Clockwork Orange: The Dutch RoboSoccer Team

Marco Wiering[1], Matthijs Spaan[2], Robert Bartelds[3], and Frank de Jong[3]

[1]Intelligent Systems Group, Utrecht University, the Netherlands
[2]Intelligent Autonomous Systems Group, University of Amsterdam, the Netherlands
[3]Pattern Recognition Group, Delft University of Technology, the Netherlands

1 Introduction

The Dutch RoboSoccer Team, called "Clockwork Orange" is a Dutch research project in which 3 universities participate: the University of Amsterdam, the Technical University of Delft, and University Utrecht. In this research project we study multi agent systems in general and multi robot systems in particular.

Multi Agent Systems (MAS) will play an important role in the future. There are many real-world problems involving multiple interacting agents, such as traffic control, public safety, pollution detection, fire fighting etc. Real world agents should cooperate together to come up with an optimal sensing and acting strategy. RoboCup Soccer provides us with a challenging environment for doing novel research in Multi-robot systems. First of all, the robot agents should be able to gather sensor information using camera's, laser range finders, and sonar sensors, and integrate the information about the world using communication and sensor data fusion. Furthermore, the robot agents should be provided with local intelligence, creating a distributed team in which team roles can be used for coordinating the team behavior. Such a team should be robust in case one of the team members is malfunctioning. The RoboSoccer environment also allows for learning strategies against a particular opponent, offering research possibilities for opponent modelling and machine learning. This short paper will describe our hardware (the robots) and the themes of research we are currently investigating.

2 The Robots

Our RoboSoccer team consists of two types of robots: three Nomad Scout robots (used by Delft and Amsterdam) and one Pioneer 2 robot (used by Utrecht). The Nomad Scout (see figure 1(Left)) has a camera, 16 ultrasonic sonar sensors, a tactile bumper ring, and odometry sensors for sensing and a pneumatic driven kick device as effector. The Pioneer 2 (see figure 1(Right)) uses a laser, 16 ultrasonic sonar sensors, odometry sensors, and a camera for sensing. It uses the same pneumatic kick device as effector. For communication between robots we use a wireless Ethernet system, with a data rate of max. 3 Mbps giving the robot an action radius of up to 1 km. It uses the 2.4GHz ISM band and is compliant with IEEE 802.11. Specifications of the robots are given in table 1.

Specification	Nomad Scout	Pioneer 2
Diameter	41 cm	44 cm
Height	35 cm	22 cm
Payload	5 kg	20 kg
Weight	25 kg	9 kg
Battery power	300 watt hour	252 watt hour
Battery duration in game	1 hour	30 min
Max. Speed	1 m/s	2 m/s
Max. Acceleration	2 m/s^2	2 m/s^2
Special sensors	Camera	Laser + camera

Table 1. The specifications of our robots.

Fig. 1. Our robots. Left: a Nomad Scout robot. Right: the Pioneer 2 robot.

3 Teamskills

We use a distributed shared dynamic world model for team coordination. Sharing the same world model enables the team members to reason not only about their own actions, but they can also predict the next action a team member will take. The world model is maintained locally which can cause small differences between the world models of each robot due to network lag. Communication on the team coordination level is used to improve robustness and is almost a necessity since the team is heterogeneous both in hardware and software. The Utrecht Pioneer 2 robot uses a subsumption architecture while the Amsterdam/Delft Nomad Scouts operate on a hybrid architecture. Our team uses a global team behavior (a strategy) from which each robot derives its individual behavior (also known as role).

Our teamskills module determines the current team behavior using a simple finite state machine. Each player has its individual behavior defined by the team behavior. There are about half a dozen different individual behaviors available, all of which have an attacking, defending or intercepting purpose. Utility functions based on potential fields [4, 3] are used to distribute the roles among the team members. This technique is similar to the ones used by other participants in the middle size league [2, 6]. Each robot calculates its own scores for the individual behaviors useful at the moment, communicates them and then calculates the scores of its team mates. Incoming scores from one of your team mates overwrite the scores you calculated for him. This introduces a level of redundancy which can improve the team performance. When everybody knows the scores for the individual behaviors of each team member the roles can be easily distributed.

