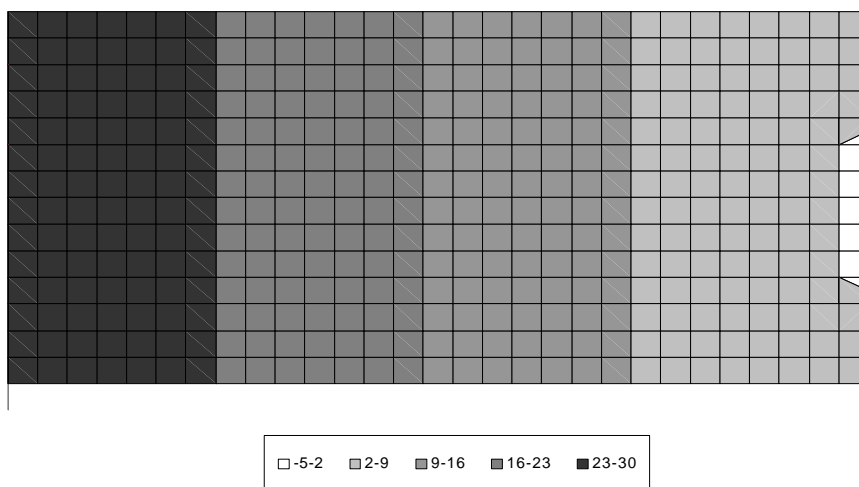


Figure 2. Base field representation. All subsequent fields build on this basic representation.

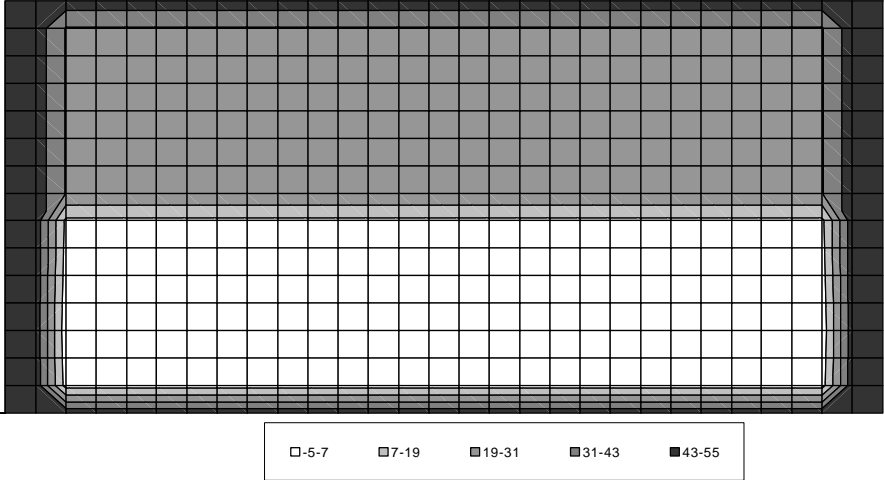


Figure 3. Robot's field position representation (right wing).

