Collision Avoidance for a Visuo-motor System Using Multiple Self-organizing Maps

Han, Min
Department of Intelligent Machinery and Systems : Graduate Student

Okada, Nobuhiro
Department of Intelligent Machinery and Systems : Associate Professor

Kondo, Eiji
Department of Intelligent Machinery and Systems : Professor

https://hdl.handle.net/2324/3288

出版情報：九州大学工学紀要. 65 (4), pp.129-142, 2005-12. 九州大学大学院工学研究院
バージョン：
権利関係：

Memoirs of the Faculty of Engineering, Kyushu University, Vol.65, No.4, December 2005

Collision Avoidance for a Visuo-motor System Using Multiple Self-organizing Maps

by

Min HAN[*], Nobuhiro OKADA[**] and Eiji KONDO[***]

(Received November 2, 2005)

Abstract

Collision avoidance for a visuo-motor system in unstructured and cluttered environment is described. The achievement of collision avoidance is based on a simplified path planning system and motion control performed by self-organizing maps. The self-organizing maps are learned to determine joint angles of a redundant manipulator. Since the learning algorithm promises to make the manipulator reach targets precisely with obstacle-free poses, the path planning system only needs to plan a collision-free path for the end effector of the manipulator in the image spaces. By means of the cooperation of two self-organizing maps, the system solves occlusion problems successfully. Simulation results are presented to demonstrate the effectiveness of the proposed approach.

Keywords: Collision avoidance, Self-organizing maps, Visuo-motor system, Redundant manipulator, Path planning

1. Introduction

Collision avoidance of redundant manipulators is becoming increasingly important in practical applications. It has been studied by many researches. Most of them generally indicate to find globally a collision avoidance path in a configuration space, in which the number of dimensions corresponds to the number of degree of freedom that the manipulator has, before performing tasks[1) 2)]. However, in these global planning approaches, exact, known and static environment models are required. In addition, the Computational cost of the planning does increase exponentially with n-DOF *(i.e.* degrees of freedom) of the manipulator. Most motion planning problems are proved computationally hard.

The self-organizing map (SOM), which is proposed by Kohonen[3)], can solve inverse kinematics problem without supervised learning. The SOMs also can be used for path-planning or trajectory formation tasks[4) 5)] . After the mapping has been established, a path is generated from any initial position to a given target, e.g., to guide an end effector of a robot manipulator in the presence of obstacles within the workspace. Using the TRN model, the

[*] Graduate Student, Department of Intelligent Machinery and Systems
[**] Associate Professor, Department of Intelligent Machinery and Systems
[***] Professor, Department of Intelligent Machinery and Systems

paper[6] showed that a locally optimized path can be determined by minimizing the Euclidean distance from the current position to a given target position. Collision checking is conducted while planning the optimized path.

On the contrary, we propose a scheme that realizes collision avoidance for an n-DOF redundant manipulator in image spaces (2 dimensional spaces), and only consider whether the path of the end effector is collision-free or not. We integrate the path planning of the end effector and self-organizing maps to achieve collision avoidance. The self-organizing maps are learned to perform motion control, by which joint angles of the manipulator are determined. The learning promises to make the manipulator arrive targets precisely with obstacle-free poses. The path planning system plans a collision-free path for the end effector from an initial point to a goal point in the image spaces. Therefore, in our system reconstructing of 3D geometry is not necessary. The approach proposed in this paper differs from them in: (1) The system only needs to plan a collision-free path for the end effector in collision avoidance; the computational cost of the path planning does not increase exponentially even for a high dimensional redundant manipulator. (2) The obstacle-free poses of the manipulator are achieved in the learning of the SOM, so collision checking is not necessary in whole path planning process. (3) Besides the number of movable joints of the manipulator, no further pre-knowledge about the manipulator and the cameras will be used.

The potential field method, which was first proposed by Khatib[7], is an efficient local path planning method and has been widely used in obstacle avoidance. The problem with this approach is to become increasingly difficult to determine good potential function with increasing dimensionality, which does not cause the planner to get trapped in local minim. In 1990 Connolly[8] proposed the use of local minima free Laplace's equation for path planning. A potential field based on Laplace's equation has no local minimal point. Therefore a path planning is performed without falling in local minima. In this paper, we use Laplace potential method to plan a path for the end effector.

In the previous researches, we realized collision avoidance for a visuo-motor system in a 3D space by using a self-organizing map[9]. In that research, a stereo camera system was used to provide 3D information. However, the occlusion problem was not addressed well; the spaces occluded by obstacles in the image spaces were untreatable for the system. In addition, in order to solve corresponding problems in a simple way, some constraints also existed between the cameras in that research.

In this paper, we employ a visuo-motor system with two related self-organizing maps and a redundant camera system. The self-organizing maps are directly connected to the cameras and learned to perform position control. Based on the visibility of targets given in the workspace, a more appropriate map is adopted. The map outputs a set of joint angle commands which make the manipulator reach the targets and make it with obstacle-free poses. By using two maps alternately, we solve the occlusion problems in cluttered environment. The learning method ensures that the manipulator moves smoothly and coherently in whole workspace no matter which map is used to control the manipulator. Furthermore, we use a data base instead of any calibrations of cameras to solve corresponding problems while planning a path. The data base is constructed automatically in the self-organizing maps’ learning stage. It does not expend an extra computation cost.