The exact specification of an individual behavior is not (and can not be) the same on both architectures. The following describes the Amsterdam/Delft approach which is similar to the one Tambe [5] described for use in the simulation league. A Markov decision process is used to search through state space of a robot's future states. First a number of possible actions and their parameters with respect to the current situation and individual behavior are generated. For each of these actions the probability of success is calculated. The impact on the world of an action and the expected concurrent actions of team mates and enemy players is evaluated. Modelling team mate actions is relatively easily compared to modelling your opponent with reasonable accuracy. The evaluation criteria both include a role evaluation (using potential fields) as well as concepts like ball possession. Now the action with the maximum expected utility can be chosen and the process can be repeated to look further ahead. One of the challenges of this approach is to calculate the probability a certain action will succeed; after an estimation by hand we want to use a technique like reinforcement learning to adjust the probabilities.

4 Behavior Based Reinforcement Learning

The Pioneer 2 robot of Utrecht University uses an extended version of the subsumption architecture [1] in which particular behaviors such as *Get_Ball, Dribble*, and *Score* compete for controlling the robot. The competition is modelled by using preconditions for the behaviors and a priority scheme for selecting a possible behavior; a behavior such as *Emergency_Brake* has the highest priority.

Learning is integrated as part of the behavior-based control architecture of the robot. This is done by providing the robot with different implementations of the same behavior. E.g. we use three *Get_Ball* behaviors instead of just one. Particular behaviors may implement fast play while allowing for errors, whereas another behavior may implement slow, but precise play. These behaviors are tested during online play against an opponent. If a particular behavior is successful against an opponent team (e.g. the *Get_Ball* succeeds in getting the ball within 10 seconds), the behavior gets strengthened, and is therefore used more often. This allows us to learn a sequence of behaviors which has the highest ex-

pected probability of success against an opponent. Since there are not too many alternative behaviors, a single RoboCup game could be sufficient to test enough behaviors so that the best behaviors can be executed during the final part of the game. To guarantee the best behavior is found, there is an exploration function which tests alternative, currently thought sub-optimal, behaviors.

Although the opponent is not explicitly modelled, which would require a lot of training data and games, our method can deal with different opponents by quickly learning the best functioning sequence of behaviors. This learning behavior may provide the robot player with additional strength, possibly leading to quite attractive play during the RoboCup games.

5 Vision

Our vision research is mostly directed at the problems of recognition and self-localisation. We can detect and track all objects typically found in a RoboCup environment. This is done via color detection in a pie chart manner in adapted HSI space. We have implemented 2 different ways to do self-localisation, a local and a global method. Both are based on detecting the lines of the field. The global method first splits the screen into multiple regions of interest (ROIs) and does a rough Hough transform on each of these. Then the method searches for the precise lines using linear edge detectors. These are then matched to the world-model giving an estimation of the position. The local method checks a 3x3x3 grid (x,y and ϕ) to estimate the best position. Per position it calculates the expected lines on the captured image. It then does a distance transform (DT) in a rectangle around the expected line and compares the results, giving a new best estimator. If the DT result becomes too big we switch back to the global self-localisation. Both methods use an inclinometer to estimate the camera tilt, since a small camera tilt can impair distance accuracy and therefore self-localisation severely.

References

1. R.A. Brooks. Achieving artificial intelligence through building robots. *MIT AI Lab Memo 899*, 1986.
2. C. Castelpietra, L. Iocchi, M. Piaggio, A. Scalzo, and A. Sgorbissa. Communication and coordination among heterogeneous mid-size players: ART99. In *Proceedings of the 4th International Workshop on RoboCup*, pages 149–158, 2000.
3. O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *International Journal of Robotics Research*, 5(1):90–98, 1986.
4. J. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, 1991.
5. M. Tambe and W. Zhang. Towards flexible teamwork in persistent teams: Extended report. *Journal of Autonomous Agents and Multi-Agent Systems*, 3(2):159–183, 2000.
6. T. Weigel, W. Auerbach, M. Dietl, B. Dümler, J. Gutmann, K. Marko, K. Müller, B. Nebel, B. Szerbakowski, and M. Thiel. CS Freiburg: Doing the right thing in a group. To appear in P. Stone, G. Kraetzschmar, T. Balch, RoboCup 2000, Springer-Verlag, 2001.