Instantaneous Inverse Kinematic Solution for Redundant Manipulators Based on Virtual Arms and Its Application to Winding Control*

Toshio TSUJI**, Seiya NAKAYAMA**,
Atsushi ARAKI** and Koji ITO***

We propose an instantaneous inverse kinematic solution for redundant manipulators based on virtual arms. The virtual arm has the same kinematic structure as the manipulator except that its end point is located on the joint or link of the manipulator. Providing that the appropriate number of virtual arms is used, the configuration of the manipulator can be represented by a set of end points of the virtual arms. First of all, we formalize the kinematics of virtual arms and derive instantaneous inverse kinematics. Then, the method is applied to winding control for hyperredundant manipulators. The proposed winding control algorithm is divided into two steps: 1) planning desired positions for virtual end points, and 2) integrating them into joint trajectory of the manipulator. The desired positions of each virtual arm can be planned in a parallel and distributed manner, and there is no necessity for considering joint space of the manipulator. Finally, computer simulations show that the winding control for a hyperredundant manipulator can be performed in 3 D-space.

Key Words: Computer Control, Mechanics, Robotics, Redundant Manipulator, Inverse Kinematics, Winding Control, Virtual Arm

1. Introduction

In the control of a multijoint manipulator with redundant joint degrees of freedom, its configuration, as well as end point trajectory of the manipulator, must be considered. In a trajectory planning problem including obstacle avoidance, for example, not only the end point but the entire arm must not collide with the obstacles. Therefore, in conventional methods of motion planning, the manipulator configurations are represented by a set of generalized coordinates called the configuration space[1)-3)]. In this space, the configuration of the manipulator can be expressed by a single point, so that it is easy to judge interference between an obstacle and the entire arm, and the multijoint manipulator can be regarded as a mobile robot. However, environments and information expressed in the task space, such as positions of obstacles and image signals from a camera, must be converted to those in the configuration space, and more complicated conversion is required with an increase of the number of joint degrees of freedom. Therefore, generally speaking, it is quite difficult to apply the configuration space approach to trajectory planning for highly redundant manipulators.

The present paper uses a concept of the virtual arm that allows us to represent the manipulator configurations in the task space. The virtual arm is a virtual manipulator that has its end point located on the joint or link of the manipulator (hereafter referred to as the actual arm), and has kinematic parameters such as link length and joint angle that are the same as those of the actual arm[4)-6)]. Providing that the appropriate number of virtual arms is used, the configuration of the actual arm can be expressed as a set of end points of the virtual arms, thus addressing the motion planning for redundant manipulators in the task space.

* Received 29th September, 1993. Japanese original: Trans. Jpn. Soc. Mech. Eng., Vol. 59, No. 558, Ser. C, 1993.

** Faculty of Engineering, Hiroshima University, 4-1 Kagamiyama 1 chome, Higashi-Hiroshima 724, Japan

*** Toyohashi University of Technology, 1-1 Hibarigaoka, Tempaku-cho, Toyohashi, Aichi 441, Japan
Lab. for Bio-Mimetic Control Systems at RIKEN; Bio-Mimetic Control Research Center.

The authors previously showed that kinematics of the virtual arm can be represented as a neural network minimizing overall network energy, and that trajectory planning for highly redundant manipulators can be performed in a parallel and distributed manner[5]. In this paper, the forward kinematics of the virtual arms are mathematically modeled and an instantaneous inverse kinematic solution is derived. Then, a trajectory generation method for winding control of a hyperredundant manipulator is proposed, and it is shown by computer simulations that the winding control can be performed in 3 D-space.

2. Virtual Arm and Its Kinematics

Let us consider an actual arm having m joints and a Cartesian task coordinate system with the base of the actual arm as its origin. Then the virtual arm that has an end point on a joint or a link of the actual arm is defined. Figure 1 shows an example of setting three virtual arms for an actual four-link arm. The parameters of the virtual arm, such as the link length, joint angles and base position, correspond to those of the actual arm. In general, n-1 virtual arms are used and the actual arm is regarded as the n-th virtual arm.