The paper is organized as follows: outline of our visuo-motor system and the learning algorithm of self-organizing maps are introduced in Section 2; approach of collision avoidance and the path planning are discussed in Section 3; simulation results are shown in Section 4; conclusions are given in Section 5.

2. Our Visuo-motor System

Our visuo-motor system is illustrated in **Fig. 1**. The system contains:

- A 4-DOF redundant manipulator moving in a 3D space
- Three CCD cameras
- Two related self-organizing maps

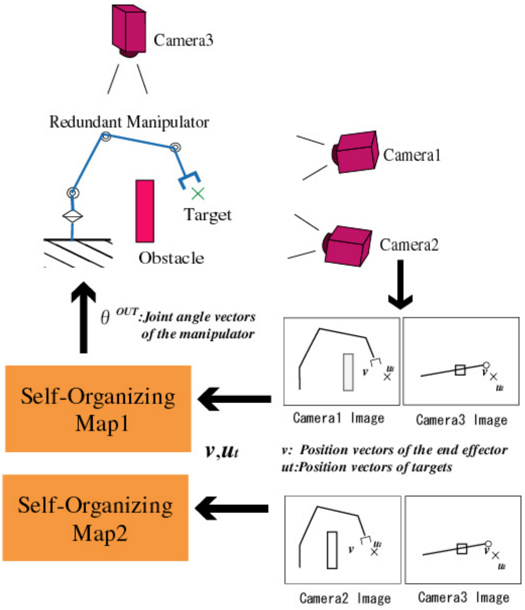


Fig. 1 Outline of the visuo-motor system.

The CCD cameras are used to know the positions of the targets, the locations of the end effector and the poses of the manipulator. They also acquire information about obstacles. Based on visual information provided by the cameras, the self-organizing maps learn projections that convert the image vectors of targets in the image spaces into joint angle vectors of the manipulator. The manipulator is commanded by a set of joint angle commands θ^{out} that are the output from the self-organizing maps.

Although a stereo camera system can provide complete 3D information, spaces occluded by obstacles cannot be dealt with well in our previous works. In this paper a redundant camera system is used to solve the occlusion problem. Two of the cameras are arranged to see the workspace from the side, and the third camera observes from the top. The valid workspace is increased obviously by adding the third camera. Consequently, two related self-organizing maps are employed in our system. As shown in **Fig.1**, the projections of a target point u_t in camera1, camera2 and camera3 are (u_R, v_R), (u_L, v_L) and (u_T, v_T). A pair of image coordinates of the camera1 (u_R, v_R) and camera3 (u_T, v_T) are combined into a 4 dimensional vector $u_t(u_R, v_R, u_T, v_T)$ which is used as the input of the map1. A pair of the image coordinates of camera2 (u_L, v_L) and camera3 (u_T, v_T) are combined into the input of the map2. Because the valid workspaces of two maps are different from the visible space of camera1 or camera2, two maps are used alternately.

Besides the number of movable joints, no further information about the manipulator and the cameras will be used in our visuo-motor system.

2.1 Self-organizing maps

As shown in **Fig. 2**, each self-organizing map consists of neurons, which are distributed in the image spaces of the cameras. Each neuron N_j has 4 parameters.

- W_i : Position of the neuron in the image spaces.
- J_i : Jacobi matrix from the joint angle space to the image spaces.
- θ_i : Joint angle vector of the manipulator at W_i .
- ξ_i : Gradient vector of the evaluation function H .

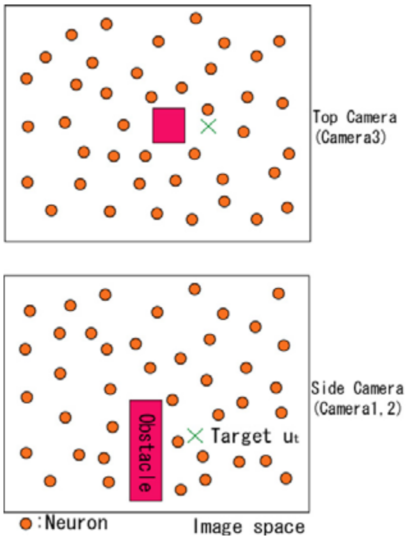


Fig. 2 A self-organizing map.

When a target u_t is given in the workspace, either map1 or map2 is chosen based on which cameras can see the target. In the chosen map the neuron N_i , which W_i is the nearest to the projection of the target u_t , is used. The joint angles θ^{out} , which conduct the manipulator end effector to the target, are calculated obeying following linear equation Eq. (1). Although it is not a linear projection for a redundant manipulator from the image spaces to the joint angle space, we think the domain of each neuron is small enough to use a linear projection as an approximation of the non-linear projection.

$$\theta^{out} = \theta_i + J_i^{\dagger}\left(u_t - W_i\right) \tag{1}$$

Where, $J_i^{\dagger}$ is a pseudo-inverse matrix of J_i .

