OBSTACLE AVOIDANCE OF REDUNDANT DISCRETELY-ACTUATED MANIPULATORS USING WORKSPACE DENSITY FUNCTIONS

Eric Lanteigne, Amor Jnifene
Department of Mechanical Engineering, Royal Military College of Canada
E-mail: Eric.Lanteigne@rmc.ca; Amor.Jnifene@rmc.ca

Received October 2009, Accepted November 2009
No. 09-CSME-44, E.I.C. Accession 3130

ABSTRACT

This paper presents an obstacle avoidance method for discretely-actuated planar manipulators with a large number of serially connected modules. The method uses a direct combination of the manipulator workspace density functions and obstacle density maps to solve the inverse kinematics of the manipulator, thus avoiding the geometrical modelling of obstacles and the computation of the nearest point distance between the manipulator and the obstacles. Simulations on a seventeen-module planar manipulator operating in a workspace with circular obstacles have demonstrated the viability of the obstacle avoidance algorithm. This paper presents a detailled description of the proposed method and a summary of the results.

Keywords: redundant manipulator; density functions; obstacle avoidance.

FONCTIONS DE DENSITÉ DE L'ESPACE ATTEIGNABLE ET LEURS APPLICATIONS À L'ÉVITEMENT D'OBSTACLE DES MANIPULATEURS REDONDANTS À ACTIONNEURS DISCRETS

RÉSUMÉ

Ce papier présente un algorithme d'évitement d'obstacle pour un manipulateur hyper-redondant actionné par des actuateurs discrets. La procédure d'évitement se caractérise par la combinaison directe des fonctions de densité de l'espace atteignable du manipulateur et des cartes de densité d'obstacles dans la résolution de la cinématique inverse du manipulateur, de tel façon que la modélisation géométrique des obstacles et du calcul du point le plus proche entre le manipulateur et l'obstacle sont évités. Des simulations sur un manipulateur comprenant dix-sept modules opérant dans un champ d'obstacles circulaires ont démontrées la viabilité de l'algorithme. L'article présente une description détaillée de la méthode proposée ainsi qu'un résumé des résultats.

Mots-clés : manipulateur redondant; densité de l'espace atteignable; évitement d'obstacle.

1. INTRODUCTION

Hyper-redundant manipulators are highly dexterous robotic devices composed of numerous serially connected modules and actuators. The large number of degrees of freedom allows the manipulator to avoid obstacles, navigate through narrow passages, and overcome many of the problems associated with conventional robotic devices; however, the high mobility increases the complexity of both the inverse kinematics and the obstacle avoidance problem. The challenge is determining a suitable solution in a reasonable amount of time.

There are various different approaches to both of these problems in the literature. Resolved-motion rate methods based on the pseudoinverse have been proposed under various formulations [1–4]. The solution yields a minimum norm solution to the primary motion task and a homogeneous solution which can be applied to a number of subtasks, depending on the degree of redundancy. Typical subtasks include singularity avoidance, joint motion restrictions, manipulability, and obstacle avoidance.

Early obstacle avoidance strategies involved maximizing the distance between the manipulator and the closest obstacle [5, 6]. The disadvantages of these methods are that obstacles must first be modelled using analytical equations, and solutions can converge to a local minimum if the manipulator approaches a concave obstacle. By simplifying the obstacle shapes as convex polyhedrons, spheres or ellipsoids [7–9], the possibility of yielding local minimum solutions can be avoided, but the complexities of the working environment may be lost.

The artificial potential field method developed for mobile robots [10, 11] was adapted to redundant manipulators [12, 13] to deal with both the inverse kinematics and the obstacle avoidance problem while eliminating the need to mathematically model the obstacles. This method is based on a field of forces interacting with the manipulator structure. The obstacles are modelled via a mesh of points that exert repulsive forces on the nodes of the manipulator [14], and the target coordinate is modelled as an attractive force to the end-effector.

Although these methods are compact and concise, they still rely on the Jacobian and its pseudoinverse which has been demonstrated to be computationally inefficient for manipulators with a large number of degrees of freedom.

The backbone curve method was developed to limit the number of parameters in the optimization problem. The inverse kinematics problem is reduced to finding the mode shape functions describing a continuous or piecewise continuous curve fixed to the manipulator structure or backbone [15, 16]. The number of mode shape functions is set according to the number of obstacles in the workspace. The shape functions are sequentially separated into ‘‘free’’ sections and obstacle avoiding sections which traverse ‘‘virtual tunnels’’ derived from the obstacle field [17]. The generation of the tunnels is non-intuitive for complex obstacle fields and not easily adaptable to time-varying obstacle fields.

