COORDINATION AND LEARNING IN A TEAM OF MOBILE ROBOTS

HUOSHENG HU, KOSTAS KOSTIADIS, ZHENYU LIU

Department of Computer Science
University of Essex, Wivenhoe Park
Colchester CO4 3SQ, UK

ABSTRACT

Research on the cooperation of multiple mobile robots has to address three main problems: (i) how to appropriately divide the functionality of the system into multiple robots, (ii) how to manage the dynamic configuration of the system in order to realise co-operative behaviours, and (iii) how to achieve coordination and learning for a team of mobile robots. This paper addresses these issues using a behaviour-based approach. More specifically, the aim of this research is to develop a team of mobile robots with coordination and learning capabilities for the robot soccer competition. The methodology to implement co-operative behaviours and the learning strategy is presented. The construction of Essex Wizards soccer robots and initial simulation results are introduced to demonstrate the feasibility.

Keywords: Mobile Robots, Multirobot Systems, Cooperative Behaviours, Learning, Soccer Robots.

1. INTRODUCTION

During the last few decades, major research efforts were focused on improving the performance of an individual mobile robot by using advanced sensors, actuators, and intelligent control algorithms. This was mainly driven by the need to perform increasingly complex tasks, that required by real-world applications. As a result, individual mobile robots have become very sophisticated. Recently, an alternative approach that uses a number of simpler robots to achieve complex tasks via co-operation has been researched [1] [2][10][17]. Such an approach offers potentially significant advantages in terms of flexibility of operation, and fault tolerance due to redundancy in the number of mobile robots available [11]. More specifically, RoboCup [15] and MiroSot [14] are international robot soccer games that have been launched recently in order to promote the research in multiple robot systems and robotic intelligence.

As the number of mobile robots in a system increases, planning and control of the system becomes increasingly complex. The methods to handle such complexity include a centralised control method and a decentralised control method. More specifically, in a centralised control method all planning and decision-making functions are handled by a single control centre [14]. Each mobile robot contains few simple sensors for control and guidance, the actuators for operation, and the communication facilities for data exchange with the control centre. All the movements of mobile robots in the system are controlled from the control centre and conflicts among multiple robots are easily solved. This method has been widely adopted in manufacturing industry and warehouses where multiple mobile robots are used to transfer parts and clean warehouses [9]. One major disadvantage of the system is that the whole system will stop functioning immediately if the control centre fails.

In contrast, a decentralised control method is to equip each mobile robot with multiple sensors and embedded computers in order to sense its environment, build maps, and plan actions [3][8]. In any unforeseen situation, the robot is able to plan a new path or find a solution without waiting for commands from the control centre if there is one. The function of the control centre is only limited to the broadcasting of traffic flow information received from all robots and the allocation of tasks in the system. Inter-robot communication becomes necessary since competition for resources should be avoided and sharing experience could improve system performance. In both control methods, to realise the co-operation of multiple mobile robots, three main issues need to be addressed: (i) how to appropriately divide the functionality of the system into multiple robots, (ii) how to manage the dynamic configuration of the system in order to realise co-operative behaviours, and (iii) how to achieve coordination and learning for a team of mobile robots. This paper is to address these issues using a centralised approach.

In the next section a brief introduction of co-operative behaviours and their implementation is presented. Section 3 describes a simulated soccer agent to demonstrate how the simulated soccer robots can learn from their experience. The design and construction of the first generation of Essex Wizards soccer robots is given in section 4. The initial simulation results of the proposed learning strategy and trajectory planning are given in section 5. Finally, brief conclusions and future work are summarised in section 6.