In an actual system, weighted sum of outputs from plural neurons around the target is used instead of Eq. (1).

$$\theta^{out} = \frac{\sum g\ \left(\theta + J^{\dagger}\left(u_t - W\right)\right)}{\sum g} \tag{2}$$

Where, g is the weight defined by the following Eq. (3)

$$g = \begin{cases} \exp(-n/\lambda) & for \quad \exp(-n/\lambda) > \varepsilon \\ 0 & for \quad \exp(-n/\lambda) \le \varepsilon \end{cases} \tag{3}$$

Where, n is the orders of the neurons determined based on the distances between the neurons and the target. It has a large value for the neuron that is near to the target, and has a small value for the one that is far from the target. The symbols λ and ε are values to define neuron numbers that can affect θ^{out} .

2.2 Evaluation function

For redundant manipulators, finding the inverse kinematics mapping for a given end effector position is hard because this is an ill-posed problem in the sense that many solutions are possible. For two adjacent target points an algorithm may select two completely different joint angle configurations when there are redundant degrees of freedom. In our system, we introduce two evaluation functions to eliminate the under-determination of the manipulator control. A joint angle configuration, which optimizes the evaluation functions in addition to reaching the target point, is selected. One is the function H_M which is defined as Eq. (4) for obtaining high manipulability; the other is the function H_O which is defined as Eq. (5) for obtaining obstacle-free poses. Evaluation function H is defined as the weighted sum of function H_M and function H_O as Eq. (6).

$$H_M = \sqrt{\det\left(J(\theta)J^T(\theta)\right)} \tag{4}$$

$$H_O = \sum_l \left(1.0 - \frac{D_0}{d + D_0}\right) \tag{5}$$

Where, d is the shortest distance from each link of the manipulator to the obstacles, and D_0 is the predefined value that affects the effective area of the potential.

$$H = \alpha_1 H_M + \alpha_2 H_O \tag{6}$$

ξ is the gradient vector of the function H . It is defined as the following equation.

$$\xi = \alpha_1 \xi_M + \alpha_2 \xi_O \tag{7}$$

Where, ξ_M is the gradient vector of the function H_M , and ξ_O is the gradient vector of the function H_O . α_1 and α_2 are weights, which are decided depending on the desirability between the manipulability and importance of obstacle-free. In the learning procedure, by updating ξ the maps become to output desirable θ^{out} that make the manipulator with high manipulability and with obstacle-free poses.

2.3 Algorithm of the self-organizing learning

Two maps are employed in our visuo-motor system. If they are learned separately, the outputs of them are different even for the same target in the workspace. This will result in that the manipulator moves incoherently when it is driven from one map's valid workspace to another map's valid workspace. In the learning algorithm, this problem has to be solved.

While a target position u_t is presented randomly in the workspace, the cameras see the target. Depending upon which camera (camera1 or/and camera2) can see the targets (camera3 can always see all the workspace), map1 or/and map2 will be learning. While both maps learn for a common visible target, the maps do not learn separately. In this case, only one self-organizing map determines the joint angles θ^{out} of the manipulator for the target. The manipulator is driven by using θ^{out} , and the cameras obtain different end effector positions v in map1 and map2 respectively. Then the map1 and map2 correct the parameters by using u_t, θ^{out} and v following the learning algorithm described below. Such a learning procedure results in that at the end of learning the neurons of two maps possess the similar value of θ^{out}, ξ and different W, J for common visible targets. Thereby, for those targets the outputs from either map1 or map2 will ensure the manipulator has the same pose. This means: while a target is given in the common visible space, the output from either map1 or map2 does not result in any change of the manipulator pose. In addition, the assignment of similar joint angles to adjacent target point is, in fact, one of the main features of the learning algorithm of the SOM. By the construction of a map, learning algorithm makes sure that adjacent target points always activate adjacent neurons in the network and the learning forces adjacent neurons to adapt their output towards similar values. At the end of the learning the output values will vary smoothly from a neuron to another neuron. These features bring about a continuous and smooth transformation from the input image spaces of targets to the output space of joint angles. According to such a learning algorithm, the maps guarantee smooth and coherent movements of the manipulator in the whole workspace.

For the *N*-th iteration, the maps amend the parameters as following learning procedure.

1) A target position u_t is given arbitrarily in the workspace. The target positions in the images obtained by the cameras, and then they are transferred to the self-organizing maps.

2) The maps sort the neurons in the order of distance between the target and W of the neurons. In other words, the neurons are sorted in the order as follows.

$$\| u_t - W_{i1} \| < \cdots < \| u_t - W_{i\lambda^n} \| \tag{8}$$

Where, λ^n is the number of the neurons updated in the *N*-th iteration.