Workspace density functions approximate the number of forward kinematic solutions of discretely-actuated hyper-redundant manipulators in the workspace. The density in the neighborhood of a particular point is an indication of how accurately a manipulator can reach that point [18]. These functions have been used in conjunction with a breadth-first search algorithm to quickly solve the inverse kinematics problem [19]. The method operates sequentially from the base of the manipulator to the end-effector module. For each module, the state resulting in the highest density value at the end-effector coordinate is selected as the optimal state and used in the computation of the subsequent module states in the manipulator chain. This method has successfully solved the inverse kinematics of manipulators with a very large number of degrees of freedom.

In this paper we present an obstacle avoidance strategy for the computation of the inverse kinematics of serial hyper-redundant manipulators using the workspace density function approach. The method extracts local obstacle positions from a two-dimensional Cartesian grid representing the working environment and applies this information to the breadth-first search inverse kinematics algorithm. These grids have been used extensively in mobile robot motion planning algorithms as a means to model obstacles from range sensor readings [20, 21]. The proposed method is tested using a simulated discretely-actuated manipulator with seventeen modules and an obstacle grid similar to those generated by actual sensors.

The following sections describe the generation of the workspace density functions, the breadth-first search inverse kinematics algorithm, and the obstacle avoidance strategy. The simulation results are presented at the end of the paper.

2. WORKSPACE DENSITY FUNCTIONS

Workspace density functions quantify a manipulator's ability to reach a given target coordinate throughout the workspace. That is, areas which have a high density can be reached with a high accuracy, and those with a low density can be reached less accurately [22]. For a hyper-redundant manipulator, the density at a particular point in space is also an indication of the amount of different configurations capable of reaching that point.

The density function ρ of a single discretely-actuated module is obtained by dividing the workspace into small volume elements and recording the number of times the module effector is present in each element. The generation of the density function is irrespective of the module structure or actuation method; modules may be composed of parallel platforms, single actuator revolute or prismatic joints, or any combination of the above. If the actuator is non-discrete, for example an electric motor, its range of motion must first be discretized before its density function can be constructed.

The accuracy of the workspace density functions depends on the level discretization of both the actuators and the manipulator workspace itself. Reducing the size of the volume elements increases the accuracy of the approximation, but also the size of the functions and the computation time required to build them. Therefore a compromise between speed and accuracy must be made.

The workspace density function of a single or several modules in a manipulator can be obtained manually by sweeping the actuators though all of their states and assigning density values in the appropriate cells of the discretized density function. This become inefficient for manipulators with a large number of modules and actuators as the number of permutations grows exponentially. As a result, the method described in [23] was developed to compute the density function of a manipulator with any number of modules from the sequential convolution of the density functions of each of its modules.

In Fig. 1, F_A and F_B are moving frames attached to the distal end of two modules described by the transformation matrices H_A and H_B. The transformation matrix H_A describes the position and orientation of F_A with respect to the fixed reference frame and H_B describes the position and orientation of F_B with respect to F_A. If the coordinates of F_B with respect to the fixed reference frame are described by $H_t = H_A H_B$, the transformation matrix H_B can be written as:

$$H_B = H_A^{-1} H_t \tag{1}$$

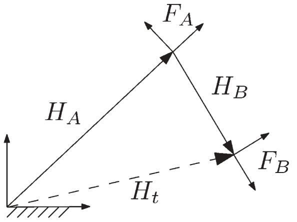


Fig. 1. Concatenation of two homogeneous transformations.

The density function $\rho_B(H_t)$ of F_B with respect to the fixed frame can be obtained from the convolution of the density functions $\rho_A(H_A)$ and $\rho_B(H_B)$ of the two modules [18, 23].

$$\rho_B(H_t) = (\rho_A * \rho_B)(H_t) = \int_{SE(D)} \rho_A(H_A)\rho_B\left(H_A^{-1}H_t\right)d\mu(H_A) \tag{2}$$

where $SE(D)$ denotes the group of rigid body motions of N-dimensional space and $d\mu(H_A)$ is the integration measure on the motion group. In the case of a planar manipulator $D = 2$, and $d\mu = dxdyd\theta$.