2. CO-OPERATIVE BEHAVIOURS

2.1 DEFINITION

A primary aim in the development of teams of co-operative mobile robots in general, soccer robots in particular, is to synthesis cooperative behaviours of multiple mobile robots. Cooperative behaviours enable a team of mobile robots to accomplish missions that cannot easily be achieved with an individual mobile robot [12]. Many cooperative behaviours can be synthesised for the co-ordination of multiple mobile

robots based on the reactive behaviour architecture proposed by Brooks [5]. In this section, several cooperative behaviours are identified:

- Safeguard behaviour – to keep a safe distance among mobile robots during coordination, and to protect robots from colliding with other objects.
- Communication behaviour – to realise either inter-robot communication or exchange information with a control centre in order to make co-operation of multiple robots feasible.
- Flocking behaviour – to enable multiple mobile robots to form a team where each robot makes its own contribution towards a particular goal.
- Convey behaviour – to implement a task cooperatively, i.e. transferring an object down a narrow jointly.
- Formation behaviour – to implement attacking/defence tasks in the robot football and to search a large area in rescue jobs and security patrols.
- Group learning behaviour – to learn from other mobile robots and to share experience each other in order to achieve high learning speed [12][19].

Many additional cooperative behaviours are to be synthesised during the next stage of our research, for instance homing behaviour and foraging behaviour. This should be easy to implement because of our modular design.

2.2 IMPLEMENTATION

The proposed robot behaviours can be implemented using two different schemes: a centralised system and a decentralised system. As described earlier, the centralised system is a simplified way to approach a multi-robot system in which a central control system supervises all the robots and control them. Figure 1 shows a centralised control scheme for a multi-robot system. Most of the behaviours are in fact implemented by the central controller. Each robot is only equipped with safeguard behaviour and communication behaviour. In this case, each robot has little intelligence that is limited to avoiding obstacles dynamically. The central control system is in charge of monitoring, planning, and decision making. If anything goes wrong with it, the whole system will stop functioning. This scheme has been used in the RobotCup small robot league [15] or MiroSot competition [14] since the size of each robot is very limited. No space is available for many sensors and no computing power is enough for planning and decision making. Instead an overhead vision system and a Pentium PC, as a control centre, are used to control a team of robots to play the game.

In contrast, the decentralised system has distributed sensors and intelligence in each individual robot, which is to be maximised in order for each robot to generate an intelligent action by itself. Figure 2 shows such a control scheme where no central control system is presented. Such a system is more robust than the centralised system in a sense that each robot is able to make decisions in face of unexpected situations. In fact, the central control system, if one exists, is only in charge of allocating the tasks and resources. This scheme has been used in the middle size robot league in the RoboCup events where no central control system is allowed, which is a big challenge for every team.

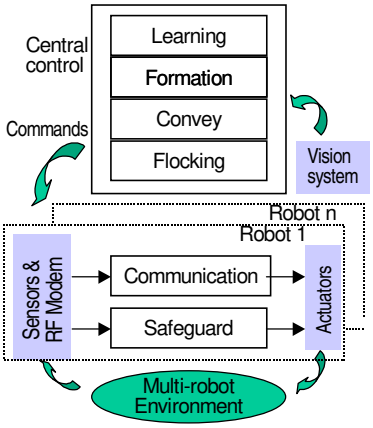


Figure 1 Cooperative behaviours in a centralised robotic system

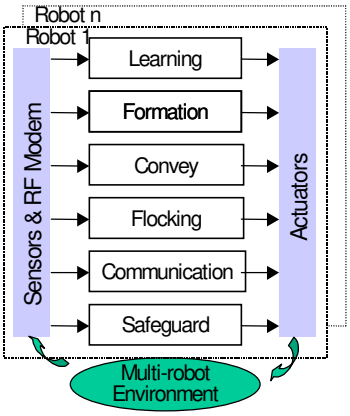


Figure 2 Cooperative behaviours in a decentralised system

It was decided that the first step of our research would be focused on building a centralised robot system for both RoboCup small-size robot and MiroSot competitions, as well as developing a software system for the RoboCup simulation league. A decentralised robot system will be built in the next stage.

3. SIMULATED SOCCER AGENTS

This section presents a simulated soccer agent currently built at Essex for the RoboCup. The learning behaviour of a team of simulated soccer robots is demonstrated.

