Self-Organization of Visuo-Motor Map Considering an Obstacle

Yuji MARUKI
Oita National College of Technology, 1666 Maki Oita, 870-0152, Japan
(Tel: +81-97-552-7443; Fax: +81-97-552-7548, E-mail: maruki@oita-ct.ac.jp)

Keywords: Neural Network, Self-Organizing Map, Robot Manipulator

Abstract: The visuo-motor map is based on the Kohonen's self-organizing map. The map is learned the relation of the end effecter coordinates and the joint angles.

In this paper, a 3 d-o-f manipulator which moves in the 2D space is targeted. A CCD camera is set beside the manipulator, and the end effecter coordinates are given from the image of a manipulator.

As a result of learning, the end effecter can be moved to the destination without exact teaching.

1 Introduction

The system, which is combined with a manipulator and a video camera for autonomous moving of the end effecter, is called a hand-eye system. The purpose of this research is to make a visuo-motor map, which is based on the Kohonen's self-organizing map, learn the relation of the end effecter coordinates and the joint angles. As a result, the end effecter can be moved to the destination without exact teaching.

In the other researches, a video camera has caught the image of the whole manipulator to get the end effecter coordinates. But when the hand-eye system is used in practice, it is much better to put the camera near the end effecter, because the more exact coordinates can be given by the image.

For this paper, a 3 d-o-f manipulator is targeted, and the position of a video camera is near the end effecter, so the first joint is out of view of a camera. Moreover, an obstacle avoidance is tried by a simple potential method.

2 Outline of system

2.1 System

In this research, the system consists of the followings, i.e. (1) a 3 d-o-f manipulator which moves in the vertical 2D space, (2) a CCD camera which catches the side view of the manipulator, (3) image processing board for getting the coordinates of the end effecter, (4) personal computer which executes a visuo-motor map program and sends commands to manipulator.

When this method is used in practice, the relation of an end effecter coordinates and the joint angles is learned by simulation at first. Then taking over the 4 parameters of neuron, the learning program is executed again, moving real manipulator. But it is shown the results of only a simulation in this paper.

The end effecter coordinates are given with processing the image of a manipulator which is caught by a CCD camera. If an LED is attached on the end effecter for example, we can get the end effecter coordinates by calculating the center coordinates of the image of the LED.

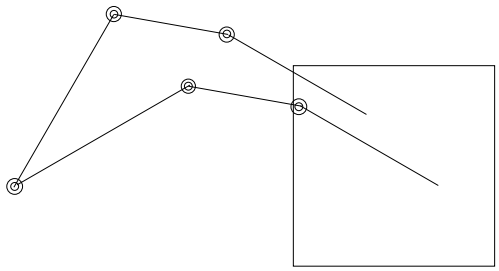

Figure 1: A Manipulator and a View of Camera

2.2 Visuo-motor map

The visuo-motor map has 4 parameters.

- W_i **:** The end effecter coordinates.
  It is equivalent to the position of neurons in the visuo-motor map. Its coordination system is equal to an image input space.
- J_i **:** Jacobian matrix which converts joint space to work space.
- ξ_i **:** The differencial vector of an evaluation functionH_i
  Various estimation functions are thought as H, but the manipulability [4] is used here, which represents the distance from a singular point.

- θ_i : The joint angles.
  When a target(i.e. the destination of the position of an end efecter u_t) is given near W_i, the map outputs angles (θ^{out}) near θ_i.

3 Learning algorithm

3.1 In case of no obstacle

The learning algorithm of a visuo-motor map without obstacles is put in order based on the method by K. Asada[1]. The nth time aroud learning procedures of the map are as follows. N is the number of neurons which are neighborhood of target.

1) At first, a target u_t^n in work space is given randomly. In this paper, a target u_t^n consists of the 2-D coordinates and angle of the end efecter.

2) Following next equation, much higher order is given to the neuron if its parameter W is much nearer to a target u_t. The next norms are sorted.

$$\|u_t^n - W_i^n\| < \cdots < \|u_t^n - W_j^n\| \qquad (1)$$

That is, $order_i$ is 0(the first order), $order_j$ is $(N-1)$(the Nth order).

