Multi-platform Soccer Robot Development System

Hui Wang, Han Wang, Chunmiao Wang, and William Y.C. Soh

Division of Control & Instrumentation, School of EEE
Nanyang Technological University
Nanyang Avenue, Singapore 639798
{p147513815, hw, p149510969, eycsoh}@ntu.edu.sg

Abstract. Robot soccer is a challenging research domain, which involves multiple agents (physical robots or "softbots") to work together in a dynamic, noisy, cooperative and adversarial environment to achieve specific objectives. The paper focuses on the design and implementation of an effective system for developing a small-size physical soccer team for the competition of RoboCup. There are three goals preset in our research, that is, robust individual behaviors, controllable cooperation and behavioral learning. The system is therefore required to collect data in real-time situations and repeat experiments accurately, which are found difficult to be achieved in pure physical or simulated case. A distinct feature of our system is that it is a close combination of three functional parts: physical test platform, simulation platform and measurement platform, which support each other to overcome many defects inherent in traditional physical system or simulation system.

1 Introduction

In recent years, robot soccer has become a popular research topic for robotics and artificial intelligence. In order for a robot team to perform a soccer game, various technologies should be incorporated, such as, robotics, multi-agent collaboration, real-time image processing, sensor fusion, and machine learning [1]. Our research focus is to develop an autonomous physical team to explore the cooperative behaviors in robot soccer.

The small-size robot soccer game contains two teams, each of which has five players. Our team is composed of three functional parts: vision module, intelligent control module and real robots. It is designed according to a well-defined processing cycle : Vision module receives the raw image sequence from a camera hung over the playing field and processes it, giving out the motion states of the ball and all robots. The result is sent to a control module, which evaluates the world state and decides what to be done in the next step based on its specific strategy. Robots' actions are motion commands (encoded in a packet) broadcasted by the control module through wireless communication. Each robot decodes its own part from the command packet based on a unique ID code and then executes corresponding actions.

In this paper, we first explain why a multi-platform system should be adopted in developing a small-size robot team and then discuss how to solve a critical issue in such a system. Details of implementation of the multi-platform system are also presented to embody some principles. Finally, experiences and results in our work are summarized and future work is presented.

2 Obstacles in Soccer Robot Team Development

Since there is still not a formal way to design and implement a cooperative small-size soccer robot team, our work heavily depends on experiments. There are three objectives in our development. Firstly, single robot is able to perform individual behaviors in a robust way, that is, it can repeat certain actions in specific situations; the second is to investigate what mechanisms can produce a controllable rather than "emergent" cooperation in a robot team, that is, in an ideal case, each robot can work together intentionally to help its teammates; finally, to study how to apply machine learning to enhancing the performance of robots including their individual behaviors and team behaviors. In order to obtain these goals, our system is at least required to have the following abilities:

– Collect data and/or perform analysis in real-time situation.
– Repeat experiments in controlled mode, such as in the same system specifications.

The first requirement is because soccer is a time-critical game. A playing strategy should deal with dynamic rather than static situations. The second requirement is because we intend to develop a collection of robots that are able to act like a team, which means the robots can present cooperative behaviors in typical situations. Such cooperation not only includes those so-called "emergent" cooperation [2][3], but also "controllable" cooperation. The key difference between them is that emergent cooperation can be achieved by adding simple rules in robot's individual behaviors, which are useful to avoid some explicit conflicts, such as a robot vies with teammates to deal with the ball, but not sufficient to guarantee that a player can help its teammates to complete a task intentionally. However, such intentional cooperative behaviors are fairly helpful in soccer games [4].

In our development, we have found that if we rely on a pure physical platform, it is hard to meet the above two requirements. Firstly, our system has to run in a real-time case. Any extra on-line data collection and analysis task may affect the system performance, which will lead to incorrect experimental results. Secondly, many unexpected factors limit the accuracy of system specifications. It is sometimes a tough task to suppress such random influences.

Therefore, a concept of "multi-platform system" is introduced, which includes several functional platforms that undertake well-defined sub-tasks in development. In our case, a simulation platform and an assistant measurement platform are added to overcome the above obstacles in a pure physical system. Different from traditional ways, our simulation platform doesn't work only for testing some ideas. It will serve to develop effective algorithms applied in physical environment. It enables us to account for specific statistics in real-time situation, which are extremely useful in comparing the feasibility and performance of each strategy. Another advantage of simulation is that it is easy for us to control the experiment conditions, that is, in each trial, we can only focus on one or several factors.