3.1 AGENT REQUIREMENTS

The robotic soccer simulator is an instance of a client/server application in which each client communicates with the server via a UDP socket [21]. The server is responsible for executing requests from each client and updating the map. At each 150ms, the server broadcasts visual information to all clients: their position, the field of their view, and their facing direction on the field. It also sends auditory information to various clients at random time intervals.

After processing the sensory data, the clients respond by sending action requests to the server from a set of primitive actions available to them. To avoid message congestion on the server, clients are allowed to send one request per cycle:

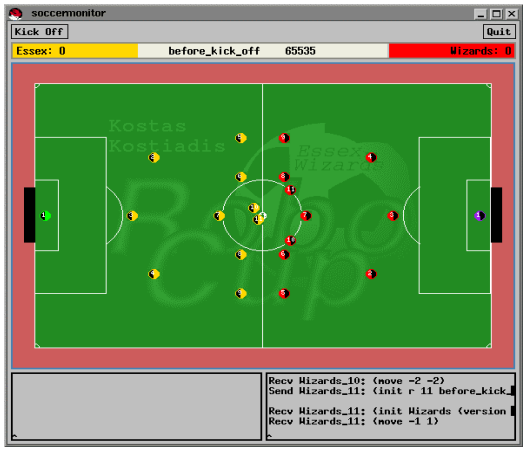


Figure 3 Robot soccer simulator

100ms. If no message is sent within this interval, the client will not perform any actions. If more than one message is send during the same cycle, the server executes only one, chosen at random. The server updates the state of the environment by serially executing each request. The results are projected on a window shown in figure 3.

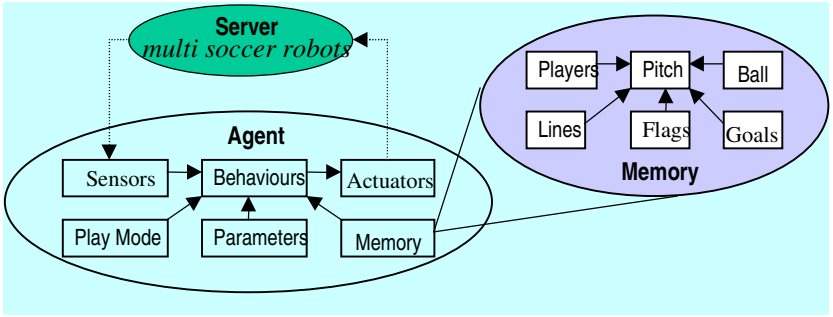


Figure 4 Block Diagram of the Simulated Soccer Agent's Structure

As shown in figure 4, the robot agent contains six different modules: sensors, behaviours, actuators, the current play mode, parameters and memory. The dashed arrows are communication links between the agent and the server. The server sends information received by the agent's "Sensors" module, and the agent sends information back to the server through its "Actuators" module. The solid arrows show dependence relationships between the modules. More specifically,

- Sensors -- are responsible for receiving and analysing the visual or auditory information transmitted by the server. Due to the limited view in each cycle the agent updates only part of the environment. Therefore the agent holds a probabilistic map for the whole pitch.
- Behaviours -- is responsible for generating actions depending on the current state of the environment. It collects information from other modules, including the agent's position and role in the team, the current team formation, the position of the ball and other agents, the current play mode and so on. It processes all this information and determines the best course of action.
- Actuators -- are responsible for receiving an action (or a set of actions) from the "Behaviours" module, and sending them to the server at every 100ms.
- Play Mode -- holds and updates the current play mode using the information received by the "Sensors".
- Parameters – hold information regarding various settings both for the server and the robot agents. The parameters affect the behaviours of the agent since they include information regarding the role of the agent in the team and most of the server parameters.
- Memory – is a representation of the whole soccer pitch that contains players, lines, flags, goals and a ball. Each of them is associated with a confidence value which represents the confidence that an object is at the current coordinates. If an object was seen in the last cycle, its confidence value is 1. This value is decreased if the object has not been seen in the current cycle. When it falls under a certain threshold the object is "forgotten".