3) One map outputs the joint angles θ_0^{out} using Eq. (9), and the manipulator is commanded by θ_0^{out} .

$$\theta_0^{out} = \frac{\sum_{i=1}^{\lambda^n} g\left(order_i^n, \lambda^n\right)\left(\theta_i^n + J_i^{n\dagger}\left(u_t - W_i^n\right)\right)}{\sum_{i=1}^{\lambda^n} g\left(order_i^n, \lambda^n\right)} \tag{9}$$

Where, $order_i^n$ is the order of i -th neuron decided in the step 2). At this time the end effector does not reach the target position for the error of neuron parameters, and the cameras obtain the position of the end effector v_0.

4) In order to reduce the error, the map changes θ_0^{out} using visual feedback information v_0 as Eq. (10) and the manipulator is commanded by θ_1^{out}. The cameras also obtain the position of the end effector v_1 .

$$\theta_1^{out} = \theta_0^{out} + \frac{\sum_{i=1}^{\lambda^n} g\left(order_i^n, \lambda^n\right)\left(\theta_i^n + J_i^{n\dagger}\left(u_t - v_0^n\right)\right)}{\sum_{i=1}^{\lambda^n} g\left(order_i^n, \lambda^n\right)} \tag{10}$$

5) The self-organizing maps update the parameters of each neuron by the method mentioned in the following sub-sections.

2.4 Update of the parameters

Each parameter of the neurons are updated using θ^{out}, u_t, v_0 and v_1.

(a) Updating W

W_i is updated by Eq.(11).

$$W_i^{n+1} = W_i^n + \varepsilon_W^n g\left(order_i^n, \lambda^n\right)\left(u_t^n - W_i^n\right) \tag{11}$$

Where, ε_W^n is the learning coefficient of W. It has a large value for early stages of learning process and has a small value for late stages. Updating W by Eq. (11), the neurons will be distributed all over the image spaces.

(b) Updating J

J_i is updated by Eq.(12).

$$J_i^{n+1} = J_i^n + \varepsilon_J^n g\left(order_i^n, \lambda^n\right)\Delta J_i^n \tag{12}$$

Where, ε_J^n is the learning coefficient of J, and it changes just like ε_W^n. ΔJ_i^n is determined by Widrow-Hoff's learning rule. The evaluation function E_J is

$$E_J = \frac{1}{2}\left\|\left(v_1 - v_0\right) - J_i^n\left(\theta_1^{out} - \theta_0^{out}\right)\right\|^2 \tag{13}$$

Then, ΔJ_i^n is determined by

$$\Delta J_i^n = -C_J \frac{\partial E_J}{\partial J_i^n} = \frac{\left(v_{01} - J_i^n \theta_{01}^{out}\right)\theta_{01}^{out\,T}}{\left\|\theta_{01}^{out}\right\|^2} \tag{14}$$

Where, $v_{01} = v_1 - v_0$, $\theta_{01}^{out} = \theta_1^{out} - \theta_0^{out}$, $C_J = 1/\left\|\theta_{01}^{out}\right\|^2$. By updating J using Eq. (12), the maps become to output more appropriate θ^{out}.

(c) Updating ξ

a) Updating ξ_M

$\xi_{M,i}$ is updated by Eq. (15).

$$\xi_{M,i}^{n+1} = \xi_{M,i}^n + \varepsilon_{\xi_M}^n g\left(order_i^n, \lambda^n\right)\Delta \xi_{M,i}^n \tag{15}$$

Where, $\varepsilon_{\xi_M}^n$ is the learning coefficient of ξ_M. $\Delta\xi_{M,i}^n$ is updated as ΔJ_i^n. The evaluation function is

$$E_{\xi_M} = \frac{1}{2}\left\|\left(H_{M,k} - H_{M,j}\right) - \xi_{M,i}^{n\ T}\left(\theta_k^{out} - \theta_j^{out}\right)\right\|^2 \tag{16}$$

Where, j is the number of the neuron that has the 1st order for the distance between it and the target, and k is the number of the neuron that has the 2nd order.

Therefore, $\Delta\xi_{M,i}^n$ becomes

$$\Delta\xi_{M,i}^n = -C_{\xi_M} \frac{\partial E_{\xi_M}}{\partial \xi_{M,i}^n} = \frac{\left(H_{M,jk} - \xi_{M,i}^{n\ T}\theta_{jk}^n\right)\theta_{jk}^{n\ T}}{\left\|\theta_{jk}^n\right\|^2} \tag{17}$$

Where, $H_{M,jk} = H_{M,k} - H_{M,j}$, $\theta_{jk}^n = \theta_k^n - \theta_j^n$, $C_{\xi_M} = 1/\left\|\theta_{jk}^{out}\right\|^2$.

b) Updating ξ_O

$\xi_{O,i}$ is updated by Eq.(18).

$$\xi_{O,i}^{n+1} = \xi_{O,i}^{n} + \varepsilon_{\xi_O}^{n} g\left(order_i^n, \lambda^n\right)\Delta\xi_{O,i}^{n} \tag{18}$$