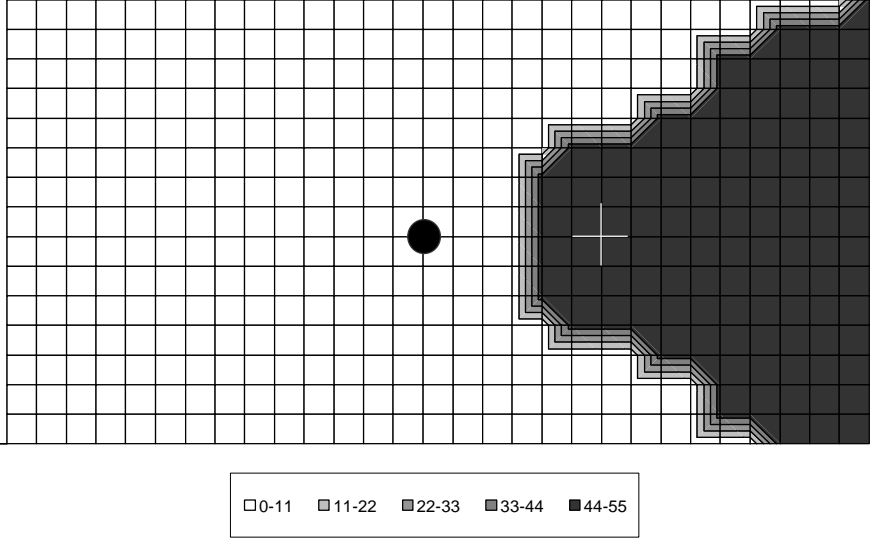


Figure 4. Clear path to ball function representation

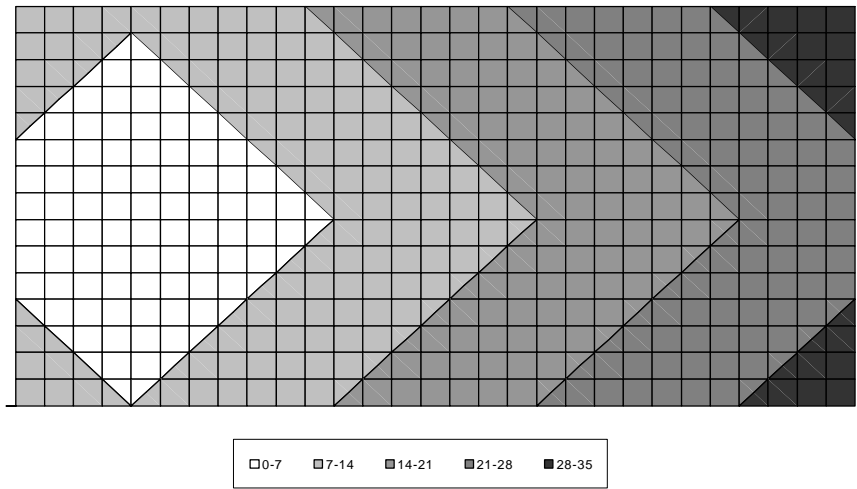


Figure 5. Distance from current position function representation.

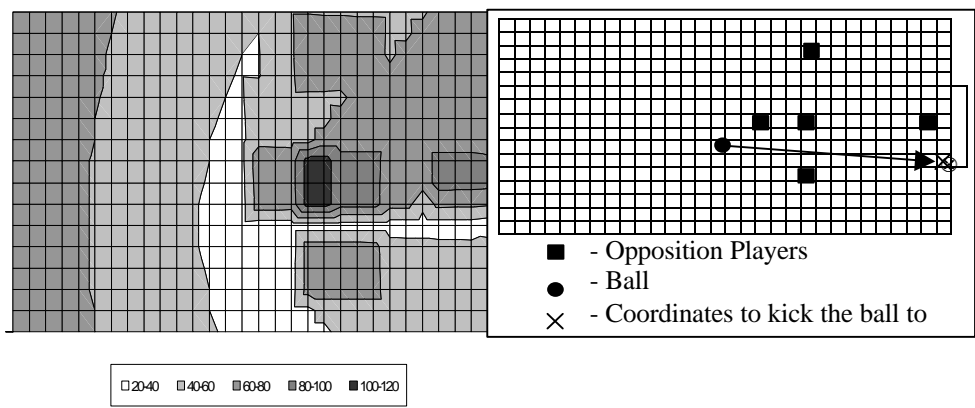


Figure 6. Contour plot and player position overlay showing an example kicking coordinate potential field.

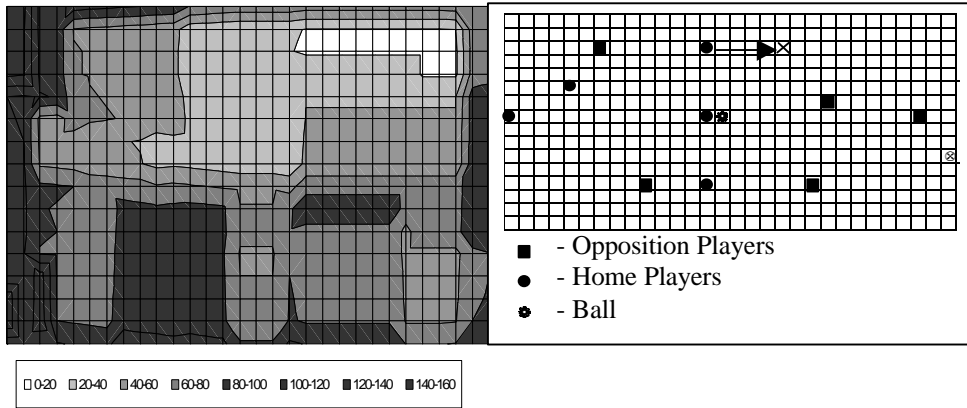


Figure 7. Contour plot and player position overlay showing an example attack coordinate potential field.

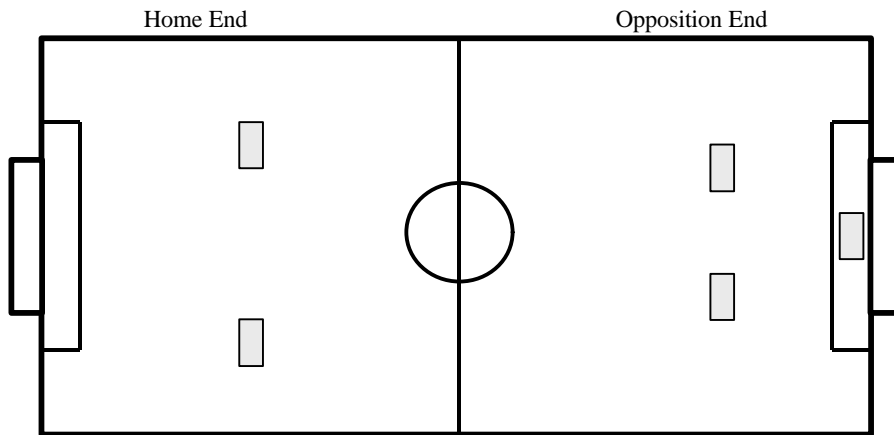


Figure 8. Opposition set up for testing.