Now, let the end-point displacement vector of the i-th virtual arm in the task coordinate be denoted as $dX^i=(dX_1^i, dX_2^i, \cdots, dX_l^i)^T$ and also let the joint displacement vector of the actual arm be denoted as $d\theta^i=(d\theta_1^i, d\theta_2^i, \cdots, d\theta_m^i)^T$, where l and m are the dimensions of the task coordinate system and the joint coordinate system, respectively. For redundant manipulators, m is larger than l. The kinematic relationship between the end-point displacement vector of the i-th virtual arm ($i=1, 2, \cdots, n$) and the joint displacement vector of the actual arm is given by

$$dX^i=J^i(\theta)d\theta, \tag{1}$$

where $J^i(\theta)\in R^{l\times m}$ is the Jacobian matrix of the i-th virtual arm (hereafter, referred to as J^i). In the case where the i-th virtual arm has its end point at the $(r+1)$-th joint of the actual arm, for example, the Jacobian matrix J^i can be expressed using the Jacobian matrix of the actual arm J^n as

$$J^i=J^n\begin{bmatrix} I_r & 0 \\ 0 & 0 \end{bmatrix}, \tag{2}$$

where

$$\begin{bmatrix} I_r & 0 \\ 0 & 0 \end{bmatrix}\in R^{m\times m}$$

and I_r is an $r\times r$ unit matrix.

When Eq. (1) is concatenated regarding all arms to express the kinematic relationships for all virtual arms simultaneously, the following equation is obtained:

$$dX_v=Jd\theta, \tag{3}$$

and

$$dX_v=\begin{bmatrix} dX^1 \\ dX^2 \\ \vdots \\ dX^n \end{bmatrix},\quad J=\begin{bmatrix} J^1 \\ J^2 \\ \vdots \\ J^n \end{bmatrix},$$

where $dX_v\in R^L$ is a concatenated end-point displacement vector, $J\in R^{L\times m}$ is a concatenated Jacobian matrix and $L=ln$ denotes the total degrees of freedom for all end points. The definition of the virtual arm indicates that the i-th virtual arm includes arm 1 to $(i-1)$, so that the matrix J has a systematic structure and can be easily computed.

On the other hand, a relationship between an end-point force/torque vector $F^i=(F_1^i, F_2^i, \cdots, F_l^i)^T$ and a joint torque vector $\tau^i=(\tau_1^i, \tau_2^i, \cdots, \tau_m^i)^T$ of the i-th virtual arm is given by

$$\tau^i=J^{iT}F^i. \tag{4}$$

Using the concatenated Jacobian matrix J, the force/torque relationship of the virtual arms corresponding to Eq. (3) can be represented by

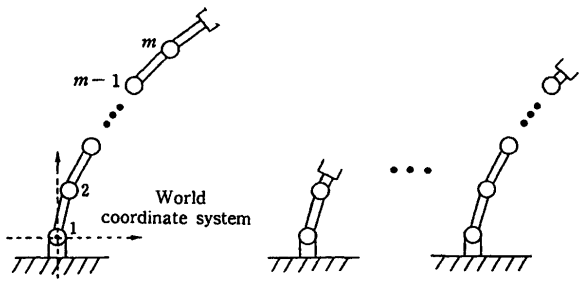
$$\tau=\sum_{i=1}^{n}\tau^i=J^TF_v, \tag{5}$$

and

$$F_v=\begin{bmatrix} F^1 \\ F^2 \\ \vdots \\ F^n \end{bmatrix},$$

where $F_v\in R^L$ is the concatenated force/torque vector of all virtual end points.