Where, $\varepsilon_{\xi_O}^{n}$ is the learning coefficient of ξ_O. $\Delta\xi_{O,i}^{n}$ is updated as ΔJ_i^n. The evaluation function is

$$E_{\xi_O} = \frac{1}{2}\left\|\left(H_{O,1} - H_{O,0}\right) - \xi_{O,i}^{n\ T}\left(\theta_1^{out} - \theta_0^{out}\right)\right\|^2 \tag{19}$$

Where, $H_{O,0}$ is the potential value for v_0, and $H_{O,1}$ is the potential valve for v_1. Therefore, $\Delta\xi_{O,i}^{n}$ becomes

$$\Delta\xi_{O,i}^{n} = -C_{\xi_O}\frac{\partial E_{\xi_O}}{\partial \xi_{O,i}^{n}} = \frac{\left(H_{O,01} - \xi_{O,i}^{n\ T}\theta_{01}^{n}\right)\theta_{01}^{n\ T}}{\left\|\theta_{01}^{n}\right\|^2} \tag{20}$$

Where, $H_{O,01} = H_{O,1} - H_{O,0}$, $\theta_{01}^n = \theta_1^n - \theta_0^n$, $C_{\xi_O} = 1/\left\|\theta_{01}^{out}\right\|^2$.

By updating ξ_M and ξ_O using Eq. (11) and Eq. (18), the maps become to output θ^{out} that achieve high manipulability and obstacle-free poses.

(d) Updating θ

θ_i is updated by Eq. (21).

$$\theta_i^{n+1} = \theta_i^n + \varepsilon_\theta^n g\left(order_i^n, \lambda^n\right)\Delta\theta_i^n \tag{21}$$

Where, ε_θ^n is the learning coefficient of θ. $\Delta\theta_i^n$ becomes

$$\Delta\theta_i^n = \theta_i^{Desire} - \theta_i^n \tag{22}$$

$$\theta_i^{Desire} - \theta_0^{out} = J_i^{n\dagger}\left(W_i^n - v_0\right) + \left(I - J_i^{n\dagger}J_i^n\right)\xi_i^n k_p^n \tag{23}$$

Where, k_p^n is a positive coefficient that meets the condition where high manipulability is achieved and the manipulator takes an obstacle-free pose. It decreases with learning times. Therefore, $\Delta\theta_i^n$ becomes

$$\Delta\theta_i^n = \theta_0^{out} - \theta_i^n + J_i^{n\dagger}\left(W_i^n - v_0\right) + \left(I - J_i^{n\dagger}J_i^n\right)\xi_i^n k_p^n \tag{24}$$

By updating θ using Eq. (21), the maps become to output θ^{out} for the manipulator with small errors of the end effector, high manipulability and obstacle-free poses.

3. Accomplishment of Collision Avoidance

3.1 Our approach to achieve collision avoidance

Our system realizes collision avoidance in the image spaces by combining two self-organizing maps and a simplified path planning system. In the image spaces the path planning plans a collision-free path for the end effector, it based on the idea, "as long as projected path does not interfere with projected obstacles in the image spaces, the path is also not collision with the obstacles in the 3D spaces". Therefore, it is not required to reconstruct a 3D model of the workspace. Consequently collision avoidance problems in 3D spaces are transformed to problems in 2D spaces. The learned self-organizing maps determine the joint angles of the manipulator so that the end effector of the manipulator reaches a target point given in the image spaces arbitrarily, and also ensures the manipulator take obstacle free poses. The path planning system uses Laplace's potential method to plan a collision-free path for the end effector from an initial to a goal position without plagued into local minima. Once the path planning system plans a collision-free path for the end effector of the manipulator, the manipulator drives the end effector from the initial to the goal along the path. Since the self-organizing maps guarantee to make the manipulator with obstacle free poses in the learning, in the process of driving the end effector it is not necessary to check the collision between the links and obstacles. This feature makes our method different from many other collision avoidance systems. Thereby we say such a path planning as a simplified one.

As mentioned above our path planning is performed in image spaces, and the system plans collision-free projected paths for the end effector in respective image space. The points on the projected paths are extracted and passed to the self-organizing maps as inputs. We have to match two different points in the images of a single point in the 3D space. Correspondence problem between the images is occurred. Some researchers use epipolar constraint to settle this problem[10)]. In our previous study, we settle the correspondence problem in a very simple way; the cameras were set with the same height. In this study, we proposed to use a data base instead of any calibrations of the camera system to solve the problem. The data base is composed of the example targets which constructed automatically in the self-organizing maps learning, so it does not expend an extra computation cost.

