Evolving RoboCup2D Agents Based on PSO

Liu Zhao
College of Computer Science and Technology
Wuhan University of Science and Technology
Wuhan, China
liuzhao@wust.edu.cn

YE Shaoxiong
College of Computer Science and Technology
Wuhan University of Science and Technology
Wuhan, China
625517303@qq.com

Dong Chunchen
College of Computer Science and Technology
Wuhan University of Science and Technology
Wuhan, China
2256397010@qq.com

Yang Yang
College of Computer Science and Technology
Wuhan University of Science and Technology
Wuhan, China
youngyang_li@foxmail.com

Abstract—In order to build intelligent robots to accomplish soccer game tasks, this paper introduces evolutionary computing in agent architecture for perception, planning, and action: (1) an architecture based on PSO is proposed, which made up of 4 levels: atomic action, combo action, behavior and policy. (2) by offline training, agents format perception rules and relevant parameters, to optimize perception method for the position, orientation and other information; (3) according to the granularity, functions, and parameters manually specified, PSO builds a set of combo actions, which described by atomic actions, parameters and execution results; (4) according to game environment and a few task rules, PSO searches for task, behavior, and combo actions, as a whole, to accomplish the game tasks. The simulation experiments on RoboCup2D platform show that, agent based on PSO is a robust and flexible robot control method: given evaluation methods and implementation frames, it is able to learn rapidly in real environment, and displays planning behavior without the use of classical planning techniques.

Keywords-RoboCup2D; agent; architecture; planning; particle swarm optimization

I. INTRODUCTION

RoboCup and other robot soccer games provide challenging platforms for robotics and AI research[1]. Among them, the RoboCup2D, a platform enables for two teams of 11 simulated autonomous robotic players to play soccer, is very suitable to research: a) agent architecture, b) machine learning, especially agent cooperation and confrontation, and c) real-time agent planning[2].

According to perception, planning, action and their relationships, agent architectures are divided into: a) hierarchical structure, b) reactive structure, and c) hybrid hierarchical/reactive structure. For soccer robot or most other agent applications, the hybrid architecture is more successfully than hierarchical or reactive structures. As a typical instance of hybrid architecture, a triple-tower architecture and teleo-reactive programs[3] have been proposed in 2001 by Nils Nilsson as a robust agent control structure. After that, they have been successfully applied to a variety of domains. As shown in Fig.1, triple-towers means: a) perception tower, b) model tower and c) action tower. Main features of the architecture shown below:

(1) Teleo-reactive programs were applied in the action tower. It is a mixture of top-down hierarchical control structure and agent-based bottom-up approach, using a tree data structure called the TR. The agent is directed towards a goal based on continuous evaluation of perceptual inputs. During each iteration cycle, searching start from the root task, value the condition of each task in turn until it finds the first established the tasks node, and then immediately executed action or action sequence associated with it.

(2) Perception rules applied to perception tower. They create increasingly abstract descriptions of the current environmental situation starting with the primitive predicates produced by the robot's sensory apparatus

(3) Truth maintenance system, or TMS, applied to model tower, which continuously kept faithful to the current environmental situation.

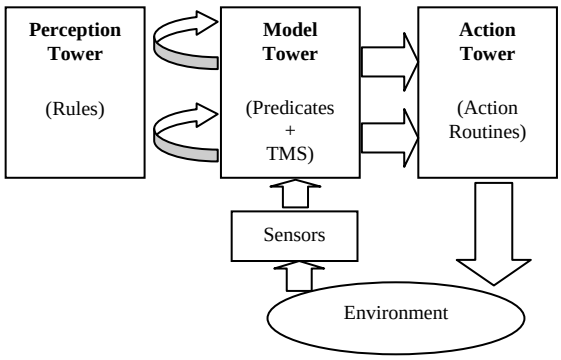

Figure 1. Nilsson's triple-tower architecture

The designs of rules and actions in the architecture are the key and the hardest work in such applications. Besides Nilsson, many researchers [4,5] have researched the automation of them. In 2003, genetic programming was applied in the architecture for block stacking robots by Kochenderfer[6]. As an evolutionary computing method, genetic programming is proposed to generate teleo-reactive rules and actions of robots. Evolving teleo-

reactive programs not only improve the adaptability of the system, but also proved to be a new approach to gain knowledge of block stacking actions.