Figure 2 shows the kinematic relationships



(a) Actual arm (b) Virtual arms

Fig. 1 Examples of virtual arms for m-joint planar manipulator

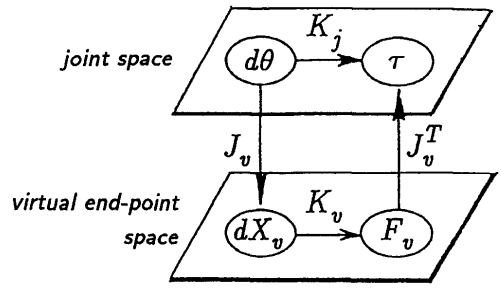


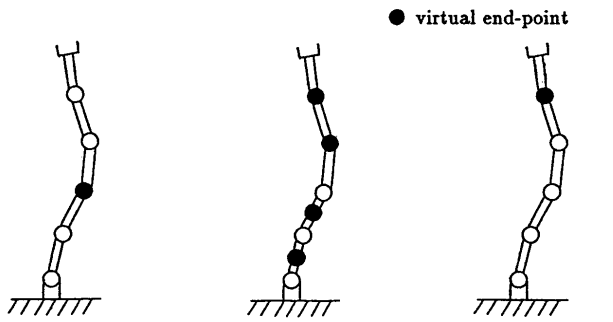
Fig. 2 Relationship between joint space and virtual end-point space

between the joint space of the actual arm and the end-point space of the virtual arms, where $K_j \in R^{m\times m}$ and $K_v \in R^{L\times L}$ denote a joint stiffness matrix of the actual arm and a concatenated end-point stiffness matrix of the virtual arms, respectively. It can be seen from the figure that there are two approaches to control the actual arm. The first approach is a kind of a joint feedback control using a desired joint displacement $d\theta^*$ converted from a desired virtual end-point displacement dX_v^* under consideration of the feasibility of the desired displacement[4),(5)]. The second approach is based on feedback control in the Cartesian task space in which the joint control torque τ is computed from the virtual end-point force F_v[6)]. In the following section, we focus on the joint feedback control and discuss the inverse kinematics that transform the virtual end-point displacement vector into the joint displacement vector.

3. Inverse Kinematics of Virtual Arm

Now, let us assume that the desired virtual end-point displacement dX_v can be obtained according to the given task. We would like to solve the kinematic equation (3) for the joint displacement $d\theta$ of the actual arm.

Depending on the joint degrees of freedom of the actual arm and the locations of the virtual end points, the kinematic equation (3) may be redundant, over-constrained or singular[4)]. Figure 3 shows an example of the three cases, where the actual arm is a five-link planar arm ($m=5$) and the dimension of the task space includes two translations ($l=2$). Locating a virtual end point on the third link as shown in Fig. 3(a), the desired concatenated virtual end-point displacement includes four elements ($L=4$), since there are two degrees of freedom for each end point. Then, the actual arm is still redundant, because L is less than m. Therefore, the kinematic equation (3) is underconstrained and the concatenated Jacobian matrix J is of full row rank, as long as the actual arm is not in a singular configuration.



(a) Redundant case (b) Over-constrained case (c) Singular case

Fig. 3 Three cases of virtual arms

In Fig. 3(b), four virtual arms are set ($n=5$). The desired virtual end-point displacement includes ten elements in this case, so that the manipulator becomes overconstrained ($L>m$). The concatenated Jacobian matrix J is of full column rank, and no joint displacement of the actual arm satisfies Eq. (3).

On the other hand, Fig. 3(c) shows the case where a virtual end point is located on the fifth link ($n=2$). At first sight, the manipulator still seems to be redundant because the joint degrees of freedom are more than the dimension of the desired concatenated virtual end-point displacement. In this case, however, since there is only one joint between the actual and virtual end points, it is impossible to control the positions of both the actual and virtual end points at the same time. The concatenated Jacobian matrix J is not of full rank in this case. Consequently, it can be seen that the rank of the concatenated Jacobian matrix dominates the kinematic equation (3).