3.2 Path planning by laplace potential method

In this study, we utilize a solution of Laplace’s equation to generate a path for the end effector. In the following parts we explain a procedure for path planning using Laplace’s equation[11)]. The procedure is divided into three steps:

1) Calculating a numerical solution of the Laplace’s equation using the Gauss-Seidel iterative method. Laplace’s differential equation Eq. (25) can be replaced by a simple discrete formula as Eq. (26)

$$\frac{\partial^2\Phi}{\partial x^2}+\frac{\partial^2\Phi}{\partial y^2}=0 \tag{25}$$

$$U_{i,j}=\frac{1}{4}\left(U_{i+1,j}+U_{i-1,j}+U_{i,j+1}+U_{i,j-1}\right) \tag{26}$$

Equation (26) illustrates that a potential value on a mesh point is the mean of the adjacent points. In order to satisfy the relationship of Eq. (26) over the whole region, Gauss-Seidel iterative method is used with boundary conditions as Eq. (27)

$$\Phi^{n}_{i,j}=\frac{1}{4}\left(\Phi^{(n)}_{i,j-1}+\Phi^{(n-1)}_{i,j+1}+\Phi^{(n)}_{i-1,j}+\Phi^{(n-1)}_{i+1,j}\right) \tag{27}$$

Where, $\Phi^{(n)}_{i,j}$ is a numerical solution on the mesh point (i,j) obtained from the *n*-th iterative of Eq. (27).

2) In order to obtain a continuous potential field from discrete one, we interpolate the discrete potential field by the method of weighted average as follows. A continuous potential field $\Phi(x,y)_{i,j}$ at a point $(x,y|i\le x<i+1,j\le y<j+1)$ is obtained by

$$
\begin{aligned}
\Phi(x, y)_{i,j} &= \Phi_{i,j} \cdot (i+1-x) \cdot (j+1-y) \\
&+ \Phi_{i+1,j} \cdot (x-i) \cdot (j+1-y) \\
&+ \Phi_{i,j+1} \cdot (i+1-x) \cdot (y-j) \\
&+ \Phi_{i+1.j+1} \cdot (x-i)(y-j)
\end{aligned}
\tag{28}
$$

Where, the subscript i and j are coordinate of the nearest point to the origin of the square region including a point (i, j) and $\Phi_{i,j}$ is the potential value on the mesh point obtained through the Gauss-Seidel method.

3) Searching a collision-free path in the calculated potential field, the path planning system traces the potential valley from the initial point to the goal point according to the method of steepest descent.

$$x^{(n+1)} = x^{(n)} - \frac{s}{l} \cdot \frac{\partial \Phi}{\partial x} \tag{29}$$

$$y^{(n+1)} = y^{(n)} - \frac{s}{l} \cdot \frac{\partial \Phi}{\partial y} \tag{30}$$

Where, $l = \sqrt{\left(\frac{\partial \Phi}{\partial x}\right)^2 + \left(\frac{\partial \Phi}{\partial y}\right)^2}$, $x^{(n)}$ and $y^{(n)}$ are the coordinates of the n-th point on the collision avoidance path, and s is the moving distance of one step.

4. Simulation Results

The proposed method has been successfully validated in simulations. We assume a 4-DOF redundant manipulator moving in a 3 dimensional workspace with an obstacle. Three CCD cameras, arranged as **Fig.1**, see the workspace. Each camera has 512×512 Pixel resolution.

4.1 Coordination of the visuo-motor system

In the simulations, the self-organizing maps were learned with 15,000 targets given in the workspace and each map consisted of 240 neurons. The time required to complete the learning was about 5 minutes (A PC with 3.0GHz Pentium4 was used.).

After learning, 80 targets were given randomly to test the positioning accuracy of the end effector and to confirm the poses of the manipulator. In view of the visibility of the targets, the map1 or map2 output the joint angles θ^{out} of the manipulator for them. **Figure 3** and **Figure 4** show the positions of the end effector and poses of the manipulator seen in each camera. **Figure 3** shows the case that the targets were visible in two maps, and both map1 and map2 can output joint angles for them. **Figure 4** shows the case that the targets were occluded by the obstacle or cross the obstacle in one front camera image. In this case only map1 or map2 is available to output the joint angles of the manipulator for them. In the presented figures, we can see that the manipulator takes obstacle-free poses and the end effector reaches the targets precisely. The average positioning error of the end effector is 0.22 pixels. It is approximately 0.7 percent of the linear dimension of the workspace.

