Manipulation Planning for Object Re-orientation based on Randomized Techniques

Masahito Yashima
Dept. of Mechanical Systems Engineering
National Defense Academy
Yokosuka 239–8686, JAPAN
Email: yashima@nda.ac.jp

***Abstract*— This paper presents randomized manipulation planning for re-orientation of an object in order to achieve dexterous manipulation by a robotic multi-fingered hand. A key issue is to develop an efficient algorithm to search for feasible trajectories of a manipulation system between two arbitrary grasps. The reasons for this are that aspects inherent in manipulation planning have high dimensional search spaces and complicated contact constraints. The present manipulation planning technique is based on the generation of subgoals for object orientations and the connection between subgoals. The generation of a random orientation, which should take into account the physical features of a rigid body orientation in a three-dimensional space, is also discussed. Furthermore, simulation examples for manipulation planning are presented and discussed.**

I. Introduction

Manipulating an object by a robotic, multi-fingered hand has attracted considerable attention. Most research is focused on local manipulation planning, which deals with the instantaneous kinematic and dynamic analysis of dexterous manipulation: rolling contacts based manipulation [2], [7], [8], [11] and sliding contacts based manipulation [3]. Local manipulation planning is definitely an important issue for a real robotic hand system to obtain control inputs in real time. In contrast, relatively less research is undertaken on global manipulation planning, which involves the planning of manipulation motions between two specified grasps [1], [12]. For example, we proposed an algorithm of dynamic manipulation planning by changing contact modes. The dimension of search space is reduced by utilizing an object nominal trajectory and randomly sampled switching times [12]. A key issue is to develop an efficient algorithm to search for feasible trajectories of a manipulation system between two arbitrary grasps. This is attributed to the feature of motion planning for a manipulation system. One aspect inherent in manipulation planning is the high dimensional search space. The state space of a manipulation system consists of the states of the object, robotic, multi-fingered hand, contact configurations, and their respective velocities. In addition, there are complicated contact constraints such as frictional constraints as well as holonomic and non-holonomic constraints due to sliding and rolling contacts.

Randomized techniques that can compute collision-free kinodynamic trajectories has attracted keen interest recently [4], [5], [6]. The basis of this approach is the incremental construction of a tree in the state space by generating subgoals at random and using a local planner to connect pairs of subgoals. This randomized planning has been shown to be valid for high degree-of-freedom systems with non-holonomic and/or dynamic constraints. These advantages motivated me to consider randomized techniques for manipulation planning of an object by a robotic multi-fingered hand [13]. In applying the randomized techniques to manipulation planning, we should note that several inherent issues need to be solved due to the complicated contact constraints.

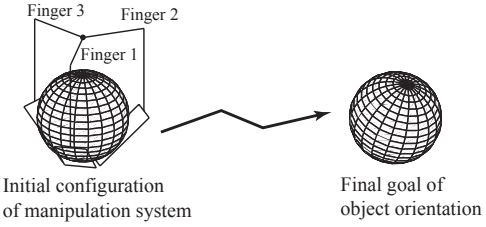

Fig. 1. Initial configuration of the manipulation system and the final goal of object orientation.

This paper presents manipulation planning for re-orientation of a three-dimensional object by a robotic multi-fingered hand, based on randomized techniques. The present manipulation planning involves finding feasible joint trajectories for a robotic multi-fingered hand in order to manipulate an object from a given initial orientation to a predetermined final goal orientation. As shown in Fig.1, we assume that the initial configuration of a manipulation system (defined by the configurations of the object and the multi-fingered hand) and the final goal of object orientation are specified. In particular, this paper discusses manipulation planning that takes into account the physical features of a rigid body orientation in a three-dimensional space in order to generate a random orientation.

