Middle Robots League, Team *BengKiat*, pages 107–112
http://www.ep.liu.se/ea/cis/1999/006/18/

Alpha++

BengKiat

FookSeng Yong, BengKiat Ng, KwokMun Loh, EngCheong Ong, KokKiaw Teo, KiaMeng Loh

FY, BN, KL, EO, KT, KL: Ngee Ann Polytechnic

Abstract. *Robotic soccer is an interesting game which involves two teams of 5 robots each. Each team of robots must cooperate with each other in order to put as many balls as possible into the opponent goal. This paper describes Alpha++, the team that we have developed for the Pacific-Rim RoboCup-98 competition, held in November 1998, in Singapore. Alpha++ is a collaboration between Alpha Centre of Ngee Ann Polytechnic and Alpha Innovations Pte Ltd. Due to the lack of time and experience, we purchased five Mach-5 robots from Newton Research Labs and concentrated our effort on the software aspect of the game.*

1 Introduction

As this was our first attempt at the RoboCup middle-sized league, we had many basic problems to solve before we could proceed to higher-level tasks such as multi-agent control and dynamic learning. Our immediate goals were robot communications, motion control and localisation. After having implemented these, we then proceeded to add a simple game strategy and we were ready for competition.

In this paper, we will first present a brief description of the physical hardware of Alpha++ and describe our solutions to the robot communications, motion control and game strategy.

2 Alpha++ Hardware

2.1 System Setup

Alpha++ comprises a host PC, four player robots and a goalie robots. All player robots are identical and run the same software. The goalie is slightly different from the player as its main purpose is to block incoming ball. Hence, its camera mounting is such that it is able to move sideways while looking forward. All robots and the host PC are fitted with a Radiometrix RF transceiver for communications. We used an Intel-based PC to function

Figure 1: The Mach5 Robots

as a host, which is primarily used to start and stop the game. The host PC is also used to command individual robots to take a kick-off and penalty shots when required. Debugging features were also implemented into the host PC such as the monitoring of the RF signals sent by the individual robots. In this manner we could easily track the status of each individual robot during gameplay.

The host is used to initiate communication by polling the robots regularly. In the event that the host goes down, the robots would continue to communicate among themselves in a round-robin manner. Starting or stopping the robots would be difficult though without the host PC.

2.2 The Mach5 Robots

The Mach5 is a robust, fast-speed robot from Newton Labs, which size and features complying with the rules of the middle-size league RoboCup competition. It is powered by two DC-motors in a wheel-chair configuration one of the most popular motion configuration for mobile robots.

The goalie is essentially the same robot, specially modified to move horizontally which is more useful considering the role of a goal-keeper.

2.3 The Cognachrome Vision System

Within each Mach5 lies the Cognachrome Vision System from Newton Labs. Based on a 16Mhz Motorola 68332 microcontroller, the system tracks objects at a rate of 60Hz. It can be trained to recognise up to 3 distinct colours at a time.

Besides processing vision data, the 68332 microcontroller is also used to control the motion of the Mach5 robot. The Mach5 comes with a multi-tasking capable kernel which we found to be very useful in implementing our game strategy.

Much as we would like to extract as much colour information from the environment as possible, the Cognachrome Vision System has a 3-colour limitation. As such, we had to carefully select the 3 objects we would like

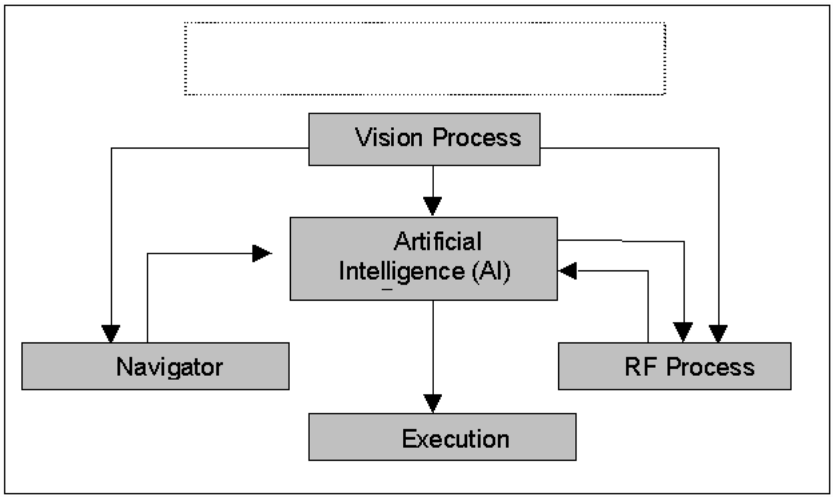

Figure 2: The software/artificial intelligence of the robot is divided into five different processes as shown above.

to see. We therefore selected to track the 3 most important colours the red ball and blue and yellow goal posts. This means that we are unable to see the opponent players. In future, as we progress, we would prefer to use the vision system to recognise the opponent's colour.