3.2 REINFORCEMENT LEARNING

Based on machine learning techniques and agent technology [6][18], an agent can significantly improve its performance from earlier experience. Since co-operation in adversarial domains is a very challenging task, the proposed learning algorithm assigns each agent a role to play to achieve a certain goal. By distributing the responsibilities among the agents, an efficient way of co-operation emerges. Reinforcement learning (RL) differs from the more widely studied problem of supervised learning in that the agent is not provided with input/output pairs. Instead, after choosing an action the agent is told how well it performed, but is not told which was the best action. Hence it is necessary for the agent to gather useful experience about system states, actions, transitions and rewards in order to act optimally. As we know, the standard RL model consists of: (i) a discrete set of states $s_i \in S$; (ii) a discrete set of actions, $\alpha_i \in A$; (iii) a set of reinforcement signals, either {0, 1} or the real values $r_i \in R$. The interaction of the agent with its environment can be described by the following diagram:

$$s_0 \xrightarrow{\alpha_0} r_0 \ldots s_i \xrightarrow{\alpha_i} r_i \ldots s_n \xrightarrow{\alpha_n} r_n$$

where $\{s_0, s_1, \ldots, s_n\}$ is a set of states, and $\{\alpha_0, \alpha_1, \ldots, \alpha_n\}$ is the set of actions. The real values r_i indicate the immediate reward an agent receives when performing α_i from s_i.

Apart from the immediate reward an agent receives after each action, there is also a delayed reward arriving from the subsequent (state, action) pairs the agent encounters. For the diagram above, the total reward the agent receives is:

$$r_0 + \gamma r_1 + \gamma^2 r_2 + \ldots \gamma^N r_N = \sum_{i=0}^{N} \gamma^i r_i \qquad (1)$$

where $\gamma \in [0,1)$ is a constant that determines the relative value of delayed versus immediate rewards and N is the expected lifetime of the agent. N might be unknown or ∞.

The agent's task is to learn a control strategy, or policy, $\pi : S \to A$, which maximises the expected sum of these rewards, with future rewards discounted exponentially by their delay. Before choosing a behaviour that can learn to act optimally, a model of optimality needs to be established. As described in [19][22], there are mainly 3 models that have been the subject of the majority of the work in this area:

- The finite-horizon model *(N=h)*
- The infinite-horizon discounted model *(N=∞)*
- The average-reward model *($N \to \infty$ as $\gamma \to 1$)*

where *N* and γ are the agent's expected lifetime and reward discount factor respectively.

The finite-horizon model -- the agent should optimise its expected reward for the next *h* steps,

$$R = E\left(\sum_{i=0}^{h} r_i\right) \qquad (2)$$

and should not worry what would happen after that. Here r_t represents the reward received t steps into the future. This model has two variations. Firstly, the agent has a non-stationary policy, i.e. one that changes over time. The agent will choose what is termed an h-step optimal action that is the best action available, given that the agent has h steps remaining in which it can act and gain reinforcement. Then the agent will choose an (h-1) optimal action, and so on until it finally takes a one-step optimal action and terminates. Secondly, the agent does receding-horizon control, in which it always performs the h-step optimal action. The agent always uses the same policy, but h limits how far ahead the agent looks in choosing its actions. The finite-horizon model is not always appropriate, especially in situations where the precise length of the agent's life is not known in advance.

The infinite-horizon-discounted model -- takes the long-run reward of the agent into account, but rewards received in the future are discounted exponentially by a factor $\gamma \in [0, 1)$:

$$R_i = E\left(\sum_{i=0}^{h} \gamma^i r_i\right) \qquad (3)$$

This model is similar to the receding-horizon control, but the discounted model gives a better mathematical interpretation than the finite-horizon model. This is the reason it has been chosen in this project.

