A Rule-Driven Autonomous Robotic System Operating in a Time-Varying Environment

Jia Jianqiang, Chen Weidong, and Xi Yugeng

Institute of Automation, Shanghai Jiaotong University
200030 Shanghai, P.R. China
mr_jia@sina.com, {wdchen,ygxi}@sjtu.edu.cn

Abstract. In this paper, the problem concerning how to coordinate concurrent behaviors, when controlling autonomous mobile robots (AMRs), is investigated. We adopt a FSM (finite state machine)-based behavior selection method to solve this problem. It is shown how a hybrid system for an AMR can be modeled as an automaton, where each node corresponds to a distinct robot state. Through transitions between states, robot can coordinate multiple behaviors easily and rapidly under dynamic environment. As an illustration, a soccer task was finished by an AMR system with this method. The robot performed well in the soccer games and won the game in the end.

1 Introduction

For an autonomous mobile robot, the ability to function in, and interact with a dynamic, changing environment is of key importance [1, 2]. A successful way of structuring the control system in order to deal with this problem is within hybrid architecture [3, 4]. This way of structuring the control system has the major advantage that it makes the system own planning ability, and at the same time, the system can react rapidly.

However, within this framework, a number of design issues need to be addressed. An important issue is how to deal with multi-behavior coordination problem. For instance, given a reactive obstacle-avoidance behavior, how should an approach–target behavior be designed and combined with it? There are two main methods to manage this problem: one is arbitration [5, 6, 7]. That is, select one behavior between several ones based on priority or through competition. This kind of method has definitude meaning and new behaviors can be added easily. But it has the major disadvantage that it both affects the performance in a negative way, not allowing for the smooth performance that concurrently active behaviors produce, and that it increases the risk of introducing chattering into the system [8]. Another method is to work with concurrently active behaviors. Different controllers affect the system simultaneously, resulting in a smooth overall performance [9, 10, 11]. However, in this case, the analysis of the system is becoming difficult as new behaviors are added. Egerstedt proposes regularization techniques to improve the first method. But he pays more attention to smoothness between behavior transitions and ignores rapidness of the system. For tasks under dynamic environment, simple, rapid, and efficient control method is necessary.

D. Polani et al. (Eds.): RoboCup 2003, LNAI 3020, pp. 487–494, 2004.

Now, autonomous soccer robot is a hot topic in AI and robot fields [12, 13]. The game has been a standard platform for theory study under dynamic, uncertain environment. In this paper, we first introduce an AMR system applicable under dynamic environment. Then, an FSM-based behavior selection method is adopted and used in the robot soccer game successfully. With this method, the robot can coordinate multiple behaviors easily and react rapidly under dynamic environment. The result of 1st CRC games (the First Chinese Robot Competition) shows the validity of the method.

2 System Description

The JiaoLong robots are constructed as part of a project to build inexpensive, autonomous robots for the study of multi-robot systems. We made three robots of the same type and one goalkeeper robot as shown in Fig.1. The hardware system consists of motion platform, sensing system, communication system and control system.

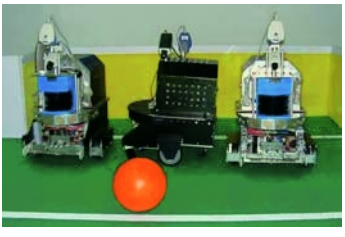

Fig. 1. JiaoLong soccer robot

The motion platform is a 45×45×60cm, differential driven car. Each robot employs a LRF (laser range finder) and two cameras as its main sensors. The whole group utilizes a wireless LAN device for communication. Every robot acts as a node in the LAN and has a wireless card that can transfer data at the speed of 11Mbps.

For software design, we propose a distributed architecture based on priority. There are one main process and three assistant processes sharing one CPU. Main process makes decision and three assistant processes dispose sensor information of LRF and two cameras. Information communication between main process and assistant process is realized through IPC (inter-process communication) mechanism. Also, this mechanism can ensure the communication between robots.