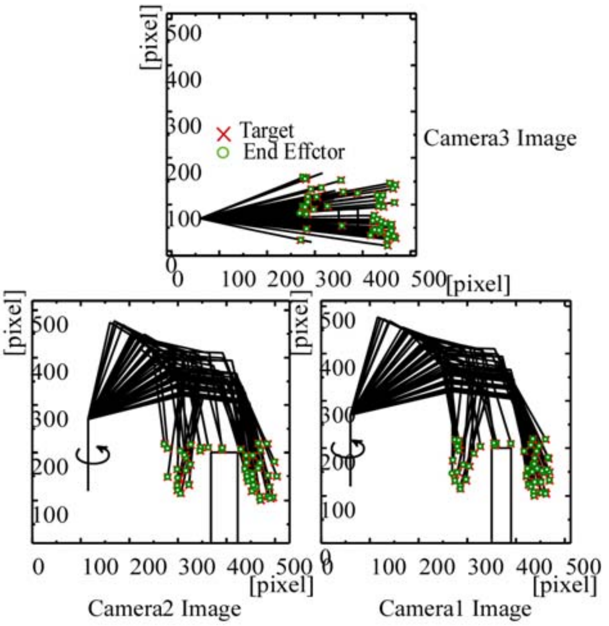
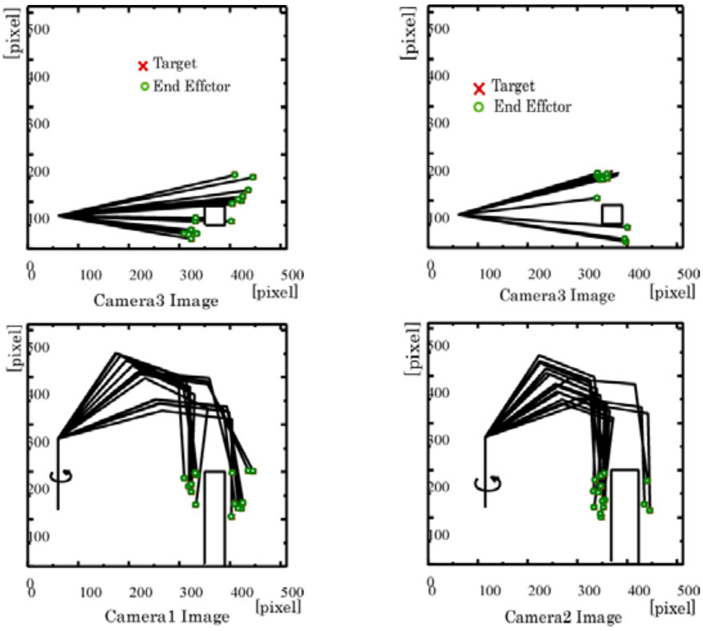


Fig. 3 Targets can be observed in two maps.



a) Using map1 b) Using map2

Fig. 4 Targets can be observed in only one map.

4.2 Accomplishments of collision avoidance

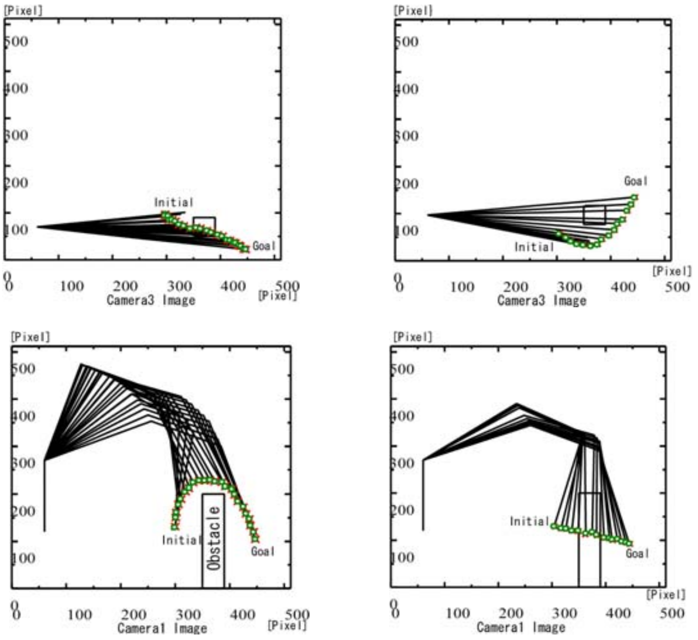
In the learning stages of the maps, 15,000 targets were given arbitrarily in the workspace. These targets were also used as a data base while planning a path. The path was searched in this data base space. Thereby, corresponding problems were solved simply without any calibrations of the camera system.

In our previous researches[9)], the path planning planed a collision-free path for the end effector which avoids obstacles in both images. In fact, as the projected path in one image is collision-free from the obstacles, the path in the 3D space is also free from the obstacles. Therefore, in this paper the path planning system planed a collision-free path for the end effector in one side image and calculated a shortest projected path in another image.

4.2.1 Collision avoidance completed by one map

When the initial position of the end effector and a goal position can be observed in one front camera, the collision avoidance of the manipulator can be performed by using one map. Here, some simulation results were presented to show the process of collision avoidance in this case.

Figure 5 shows the process of collision avoidance seen in the camera1 and camera3 respectively. The initial position of the end effector was obtained in camera3 with (300,130) and in camera1 in (300,100). A goal position was projected in the camera3 and camera1 with ordination (450, 90) and (450, 20). The path planning system planed a collision-free path in the camera1 image space using Laplace potential method. In the camera3 image space, the shortest path was calculated. Then, the system searched mid-targets along the planned path in the data base (example targets) space and passed them as the input of the self-organizing map. Map1 was chosen to output joint angles θ^{out} in this case. The manipulator was commanded by θ^{out}, and it reached the goal without collision. As shown in **Fig. 5 (a)** a collision-free path was planned in the image of the camera1. The path planning system pursued the shortest path in the camera3 image space. Alternatively, the system also can plan a collision-free path in the image of the camera3, and calculate the shortest path in camera1. Accomplishment of collision avoidance was shown in **Fig.5 (b)**.