Given the transformation matrices H_A and H_t

$$H_A = \begin{bmatrix} \cos\alpha & \sin\alpha & \xi \\ -\sin\alpha & \cos\alpha & \eta \\ 0 & 0 & 1 \end{bmatrix} \quad \text{and} \quad H_t = \begin{bmatrix} \cos\theta & \sin\theta & x \\ -\sin\theta & \cos\theta & y \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

the convolution (2) can be written as

$$\begin{aligned}(\rho_A * \rho_B)(x, y, \theta) = \iiint_{x,\, y,\, \theta} & \rho_A(\xi, \eta, \alpha)\rho_B((x-\xi)\cos\alpha + (y-\eta)\sin\alpha, \\ & -(x-\xi)\sin\alpha + (y-\eta)\cos\alpha,\ \theta-\alpha)dxdyd\theta \end{aligned} \tag{4}$$

where (ξ, η, α) are the coordinates of the moving frame F_A, and (x, y, θ) are the coordinates in the fixed reference frame.

In the numerical simulations, the workspace density functions are stored as 3-dimensional piece-wise constant histograms representing the x and y position, and the θ orientation of the end-effector as shown in Fig. 2a. The convolution (4) can then approximated by a Riemann-Stieltjes sum [18]:

$$\begin{aligned}(\rho_A * \rho_B)(x, y, \theta) \simeq \sum_{l=1}^{L}\sum_{m=1}^{M}\sum_{n=1}^{N} & \rho_A(\xi_l, \eta_m, \alpha_n)\rho_B(x-\xi_l)\cos\alpha + (y-\eta_m)\sin\alpha, \\ & -(x-\xi_l)\sin\alpha + (y-\eta_m)\cos\alpha,\ (\theta-\alpha_n)\mathrm{mod}2\pi)\Delta x\Delta y\Delta\theta, \end{aligned} \tag{5}$$

where L, M, and N represent the size of the workspace density function histogram along x, y, and θ respectively.

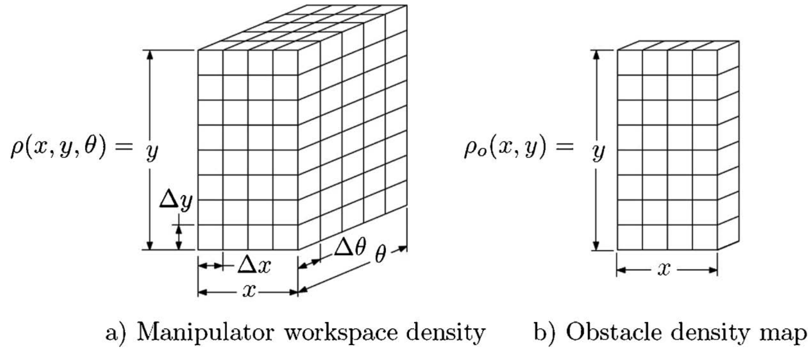


Fig. 2. Workspace density functions.

The discretization level is determined by the size of the volume elements (represented by Δx, Δy, and $\Delta\theta$ in Fig. 2a), and is dependent on the desired accuracy and the available memory. In the present simulations, the discretization of the workspace was chosen to be 10% of the length of one unit-length module, or $\Delta x = \Delta y = 0.1$. The discretization of the orientational component was chosen to be $\Delta\theta = 6^\circ$ to obtain a reasonably accurate angular resolution of each actuator state.

The workspace density function of a manipulator with P modules can then be obtained from the sequential convolution of the density functions of each module in the manipulator, from the base module density $\rho_{0,1}$ to the end-effector module density $\rho_{P-1,P}$.

$$\rho_{0,\,P}(x,\,y,\,\theta) = \left(\rho_{0,1} * \rho_{1,2} * \cdots * \rho_{P-1,P}\right)(x,\,y,\,\theta) \tag{6}$$

