Proceedings of the 2008 IEEE
International Conference on Information and Automation
June 20 -23, 2008, Zhangjiajie, China

Application of Reinforcement Learning Based on Neural Network to Dynamic Obstacle Avoidance

Junfei Qiao, Zhanjun Hou, Xiaogang Ruan
College of Electronic Information and Control Engineering
Beijing University of Technology
Pingleyuan No.100, Chaoyang District, Beijing, China
adqiao@sina.com; houzhanjun@hotmail.com

***Abstract*- This paper focuses on the application of reinforcement learning to obstacle avoidance in dynamic environments. Behavior-based control architecture is more robust and better in real-time performance than conventional model based architecture in the control of mobile robot. An intelligent controller is proposed by integrating reinforcement learning with the behavior-based control architecture and applied to the obstacle avoidance. Neural network is used to approximate the Q-function to store the Q-value. By using the reinforcement learning, the mobile robot can learn to select proper behavior online without knowing the exact model of the system. In experiments, dynamic and static obstacles are placed in the environments separately. Experiment results show that the mobile robot can get to the target point without colliding with any obstacle after a period of learning.**

I. INTRODUCTION

Autonomous mobile robots have a wide range of application, such as in hospitals, houses, industries, space exploration, nuclear plants, etc. Generally speaking, obstacle avoidance is the main aspect of mobile robot. The purpose of obstacle avoidance is to enable mobile robots to get to the target point without colliding with any obstacle in unknown or dynamic environments. A lot of approaches are developed to solve this problem. Most of the researches are concentrated on obstacle avoidance in static or known environments. However in real world, the environments of the mobile robot is unknown or contain dynamic obstacles, the obstacle avoidance in dynamic environments becomes a difficult task. The conventional control technique is model based called planning architecture. Artificial potential fields and cell decomposition belong to this type. There are many shortcomings of the architecture: (i) it needs an exact model which is difficult to be acquired. (ii) it take a long time to execute an action for that the agent should execute every sub model before. (iii) This architecture is limited to known environments. Moreover, the robustness is not good. This technique cannot meet lots of requirements, such as learning ability, capability to manage avoiding dynamic obstacles, robustness and increased autonomy, etc. Behavior-based architecture [1] is different from planning architecture and has many advantages, such as better robustness, quickness, etc. In order to adapt this architecture to the complex and dynamic environments without the further intervention, learning ability should be adopted to improve the autonomy of mobile robots. Reinforcement learning [2][3] is used widely and can be applied to the behavior-based architecture. Reinforcement learning has been proved to be a practical computational tool that permits autonomous systems to improve performance with experiences [4]. One reinforcement learning method named Q-learning is used in this paper. As the neural network has a good generalization and is used to approximate the Q-function.

This paper attempts to apply behavior-based architecture with reinforcement learning based on neural network to the obstacle avoidance in dynamic environments. The task is to let the mobile robot arrive at the target without colliding with any obstacles including dynamic obstacles after a period of time learning all by itself. This paper is organized as follows. Section II described the design of behavior-based architecture with reinforcement learning called intelligent controller. Section III describes the reinforcement learning based on neural network. In section IV, the method talked above is executed in simulated environments. Finally, the paper is concluded in section V.

II. DESIGN OF INTELLIGENT CONTROLLER

There are two control techniques for mobile robots: Planning architecture and Behavior-based architecture. Now behavior-based in used widely for its advantages over the planning architecture. Behavior-based architecture emphasizes on the use of distributed, parallel and primarily reactive control processes, the emergence of complex behavior through the interaction of these process with each other and the environment, cheap computation and multi-layer of behaviors. Although the behavior-based has made the robot work better, it lacks of learning ability. Thus, it is necessary to add the learning ability to the architecture in order to improve the intelligence of the mobile robot. Reinforcement learning (RL) is a learning algorithm developed well and is suitable for the principle of behavior-based architecture. So an intelligent controller is designed on the base of the behavior-based architecture and the reinforcement learning as shown in Fig. 1. The controller contains the following modules: Perception, Basic behaviors, Reinforcement learning, Action selection and Actuators.