3 System Architecture and Behavior Design

3.1 Hybrid Architecture for Dynamic Environment

Robot architecture can be divided into three kinds: deliberative, reactive and hybrid architecture [3]. The first one always need precise information and is not suitable for dynamic environment. The second one makes the system react quickly and robustly, but it is not suitable for complex task. The hybrid one is suitable for dynamic environment and the system can react rapidly. Our architecture is a hybrid one, but differs from others in two aspects (Fig. 2). First, we add short-time memory in behavior-based control. Secondly, our system only needs local path-planning. The whole task will be divided into several subtasks. Robot schedules tasks according to his perception. In every time cycle, there is only one subtask executed, which generates one behavior. More details will be introduced in 3.2.

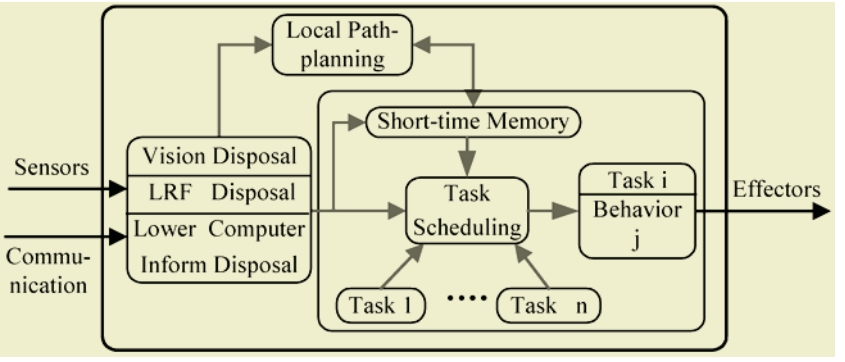

Fig. 2. Planning and control diagram

The reactive architecture characterizes having no memory and reacting rapidly. The long-time memory will ask for better hardware and lead to wrong decision due to the uncertain data. We add short-time memory in the behavior-based control according to the task. Tab. 1 shows an example of the memory content. Fig. 3 shows the meaning of the angle θ. In Fig. 3, X-axis represents robot's heading. θ means the angle between ball centre and Y-axis, and is determined by camera visual angle.

Whenever the robot sees ball, he will update the value of SIGN in Tab 1. If the ball disappears suddenly, robot could find it rapidly according to SIGN. For example, if the ball disappears at time of T, and the value of SIGN at time T-1 is positive, then the robot turns to left. Thus the ball can be found in the shortest time. If ball disappears for a long time (longer than 20 seconds), the SIGN will be set to zero until ball appears again.

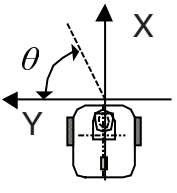

Fig. 3. Short-time memory

Table 1. Short-time memory

θ / SIGN	60≤θ≤90	90<θ≤120
Ball	positive (ball on left)	negative (ball on right)
Note: Lose ball for a long time, the SIGN will be zero.		

The global path-planning has some disadvantages when using under dynamic environment. It needs precise environment model, and will make system complex. For example, in football game, the robot always be put under a dynamic environment, a proper path at this time will be improper during a short time, especially when many robots stay together. So it is not worthy of planning globally. Based on this opinion, our system only makes local path-planning. Path-planning lets robot reach a place behind the obstacle, and only needs local information about obstacles. Experiment and games verified that local path-planning could improve the robot's decision ability, and make robot react rapidly.

We also define some special behaviors. For example, when the ball stays on a corner on the playground or made by other robots, our robot will turn suddenly, thus, the ball will be moved out of the corner. Through this behavior, the ball will not be out of play, and robot can show more intelligence.

3.2 Primary Behaviors and Their Synthesis

Soccer is a complex task for robot, especially for autonomous robot. In the game, robot should search ball, move ball and shoot ball autonomously. Moreover, when he meets an obstacle, he must coordinate multiple behaviors. Generally, we decompose the whole task into several subtasks associated with ball. Behaviors are divided into two kinds: manipulating ball and avoiding obstacle. Every subtask generates a behavior associated with the ball. The final behavior should be made through arbitration or fusion. That is, we make a two-step decision: in the first step, every subtask generates a behavior, and then system decides the final behavior. From the view of human's behavior, when we do such a work, there is just a natural behaviors transition but not such a layered decision.