The inverse kinematics method described in the following section uses the workspace density functions $\rho_{1,P}$ to $\rho_{P-1,P}$ of each manipulator section, as opposed to the density function of the entire manipulator $\rho_{0,P}$. If the all the modules in the manipulator are identical, the density functions $\rho_{0,1} = \rho_{1,2} = \ldots = \rho_{P-1,P}$ are equal, and the convolution products of each manipulator section can be found using the following iterative method:

$$\begin{aligned}
\rho_{P-1,P}(x,\,y,\,\theta) &= \rho_{0,1}\\
\rho_{P-2,P}(x,\,y,\,\theta) &= \left(\rho_{0,1} * \rho_{P-1,P}\right)(x,\,y,\,\theta) = \left(\rho_{0,1} * \rho_{0,1}\right)(x,\,y,\,\theta) = \rho_{0,2}(x,\,y,\,\theta)\\
\rho_{P-3,P}(x,\,y,\,\theta) &= \left(\rho_{0,1} * \rho_{P-2,P}\right)(x,\,y,\,\theta) = \left(\rho_{0,1} * \rho_{0,2}\right)(x,\,y,\,\theta) = \rho_{0,3}(x,\,y,\,\theta)\\
&\vdots\\
\rho_{1,P}(x,\,y,\,\theta) &= \left(\rho_{0,1} * \rho_{2,P}\right)(x,\,y,\,\theta) = \left(\rho_{0,1} * \rho_{0,P-2}\right)(x,\,y,\,\theta) = \rho_{0,P-1}(x,\,y,\,\theta)
\end{aligned} \tag{7}$$

where the density $\rho_{0,1}$ is constructed manually. The convolutions in (6) and (7) become computationally unfeasible for manipulators with a large number of modules since the density function size increases with the number of modules. Several methods, such as the conversion of the density functions to Fourier transforms [23] and the use of a diffusion-type equation [24], have been developed to overcome this problem.

3. BREADTH-FIRST SEARCH INVERSE KINEMATICS

The breadth-first search inverse kinematics algorithm involves determining and fixing the state of each module sequentially, starting at the base module of the manipulator, while maximizing the density of the remaining modules at the desired coordinates for each step [18]. Assuming all the modules in the manipulator have identical structures, the manipulator configuration can be found using the following steps. Given the desired target coordinate (x_d, y_d, θ_d), the density value

$$\rho_{1,P}\left(H_{1,P}^{S}\right) \tag{8}$$

at $H_{1,P}^{S} = \left(H_1^S\right)^{-1} H_d$ is found for each state S of the first module, where H_1^S is the transformation matrix of one state of the first module at the base of the manipulator, and H_d is the transformation matrix of the target coordinates. The density value obtained from the workspace density function $\rho_{1,P}$ at the coordinates given by $H_{1,P}^{S}$ represents the probability of finding a suitable manipulator configuration for the remaining $P-1$ modules if the fixed reference frame is moved to the position and orientation of H_1^S. The state of the first module which results in the highest density value is selected since it represents the highest probability of finding a suitable manipulator configuration starting from this point.

The state of the second module can be found by locating the density value

$$\rho_{2,P}\left(H_{2,P}^{S}\right) \tag{9}$$

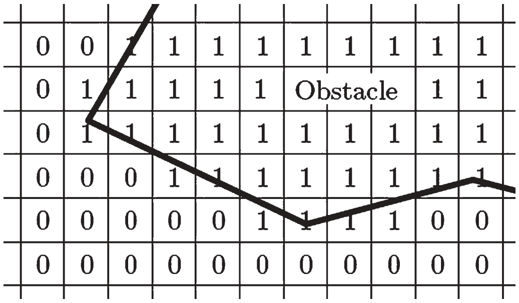
at $H_{2,P}^{S} = \left(H_1^C H_2^S\right)^{-1} H_d$ for each state S of the second module, where the transformation matrix H_1^C is the state of the first module found from (8) and H_2^S is the transformation matrix of one state of the second module. The configuration $H_2^C = H_2^S$ which results in the highest density value is selected. Generalizing (9), the state of the ith module can be found by locating the maximum value of

$$\rho_{i,P}\left(\left(H_1^C H_2^C \dots H_{i-1}^C H_i^S\right)^{-1} H_d\right) \tag{10}$$

from each state S of the ith module. The process is repeated for the first $P-1$ modules. The state of the Pth module is simply determined by minimizing the distance between the end-effector and the target coordinates, and ensuring no collisions ensue. Using this method, the computation time of the inverse kinematics is proportional to the number of modules in the manipulator.