The average-reward model -- favours actions that optimise the long-run average reward when $h \to \infty$:

$$\overline{R} = E\left(\frac{1}{h}\sum_{i=0}^{h} r_i\right) \qquad (4)$$

The learning task -- the agent senses the current state s_I, and chooses an action α_t at each discrete time step i. The environment responds by rewarding the agent $r_i = r(s_i, \alpha_i)$, and producing the succeeding state $s_{i+1} = \delta(s_i, \alpha_i)$. Note that the functions r and δ are not known to the agent. The agent's task is to learn a policy $\pi: S \to A$, for selecting actions based on the current state. Needless to say, the agent should look for a policy that will maximise the discounted cumulative reward achieved from an initial state s. In other words, the agent's learning task is to find a policy π, which will maximise the discounted cumulative reward for all states. Such a policy is called an optimal policy, and is denoted by π^*. The initial results of this learning strategy are presented in section 5 and more details can be found in [13].

4. REAL SOCCER ROBOTS

4.1 ROBOT CONFIGURATION

Figure 5 presents the block diagram of one of Essex Wizards soccer robots currently constructed. The robot is physically small and it measures 75mm in width, 75mm in length and 75mm in height. Two front wheels are differentially driven by two small D.C. motors with in-line gearboxes and optical encoders. The back of each robot is supported by a single castor wheel. The motor control is implemented with pulse width modulation, providing a speed of up to 0.5m/s and directional control.

The design of the robot is modular. Each robot has two layers, namely L1 and L2. As shown in figure 5, L1 accommodates a RF receiver, and L2 has a microcontroller and 4 pairs of infrared sensors. Under L2 there are 4 batteries to provide 6V DC power. An 8MHz Motorola 68HC11E2 microcontroller is on board L2, which is used for low-level motion control and communication. A RF (Radio Frequency) receiver in layer 1 is used for receiving commands from the central controller. The microcontroller receives an interrupt when the communication system receives a byte of data from the central control station, and then takes an action accordingly. In contrast, all infrared sensors are used for detecting other robots and walls.

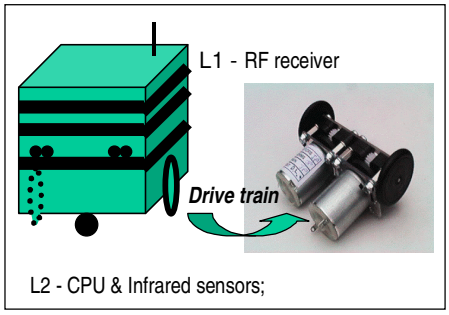


Figure 5 Essex Soccer robot & its drive train

As can be seen from figure 6, each pair of infrared sensor has one infrared LED as an emitter and one phototransistor as a receiver. The emitter generates a modulated light, and the receiver detects the reflected light from any object within 10cm. A modulator and a demodulator are used to distinguish the emitted light by other robots.

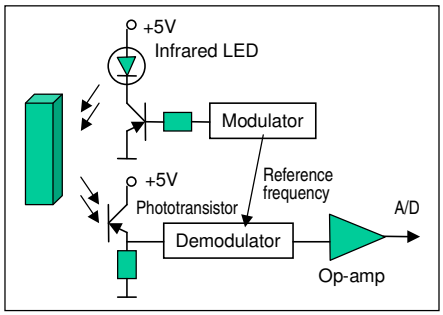


Figure 6 Infrared sensor module for obstacle detection

4.2 INTER-ROBOT COMMUNICATION

Communication behaviour is a key design issue for multi robot systems [23]. The communication may take place directly via explicit communication facilities such as radio link or indirect (pseudo communication method) through one robot sensing a change in other robots or its environment. It should be noted that the communication mechanism utilised is critical for the successful implementation of multiple mobile robots.

In general, audio or ultrasonic sound region media has rather low data rates. An infrared communication system with multiple channels could be used for inter-robot communications of soccer robots thus achieving high efficiency and reliability [12]. But it cannot function well under light noises from the robot's environment. Instead, RF communication has been used here since it isn't affected by any light and doesn't have to align the transmitters and the receivers in any way. Information (motion commands to a team of soccer robots) from a Pentium PC is frequency-modulated and transmitted by the RF transmitter. The decoding process is carried out within the individual soccer robot using a RF receiver. All robots will follow the motion commands being received to play a role in the game.