3) The joint angle outputs θ_0^{out} for making the end effecter reach the target are calculated approximately by the following equation, and the manipulator is moved. The position v_0 of the end effecter is observed by a CCD camera.

$$\theta_0^{out} = \frac{\sum_{i=1}^{N} g(order_i, \lambda_{out}^n)(\theta_i^n + J_i^{n+}(u_t^n - W_i^n))}{\sum_{i=1}^{N} g(order_i, \lambda_{out}^n)} \qquad (2)$$

That is, J_i^{n+} is a quasi-inverse matrix, and the denominater is a value for normalization.
$g(\cdot)$ is given by next equation. By this function, the weight of a neuron is corrected more largely, if the order of a neuron is higher.

$$g(order, \lambda_{out}^n) = \exp\left(-\frac{order_i}{\lambda_{out}^n}\right) \qquad (3)$$

λ_{out}^n is the parameter which decides the range of number to be updated. It is changed according to learning times by the following equation. Consequently it is sufficient to be given the λ_{out}^n th order of norms by sorting.

$$\lambda_{out}^n = \lambda_{out}^{init}\left(\frac{\lambda_{out}^{final}}{\lambda_{out}^{init}}\right)^{\frac{n}{T_{max}}} \qquad (4)$$

That is, $\lambda_{out}^{init} and \lambda_{out}^{final}$ are respectively the initial value and the final value of λ_{out} for learning, and those are given before learning. T_{max} is a maximum learning times.

4) To reduce the positioning error by the joint angle outputs θ_0^{out}, the visual feedback is done by the following equation. The position v_1 of the end effecter by the joint angle θ_1^{out} is observed by a CCD camera.

$$\theta_1^{out} = \theta_0^{out} + \frac{\sum_{i=1}^{N} g(order_i, \lambda_{out}^n)(J_i^+(u_t - v_0))}{\sum_{i=1}^{N} g(order_i, \lambda_{out}^n)} \qquad (5)$$

5) The 4 parameters of neuron are learned according to the information which is given in the procedures from 2) to 4).

a) Learning of W
W is learned preserving a topology of the input space according to the following equation.

$$W_i^{n+1} = W_i^n + \varepsilon_W^n g(order_i^n, \lambda_W^n)(u_t^n - W_i^n) \qquad (6)$$

That is, $g(\cdot)$ is the function to decide the amount of W to be corrected, and it is given by the equation same as Eq.(3). ε is a learning ratio, which is given by the equation same as Eq.(4).

b) Learning of J
J is learned by the following equations.

$$J_i^{n+1} = J_i^n + \varepsilon_J^n g(order_i, \lambda_J^n)\Delta J_i^n \qquad (7)$$

ΔJ_i^n is decided by the Widrow-Hoff's learning rule. Estimation function is represented by the following equation.

$$E_J = \frac{1}{2}\left\{(v_1 - v_0) - J_i^n(\theta_1^{out} - \theta_0^{out})\right\}^2 \qquad (8)$$

Therefore, ΔJ_i^n is given by the following equation.

$$\Delta J_i^n = \frac{(v_{01} - J_i^n\theta_{01}^{out})\theta_{01}^{out\,T}}{\|\theta_{01}^{out}\|^2} \qquad (9)$$

Where, $v_{01} = v_1 - v_0,\ \theta_{01}^{out} = \theta_1^{out} - \theta_0^{out}$

c) Learning of ξ
ξ is a gradient vector $(\partial H/\partial\theta)$ of manipulability.

$$\xi_i^{n+1} = \xi_i^n + \varepsilon_\xi^n g(order_i, \lambda_\xi^n)\Delta\xi_i^n \qquad (10)$$

$\Delta\xi_i^n$ is updated by the Widrow-Hoff's learning rule, same as ΔJ_i^n. Estimation function is as follows.

$$E_\xi = \frac{1}{2}\left\{(H_k - H_j) - \xi_i^{n\,T}(\theta_k^{out} - \theta_j^{out})\right\}^2 \qquad (11)$$

That is, j is the neuron number which has the nearest W to a target, and k is one which has the secondarily near W to a target. H_i is the manipulability of the neuron i, and is given by $\sqrt{\det(J_i^n J_i^{n\,T})}$. So,

$$\Delta\xi_i^n = \frac{(H_{jk} - \xi_i^{n\,T}\theta_{jk}^n)\theta_{jk}^{n\,T}}{\|\theta_{jk}^n\|^2} \tag{12}$$