4. OBSTACLE DENSITY MAPS

Obstacles can be represented in a two-dimensional Cartesian grid using the data generated from ultrasonic and/or infrared range sensors aboard a mobile robot or manipulator. There are several different methods for producing these maps in the literature. An example is the vector field histogram method which uses the onboard ultrasonic range sensors of a mobile robot to produce a map of the working environment [10]. From the Cartesian grid, a polar histogram function quantifying the obstacle density around the robot is generated, and the direction with the lowest obstacle density is chosen [20, 21].



0	0	1	1	1	1	1	1	1	1	1
0	1	1	1	1	1	Obstacle			1	1
0	1	1	1	1	1	1	1	1	1	1
0	0	0	1	1	1	1	1	1	1	1
0	0	0	0	0	1	1	1	1	0	0
0	0	0	0	0	0	0	0	0	0	0

Fig. 3. Obstacle density map ρ_o.

The structure of the two-dimensional obstacle grid used in the present simulations is shown in Fig. 2b, and the obstacle representation within this grid is shown in Fig. 3. The cells containing an obstacle have a value of one and cells in free areas have a value of zero.

5. OBSTACLE AVOIDANCE USING WORKSPACE DENSITY FUNCTIONS

The proposed obstacle avoidance method combines the manipulator workspace density functions and the obstacle density maps to solve the inverse kinematics problem. The computed density value of each state of the current module is multiplied by an obstacle density factor representing the density of obstacles surrounding the module. If, for a particular state, the module is located at or close to an obstacle, the obstacle density factor reduces the density value for that state allowing a lower-density but obstacle-free state to be chosen.

The searched area for one state of the current module is shown in Fig. 4. The three sectors denoted by the letters *A* in black, *B* in gray, and *C* in light gray represent areas of high, medium, and low concern respectively. The *A* sector is located at the current module state in such a way that obstacles in this region would result in a definite collision. The *B* sector is located in the immediate vicinity of the current module so that a collision with the next module might occur if an obstacle is present. Finally, the *C* sector is located at a distance of two module lengths from the current module.

The algorithm queries a maximum of ten cells along the length of each module from the obstacle density map. The location and the number of queried cells along the length of the module are determined using the forward kinematics and the geometry of the module. This number reflects the ratio between the length of the module (for the present simulation, the width

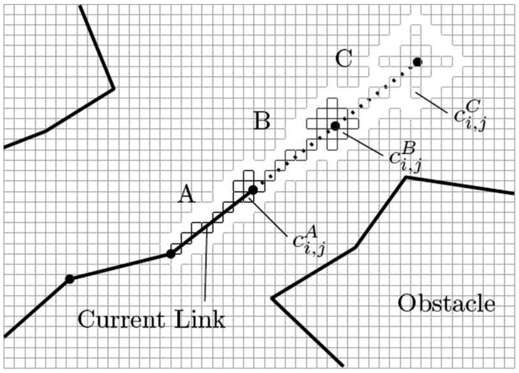


Fig. 4. Active cells of the obstacle density map for one module state.

is assumed to be negligible) and the size of the discretization of the workspace. Between four and sixteen cells in cross-shaped patterns located at the joints of distant modules are also queried for obstacles. The number of cells queried at the each module joint is determined based on the area swept by the current module and the distance of each module to the current joint.

The forward kinematics are computed for each state of the current module and the cell values $c_{i,j}$ of each sector A, B, and C of each state are determined from the obstacle density map. The obstacle strength k_o^S is then computed for each state from the sum of the cells $c_{i,j}$ of each sector.

Three variations of the above method were tested in the present simulation. The algorithms **A**, **AB**, and **ABC** represent the active cells used to perform the obstacle avoidance. For example, only the cells in sector A are searched for obstacles in algorithm **A**. The obstacle strengths for each variant are given by:

$$\mathbf{A} \quad : \quad k_o^S = p_1 \sum_A c_{i,j}^A \tag{11a}$$

$$\mathbf{AB} \quad : \quad k_o^S = p_1 \sum_A c_{i,j}^A + p_2 \sum_B c_{i,j}^B \tag{11b}$$

$$\mathbf{ABC} \quad : \quad k_o^S = p_1 \sum_A c_{i,j}^A + p_2 \sum_B c_{i,j}^B + p_3 \sum_C c_{i,j}^C \tag{11c}$$