Thus, we decomposed the whole task into four subtasks corresponding to four states: Get_Ball, Move_To_Ball, Search_Ball and Avoid_Obstacle. If robot is in a state of the first three ones, he can manipulate ball directly and needs not to avoid obstacle. If the robot has to avoid obstacle, he will be in the forth state. In this state, a behavior generated by local path-planning will help him leave this state. Different from common avoiding behavior, robot does not care about where the ball locates, he only moves directly to the target generated by path-planning at highest speed, that is, there is only one behavior. The result is that robot escapes from the situation where multi-behavior conflicts, and can manipulates ball directly. This simple behavior makes robot transit from the forth state to one of the other three ones. Moreover, through such task decomposition, there is only one behavior generated in every state at a time, and this is the final behavior of the robot. Robot needs not to select a behavior from multi behaviors. From above, we can see that this method can simplify the control architecture, and new behaviors can be added randomly without making control system more complex.

Every robot has seven basic behaviors: MoveToBall, MoveToTarget, SearchBall, Defend, Shoot, AvoidObstacle and SearchGoal. The robot's program for accomplishing the task can be represented as a FSM diagram as shown in Fig. 4.

In this approach, each state corresponds to a suite of activated behaviors for accomplishing that step of the task. Transitions between states are initiated by real-time perception.

We will give an example to illuminate the algorithm. Suppose the robot is in the state of Move_To_Ball, at this time, an obstacle suddenly appears in the way, the robot then will transfer to the state of Avoid_Obstacle. The path-planning module generates a target for the robot as shown in Fig.5. DIST_1, DIST_2 is related to the robot's size and can be modified. During this period, if the ball moves to the place where it can be manipulated directly by the robot, robot will transfer to other state, otherwise, he will move to the target at all times. During this period, robot will not avoid obstacles except that an obstacle appears in the way and is very near. When robot reaches the target, if he cannot see the ball, he will turn according to his short-time memory. Thus the states transition is realized.

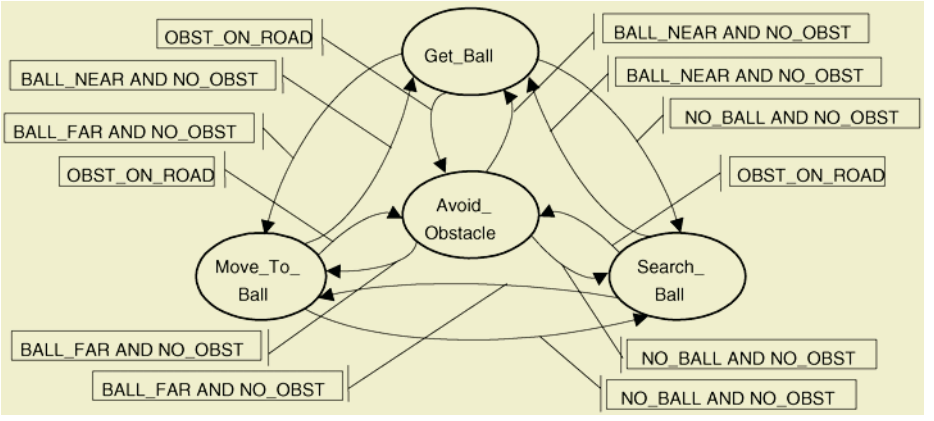

Fig. 4. State transition

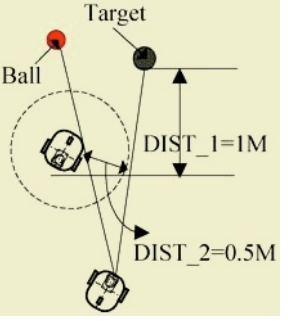

Fig. 5. Local path planning

Formula 1,2,3 can calculate the translational velocity (V) and rotational velocity (ω). In formula 1, DIST means the distance between robot and the goal calculated by visual information. The units of V and ω are mm/sec, degree/sec. P_1, P_2, P_3 are parameters. Ang means the angle of the goal position in robot's local coordinate as shown in Fig. 3.