That is, $H_{jk} = H_k - H_j,\ \theta_{jk}^n = \theta_k^n - \theta_j^n$.

d) Learning θ
θ is learned by the following equations.

$$\theta_i^{n+1} = \theta_i^n + \varepsilon_\theta^n g(order_i, \lambda_\theta^n)\Delta\theta_i^n \tag{13}$$

$\Delta\theta_i^n$ is corrected to be more operationable acording to the following equation.

$$\dot{\theta}_i = J_i^+(u_t - W_i) + (I - J_i^+ J_i)\xi_i K_p \tag{14}$$

That is, K_p is the coefficient for satisfying the conditions that $\dot{\theta}_i$ isn't too much, and the manipulability is made larger. In this paper, K_p is changed according to the learning times. $\Delta\theta_i^n$ is as follows.

$$\Delta\theta_i^n = \theta_i^{Desire} - \theta_i^n \tag{15}$$

$$\theta_i^{Desire} - \theta_0^{out} = J_i^{n+}(W_i^n - v_0) + (I - J_i^{n+}J_i^n)\xi_i^n K_p \tag{16}$$

Therefore,

$$\Delta\theta_i^n = \theta_0^{out} - \theta_i^n + J_i^{n+}(W_i^n - v_0) + (I - J_i^{n+}J_i^n)\xi_i^n K_p \tag{17}$$

6) The procedures from 1) to 5) are repeated until the average of errors is reduced.

3.2 In case that one obstacle exists

In case that one obstacle exists in the vision space, some processes are different from the former procedure 5). If an obstacle dosen't exist, the parameter W will be learned to approach a target. But when an obstacle exists, the parameter W will be made learn to go away from a target which is near the obstacle.

In this paper, it assumed that the position and the size of a obstacle are given by the image from a CCD camera. In order to make the end effecter avoid an obstacle, first, the coordinates of targets mustn't be given in the area of a obstacle. Next, the repellent potetials are given to an obstacle, and the weight W of neurons are learned by the following algorithm.

[1] For the points u_o on the outline of an obstacle

$$W_i^{n+1} = W_i^n - \mu_i\varepsilon_W^n P(d)(u_i^n - W_i^n) \tag{18}$$

[2] For a point u_i inside an obstacle

$$W_i^{n+1} = W_i^n - \mu_i\varepsilon_W^n P(d)(u_i^n - W_i^n) \tag{19}$$

That is, μ_o and μ_i are strength of potential, and μ_o is half of μ_i here. $P(d)$ is a simplified potential function.

$$P(d) = \begin{cases} 1/d - 1/d_0 & \text{for } d \le d_0 \\ 0 & \text{for } d > d_0 \end{cases} \tag{20}$$

That is, d is a minimum distance from an obstacle, and d_0 is an effective range of the potential.

4 Simulation

4.1 In case of no obstacle

The link lengthes of a small 5 d-o-f industrial robot are introduced as those of manipulator in this paper. The size of image which is given by a CCD camera is thought as 512 x 512 pixels. The unit of the link length is pixel in the learning program. In the simulation, the end effecter coordinates are calculated from the joint angles θ_i and the link lengthe, instead of getting them from an image by a CCD camera.

Total number of neurons is 500 in this paper. 7000 targets are given randomly in the square vision space. The parameters of neurons are learned with repeated calculation. The learning ratio ε and the number of neurons $g(\cdot)$ which is learned in one calculation, are decreased as the learning process progresses. Moreover, the procedures from b) to d) are repeated several times in the procedure 5). In case that obstacles don't exist, the positioning errors decreased uniformly.

Fig.2 shows the positions of the neurons in the vision space after learning.

According to the relative position of a manipulator and a camera, it can be thought that the end effecter can't move all over the vision space. In Fig.3, the targets were given in the fan shape which is a representative shape as the work space of a manipulator.

The neurons are spreading in the fan shape, but the position error has increased.

4.2 In case that one obstacle exists

When an obstacle exists in the vision space, the neurons which are near to an obstacle before learning are learned to be separated from it by repellent potential.

Fig.3 shows the case that a round obstacle exists. The neurons became to avoid an obstacle. But the positioning errors have increased.