The values of the weights p_i are determined such that the effect of obstacles diminish as the distance from the current module increases. In the present simulation, the values of the weights are set to $(p_1, p_2, p_3) = (10^{10}, 10^1, 10^0)$. The large magnitude of p_1 ensures that the inverse of the obstacle strength, given by the obstacle density factor K_o^S, is reduced to zero in the presence of obstacles near the current module state region of sector A.

$$K_o^S = \begin{cases} 1/k_o^S & \text{if } k_0^S > 0 \\ 1 & \text{if } k_o^S = 0 \end{cases} \tag{12}$$

The obstacle density factor K_o^S is computed for each state of the current module and then multiplied by the density value of each respective state. Combining (10) and (12), the state of the ith module resulting in the highest value of

$$K_o^S \cdot \rho_{i,P}\Big(\left(H_1^C H_2^C \cdots H_{i-1}^C H_i^S\right)^{-1} H_d\Big). \tag{13}$$

is selected, where H_i^S is the transformation matrix of one state S of the ith module. Since a collision at the current module state would result in a value of zero in (13), the highest value corresponds to the collision-free module state with the greatest possibility of directing the manipulator toward the target coordinates.

6. NUMERICAL SIMULATIONS

Numerical simulations were performed on a planar hyper-redundant manipulator with seventeen unit-length modules. Each discretely-actuated module is capable of reaching five equally-spaced states, as shown in Fig. 5a. In Fig. 5b, the search patterns corresponding to each of the five states are superimposed to illustrate the searched area of one module. In this paper, the obstacle density map is generated using MATLAB and has a level of discretization identical to the manipulator workspace density functions to facilitate the identification of the active cells.

The performance of the proposed method was evaluated by computing the inverse kinematics of approximately eight thousand target coordinates, bounded by $0 \leq x_d \leq 18, 0 \leq y_d \leq 12,$ and $-\pi \leq \theta_d \leq \pi$ in an obstacle field consisting of uniformly-distributed circular objects. Since the states of the discretely-actuated revolute joint of Fig. 5a are symmetric about the x-axis, the manipulator density functions are also symmetric about the x-axis. As a result, only positive y-axis target coordinates were used to limit the redundancy of the data.

The computation time, mean end-effector error, and percent convergence were recorded for each simulation, where the convergence is defined as the ability to find a collision-free manipulator configuration capable of reaching the target coordinate. The end-effector error

$$e = \sqrt{(x_d - x)^2 + (y_d - y)^2 + L^2(\theta_d - \theta)^2} \tag{15}$$

is expressed in terms of the unit length of one module. The length scale L is used to homogenize the positional and orientational units [25]. For all simulations, the value $L^2 = 0.1$ was used. Other quantities of L may be used but this value gave a good balance between position and orientation errors.

The three algorithm variants described in Section 5 were tested on a Pentium IV 3.00GHz PC, and the results are given in Table 1. It was found that the convergence was reduced when the amount of preemptive collision checking was increased. Increasing the preemptive collision checking also had the effect of producing manipulator configurations with end-effector coordinates closer to the target coordinates, as shown by the reduced mean end-effector error of algorithm **ABC**. It is hypothesized that feasible module states near an obstacle are discarded by the more preemptive variants, such as the **ABC** algorithm, if there are other obstacles in the vicinity. For the obstacle scenario used in the present simulations, the differences between the performance of each variant are small, indicating that preemptive collision checking does not offer a significant advantage compared to the local collision checking of algorithm **A**.

The target coordinates were also subjected to the breadth-first search algorithm described in Section 3 in an obstacle-free workspace. The comparison revealed that the additional

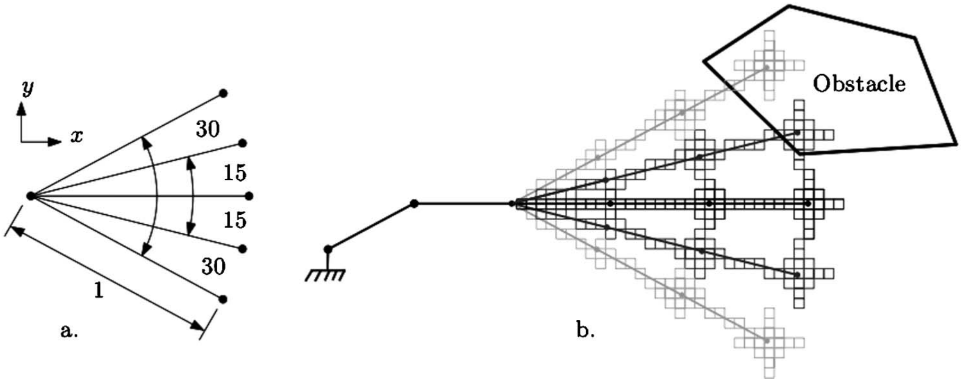


Fig. 5. Active cell search of one module.