Maximum-rank decomposition of the concatenated Jacobian matrix J gives us a unified approach to all three cases,

$$J=J_aJ_b, \tag{6}$$

where $J_a \in R^{L\times p}$ and $J_b \in R^{p\times m}$ have the same rank as J: rank J = rank J_a = rank J_b = p. Substituting Eq. (6) into Eq. (3), we have

$$dX_v=J_aJ_bd\theta. \tag{7}$$

The matrices J_a and J_b express an overconstrained part and an underconstrained part of the concatenated Jacobian matrix J, respectively.

To derive a general solution of the kinematic equation (6), we should solve it for a vector $J_bd\theta$ as the first step. Denoting

$$dX_b=J_bd\theta, \tag{8}$$

we have

$$dX_v=J_adX_b. \tag{9}$$

In general, the exact solution dX_b which satisfies Eq. (9) does not exist, because the matrix J_a is of full column rank. Here, an approximate solution that minimizes cost function

$$Q(dX_b)=(dX_v^*-J_adX_b)^TW(dX_v^*-J_adX_b) \tag{10}$$

is derived, where dX_v^* is a desired concatenated virtual end-point displacement vector. The weighting matrix $W \in R^{L\times L}$ is a nonsingular diagonal matrix:

$$W=diag.[w_1^1, \cdots, w_l^1, w_1^2, \cdots, w_1^n, \cdots, w_l^n], \tag{11}$$

where $w_j^i>0$ is a weighting factor for the j-th element of the desired end-point displacement of arm i, and regulating this value allows us to assign an order of priority to each virtual end point. Using the well-known optimization technique, we can obtain the solution dX_b:

$$dX_b=(J_a^TWJ_a)^{-1}J_a^TWdX_v^*. \tag{12}$$

The second step is to solve Eq. (8) for the joint

displacement vector $d\theta$ using the optimal vector dX_b. Since the matrix J_b is of full row rank, the solution $d\theta$ which satisfies Eq. (8) always exists. Using the pseudoinverse of J_b, we can obtain the general solution:

$$d\theta = J_b^+ dX_b + (I_m - J_b^+ J_b) r, \tag{13}$$

where $J_b^+ = J_b^T (J_b J_b^T)^{-1} \in R^{m \times p}$ is the pseudoinverse of J_b, I_m is a unit matrix and $r \in R^m$ is an arbitrary constant vector which may be utilized in the other subtasks[7]. In this paper, we choose r as a zero vector, then the general solution (13) gives the minimum norm solution:

$$d\theta = J_b^+ dX_b. \tag{14}$$

Consequently, if the desired virtual end-point displacement dX_v^* is given, we can obtain the optimal joint displacement $d\theta$ of the actual arm using Eqs. (12) and (13). Substituting Eqs. (12) and (14) into Eq. (7), we can also obtain the resulting virtual end-point displacement dX_v:

$$dX_v = J_a (J_a^T W J_a)^{-1} J_a^T W dX_v^*. \tag{15}$$

The inverse kinematic solution presented here can be applied to all cases shown in Fig. 3. In the redundant cases (e. g., Fig. 3(a)), since $J_a = I_L$ (an $L \times L$ unit matrix) and $J_b = J$, we can see that $dX_b = dX_v^*$. From Eq. (14), the joint displacement vector of the actual arm reduces to

$$d\theta = J^+ dX_v^*. \tag{16}$$

On the other hand, in the overconstrained cases (e. g., Fig. 3(b)), since $J_a = J$ and $J_b = I_m$, we can see that $dX_b = d\theta$. From Eq. (11), the joint displacement vector of the actual arm reduces to

$$d\theta = (J^T W J)^{-1} J^T W dX_v^*. \tag{17}$$

Obviously, we can use both Eqs. (12) and (14) in the singular case such as shown in Fig. 3(c).