4.3 VISION-BASED CONTROL CENTRE

A centralised control scheme is used for managing a team of soccer robots, shown in Figure 7. A Pentium III computer with 450MHz CPU and 128Mbyte RAM are adopted as the host computer. A CCD colour camera, *JVC TK-C1380E*, overlooks the whole football pitch from a height of over 2 meters at the pitch centre. A PCI frame grabber is used to grab the image, which is made by Ellips and can sample at a rate of 14.75/7.38MHz for PAL.

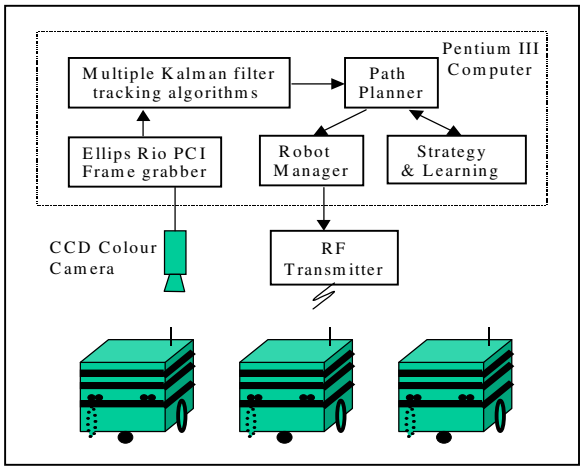


Figure 7 Block diagram of our soccer robots

At each control cycle, the central computer implements multiple Kalman filter algorithms to detect the ball's position and the positions of all soccer robots on the pitch simultaneously, including its own team and the opponent team. All the position information is passed to a path planner which does the required calculations to generate individual subgoals for each robot agent. There is a library containing decision-making strategies and the learning algorithm which are used by the path planner to generate optimal solutions. All the generated subgoals are sent to the robot manager that generates more detailed motion plans such as catching the ball and passing the ball in order to control every member of the team to move quickly.

5. INITIAL RESULTS

5.1 A LEARNING EXAMPLE

To evaluate the performance of the proposed reinforcement learning strategy, the "2 attackers versus 1 defender" case was used. As shown in figure 8 and figure 9, two attackers (players 2 and 3) are in front of the enemy goal and the one defender (player 1) is trying to stop them from scoring. The correct strategy for scoring is dependent upon the defender's position. If the defender is blocking a direct shot, a pass should be executed, see figure 8. Otherwise if the defender is blocking the pass, a direct shot is preferred, see Figure 9.



Fig. 8 Defender blocking shot



Fig. 9 Defender blocking pass

Initially a "random attacker" was used to this case. In other words, the choice between a shot and a pass is completely random and is irrelevant of the defender's position. The results of 141 trials are shown in table 1. Horizontally the table indicates the number of successful, failed, and total number of trials. Vertically the table shows the number of passing, shooting, and total attempts. Then, a "Q-learning striker" was used, which tries to learn the optimal action depending on the state it currently is. The results of 120 trials are shown in table 2, which shows the extremely high success rate.

Table 1 Results for random striker

Random striker	Pass	Shoot	Total
Success	34(55.74%)	37(46.25%)	71(50.35%)
Failure	27(44.26%)	43(53.75%)	70(49.65%)
Total	61	80	141

Table 2 Results for Q-Learning (Q-L) striker

Q-L striker	Pass	Shoot	Total
Success	52(100%)	67(98.53%)	119(99.17%)
Failure	0(0%)	1(1.47%)	1(0.83%)
Total	52	68	120

The reason for the high success rate lies in the back-propagating nature of Q learning. Initially the learner's Q' hypothesis about all state-action pairs is set to zero. Since both actions have the same Q value (i.e. zero) the agent randomly chooses an action. If it is the wrong action, no reward is received and again chooses randomly so at the next trial. However if a successful choice is made, the reward that is received makes this action optimal thereafter. Hence, once the agent finds the correct action at each state, it then acts with 100% accuracy. In other words, once the goal-state has been reached, the learning propagates backward to the previous states and so on.