Table 1. Numerical simulation results.

Algorithm	Convergence [%]	Mean computation time [s]	Mean end-effector error
A	33.8	0.0928	0.182
AB	32.8	0.0933	0.168
ABC	32.1	0.0936	0.167

computation time required to evaluate the active cells and the obstacle density factor was approximately 10ms, and that the mean end-effector error was approximately half that reported in Table 1. The comparison also shed light on the low convergence of the proposed method. The convergence of (10) in an obstacle-free workspace was only five percentage points higher than the convergence of algorithm **A** of (13) in the obstacle field of Fig. 6. The low convergence of both (10) and (13) indicates that many target coordinates, such as those located on the fridge on the workspace, were themselves unreachable and should be removed from the simulation. An example of typical manipulator configurations obtained using the **ABC** algorithm is shown in Fig. 6.

7. CONCLUSIONS

This paper presents an obstacle avoidance strategy for hyper-redundant manipulators using workspace density functions. Simulations on an obstacle-laden workspace demonstrated that the proposed method effectively solves the obstacle avoidance problem of a manipulator with a large number of modules. It is important to note that the sequential nature of the proposed algorithm could potentially lead to problems in certain scenarios. These problems would arise from the fact that the base modules must commit to particular states without any knowledge of the obstacles in the end-effector region. In this case, a collision-free manipulator configuration could easily be found by changing one of the base module states and reiterating the obstacle avoidance process described in this paper.

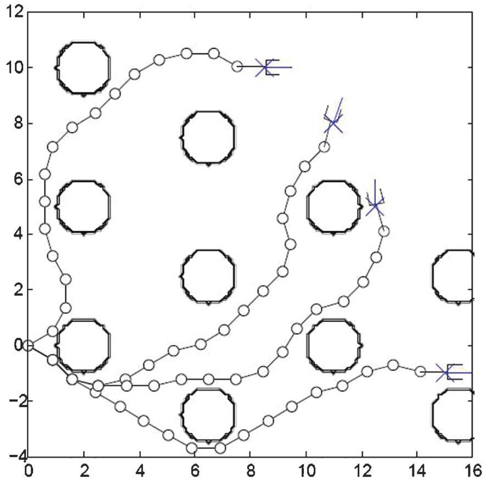


Fig. 6. Manipulator configurations in an obstacle field.