A. Perception of Environment

Perception of the environments depends on the sensors of the mobile robot. The mobile robot used in this experiment is a multi-sensor system which is placed 6 sonar sensors and a

978-1-4244-2184-8/08/$25.00 © 2008 IEEE.

CCD camera. Sonar sensors are on the front of mobile robot and they can inform the robot how long the distance between the robot and the static obstacles as shown in Fig. 2. Set $d_{rs} = \{d_{rs1}, d_{rs2}, d_{rs3}, d_{rs4}, d_{rs5}, d_{rs6}\}$ denote the distance.

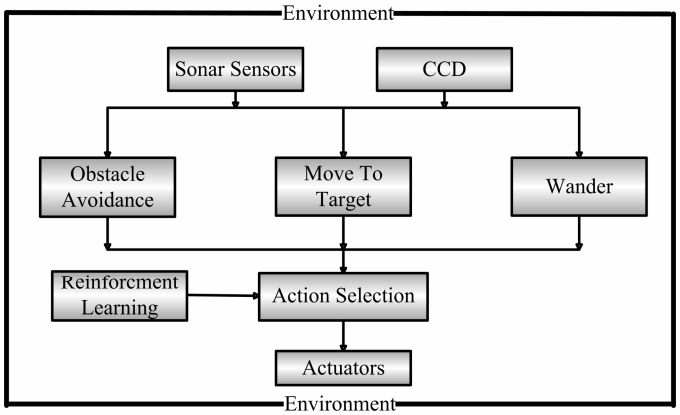

Fig. 1 Design of intelligent controller.

Besides, this figure also shows the perception of the target point. The CCD camera can search for the target point and get the position in order to arrive at the point in the end. Two data are needed in the experiment: distance between the robot and the target d_{rt}, angle between the current direction of robot and line from robot to the target θ_{rt}.

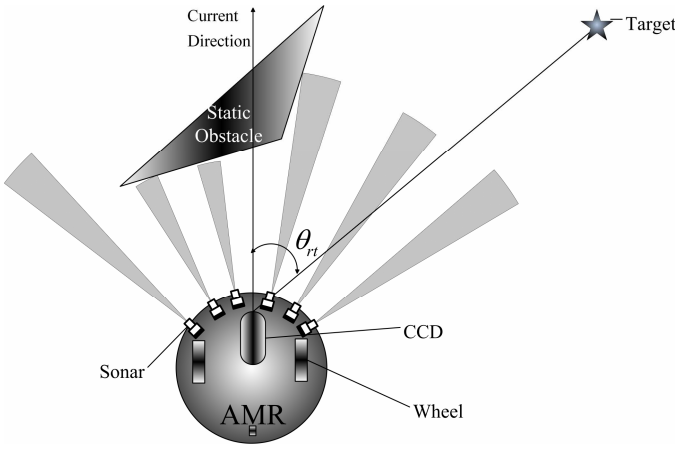

Fig. 2 Perception of Static obstacles and target.

In the environments of the mobile robot not only static obstacles but also dynamic obstacles exist. As shown in Fig. 3, we suppose that the dynamic obstacle is moving at a velocity v_{do} and can be acquired trough the camera. Meantime, an angle θ_{rd} of moving direction of dynamic obstacle relative to the direction of robot is obtained. Besides, the distance d_{rd} between robot and dynamic obstacle is needed in the experiment.

B. Basic Behaviors

Three basic behaviors are set in the intelligent controller, including *Obstacle Avoidance*, *Move To Target* and *Wander*. If other use is needed, new behavior can be added. In the running of the mobile robot, the behaviors are divided into some actions. In experiment, several basic actions can be set, such as *Move Forward, Turn Left 10°, Turn Left 30°, Turn Left 50°, Turn Right 10°, Turn Right 30°, Turn Right 50°*. By executing these actions, the mobile robot can accomplish the task of basic behaviors.