Using the method presented here, the planning of the virtual end-point displacements can be performed in the task space and the order of priority can be assigned to each virtual end-point motion by adjusting the weighting matrix W according to the task environment. Previously, Hanafusa et al. (1983) showed the inverse kinematic solution for redundant manipulators considering the order of priority of subtasks[7]. Under multiple preordered subtasks (desired trajectories in their method), the method can generate a joint trajectory satisfying lower-ordered subtasks as much as possible while exactly performing higher-ordered subtasks. The desired virtual end-point displacements used in this paper may also be considered a kind of such pre-ordered subtasks, except that our method can function properly even in the cases where kinematic redundancy is utilized completely for higher-ordered subtasks or there are some subtasks with the same priority. In the next section, the instantaneous inverse kinematic solution using virtual arms is applied to winding control for hyperredundant manipulators.

4. Trajectory Generation Method for Winding Control

In animal movements, motions of winding around an object by active transformation of its body are observed, such as motions of an elephant's trunk and a snake. Umetani and co-workers called this kind of organism an active cord mechanism, and have been studying its motion control[8]-[10]. Also, they developed several active cord mechanisms; for example, ACM 7 is an elephant's-trunk-type manipulator with 8 links and 24 joint degrees of freedom[10]. Wada and Kuba[11] proposed a winding grasp using a lateral-inhibition-type control scheme developed by Hirose and Umetani[9]. This method can control a winding motion performed by cooperation of joint motion based on tactile information of the arm. Since their method dealt with only the case where the rotational axis of each joint was always parallel to each other, the winding motion realized by their method reduced to a planar movement. In contrast to Wada and Kuba's method, this paper shows that a winding control in 3 D space can be controlled by using virtual arms.

4.1 Winding motion

Let us consider a hyperredundant manipulator contact with a convex object at its k-th link in 3 D space ($l=3$), as shown in Fig. 4. Our purpose is to control the manipulator to wind around the object using the arm from the k-th link to the actual end point, assuming that the length of each link is sufficiently short relative to the size of the object it winds around. If a priori knowledge about the object such as size, shape and position is given, it may be easy to realize winding control by setting a desired position for each virtual end point on the surface of the object and using the inverse kinematic solution of the virtual arms presented in the previous sections. However, assuming that such a priori knowledge about the task environment is not available, we propose a winding control algorithm based on tactile information.

It is assumed that a contact vector $c^i \in R^3$ is available as tactile information. The contact vector represents a direction from which the object makes contact with the i-th link (see Fig. 4), and $c_i = 0$ means that the i-th link is not in contact with the object. Also, for simplicity, virtual end points are set at the end of each link. Therefore, the number of virtual arms is the same as the number of links (n).

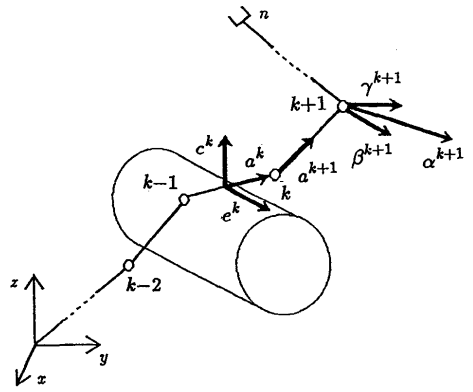


Fig. 4 Redundant manipulator contact with an object

4.2 Planning of desired virtual end-point displacements

The following strategy is used for planning the desired virtual end-point displacements dX_v^* for winding control.

(1) The links and the object may come apart, if one of any links from 1 to k moves greatly. Therefore the desired virtual end-point displacement for arm i ($i=1, \cdots, k$) is set to zero: $dX^{i*}=0$. Also, the highest priority is given to those virtual end points by making the corresponding elements of the weighting matrix W large (see Eq. (11)).