(a) (b)

Fig. 5 Collision avoidance completed by one map.

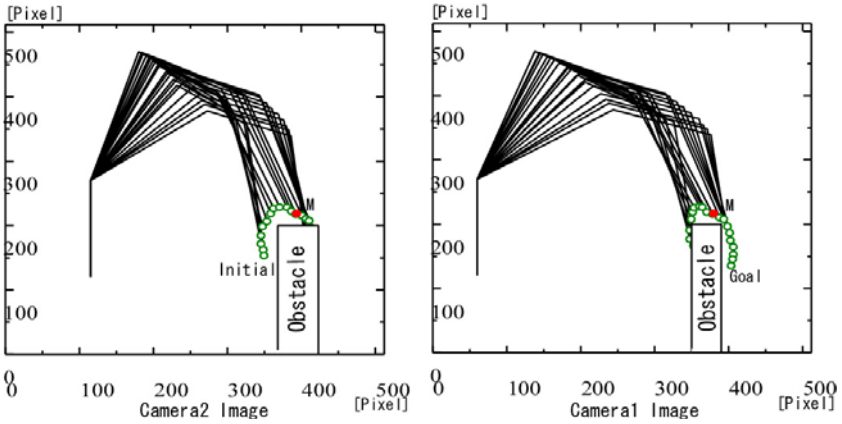


Fig. 6 Collision avoidance completed by the cooperation of two maps.

4.2.2 Collision avoidance completed by the cooperation of two maps

When the initial position of the end effector and a goal position can not be observed in one front camera at one time, the collision avoidance of the manipulator is performed by the cooperation of two maps. **Figure 6** shows such a case that there exists changeover of the self-organizing maps in collision avoidance. Since the initial point and the goal point were visible only in one side front camera, a middle point M was set as the changeover point of two maps. The path from the initial point to the middle point was planned in camera2 and 3 image spaces, and then the manipulator was driven by the outputs θ^{out} of map2. The path from the middle point M to the goal was planned in camera1 and camera3 images, and map1 was used to output the joint angles θ^{out} of the manipulator. In section 2.3, it has been discussed that the learning algorithm guarantees the manipulator move smoothly and efficiently in the whole workspace no matter which map outputs joint angles θ^{out} . As shown in the **Fig. 6**, although at the middle point M the maps were switched to output joint angles of the manipulator, the manipulator proceeded smoothly and coherently without sudden pose changes.

5. Conclusion

The implementation of collision avoidance for a visuo-motor system, which is composed of a redundant manipulator, three cameras and two self-organizing maps, was demonstrated. The achievement of collision avoidance is based on a simplified path planning and motion control by the cooperation of the self-organizing maps. In contrast to previous studies, path planning system plans a collision-free path of the end effector in much simpler and more efficient way. In addition, by using a data base, which was constructed in the self-organizing learning stage, the system solved the corresponding problem without any camera system calibration. The proposed method was proved effectively to achieve collision avoidance for the visuo-motor system by the simulations. It is expected to be verified in a real experiment in our future works.

References

1) J. Barraquand and J. C. Latombe, Robot motion planning: A distributed representation approach, Int.J.of Robotics Research, Vol.10, No.6, pp.628-649, (1991).

2) T. Lozano-Perez, Spatial planning: A configuration space approach, IEEE Trans. On Computers, Vol.32, No.2, pp.108-120, (1983).

3) T. Kohonen, Self-organizing maps and associative memory, Second Edition, Springer in Information Sciences, Vol.8, pp.43-48, (1988).

4) R. Glasius, A. Komoda and S. Gielen, A biologically inspired neural net for trajectory formation and obstacle avoidance, Biological Cybernet, Vol.84, pp.511-520, (1996).

5) S.X. Yang and M. Meng, Neural network approaches to dynamic collision free trajectory generation, IEEE Trans. Systems Man Cybernet, Vol.31, No.3, pp.302-318, (2001).

6) M. Zeller, R. Sharma and K. Schulten, Motion planning of a pneumatic robot using a neural network, IEEE Control Systems Mag, 17, pp. 89-98, (1997).

7) Khatib, S. Quinlan, and I. D. William, Robot planning and control, Robotics and Autonomous System, Vol. 21, pp. 249–261, (1997).

8) C.I. Connolly, J.B.Burns and R. Weiss, Path planning using laplace’s equation, In. Proc. ICRA, pp. 2102-2106, (1999).

9) M. Han, N. Okada and E. Kondo, Collision avoidance for a visuo-motor system using a self-organizing map in a 3D space, 6th Japan-France Congress on Mechatronics, pp.495-500, (2003).

10) K. Hosoda, K. Sakamoto and M. Asada, Trajectory generation for obstacle avoidance of uncalibrated stereo visual servoing without 3D reconstruction, J. of the Robotics Society of Japan, Vol.15, No.2, pp290-295, (1997), [in Japanese].

11) K. Sato, Global motion planning using a laplacian potential field, J. of the Robotics Society of Japan, Vol.11, No.5, pp702-709, (1993), [in Japanese].