Due to the lack of insufficient sensor information, self-localisation is difficult. Our solution to that is to use the encoder readings of the wheels to track positional data. However, wheel skidding and collisions caused this method to be inaccurate as time progressed. We periodically use the goal posts to re-calibrate positional information but this is of limited accuracy.

3 Game Strategy

3.1 The Vision Process

The Vision Process captures visual data such as the ball and the goal posts from the mounted camera, decodes the data and store them into respective variables for the AI Process to use. Software "filters" are also placed here if necessary. These software "filters" filter away unnecessary information or noise a spectator wearing a bright red T-shirt, which is of the same colour as the ball.

3.2 The Navigator Process

The Navigator Process tracks navigational information. It monitors the wheel encoders, reads visual data from the Vision Process and calculates the orientation and position of the robot .

3.3 The RF Process

The RF Process basically does, well, RF communications. Players communicate with one another, and with the host, using radio frequency. Each robot would take turn to broadcast its current location on the field, what it sees and its orientation. This message is received by the other robots, and using this information, they will decide amongst themselves who should go for the ball. This form of communication provides for cooperation among the robots. Thus, we should not see all four players attacking the ball at the same time, which would result in them blocking and crashing into each other.

3.4 The Execution Process

The Execution Process controls the movement of the robot. This process deals directly with the motor interface and controls the robot movement according to its current state. The state of the robot is determined by the AI process.

3.5 The AI Process

The AI Process is the "heart" of the software. It receives information from the Vision Process, Navigator Process and RF Process and decides what course of action to take.

At any one time, the robot will be in one of these states:

- Search State
- Strike State
- Maneuvering State
- Clear Ball State
- Repositioning State
- Penalty State

The AI Process will decide on which state should the robot go into, based on the information it receives from other processes.

3.5.1 Search State:

Normally when the ball is not in sight, this state will be invoked by the AI. In this state, the robot search for the ball until found. When ball is in site, the AI would *usually* invoke another state. However, if another player is currently engaging the ball, the robot would remain in the Search State and does nothing but tracks and looks at the ball, not allowing it to go out of site.

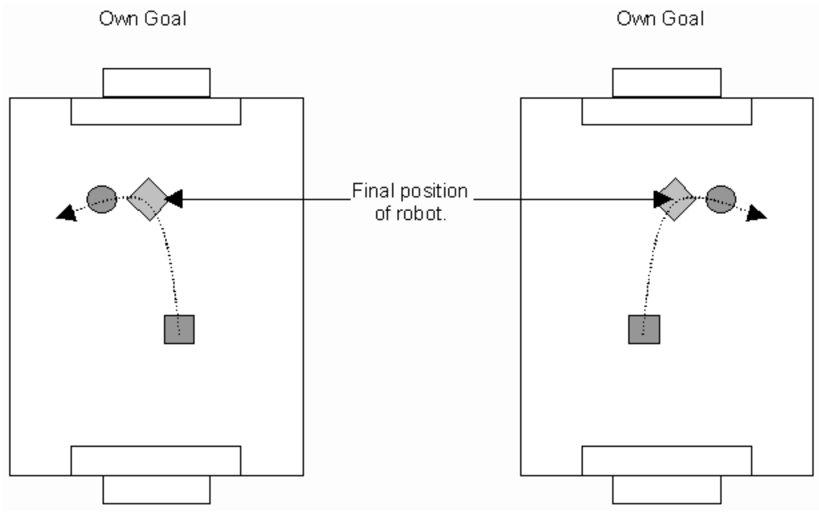

Figure 3:

3.5.2 Strike State:

AI would come into this state if the robot, ball and opponents goal are in line. The robot would then move forward at high speed, simulating a forceful kick at the ball and trying to score for a goal.

3.5.3 Maneuvering State:

In this state, the robot will try to drive the ball towards the opponents goal in order for the ball to be in line with the goal and the robot, so that the robot will be able to jump to Strike State and kick the ball into the opponents goal. When the ball is on the left of the goal, the robot will try to move the ball right. Similarly when the ball is on the right, it will try to move the ball to left.

3.5.4 Clear Ball State:

AI would enter this state if it sees that the ball is going to score an own-goal. The robot would then attempt to sweep the ball out of danger by doing a curve around the ball.

3.5.5 Repositioning State:

Players are positioned so that they are spread out in the field. Some players are assigned as defenders while some as strikers (just as in real-life soccer). If a player is too far in the opponents side of the field, and it doesnt have a chance to score a goal or some other player has a higher chance of scoring, AI will switch to Repositioning State and the robot will return to its original position.

3.5.6 Penalty State:

The Penalty State is for the robot taking a penalty shot. The AI does not automatically switch into this state at any time, but is triggered to go into this state by the host PC which would send out a "penalty state" message to the corresponding robot taking the penalty shot.

4 Conclusion

The project was completed in time for the competition. Alas, there were only 2 teams competing in the middle-size league and we did not have much opportunity to test our team thorough. One of the main constrains doing this project is the unavailability of space that can accommodate the playing field. Most of the time, we had only space to setup a goal area and test one or two robots at a time.

We hope to improve our team further, especially in the area of self-locations. We believed that this is the key to fast, robust and accurate play.