3 Challenge in a Multi-platform Development System

Since there are more than one experiment platform in our system to develop physical robots, a subsequent critical problem is "How to keep coherence between them?", that is, a question must be answered: "If an algorithm is developed in simulation platform, can it be used in physical environment without any radical change? If not, what should be noticed?". In fact, some scholars once developed simulation platform to study the intelligent robots and try to put the results into practice. However, there was few report about the successful application based on those simulation results ([6] and [7] may be two exceptions.). We believe that there are two main reasons: 1) Overly specific assumptions are applied in simulation, so that the results are hardly extended to general cases and 2) There is no clear work decomposition and rigorous development process for those platforms. In our system, the simulation platform is not an independent part. It is designed to facilitate the physical test platform to develop robots' high-level strategy. The point is each platform only undertakes the work, at which they are good. In addition, their interactions are rigidly defined (refer to Fig. 1):

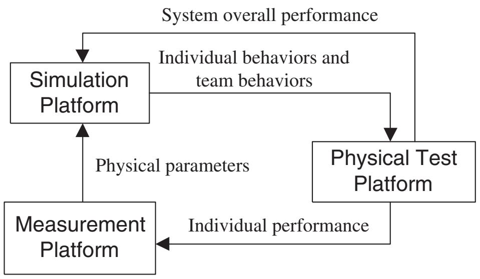


Fig. 1. Interactions among three functional modules in our multi-platform development system.

In addition, there are three methods taken to keep the coherence between simulated and physical experiments: First, the kinetic model of robot in the simulation platform is the same as that in the physical platform. All related motion parameters, such as max/min speed and max acceleration, are kept the same. They are measured on measurement platform. Second, the same system architecture (refer to Fig. 2) is adopted in both platforms. It provides convenience for simulation program to test the performance of each component in the physical system by altering some parameters, so that we can determine which factor of what component how to affect the system's overall performance. Finally, algorithms and software frame in high-level strategy are the same in both platforms. It makes sure that the computation complexity is identical and thus the structure of decision process developed in the simulation platform can be easily transferred to the physical environment.

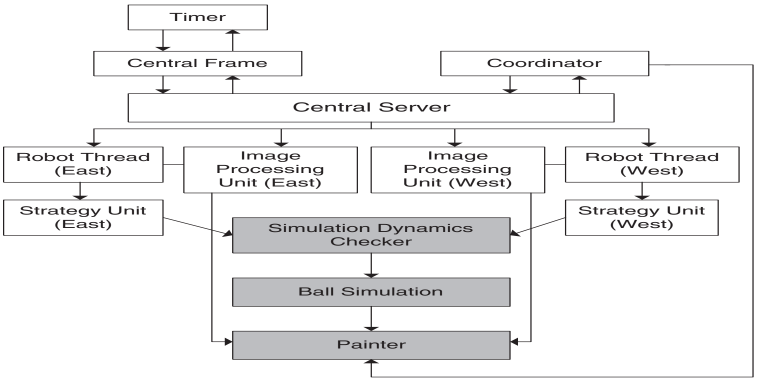


Fig. 2. Structure of simulation platform.

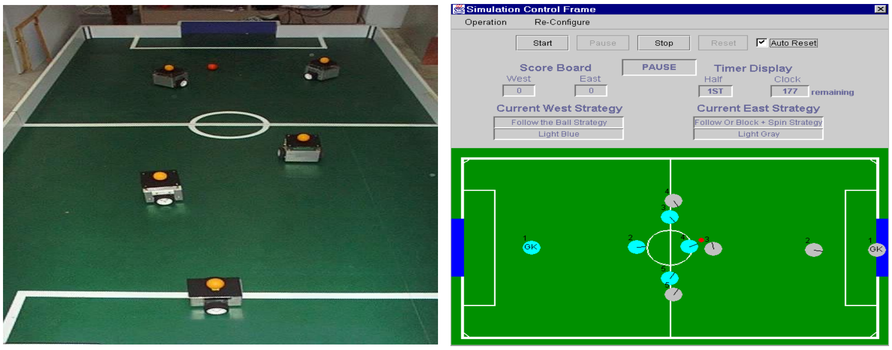
4 Implementation of a Multi-platform Development System

Fig. 3(a) and 3(b) are the photos of physical test platform and simulation platform. In this section, we will present related details of implementation in the view of function rather than entity. For both physical and simulated cases, our system adopts a uniform modeling as follows to keep coherence among multiple development platforms:

– A single robot is represented as a set of physical parameters: size, max. speed, max. acceleration and a table mapping wheel speeds to motion commands, all of which are determined by measurement platform and applied in simulation.
– Moving objects (robots and the ball) in soccer game are represented as a motion state set and behavior set, where motion state is a vector [x, y, dx, dy, theta, ID], that is, position, velocity, orientation and identification; behavior is a situation-action pair (a situation is a designer-determined motion state subset; action is only for robot, which has four basic types, that is, forward/backward linear move with quick start, immediate stop, turning in a fixed angle and spin).

System software frame are summarized in Fig 2. It is implemented based on a typical multi-thread client-server model: (1) Central Server waits for the inputs from user through its interface - Central Frame and will inform each of five clients, that is, two Robot Thread, two Image Processing Unit and Painter. (2) Coordinator is basically responsible for monitoring the game state, such as goal.