II. Manipulation planning

This section outlines the algorithm of manipulation planning. Fig.2 illustrates a sequence of manipulation motions that were obtained by the manipulation planning presented in this paper. A robotic multi-fingered hand consists of three fingers. Each finger has three links with revolute joints and its third flat rectangular link makes contact with the object. Starting from an initial configuration of the manipulation system at position

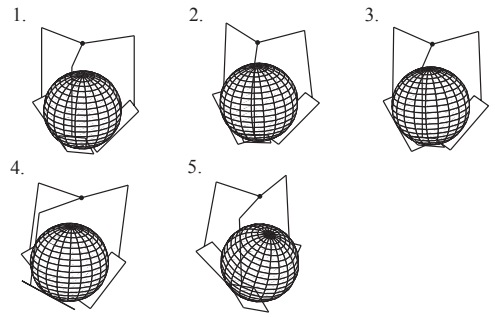

Fig. 2. Re-orientation of object with three-fingered hand.

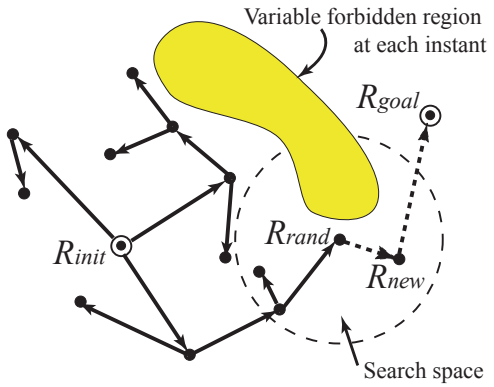

Fig. 3. Iterative process of generating subgoals in the configuration space for object orientation.

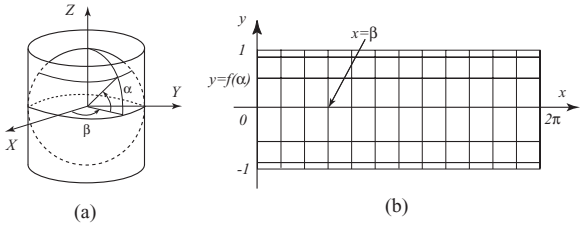

Fig. 4. Unit sphere and Lambert's cylindrical equal-area projection.

1, the object goes through intermediate subgoals in $2 \sim 4$, and reaches the final orientation goal in 5.

The present manipulation planning techniques are based on the generation of subgoals and the connection between subgoals. For the tasks of generation and connection, an algorithm for manipulation planning consists of the GENERATE_SUBGOAL and the CONNECT functions, respectively. Although subgoals are actually generated in the configuration space of the manipulation system, the iterative process of generating subgoals in the configuration space for object orientation is discussed in order to make easily to understand. Fig.3 illustrates a directed graph in the configuration space for object orientations. The nodes, expressed by black dots, refer to subgoals. The nodes, $\boldsymbol{R}_{init}$ and $\boldsymbol{R}_{goal} \in SO(3)$, show the rotational matrices that indicate the object's initial and final orientations with respect to the base frame, respectively. A pair of subgoals, $\boldsymbol{R}_A$ and $\boldsymbol{R}_B$, which is reachable from a parent subgoal, $\boldsymbol{R}_A$, to a child subgoal, $\boldsymbol{R}_B$, is connected by a directed tree. The GENERATE_SUBGOAL function randomly selects a subgoal, $\boldsymbol{R}_{rand}$, from the existing subgoals. The planner selects the candidate for the subgoal, $\boldsymbol{R}_{new}$, at random in the search space defined around $\boldsymbol{R}_{rand}$ (see dashed circle in Fig.3). Then, the CONNECT function generates the trajectory for object orientations, which connects $\boldsymbol{R}_{rand}$ to $\boldsymbol{R}_{new}$. Reachability between $\boldsymbol{R}_{rand}$ and $\boldsymbol{R}_{new}$ is checked by the local planner that solves an inverse problem for the specified object trajectory. If the connection is feasible, the nodes $\boldsymbol{R}_{rand}$ and $\boldsymbol{R}_{new}$ are connected by a directed tree (see dashed line in Fig.3), and then the local planner tries once again to connect $\boldsymbol{R}_{new}$ to $\boldsymbol{R}_{goal}$. When this connection is feasible, the search is terminated successfully. The algorithms mentioned above are described in section IV.