REFERENCES

1. Chiaverini, S., Siciliano, B. and Egeland, O., ‘‘Review of the damped least-squares inverse kinematics with experiments on an industrial robot manipulator,’’ *IEEE Transactions on Control Systems Technology*, Vol. 2, No. 2, pp. 123–134, 1994.
2. Baker, D.R. and Wampler II, C.W., ‘‘On the inverse kinematics of redundant manipulators,’’ *International Journal of Robotic Research*, Vol. 7, No. 3, pp. 2–21, 1988.
3. Deo, S. and Walker, I.D., ‘‘Minimum effort inverse kinematics for redundant manipulators,’’ *IEEE Transactions on Robotics and Automation*, Vol. 13, No. 5, pp. 767–775, 1995.
4. Nakamura, Y., Hanafusa, H. and Yoshikawa, T., ‘‘Task-priority based redundancy control of robot manipulators,’’ *The International Journal of Robotics Research*, Vol. 6, No. 2, pp. 3–9, 1987.
5. Maciekewski, A.A. and Klein, C.A., ‘‘Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments,’’ *The International Journal of Robotics Research*, Vol. 4, pp. 109–117, 1985.
6. Sciavicco, L. and Siciliano, B., ‘‘A solution algorithm to the inverse kinematic problem for redundant manipulators,’’ *IEEE Journal of Robotics and Automation*, Vol. 4, pp. 403–410, 1988.
7. Mao, Z. and Hsia, T.C., ‘‘Obstacle avoidance inverse kinematics solution of redundant robots by neural networks,’’ *Robotica*, Vol. 15, pp. 3–10, 1997.
8. Cheng, F.-T., Lu Y.-T. and Sun Y.-Y., ‘‘Window-shaped obstacle avoidance for a redundant manipulator,’’ *IEEE Transactions on Systems, Man, and Cybernetucs*, *Part B*, Vol. 28, pp. 806–815, 1998.
9. Glass, K., Colbaugh, R., Lim, D. and Seraji, H., ‘‘Real-time collision avoidance for redundant manipulators,’’ *IEEE Transactions on Robotics and Automation*, Vol. 11, pp. 448–457, 1995.
10. Borenstein, J. and Koren, Y., ‘‘Real-time obstacle avoidance for fast mobile robots in cluttered environments,’’ *IEEE Journal of Robotics and Automation*, Vol. 7, No. 3, pp. 278–588, 1991.
11. Ge, S. S. and Cui, Y. J., ‘‘New potential functions for mobile robot path planning,’’ *IEEE Transactions on Robotics and Automation*, Vol. 16, pp. 615–620, 2000.
12. Khatib, O.Y., ‘‘Real-time obstacle avoidance for manipulators and mobile robots,’’ *The International Journal of Robotics Research*, Vol. 5, No. 1, pp. 90–98, 1986.
13. Wang, C.-C., Kumar, V. and Chiu, G.-M., ‘‘A motion control and obstacle avoidance algorithm for hyper-redundant manipulators,’’ *Proceedings of the 1998 International Symposium on Underwater Technology*, pp. 466–471, 1998.
14. Agirrebietia, J., Avilés, R., de Bustos, I.F. and Ajuria, G., ‘‘A method for the study of position in highly redundant multibody systems in environments with obstacles,’’ *IEEE Transactions on Robotics and Automation*, Vol. 18, No. 2, pp. 257–262, 2002.
15. Chirikjian, G.S. and Burdick, J.W., ‘‘A modal approach to hyper-redundant manipulator kinematics,’’ *IEEE Transactions on Robotics and Automation*, Vol. 10, No. 3, pp. 343–352, 1994.
16. Zanganeh, K.E. and Angeles, J., ‘‘The inverse kinematics of hyper-redundant manipulators using splines,’’ *IEEE International Conference on Robotics and Automation*, Vol. 3, pp. 2797–2802, 1995.
17. Chirikjian, G.S. and Burdick J.W., ‘‘An obstacle avoidance algorithm for hyper-redundant manipulators,’’ *IEEE International Conference on Robotics and Automation*, Vol. 1, pp. 625–631, 1990.
18. Chirikjian, G.S. and Ebert-Uphoff, I., ‘‘Numerical convolution on the Euclidian group with applications to workspace generation,’’ *IEEE Transactions on Robotics and Automation*, Vol. 14, No. 1, pp. 123–136, 1998.

19. Ebert-Uphoff, I. and Chirikjian, G.S., ‘‘Inverse kinematics of discretely actuated hyper-redundant manipulators using workspace densities,’’ *IEEE International Conference on Robotics and Automation*, pp. 139–145, 1996.
20. Borenstein, J. and Koren, Y., ‘‘The vector field histogram-fast obstacle avoidance for mobile robots,’’ *IEEE Transactions on Robotics and Automation*, Vol. 7, No. 3, pp. 278–288, 1991.
21. Ulrich, I. and Borenstein, J.Y., ‘‘VFH+: reliable obstacle avoidance for fast mobile robots,’’ *IEEE International Conference on Robotics and Automation*, Vol. 2, pp. 1572–1577, 1998.
22. Kyatkin, A.B. and Chirikjian, G.S., ‘‘Synthesis of binary manipulators using the Fourier transform on the Euclidean Group,’’ *Journal of Mechanical Design*, Vol. 121, No. 1, pp. 9–14, 1999.
23. Chirikjian, G.S. and Kyatkin, A.B., *Engineering applications of noncommutative harmonic analysis*, CRC Press, ch. 12, 2000.
24. Wang, Y. and Chirikjiam, G.S., ‘‘Workspace generation of hyper-redundant manipulators as a diffusion process on SE(N),’’ *IEEE Transactions on Robotics and Automation*, Vol. 20, No. 3, pp. 399–408, 2004.
25. Suthakorn, J. and Chirikjian, G.S., ‘‘A new inverse kinematics algorithm for binary manipulators with many actuators,’’ *Advanced Robotics*, Vol. 5, No. 2, pp. 225–244, 2001.