(2) To wind around the object, the next step is to make the $(k+1)$-th link come into contact with it. Thus, a desired displacement toward the object is used for the $(k+1)$-th virtual arm.

(3) None of the movements of the links from $k+2$ to n has a direct effect on winding control. Here, the desired displacements for those virtual end points are set to zero, $dX^{i*}=0$, and the corresponding order of priority is adjusted using the weighting matrix.

Using the desired virtual end-point displacement dX_v^*, the manipulator is kinematically controlled by Eqs. (12) and (14). When the $(k+1)$-th link comes into contact with the object, the index k is replaced by $k+1$ and the same control strategy is repeated until k becomes n.

Following the above strategy, the desired virtual end-point displacement $dX^{i*} \in R^3$ ($i=1, \cdots, n$) is computed by

$$dX^{i*}=\eta a^i(1-M^i)(M^{i-1}-M^i), \tag{18}$$

where η is a positive constant that determines the amplitude of the desired displacement, and $a^i \in R^3$ is a vector representing a winding direction in the task space. Also, M^i indicates whether the i-th link is in contact with the object: $M^i=1$ means contact and $M^i=0$ means noncontact. Under Eq. (18), an equilibrium state wherein all dX^{i*} ($i=1, \cdots, n$) become zero simultaneously is achieved from only the following two different mechanical states between the manipulator and the object:

[1] winding state: $M^0=M^1=\cdots=M^{k-1}=0$, and $M^k=M^{k+1}=\cdots=M^n=1$,

[2] free (noncontact) state: $M^0=M^1=\cdots=M^n=0$.

Therefore, using the appropriate winding direction vector a^i, once the manipulator comes into contact with the object at the k-th link, the arm starts to wind around the object and stops at the equilibrium state.

The winding direction vector a^i for the i-th virtual end point is determined as

$$a^i=\beta^i+\gamma^i, \tag{19}$$

$$\beta^i=s^i e^{i-1}, \quad \gamma^i=a^i \times e^{i-1}, \tag{20}$$

$$s^i=\begin{cases} 1 & \text{if } (a^i, e^{i-1})<0 \\ 0 & \text{if } (a^i, e^{i-1})=0, \\ -1 & \text{if } (a^i, e^{i-1})>0 \end{cases} \tag{21}$$

$$e^{i-1}=a^{i-1} \times c^{i-1}, \tag{22}$$

where $\times$ and $(*, *)$ denote a cross and inner product, respectively; and $a^i \in R^3$ is a unit vector representing the direction of the $(i-1)$-th link (see Fig. 4). When the $(i-1)$-th link is in contact with the object, the direction vector a^i of Eq. (19) has a nonzero amplitude; otherwise it becomes a zero vector.

When the k-th link is in contact with the object, the direction vector for the $(k+1)$-th virtual end point, a^{k+1}, is computed as follows. Since $c^k \neq 0$, the vector $e^k \in R^3$ of Eq. (22) represents a direction orthogonal to the plane P determined by a^k and c^k as shown in Fig. 4. Therefore, the vector $\beta^{k+1} \in R^3$ of Eq. (20) represents the direction toward the plane P from the $(k+1)$-th virtual end point (that is, the $(k+1)$-th joint). On the other hand, the vector $\gamma^{k+1} \in R^3$ of Eq. (20) gives the direction in which to rotate the $(k+1)$-th link toward the object on the plane P. As a result, it can be seen from Eq. (19) that the winding direction vector a^{k+1}, is determined in order to come into contact with the object while maintaining the k-th and $(k+1)$-th links in the same plane P as much as possible.

4.3 Simulation experiments

To show the validity of the proposed method, simulation experiments using the hyperredundant manipulator shown in Fig. 5 are performed. The manipulator includes 17 links and 49 joint degrees of freedom, and three adjoining joints, e.g., joints (1, 2, 3) or (4, 5, 6), represent ball joints. The length of each link is 0.06 m. Also, 17 virtual end points are set at the end of each link ($n=17$).

