Team Erika

Takeshi Matsumura

Department of Information and Computer Science, Waseda University

1 Introduction

Team Erika's main focus is on the facilatation of the design of agent behavior. The behavior code is generated by a graph editor which process transition diagram like graph. Since the concept is represented visually, high design efficency can be achieved. Besides, people other than computer scientist can design the behavior easily without understanding the underlying stucture.

Future work is to improve the graph editor to process graph which represents a cooperation among a few agents in single graph.

2 Team Development

Team Leader and only a member: Takeshi Matsumura

- affiliation: Waseda University
- country: Japan
- position: graduate student
- did or did not attend the competition: attended

Web page `http://www.futamura.info.waseda.ac.jp/~matsu/erika`

3 World Model and Communication

World model which is situated in the lower layer of the two layer structure agent was created by the sensor. The internal functions inside the agent invoked every 100ms update the world model by using the information received from the server. Agent processes all of the visual information and update the proper object inside the world model by examining the time stamp. Newly created object always supersedes the old one which is found by comparing the time stamps of each objects. (e.g. If agent had a ball object with time 14 and he got a ball information when time 16, then the time 14 ball object information is thrown away.) An agent stop its current computation immediately, reset the states and restart decision making when it receives a referee message in order to increase reactivity.

Agent used in Stockholm had no communication. Communication is being installed in current work to synchronize the computation of agents when a cooperation, like one-two pass or attack from corner kick, starts.

M. Veloso, E. Pagello, and H. Kitano (Eds.): RoboCup-99, LNAI 1856, pp. 554–557, 2000.

4 Skills

Agent has the ball prediction routine which use the world model update routine to predict the ball's future postion. The prediction is not very accurate because only the last information of the ball was being used. In future work, history of the ball position and the sequence of the ball movement will also be used in order to improve accuracy.

5 Strategy

All agents always try to monitor the ball's action even when the agent is far away from the ball because important events usually occur around the ball but basically only the agent closest to the ball goes to get it. The only exception is that when the agent is returning to his home position. In this case he will monitor the ball every 5 seconds.

6 Special Team Features

6.1 Graph editor for one player

An transition diagram like graph shown in fig.1 is used to represent each agent's play style. Each agent's computation moves from node to node, depends on the conditions described on each arcs.

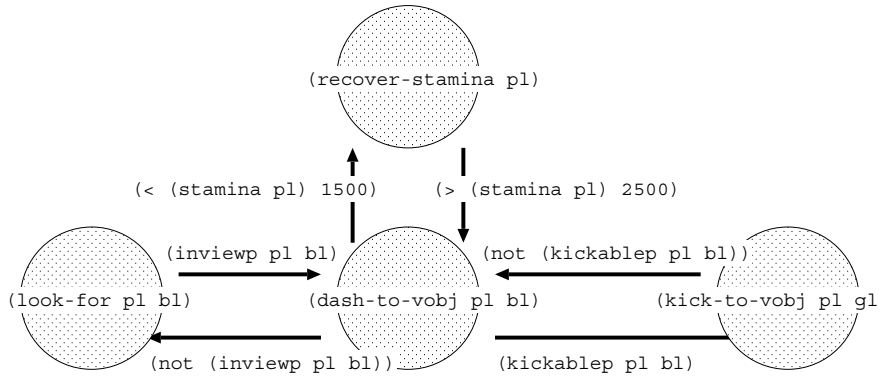

Fig. 1. graph styled flow-chat represents dango-play

Agent's computation begins at the leftmost node, for instance, node 1. At first the agent does the action "(look-for pl bl)", where the variables pl and bl are bounded to the agent himself and a ball object respectively. Then he checks the condition "(inviewp pl bl)" on the arc derived from node1. If the

condition "(inviewp pl bl)" is true, which means that ball come into his view, then his computation will be moved to the center node. Otherwise, he takes the "(look-for pl bl)" action and attempt to evaluate the condition again. If the computation is at the node which has no arcs, then the action on it is taken and the computation stops. In the example, however, the computation continues permanently because there are no nodes without any arcs if there is no referee message.

After moved to another node, the action which is described on the node is absolutely taken before evaluation of conditions.

We us a graph editor to create this kind of graph and use it to generate lisp lists which are embedded inside an agent. The interpreter inside each agent's upper layer analyse the diagram and then the agent behavior is set.

6.2 Graph Editor for cooperation

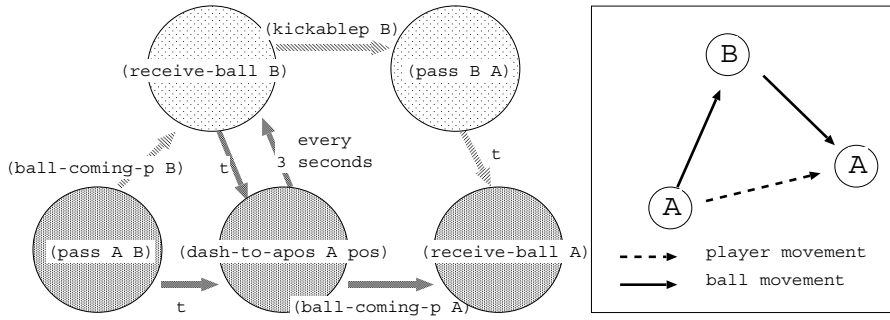

Fig. 2. one-two pass cooperation graph between two agents

Cooperation between agents can also be represented by a transition diagram like graph. In Fig.2, the left graph shows an example of the cooperation between two agents acting a one-two pass as shown in the right figure.

There are two roles. Role A who kicks a ball to role B and receives it later at another position; role B who receives the pass from role A and kicks it back towards A immediately.

The computation shown in Fig.2 will is similar to Fig.1. The most major difference is that each node and arc has a a color which is assigned to each role. We assume that an agent's computation is on a node N colored c_n. If the agent is playing the role r_n, which is assigned to the color c_n, then he does the action described on N. Otherwise, he observes another agent playing the role r_n who are assigned the color c_n instead of acting it himself.

An agent evaluates the conditions on the arcs colored the same color as the agent. If there are no such an arc, he stops his computation.

For instance, the graph of Fig.2 has two colors black and gray which are assigned to the role A and B respectively. Assumed that both A and B's computations are on the left most node.

A's computation sequence will be as follows:

1. act "(pass A B)" on the black node (node1)
2. evaluate "t" (it means true in lisp) and move to the center node (node2).
3. act "(dash-to-apos A pos)", where variable pos is a proper position.
4. if ball comes towards A, go to 6.
5. computation moves to upper left gray node (node3) every 3 seconds. Because his color is not gray, he does not act "(receive-ball B)" but observe the other agent who plays role B. A's computation returns to node2 immediately because the evaluation "t" on the black arc is derived from the node3.
6. he receives the ball and stops computation.

B's computation goes in the same manner. One-two pass cooperation will go well because of the observations of other players.

7 Conclusion and future work

With the introduction of the graph editor, behavior design of the agent is facilitated. Future work will include the improvement of graph editor to generate eager execution code for the evaluation of conditions in order to shorten the response time.