Similar to Nilsson's triple-tower architecture, the most popular approach to building RoboCup2D agents is hybrid hierarchical/reactive architecture, shown as Fig. 2.

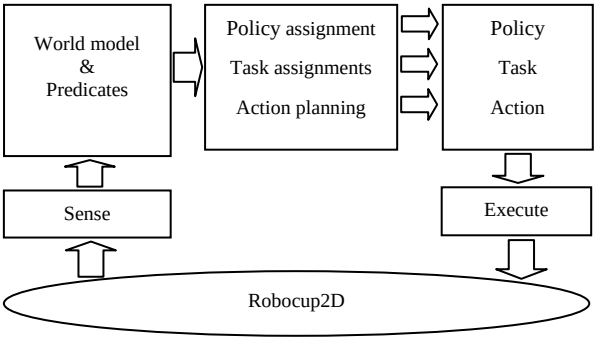

Figure 2. A traditional soccer agent architecture

Policy layer is the highest layer among decision modules. For global team interests, it determines the behavior of the agent and the general tactics. Task layer is to develop specific tasks, including agent moving, ball catching, ball intercept, ball dribbling, ball passing, ball shotting, ball bashing, and so on. Action planning is arrangement basic actions which is acceptable for platform to execute.

There are some disadvantages in the traditional soccer agent architecture. Firstly, knowledge and preconditions about actions are very difficult to acquire. Althought some methods, such as grammatical inference[], knowledge mapping, Q-learning, and reinforcement learning methods have been used, flexible and effective knowledge and veracious preconditions often fail to meet the requirements. Secondly, the architecture could not timely response to dynamic environment, and it is rigid when the top policy changed from one to another. Most of scenarios in the games are arranged by the programmer. Thirdly, and the most important is that, it didn't take account of the interactive relation between the layers. Infact, it is not reasonable make the top policy without calculating of advance action layer and basic action layer, especially when the related information is deficient.

II. ARCHITECTURE

To get enough action knowledge which is essential for the agent decision making, agent actions are broken down into 4 levels: atomic action, combo action, behavior and policy. Soccer robot architecture based on PSO made up of two parts: offline learning architecture, and online strategy architecture. PSO is used to perception training, action training, and decision making.

A. *Offline Learning Architecture*

Perception training and action training are conducted under offline condition. By sense and behavior in a specified environment, PSO can get perception rules, perception filter, and sense parameter. Offline learning architecture is shown as Fig.3.

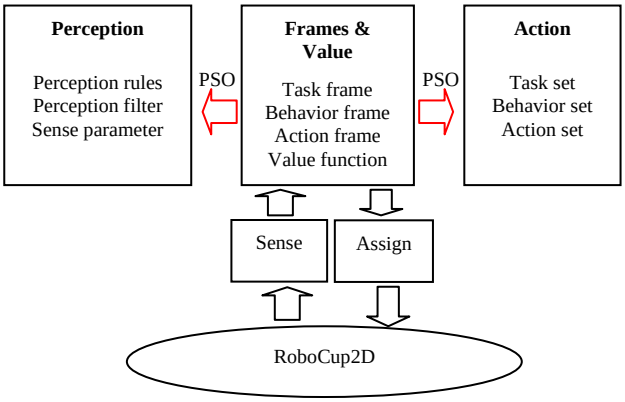

Figure 3. Offline learning architecture

B. *Online Strategy Architecture*

Decision-making can be seen as a multivariable optimization process. Multivariable means a combination of task, behavior, and action, which determine the correctness and efficiency of action results. Although the component of the solution, include task, behavior and combo action, is hierarchical and related to each other, but PSO can process them according to the final action results.

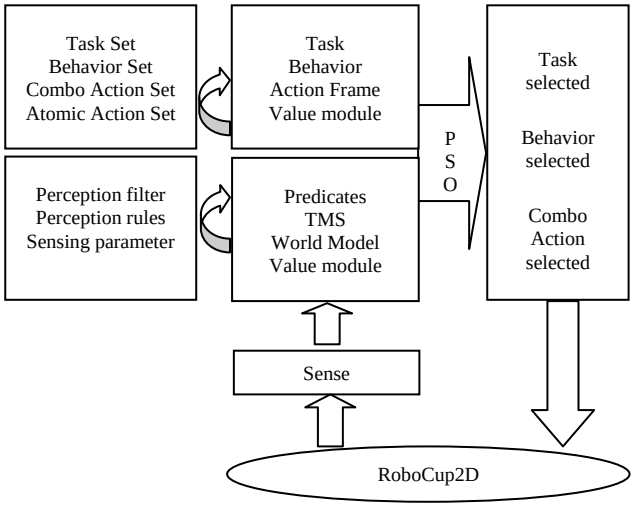

Figure 4. Online strategy architecture

III. COMBO ACTION LEARNING BASED ON PSO

Combo action is designed to get general action skills through training. Combo action learning is to get action sequence, action precondition, action duration, and action results.