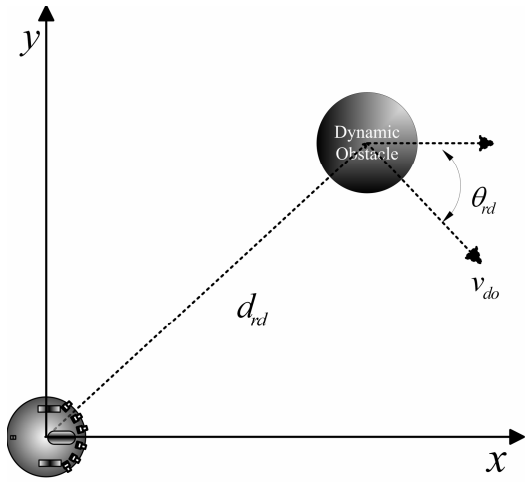

Fig. 3 Perception of dynamic obstacles

C. Reinforcement Signal

Reinforcement learning aims to solve the problem of learning to maximize a numerical reward signal over time in a given environment [5]. The reward signal is the only feedback obtained from the environment and plays an important role in the learning. The signal in the research contains three parts: the change of distance between the robot and static obstacle avoidance, the change of distance between the robot and the dynamic obstacles, the change of distance between the robot and the target point. r_{rs}, r_{rd}, r_{rt} are used to denote three reinforcement signal respectively. The total reinforcement signal is the sum of three signals.

D. Strategy of Action Selection

In reinforcement learning, processes of exploration to exploitation play an important role. At the beginning of the Q-learning, the Q-value is initialized at random and they do not stand for the best state-action pairs and is meaningless. So in order to explore all possible actions, the Boltzmann probability distribution is employed to select all possible actions. The Boltzmann distribution looks as follows:

$$P(a_i \mid s) = \frac{e^{Q(s,a_i)/T}}{\sum_{a \in A} e^{Q(s,a)/T}} \quad (1)$$

where is the virtual temperature that regulates the extent of randomness of the selection with the largest Q-value,.

After a period of learning, the Q-valued is converged to the expected value and it stands for the proper state-action pair.

The greedy policy is used to select the best action for each state according to the Q-value.

$$a = \arg\max_{b \in A} Q(s, b) \tag{2}$$

E. Actuators

The autonomous mobile robot used in this research is similar to the Poineer3 which has three wheels including two drive wheels for controlling the speed and a caster for balance. The distance between two drive wheels is 0.2m and we can change the velocity and direction of the robot by changing velocity of each drive wheel separately.

III. Q-LEARNING BASED ON NEURAL NETWORK

Reinforcement learning is used to learn the internal structure of the behaviors by mapping the perceived states to control actions while maximizing the accumulated reinforcement reward and is attractive to researchers. In RL process, the agent perceives the state of the environment and receives an input that describes the state. Then the agent chooses an action from the action set to execute immediately and then the agent possibly moves to a new state. After that, the agent receives and evaluation feedback r called reinforcement as well as perceives the new state. The purpose of the learning system is to find a control policy that maximizes the expected amount of reward during the whole learning period. The value function is can be described as follows:

$$V^{\pi}(s_t) \equiv r_t + \gamma \cdot r_{t+1} + \gamma^2 \cdot r_{t+2} + \ldots \tag{3}$$

where r_t is the reward received in transition from state s_t to state s_{t+1}, γ $(0 < \gamma < 1)$ is the discount factor. $V(x_i)$ is the discounted amount reward received by the learning system since the time t. It depends on the sequence of action chosen which are determined by the strategy of control. The system has to find a control strategy that maximizes $V(x_i)$ for each state. The model of RL is shown in Figure 4.

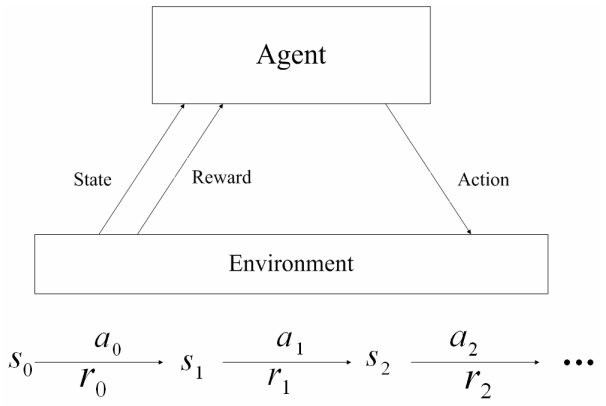

Fig 4. Model of RL

Q-learning is a temporal difference algorithm based on the Markov decision processes. Q-learning is an off-policy algorithm that has the ability to learn without following any particular strategy when actions are selected. This feature enables the robot to choose the proper action after executing actions that caused collision.