When the obstacle is very near the robot, the line speed is calculated by formula 3. OBST_DIST is distance between robot and obstacle calculated according to LRF.

$$V = \begin{cases} DIST^2 / P_1 & DIST < 1m \\ 1000 & DIST >= 1m \end{cases} \tag{1}$$

$$\omega = \begin{cases} -40 & Ang >= 100^\circ \\ (90 - Ang) / P_2 & 80^\circ < Ang < 100^\circ \\ 40 & Ang <= 80^\circ \end{cases} \tag{2}$$

$$V = -P_3 / OBST\_DIST \tag{3}$$

4 Experiments and Results

Fig. 6 shows a standard action sequence of our robots played in the "CRC Soccer Robot in China".

The playground is set according to RoboCup rules. As there is only 1vs1 and 2vs2 game, the ground is a 6×5m place. The two teams are both autonomous robots. They can be started from outside but cannot be interfered during the competition.

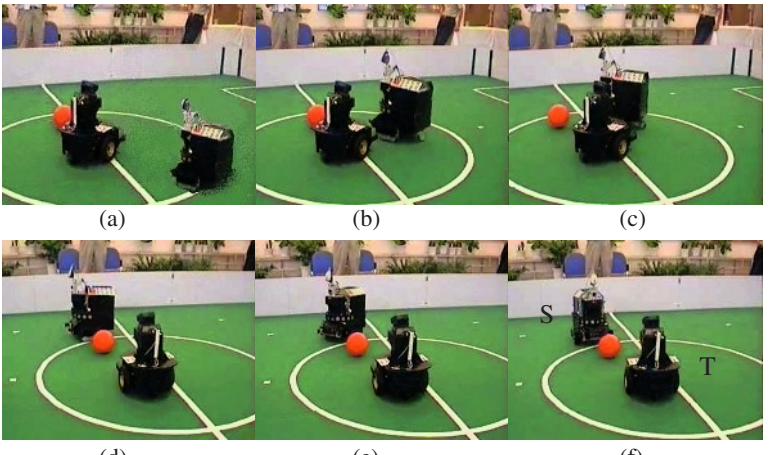

Fig. 6. Action sequence

"S" represents our robot, and "T" is the opponent. In Fig. (a), robot saw the ball, but could not manipulate it directly because of the obstacle in the way. The path-planning module generated a target behind the obstacle. Through actions in Fig. (b)-(c)-(d), robot moved to the target. At this time, robot could not see the ball, so he turned to left according to his short-time memory. Thus, after action sequence in Fig. (e)-(f), robot returned to the state of Get_Ball. Through such task decomposition, robot could finish task rapidly.

Fig.7 shows trajectories of the robot in different situations. In the figure, 2 represents dashed line. 1 and 3 represent straight line. We can divide the whole process into three periods corresponding to three lines. In the first period, the trajectory is generally similar during many cases. The robot moves to the target rapidly and directly. In the second period, the robot moves at a certain translational speed and rotational speed. That is, the trajectory is an arc, and this ensures the

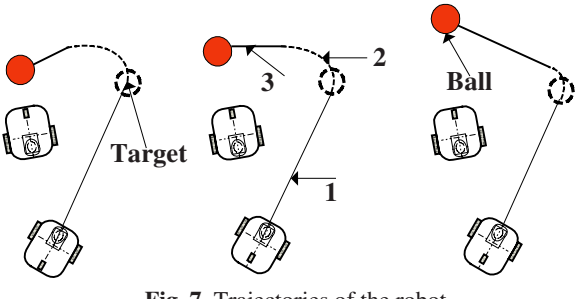

Fig. 7. Trajectories of the robot

smoothness of the behavior transition. After searching for some time, robot will see the ball and move to ball directly. This is the third period.

Fig.8 shows the trajectory of a real robot's motion. The environment is set like figure 7. The dark circle means an obstacle. The unit of X-axis and Y-axis is centimetre.