A. *Atomic Action Set*

In RoboCup2D, agent has some basic actions, such as dash, turn, kick, tackle, catch, move, turn_neck, change_view, say, point to, attention to. When asigned parameters, those actions could be output and executed. They are simplified and formulized, form an atomic action set, as Tab.1 show.

TABLE I. ATOMIC ACTION SET

Action	Action Parameters	
	1st	*2nd*
Dash	Power range (-100,100)	null
Kick	Power range (0,100)	Direction range (-100,100)
Tackle	Power range (-100,100)	null
Turn	Angle range (-180,180)	null

B. Combo Action Frame

Frame design for PSO is very important. Frames should not only be easy to apply existing knowledge, but also must be prepared to form new knowledge.

Based on atomic action set, seven combo action frames are designed according to soccer game tasks and agent environment:

(1) Hard_kick.
(2) Dribble_ball.
(3) Dribble_pass.
(4) Location_arrive.
(5) Location_rushing.
(6) Block_robot.
(7) Block_tackle.

Frame No.:1-7
Granularity: 20 loop periods
Precontion:
    Ball velocity: (dx,dy)
    Agent location:(x,y)
    Agent stamina:(s)
Desire:
    Ball velocity: (dx,dy)
    Agent location:(x,y)
    Agent stamina:(s)
Action plan:
    Period gap:(t_gap)
    Act list:(act[t_gap])
    Act paramiter:c1[t_gap],c2[t_gap]
Action value:
    Effect value:value_effect
    Time value:value_time
    Comprehensive:value_all
Action modify:
    New period gap:(t_gap)

Figure 5. Combo Action Frame

C. Evolving Combo Actions Based on PSO

When combo actions were specified granularity, functions, and parameters by combo action frames, PSO is used to optimize action sequence and results. As Fig.6 shown, combo actions were arranged to be trained one by one.

Some outstanding RoboCup2D agents, such as Brainstorm, WrightEagle, Gliders, Oxsy and so on, were used as sparring agents. The training takes the form of 2vs2, which means a goalkeeper and a general player per team to completion.

After the 7 action frameworks were designed, for each combo action, initialize n frameworks as a population of evolutionary computation. Execute every combo action, and evaluate performance and efficiency. Then, some bad frameworks are phased out, and nice frameworks can be crossover and mutated. New frameworks form a new population, a new loop repeated if the combo action result is not satisfactory.

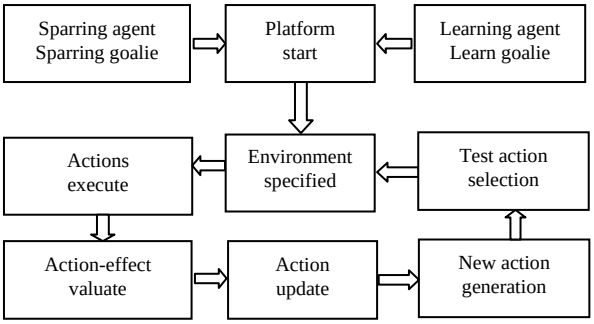

Figure 6. Evolving Combo Actions Based on PSO

IV. ONLINE STRATEGY BASED ON PSO

We integrate planning layers with each other in the particles, which means, different planning layers be valued as a whole, but be initialized and evolved respectively. During valuating, some knowledge about the relationship between layers is taken into account. Fig.7. shows soccer robot strategy architecture based on PSO.