5.2 TRAJECTORY PLANNING

Each robot has a trajectory planner that generates a smooth trajectory for its motion inside the pitch. The planned trajectory is mainly composed of straight lines and arcs. To generate a smooth trajectory, we use the improved polar polynomials by fitting splines or polynomials between those line segments that have different slopes [7]. Figure 10 presents a typical trajectory with a line segment and an arc.

To avoid obstacles, each soccer robot is embedded with the safeguard behaviour. Two graphs in figure 11 presents the simulation result to demonstrate this ability. The left graph shows the soccer robot is commanded to navigate around two obstacles from point A to point B. The right graph presents the actual trajectory generated.

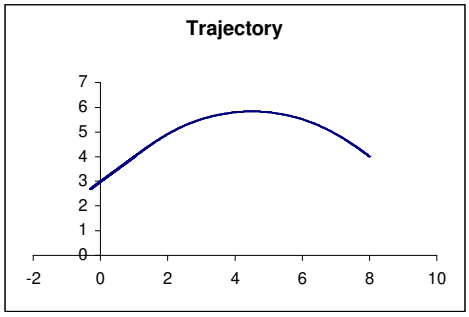


Figure 10 A example trajectory

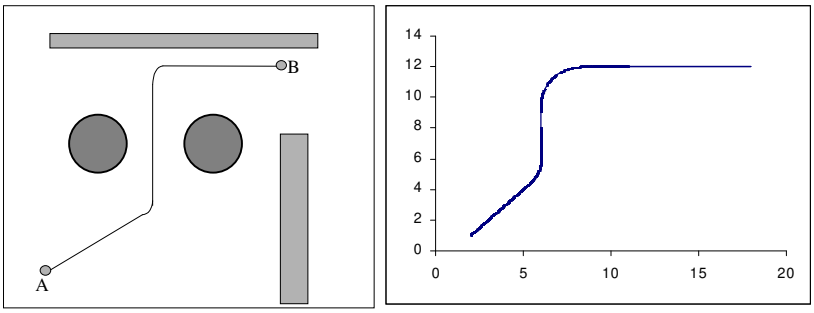


Figure 11 Avoiding two obstacles

6. CONCLUSIONS

The advantages of cooperative mobile robots over single complex mobile robot include the potential for increased fault tolerance, simpler robot design, and wide application domain. This paper presents a multi-agent system in the robot soccer domain to investigate both simulated and embodied intelligence. Each soccer robot requires sensor-motor-skills to shoot the ball, to dribble and pass the ball, to avoid serious crashes with other players, and so on. This is very challenging task, as the strategies and behaviours of the opponents are uncertain. This task is significant important for applications-oriented research.

The next stage of our research is to investigate other cooperative behaviours towards real world applications. More specifically, we will address: (i) how adaptation and learning algorithms should be improved to make robotic systems more flexible in a dynamic environment? (ii) what data fusion algorithms are required to capture environment features effectively and deal with uncertainties? (iii) how low-level behaviours such as kicking, passing are effectively integrated with high-level behaviours, i.e. team formation?

ACKNOWLEDGEMENTS

We wish to thank Malcolm Lear and Robin Dowling for building soccer robots and the football pitch. This research is supported by the University Research Office under the Research Promotion Fund DDP940.

REFERENCES

[1] R. Alami, S. Fleury, M. Herrb, F. Ingrand, F. Robert, Multi-robot cooperation in the MARTHA project, *IEEE Robotics and Automation Magzine*, Vol. 5, No. 1, pages 36-47, 1998.

[2] R.C. Arkin, Integrating behavioural, perceptual and world knowledge in reactive navigation, *Journal of Robotics and Autonomous Systems* Vol. 6, 1990.

[3] H. Asama, Operation of cooperative multiple robots using communication in a decentralised robotic system, *Proc. Conf. on From Perception to Action*, Switzerland, 7-9 Sept. 1994.