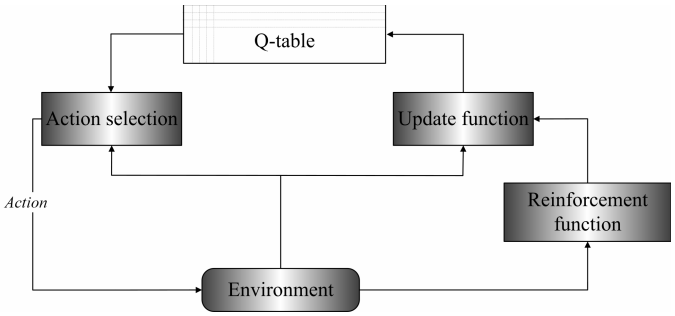

Fig. 5 Q-learning algorithm

As shown in Fig. 5, the agent perceives the environment and then executes an action. After that the agent move to a new state and a reinforcement signal is generated, and then the Q-value is updated according to the equation (4) and is stored in the Q-table. If state-action pairs are continually visited, the Q-values converge to the expected value and the greedy policy is used to select the optimal action.

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha \times (r_t + \gamma \times \max_{b \in A} Q(s_{t+1}, b) - Q(s_t, a_t)) \tag{4}$$

where γ is discount rate and fall in [0 1]. It concerns the maximization of future rewards. If γ=0, the agent maximizes only immediate rewards. α is the learning rate falls in [0 1]. s_t is the perceived state of the agent, a_t is the action selected, A denotes all possible actions.

Some researches have applied reinforcement learning in robot by using the discrete input signal [6]. As the perceived state of robot is continuous and large space is need to store all the possible state-action pair. In order to solve this large storage problem, we should find a method to approximate the Q-function. In paper [7], a method is used to deal with this problem called statistical cluster analysis. There are some limitations of this method: (i) parameters tuning for forming semantically significant clusters is complicated; (ii) a cluster once formed can not be split later. Artificial neural network has a good generalization and can approximate the Q-function, so a neural network is used to address the generalization of the Q-learning. The information of CCD and sonar sensors are the inputs of the neural network and the output is the Q-value. Sigmoid function is used in hidden layer. The expect output which is used to train the neural network is obtained using the function described as (5). Q-learning based on neural network is described as Fig. 6.

$$Q^{target}(s_t, a_t) = r_{t+1} + \gamma \max_{a_{t+1} \in A} Q(s_{t+1}, a_{t+1}) \tag{5}$$

The Q-learning based on neural network algorithm is implemented with the following steps.

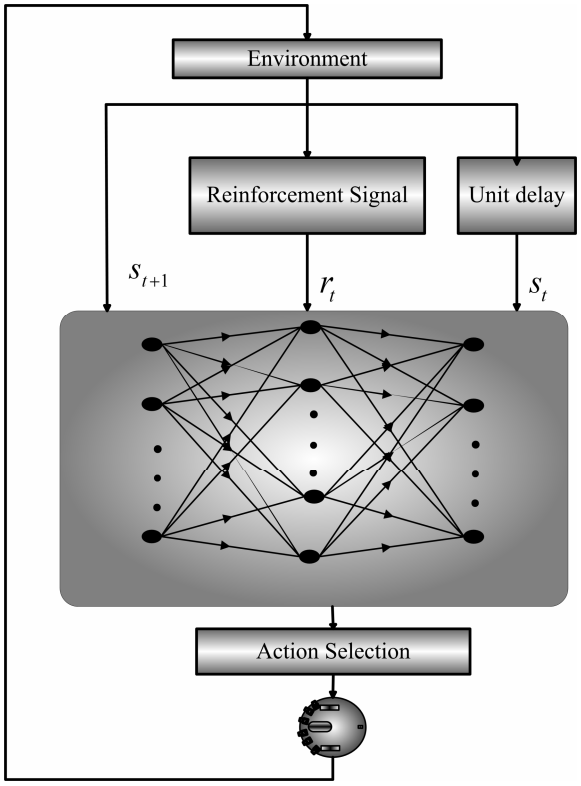

Fig. 6 Q-learning based on neural network

Step1 Initialization.