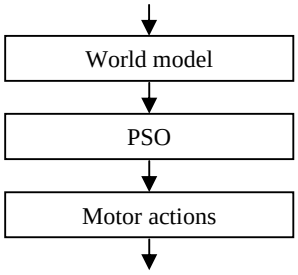

Figure 7. Soccer robot strategy based on PSO

TABLE II. PARTICLE STRUCTURE DESIGN

Particle Structure
……
applied layer no. list
particle fitness
……
top policy layer
……
behaviors layer structure_1
behaviors layer structure_2
……
basic actions layer structure_1
basic actions layer structure_2
……

In the soccer robot strategy, we found that, without existing knowledge, it is not feasible to search solutions in real time. So we decompose the main goal into some tasks and apply related knowledge in the strategy. To make the

swarm particles chase after different tasks at different situation, the structure of it is designed as Tab.2 shows.

With every planning layer structure designed as Tab.3 shows:

TABLE III. PLANNING LAYER STRUCTURE DESIGN

Planning Layer Structure
……
parameters of the structure
velocity vector of the structure
position vector of the structure
……

In the PSO, particles presented different solutions to different tasks. The structures be used, value-function, initialized and evolved methods are varied according to the situation of the game.

V. SIMULATION STUDY AND CONCLUSION

We implemented the soccer robot strategy based on PSO on RoboCup2D platform. The simulation experiments show that, it is a robust and flexible soccer robot control method.

Statistics of instantaneous working particles kinds show as Tab.4. Results show that, there are always more than one tasks being prepared simultaneously, but only the best one be selected and executed. The turning from one task to another is smooth and efficient.

TABLE IV. WORKING PARTICLES FREQUENCY

Table Column Head		
Particles	***Tasks***	***Frequency***
1	Hard_kick	23
2	Dribble_ball	35
3	Dribble_pass	46
4	Location_arrive	44
5	Location_rushing	36
6	Block_robot.	11
7	Block_tackle	15

ACKNOWLEDGMENT

This work was supported by the National Natural Science Foundation of China (Grant No. 51174151).

REFERENCES

[1] Asada, M., Kitano, H., Noda, I., Veloso, M. “Robocup: Today and tomorrow-what we have learned,” Artificial Intelligence,vol.110, 1999, pp.193-214.

[2] Stone, P., & McAllester, D. An architecture for action selection in robotic soccer. In Proceedings of the Fifth International Conference on Autonomous Agents, 2001, pp. 316–323.

[3] Nilsson, N., “Teleo-Reactive Programs and the Triple-Tower Architecture,” Electronic Transactions on Artificial Intelligence, vol. 5 , 2001, Section B, pp. 99-110.

[4] Fu J., Tanner H.G., Heinz J.N., Karydis K., Chandlee J., Koirala C. “Symbolic planning and control using game theory and grammatical inference,” Engineering Applications of Artificial Intelligence, Volume 37, 1 January 2015, Pages 378-391.

[5] Quintero C., Rodríguez S., Pérez K., López J., Rojas E., Calderón J. “Learning soccer drills for the Small Size League of RoboCup,” Lecture Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Science), vol. 8992, 2015, Pages 395-406.

[6] Kochenderfer, M., “Evolving Hierarchical and Recursive Teleo-reactive Programs through Genetic Programming,” Lecture Notes in Computer Science, vol.2610, 2003, pp. 83-92.

[7] Mortazavi A., Arshadi Khamseh A., Azimi P. “Designing of an intelligent self-adaptive model for supply chain ordering management system,” Engineering Applications of Artificial Intelligence, Volume 37, 1 January 2015, Pages 207-220.

[8] K. L. Clark and P. J. Robinson, Programming Robotic Agents: A Teleo-Reactive Multi-Tasking Approach, to appear, 2015, Springer.

[9] Broda K., Hogger, CJ., “Designing Effective Policies for Minimal Agents,” The Computer Journal 53: 1184-1209, 2010.

[10] Dongol B., Hayes I.J.,Robinson P.J., “Reasoning about goal-directed real-time teleo-reactive programs,” Formal Aspects of Computing, 2013, pp. 1-27.

[11] Hawthorne J., Anthony R. “Using a Teleo-Reactive approach in building self-managing systems,” International Journal of Autonomous and Adaptive Communications Systems, 5 (3), pp. 255-273, 2012:

[12] Soto F., Sánchez P., Mateo A., Alonso D., Navarro P.J., “An Educational Tool for Implementing Reactive Systems Following a Goal-Driven Approach,” Computer Applications for Engineering Education, DOI: 10.1002/cae.21568, 2012.

[13] Hawthorne J., Anthony R., Petridis M., “Improving the Development Process for Teleo-Reactive Programming Through Advanced Composition,” In: Proceedings of ICAS-11, 2011.