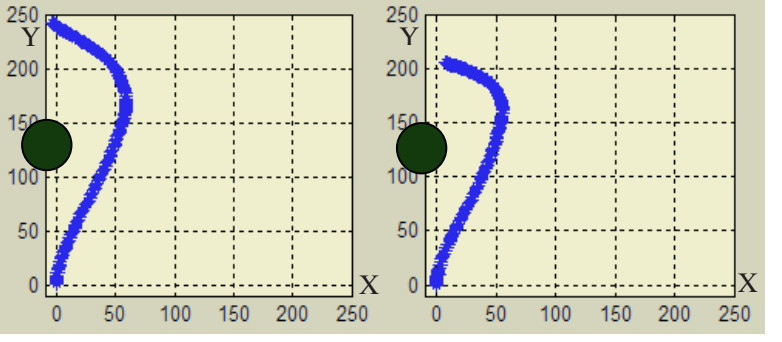

Fig. 8. The robot's motion trajectory

Tab. 2 shows the time of state transition from Avoid_Obstacle state to Search_Ball state to Move_To_Ball state. These states are corresponding to the three periods of the trajectory as shown in Fig.7. The highest speed of robot is 8000mm/sec and the robot is stationary at the beginning. For each case (a and b), we made experiment for 10 times. The value in the table is an average one. From Fig.8 and Tab.2, we can see that the robot reacts rapidly and the trajectory is smooth.

Table 2. The time of motion

	Period 1	Period 2	Period 3	Total time (second)
Fig.8 (a)	3.25	1.18	1.02	5.45
Fig.8 (b)	3.23	1.15	0.82	5.2

In a word, the proposed method for multi-behavior coordination has two advantages. From the high level, there is only one behavior generated in one state and this is just the final behavior. This behavior is easily understood as this is only related to one goal or obstacle but not a coordinated one. This means the control process is simple and easy to test. From the low level, in the Avoid_Obstacle state, the robot always makes a straight-line motion and can keep a high speed in avoiding obstacles. But in some fusion method, the robot's speed is a low one when the robot is near the obstacle.

5 Conclusion

This paper introduces an AMR system applicable under dynamic environment, and adopts an FSM-based behavior selection method to solve multi-behavior coordination problem. With this method, the robot can coordinate multiple behaviors easily and react rapidly under dynamic environment. At the same time, this method simplifies

the control architecture. The results of experiments and games show the validity of the method. In the future, this work should be extended to study more complex tasks such as multi-robot learning and coordination under dynamic environment.

References

1. K. Watanabe, J. Tang and M. Nakamura et al., "A fuzzy-Gaussian neural network and its application to mobile robot control," *IEEE Trans. Control Systems Technology*, Vol. 4, pp. 193-199, 1996
2. Weimin Shen, J. Adibi and R. Adobbati et al., "Building integrated mobile robots for soccer competition," *Proc. IEEE Int. Conf. Robotics and Automation*, 1998
3. J. Connell, "SSS: a hybrid architecture applied to robot navigation," *IEEE Int. Conf. Robotics and Automation*, 1992
4. 4. Changhong Fan, Weidong Chen and Yugeng Xi, "Automonous Robot Soccer System Based on Hybrid Architecture," *4th World Congress on Intelligent Control and Automation*, Shanghai, P.R.China, June 10-14, 2002
5. R. A. Brooks, "A robust layered control system for a mobile robot," *IEEE Journal of Robotics and Automation*, Vol. 2, pp. 14-23, March 1986
6. R. C. Arkin, "Towards the unification of navigational planning and reactive control," *AAAI Spring Symposium on Robot Navigation*, March 1989
7. J. Kosecka and R. Bajczy, "Discrete event systems for autonomous mobile agents," *Proc. Intelligent Robotic Systems*, pp. 21-23, July 1993
8. M. Egerstedt and Xiaoming Hu, "A hybrid control approach to action coordination for mobile robots," *Automatica*, Vol. 38, pp. 125-130, 2002
9. J. Riekki and J. Roning, "Reactive task execution by combining action maps," *Int. Conf. on Integrated Robots and Systems (IROS)*, pages 224-230, Grenoble, France, 1997
10. R. C. Arkin, "Motor schema-based mobile robot navigation," *International Journal of Robotics research*, Vol. 8, pp. 92-112, 1987
11. G. Schoner and M. Dose, "A dynamics systems approach to task level systems integration used to plan and control autonomous vehicle motion," *Robotics and Autonomous System*, Vol. 10, pp. 253-67, October 1992
12. M. Asada, H. Kitano, I. Noda, M. Veloso, "Robocop: Today and Tomorrow What we have and learned," *Artificial Intelligence*, 1999
13. M. Asada and H. Kitano, "The Robocop Challenge," *Robotics and Autonomous Systems*, Vol. 29, pp. 3-12, 1999