Initialize the neural network at random and other parameters used in the experiment.

Step 2 Execute action

Select an action according to the action selection strategy and execute it. If collision happens, start from the initial point again.

Step 3 Generate the reinforcement signal

After executing the action selected, the new state can be obtained. Then compare the current state to the last state and compute the reinforcement signal. r_{rt} r_{rd} r_{rs} denote reinforcement signal. The total reinforcement signal is the sum of the three parts.

$$r_{rt} = \begin{cases} +0.3, & closer\ to\ the\ target \\ -0.3, & faraway\ from\ the\ target \\ 0, & otherwise \end{cases} \quad (6)$$

$$r_{rd} = \begin{cases} -0.5, & closer\ to\ dynamic\ obstacle \\ +0.5, & faraway\ from\ dynamic\ obstacle \\ -1.0, & collide\ with\ obstacle \\ 0, & otherwise \end{cases} \quad (7)$$

$$r_{rs} = \begin{cases} -0.6, & closer\ to\ static\ obstacle \\ +0.6, & faraway\ from\ static\ obstacle \\ -1.2, & collide\ with\ static\ obstacle \\ 0, & otherwise \end{cases} \quad (8)$$

Step 4 Update the neural network

Update the weights of the neural network according to the back-propagation algorithm and Q-function.

Step 5 Repeat

Repeat *step2-step4* until the robot learns it.

IV. Experiment Results

We did the experiments in a platform of mobile robot called mobotsim. In this platform, the obstacles can be set and parameters of the mobile robot can be changed easily. Two experiments were executed in the environments that contain static obstacles and dynamic obstacles separately.

In the static environments, the input signal is the sonar values that stand for distance of the robot between the robot and static obstacles. Another input signal is the angle between the current direction robot and the target point. Then the input layer contains 7 neurons for the input signal. The output layer contains 7 neurons according to the 7 basic actions. The hidden layer of the neural network contains 18 neurons. At the beginning of the experiments the robot moves randomly because the Q values are initialized at random. Thus the robot often collides with obstacles without arriving at the target point. After bout 300 epochs of learning, the robot can avoid obstacles and get to the target point as shown in Fig, 7.

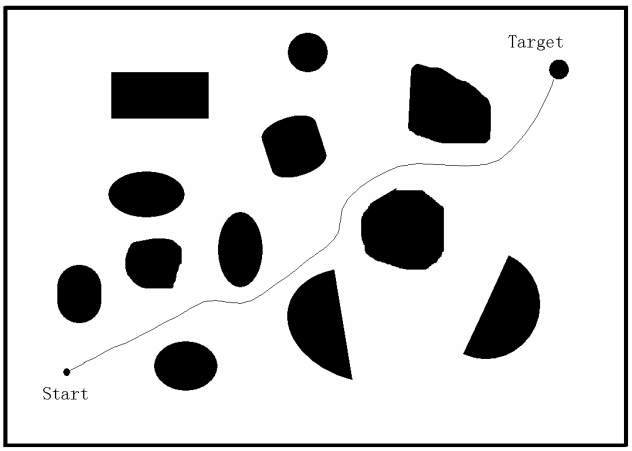

Fig. 7 Static obstacle avoidance

Another experiments is executed in the environment that with a dynamic obstacle. In the platform, only static obstacles are prepared. So another mobile robot is used to stand for the dynamic obstacle. The dynamic obstacle is moving forward at a small speed. The input signal contains three parts: distance between the robot and the dynamic obstacle, angle between the direction of mobile robot and the direction of the dynamic obstacle, angle between the direction of mobile robot and the direction of target point. In the experiment, the neural network

is set as follow: 3 neurons in input layer for 3 input signal, 10 neurons in hidden layer and 3 neurons in output layer for 3 basic actions used here. At the beginning the mobile robot moves randomly and sometimes collide with the dynamic obstacle. In general, the mobile robot can not get to the target point. With the learning goes on, the robot collides with the dynamic obstacles less and less and can get to the target sometimes. After a long period of learning, the robot can successfully get to the target point without colliding with dynamic obstacle. When the mobile robot meets the dynamic obstacle on the way to the target, it avoids the obstacle at first, as shown in Fig. 8.