[4] M. Brady and H. Hu, The Mind of a Robot, *Philosophical Transactions: Physical Sciences and Engineering*, the Royal Society, Vol. 349, No. 1691, pages 15-28, London, 1994

[5] R.A. Brooks, A robust layered control system for a mobile robot, *IEEE Journal of Robotics and Automation*, Vol. RA-2, No. 1, pages 14-23, 1986.

[6] D.E. Goldberg, *Genetic algorithms*, Addison Wesley, 1989.

[7] D. Gu, H. Hu, M. Brady, F. Li, Navigation System for Autonomous Mobile Robots at Oxford, Proc. Int. Workshop on Recent Advances in Mobile Robots, Leicester, England, pages 24-33, 1-2 July 1997

[8] H. Hu, D. Gu, M. Brady, A Modular Computing Architecture for Autonomous Robots, *Int. Journal of Microprocessors and Microsystems*, Vol. 21, No. 6, pages 349-362, March 1998

[9] H. Hu and M. Brady, Towards Advanced Mobile Robots for Manufacturing, a Chapter in the Book ``*Artificial Intelligence and mobile robots*", Editors D. Kortenkamp, P. Bonasso, and R. Murphy, pages 297-321, MIT Press, March 1998

[10] H. Hu, I. Kelly, D. Keating, D. Vinagre, Coordination of Multiple Mobile Robots via Communication, *Proc. SPIE'98, Mobile Robots XIII Conf.*, Boston, pp. 94-103, Nov. 1998

[11] Y. Ishida, *et. al*., Functional complement by co-operation of multiple autonomous robots, *Proc. IEEE Int. Conf. On Robotics and Automation*, pages 2476-2481, 1994

[12] I.D. Kelly and D.A. Keating, Faster learning of control parameters through sharing experiences of autonomous mobile robots, *Int. Journal of System Science*, Vol. 29, No. 7, pages 783-793, 1998

[13] K. Kostiadis and H. Hu, Reinforcement Learning and Co-operation in a Simulated Multi-agent System, *IEEE/RSJ Int. Conf. on Intelligent Robotics & Systems*, Korea, Oct. 1999

[14] K.H. Kim, K.W. Ko, J.G. Lee and H.S. Cho, The Development of a Micro Robot System for Robot Soccer Game, *Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 644-649, New Mexico, April 1997

[15] H. Kitano, M. Tambe, P. Stone, M. Veloso, et all., The RoboCup Synthetic Agent Challenge, 97. Proc. of *Int. Joint Conference on Artificial Intelligence (IJCAI97)*, 1997.

[16] J. M. Mataric, *Interaction and Intelligent Behaviour*. PhD Thesis, Massachusetts Institute of Technology, 1994.

[17] M.J. Mataric, Learning in Multi-Robot Systems, In *Lecture Notes In Artificial Intelligence -- Adaptation and Learning in Multi-Agent Systems*, Vol. 1042, Springer-Verlag, 1996.

[18] T.M. Mitchell, *Machine Learning*. McGraw-Hill International Editions, Singapore, pp. 367-388.

[19] R.J. Mitchel, D.A. Keating, C. Kambhampati, Learning strategy for a simple robot insect, *Proc. IEE Control'94 Conference*, pages 492-497, 21-24 March 1994.

[20] K. Narendra and M. Thathachar, *Learning Automata: an introduction*, Prentice-Hall, 1989.

[21] I. Noda, Soccer Server: A Simulator for RoboCup. *JSAI AI-Symposium 95: Special Session on RoboCup*, Dec. 1995

[22] I. Noda, H. Matsubara, K. Hiraki, Learning Co-operative Behaviour in Multi-Agent Environment -- A case study of Play Plans in Soccer, *Proceedings of 4th Pacific Rim International Conference on Artificial Intelligence*, pp. 570--579, Cairns, Australia, Aug. 1996.

[23] J. Wang, On signboard inter-robot communication in distributed robotic systems, *Proc. IEEE Int. Conference on Robotics and Automation*, pages 1045-1050, May 1994.