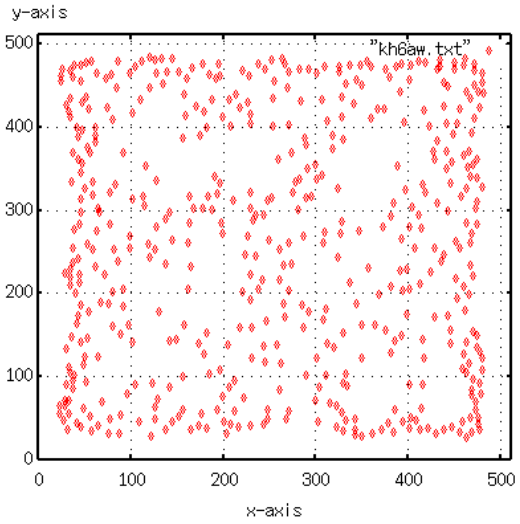

Figure 2: Position of Neurons after Learning

5 Conclusions

In case that an obstacle doesn't exist, the learning process of the viso-motor map has progressed well, and the positioning error has reduced, even if the first joint of manipulator is out of view of a camera.

In case that one obstacle exists, a part of the neurons had become to avoid an obstacle. But the positioning errors had increased, and there is the case that a link of the manipulator pierces an obstacle. So the method to give the potential to an obstacle must be improved.

After this, I'm going to increase the number of cameras, and to expand the learning program to move manipulator in 3D space.

Acknowledgement

The author gratefully acknowledge the profitable advices of Dr. Okada in Kyushu Univ., Japan.

References

[1] K. Asada et al., "Self-organization of a Task-oriented Visuo-Motor Map for a Redundant Manipulator", Technical Report of IEICE(Japanese), NC95-76, pp.15-22, 1995

[2] S. Kawamura, "Planning and Control of the Robot Motion Based on the Concept of Potential", Measuement and Control(Japanese), Vol.29, No.3, pp.237-240, 1990

[3] J. Furusyou et al., "Manipulability of Robotic Manipurators Considering the Influence of Obstacles, JRSJ(Japanese), Vol.6, No.3, pp.12-20, 1988

[4] T. Yoshikawa, "Manipulability of Robot Arm", JRSJ(Japanese), Vol.2, No.1, pp.63-67, 1984

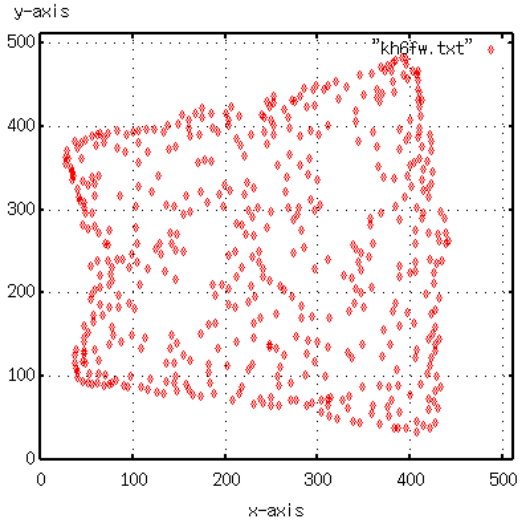

Figure 3: Position of Neurons in Fan Shape

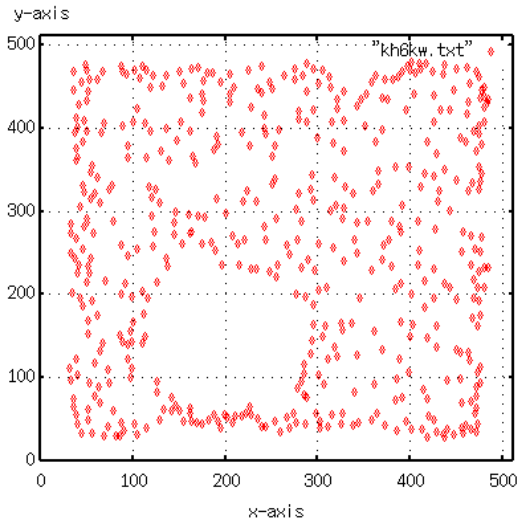

Figure 4: Result of Learning with an Obstacle