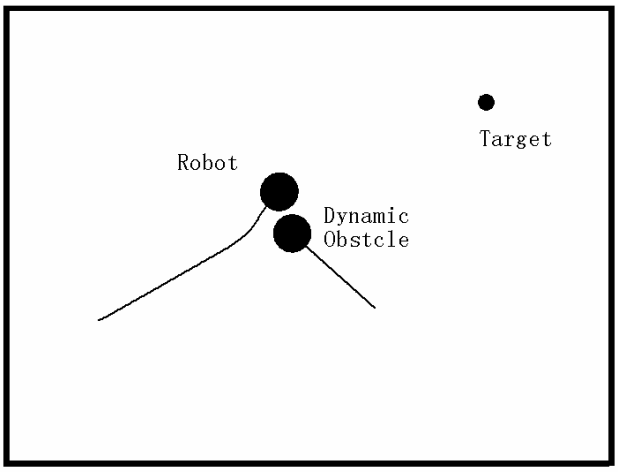

Fig. 8 Dynamic obstacle avoidance

After avoiding the dynamic obstacle, the robot moves to the target point as shown in Fig. 9. Some other experiments were also done and results are good.

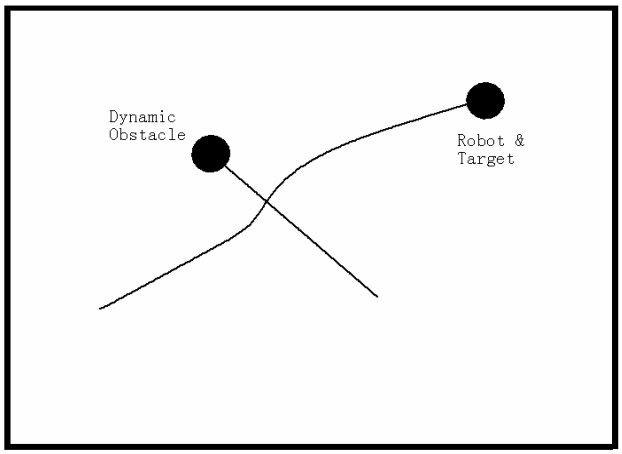

Fig. 9 Moves to the target after avoiding dynamic obstacle

V. CONCLUSIONS

This paper attempts to apply reinforcement learning for the obstacle avoidance in dynamic environment. Because behavior-based control architecture lacks learning ability, an intelligent controller is proposed by integrate reinforcement learning with behavior-based architecture. Q-learning is used in the research. As the Q-learning solve the problem of discrete states and actions, a multi-layer neural network is used to approximate the Q-function. Experiments are executed in both static environment and dynamic environment. Experiments results show the effectiveness of the method. In the experiment, BP neural network is used and the convergence needs a long time. In the future, self-organizing neural network will be tested for the purpose of better control.

ACKNOWLEDGMENT

The authors would like to acknowledge financial support from National Natural Science Foundation of China (No. 60375017), Beijing Subsidy Funds of Fostering Excellent Talents (2006D0501500203) and Science & Technology Development Plan Project of Beijing Education Committee (KM200610005019).

REFERENCES

[1] Brooks R, A robust layered control system for mobile robot, IEEE Journal of Robotics and Automation, RA-2, No.1, pp.14-23, 1986.
[2] D. Bertsekas, J. Tsitsiklis, Neuro-Dynamic Programming, Athena Scientific, Belmont, Ma, 1996.
[3] Kaelbling L P, Reinforcement learning: A Survey, Journal of Artificial Intelligence Research 4(3), pp.237-285, 1996
[4] Kumar S, Neural network: A classroom approach, International edition, McGraw-Hill Publishing Company Limited, 2005.
[5] Sutton, R., & Barto, A. Reinforcement learning: An introduction. Cambridge, MA: MIT Press., 1998.
[6] L J Lin, Self-improving reactive agents based on reinforcement learning, planning and teaching, Machine Leaning, 8, pp293-321, 1992.
[7] Mmahadevan S., Connelly J., Automatic programming of behavior-based robots using reinforcement learning, Artificial Intelligence, 55,2, pp311-256, 1999.