III. Generation of Random Orientation

In this manipulation planning, the GENERATE_SUBGOAL function generates an object orientation at random. In contrast to a position which can be described by a vector, we should note that an orientation is represented by a matrix form that belongs to $SO(3)$. This section takes into account the property of rotation, and proposes a novel algorithm for generation of a random orientation.

According to Euler's theorem on rotation, every orientation can be represented by a rotation about an equivalent axis, $\widehat{\boldsymbol{k}} = [k_x,\ k_y,\ k_z]^T \in \Re^3$, by an angle $\phi \in \Re^1$. The equivalent axis, $\widehat{\boldsymbol{k}}$, can be represented by points on the surface of a unit sphere. In order to determine a random orientation, we choose a point uniform randomly on the surface of the unit sphere. Simultaneously, we determine a rotational angle ϕ at random.

First, a random distribution of points on the surface of a unit sphere is shown. Fig.4 (a) shows how points on the unit sphere can be described by choosing latitude, α, and longitude, β. Note that it does not suffice to choose α and β uniform randomly at the intervals $[-\pi/2, \pi/2]$ and $[0, 2\pi)$, respectively. Fig.5 shows 3,000 points selected in this manner on the unit sphere, which is seen from the direction of the $+z$-axis. The points cluster around the North Pole. The size of a surface patch on a unit sphere created by infinitesimal angles, $d\alpha$ and $d\beta$, for latitude and longitude is given by $dS_{sphere} = \cos\alpha \, d\alpha d\beta$. The surface patches become smaller in area closer to the pole. There is a high probability that points generated in high latitudes lead to clustering around the pole. Regions of equal area have the same probability to contain points. The idea in this algorithm is that we can generate uniform random points on the unit sphere if we generate points on the plane, which is represented by an equal-area map projection of the unit sphere.

Fig.4 (b) shows Lambert's cylindrical equal-area projection. The meridians of longitude are mapped on an equally-spaced straight line with a length of two, and the parallels of latitude are mapped on a straight line with a length of 2π but get closer near the poles. The meridians and parallels meet at right angles. The relation between coordinates $(\alpha,\ \beta)$ on the unit sphere and $(x,\ y)$ in the plane, which retains the property of equivalence

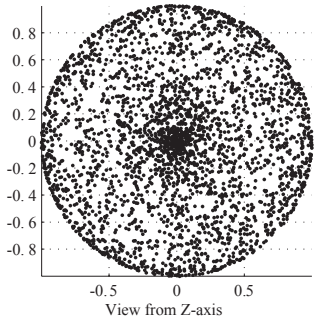

Fig. 5. Random sampling using latitude and longitude.

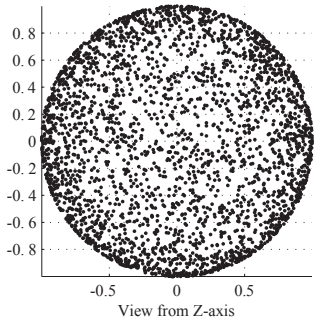

Fig. 6. Random sampling based on Lambert's cylindrical equal-area projection.

of area, can be written as

$$x = \beta \in [0, 2\pi), \qquad y = \sin\alpha \in [-1,\ 1] \tag{1}$$

Therefore, we can generate points uniform randomly on the surface of the unit sphere if we choose x and y uniform randomly in the intervals, $[0, 2\pi)$ and $[-1,\ 1]$, respectively. Fig.6 shows 3,000 uniform randomly sampled points selected in this manner. The cluster of points around the pole is found to disappear.

To summarize, the algorithm of generation of random orientation consists of the random generation of the vector of the unit equivalent axis, $\widehat{\boldsymbol{k}}$, based on the above-mentioned method, and the rotational angle, $\phi,\ |\phi| \leq \pi$. Therefore, the equivalent rotation matrix, $\Delta\boldsymbol{R}$, for the random orientation with respect to the reference frame can be given by

$$\Delta\boldsymbol{R} = \begin{bmatrix} k_x^2 v\phi + c\phi & k_x k_y v\phi - k_z s\phi & k_x k_z v\phi + k_y s\phi \\ k_x k_y v\phi + k_z s\phi & k_y^2 v\phi + c\phi & k_y k_z v\phi - k_x s\phi \\ k_x k_z v\phi - k_y s\phi & k_y k_z v\phi + k_x s\phi & k_z^2 v\phi + c\phi \end{bmatrix} \tag{2}$$

where $c\phi = \cos\phi$, $s\phi = \sin\phi$ and $v\phi = 1 - \cos\phi$.