Figure 6 shows simulated manipulator motions when a cylindrical object (diameter: 0.14 m) comes into contact with the manipulator from four different directions, where Figs. 6(a), (b), (c) and (d)

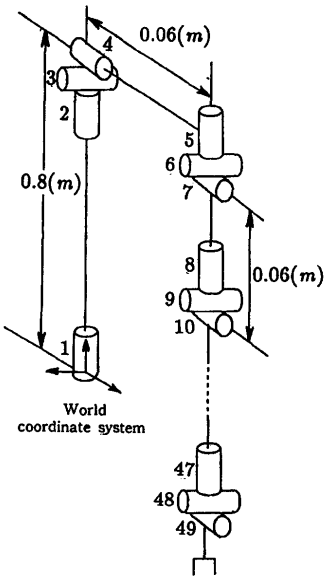
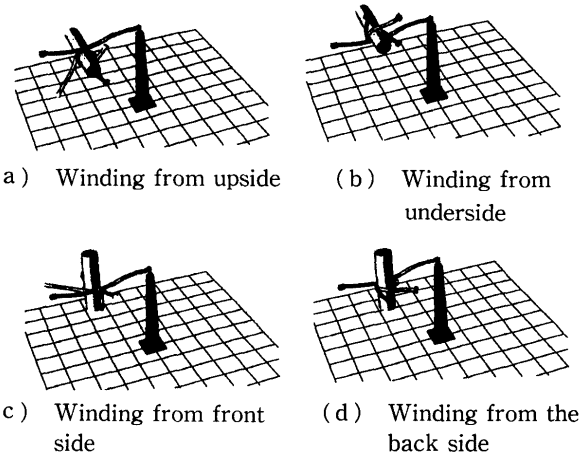


Fig. 5 A link model of a hyperredundant manipulator



(a) Winding from upside (b) Winding from underside

(c) Winding from front side (d) Winding from the back side

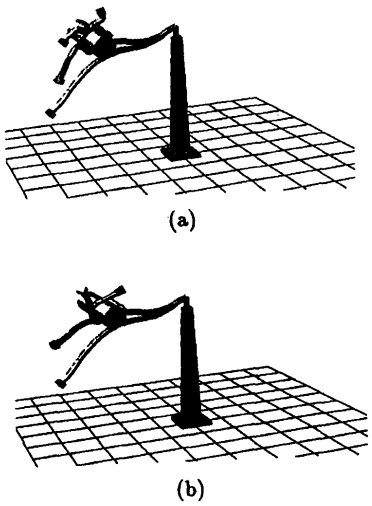
Fig. 6 Simulation results of winding control

correspond to winding motions from the upside, underside, front side and the back side, and initial configurations of the manipulator in all figures are identical. The weighting matrix $W \in R^{51\times 51}$ in Eq. (11) is set as

$$w_j^i = \begin{cases} 200.0 & (i=1, 2, \cdots, k:\ j=1, 2, 3) \\ 20.0 & (i=k+1:\ j=1, 2, 3) \\ 0.01 & (i=k+2, \cdots, 17:\ j=1, 2, 3) \end{cases}, \quad (23)$$

where k denotes the link number closest to the actual end point among links that are in contact with the object successively. Also, the parameter η is set to 2.0. It can be seen from the figure that the manipulator can wind the object from each direction in 3 D task space.

Next, Fig. 7 shows the effect of the weighting elements for the virtual end points located between the contact point and the actual end point (see section 4. 2). In Fig. 7(a), the weighting matrix W is set as



(a)

(b)

Fig. 7 Changes of the manipulator trajectory by adjusting the weighting matrix

$$w_j^i = \begin{cases} 200.0 & (i=1, 2, \cdots, k:\ j=1, 2, 3) \\ 20.0 & (i=k+1:\ j=1, 2, 3) \\ 1.0 & (i=k+2, \cdots, 17:\ j=1, 2, 3) \end{cases}, \quad (24)$$

and Eq. (23) is used for Fig. 7(b). The manipulator can wind the object after contact in both cases. However, movements of the virtual end point from $k+2$ to n are quite different. In Fig. 7(a), the virtual end points that have not yet wound around the object try to maintain their positions, since the corresponding weighting values are larger than those in Fig. 7(b). The proposed method can generate different manipulator postures by adjusting the priority for the virtual end points.