Its additional special function is to operate methods in Central Server so as not to hang on when calling objects for long time. Timer thread keeps track of time and informs Central Frame when time is up or after a second has passed. (3) Image Processing Unit processes data captured from the soccer field (i.e. the Painter). It calculates motion state of each robot (including the opponents) and ball. The Robot Thread from the same team will then pick up this processed data. (4) Robot Thread contains the number of robots requested by the user. Whenever the user activates the game, it will first try to obtain information from the Image Unit. If new data are available, it will then proceed to prompt each Strategy Unit to use these data for strategy planning. The data returned by the robots will then be kept in a vault waiting to be picked up by the Painter. (5) Simulation Dynamics Checker checks the boundary, collision and calculates the next robot step according to the speed and desired direction the strategy unit has set. Ball Simulation calculates the ball position in the next step. (6) Painter thread provides the animation and it holds on to the Simulation Dynamics Checker and Ball Simulation. Therefore, as soon as it receives the next step data from Robot Thread, the Simulation Dynamics Checker and Ball Simulation will digest these data before showing the animation.



[✓] Auto Reset

Score Board
West 0
East 0

Timer Display
Half 1ST
Clock 177 remaining

Current West Strategy
Follow the Ball Strategy
Light Blue

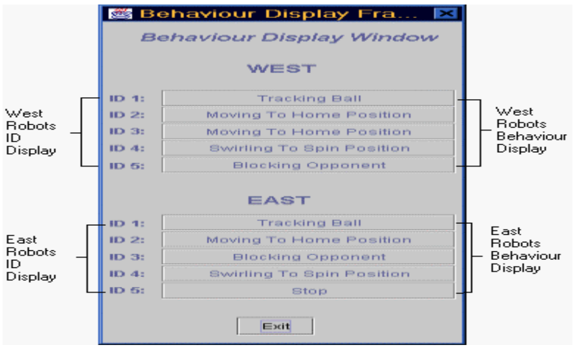
Current East Strategy
Follow Or Block + Spin Strategy
Light Gray

(a) Physical test platform (b) Simulation platform

Fig. 3. Development platform for soccer robot team

The shadow parts in Fig.2 are absent in physical platform, that is, Painter, Simulation Dynamics Checker and Ball Simulation only serve for simulation. The animation in simulation is embodied as robots' motion in physical environment. To develop a simulation platform is for behavior design and selection, we pay special attention to many special features:

- Convenient to add noise and evaluation of the performance of various filters used in image processing unit of physical system.
- Many choices in extra actions, robot's speed and response time.
- Easy to develop strategy.
- Collect and show data of robots in real-time case (refer to Fig. 4).



WEST

ID 1: Tracking Ball
ID 2: Moving To Home Position
ID 3: Moving To Home Position
ID 4: Swirling To Spin Position
ID 5: Blocking Opponent

EAST

ID 1: Tracking Ball
ID 2: Moving To Home Position
ID 3: Blocking Opponent
ID 4: Swirling To Spin Position
ID 5: Stop

Fig. 4. Robot behavior display frame

5 Conclusions

This paper describes a design and implementation of a multi-platform system for building a robot soccer team. The development task is decomposed into three interrelated platforms, each of which complete an objective and provide help for the others. Their rigidly defined interactions and additional requirements in internal structure guarantee the effectiveness of this system. At present, partial results are obtained in individual behavior design, and perception (image processing in physical system). However, more efforts are still needed to investigate the mechanism in team cooperation.

References

1. Kitano, H., Asada, M., Kuniyoshi, Y., Noda, I., Osawa, E., and Matsubara, H, RoboCup: A Challenge Problem for AI, AI Magazine , 18(1): pp73-85, 1997.
2. Ronald C. Arkin and Tucker Balch, Cooperative Multiagent Robotic System, In David Kortenkamp, R.P. Bonasso, and R. Murphy, editors, Artificial Intelligence and Mobil Robots, Cambridge, MA, 1998. MIT/AAAI Press.
3. B. Werger, The spirit of bolivia: Complex behavior through minimal control, In Proceedings of RoboCup 97, IJCAI 97, 1997.
4. M. Veloso, P. Stone, and M. Bowling, Anticipation as a key for collaboration in a team of agents: A case study in robotic soccer, In Proceedings of SPIE Sensor Fusion and Decentralized Control in Robotic Systems II, volume 3839, Boston, September 1999.
5. P. Stone and M. Veloso, Using decision tree confidence factors for multiagent control, In H. Kitano, ed, RoboCup-97: The First Robot World Cup Soccer Games and Conferences. Springer Verlag, Berlin, 1998.
6. P. Stone and M. Veloso, Task decomposition, dynamic role assignment, and low-bandwidth communication for real-time strategic teamwork, Artificial Intelligence 110(2), pp241-273, June 1999.
7. Raffaello D'Andrea, Jin-Woo Lee, Andrew Ho man, Aris Samad-Yahaja, Lars Creman, and Thomas Karpati, Big red: The cornell small league robot soccer team, In M. Veloso, E. Pagello, and H. Kitano, editors, RoboCup-99: Robot Soccer World Cup III. Springer Verlag, Berlin, pp49-60 2000.