5. Conclusions

In this paper, the redundant manipulator was modeled as a set of virtual arms, and the instantaneous inverse kinematic solution was derived with theoretical consideration of kinematic conditions determined by the number and the locations of the virtual end points. Then, a winding control method using the virtual arms was proposed as an example of the trajectory generation problems for redundant manipulators. The main results of the paper can be summarized as follows: 1) using the concept of virtual arms, configurations of the redundant manipulator can be expressed in the task space, 2) by modeling points at which the given task is performed as virtual end points, any number and location of such points can be kinematically formalized in the same way as for the actual arm, and 3) 3 D winding motions can be realized based on tactile information without any knowledge about the object.

Further research will be directed at developing a new control method for redundant manipulators based

on the features of the virtual arms, introducing a distributed control scheme and communication among the virtual arms.

Acknowledgments

Part of the work presented here was supported by the scientific research fund (01750399, 03234105) from the Ministry of Education, Science and Culture of Japan.

References

(1) Lozano-Perez, T., Automatic Planning of Manipulator Transfer Movements, IEEE Trans. on Systems, Man and Cybernetics, Vol. SMC-11, No. 10 (1981), p. 681.

(2) Lozano-Perez, T., Spatial Planning: A Configuration Space Approach, IEEE Trans. on Computers, Vol. C-32, No. 2 (1983), p. 108.

(3) Lozano-Perez, T., A Simple Motion Planning Algorithm for General Manipulators, IEEE J. of Robotics and Automation, Vol. 3, No. 3 (1987), p. 224.

(4) Tsuji, T., Nakayama, S. and Ito, K., Trajectory Generation for Redundant Manipulators Using Virtual Arms, Proc. of ICARCV (1990), p. 554.

(5) Tsuji, T., Nakayama, S. and Ito, K., Distributed Trajectory Generation for Redundant Manipulators Based on Virtual Arms, Trans. of the Society of Instrument and Control Engineers, (in Japanese), Vol. 27, No. 12 (1991), p. 1412.

(6) Tsuji, T., Nakayama, S. and Ito, K., Distribute Feedback Control for Redundant Manipulators Based on Virtual Arms, Proc. of International Symposium on Autonomous Decentralized Systems (1993), p. 143.

(7) Hanafusa, H., Yoshikawa, T. and Nakamura, Y., Redundancy Analysis of Articulated Robot Arms and Its Utilization for Tasks with Priority, Trans. of the Society of Instrument and Control Engineers, (in Japanese), Vol. 19, No. 5 (1983), p. 421.

(8) Umetani, Y. and Hirose, S., Biomechanical Study of Active Cord Mechanism with Tactile Sensors, 6th International Symposium on Industrial Robots (1976), p. CI-1.

(9) Hirose, S. and Umetani, Y., Kinematic Control of Active Cord Mechanism with Tactile Sensors, Trans. of the Society of Instrument and Control Engineers, (in Japanese), Vol. 12, No. 5 (1976), p. 543.

(10) Hirose, S., Kado, T. and Umetani, Y., Tensor Actuated Elastic Manipulator, Proc. of 6th IFToMM World Congress, New Delhi, Vol. 2 (1983), p. 978.

(11) Wada, M. and Kuba, Y., Control of Winding Grasp Using Tactile Information, Trans. of the Society of Instrument and Control Engineers, (in Japanese), Vol. 20, No. 10 (1984), p. 973.