IV. Algorithm

This section presents the algorithm of manipulation planning for object re-orientation based on the algorithm of generation of a random orientation described in section III.

A. *Generation of subgoals:* GENERATE_SUBGOAL

The aims of the GENERATE_SUBGOAL function are to generate subgoals, and find trajectories that achieve the final goal of an object orientation. The algorithm of the GENERATE_SUBGOAL function at each step are shown in Fig.7. Let $\boldsymbol{C} = (\boldsymbol{R},\ \boldsymbol{x},\ \boldsymbol{\theta},\ \boldsymbol{\eta}) \in \mathcal{C}$ denote the configuration of a manipulation system, where $\mathcal{C}$ is its configuration space, $\boldsymbol{x}$ represents the object positions, $\boldsymbol{\theta}$ represents the joint angles, and $\boldsymbol{\eta}$ represents the contact configurations.

GENERATE_SUBGOAL
1 **if** (CONNECT(C_{init}, R_{goal})=TRUE) **then** STOP
2 $C_{rand} \longleftarrow$ RANDOM_SUBGOAL
3 $R_{new} \longleftarrow$ NEW_SUBGOAL(R_{rand})
4 **if** (CONNECT(C_{rand}, R_{new})=TRUE) **then**
5 ADD_NEW_SUBGOAL(C_{new})
6 **if** (CONNECT(C_{new}, R_{goal})=TRUE) **then** STOP
7 **else goto** the 2nd step
8 **else goto** the 2nd step

Fig. 7. The GENERATE_SUBGOAL algorithm.

(Step1) As the very first step of the algorithm, the CONNECT($\boldsymbol{C}_{init}$, $\boldsymbol{R}_{goal}$) function attempts to direcrly connect the initial object orientation, $\boldsymbol{R}_{init}$, to the final goal object orientation, $\boldsymbol{R}_{goal}$, for a given initial configuration of the manipulation system, $\boldsymbol{C}_{init} \in \mathcal{C}$. If the attempt is successful, the algorithm terminates. If not, it proceeds to the next step; generating subgoals starts from $\boldsymbol{C}_{init}$.

(Step 2) The RANDOM_SUBGOAL function selects a subgoal, $\boldsymbol{C}_{rand} \in \mathcal{C}$, from the existing subgoals at random. Let $\boldsymbol{R}_{rand} \in SO(3)$ be the selected object orientation corresponding to $\boldsymbol{C}_{rand}$.

(Step 3) The NEW_SUBGOAL function randomly selects a new candidate subgoal, $\boldsymbol{R}_{new} \in SO(3)$, in the search space for the object orientation (see Fig.3), which is constructed by a rotation of $\boldsymbol{R}_{rand}$ about the equivalent axis of rotation, $\widehat{\boldsymbol{k}}$, $k_x^2 + k_y^2 + k_z^2 = 1$, by an angle $\phi,\ |\phi| \leq \phi_{max}$. The range of the search space is determined by the maximal rotational angle, $\phi_{max} \in [0,\ \pi]$. A randomly sampled rotation, $\Delta\boldsymbol{R}$ (see (2)), relative to $\boldsymbol{R}_{rand}$, can be obtained using the algorithm shown in section III. Therefore, the candidate for a rotational subgoal with respect to the base frame is given by $\boldsymbol{R}_{new}$=$\boldsymbol{R}_{rand}\Delta\boldsymbol{R}$.

(Step 4) The CONNECT($\boldsymbol{C}_{rand}$, $\boldsymbol{R}_{new}$) function checks if a local planner can connect $\boldsymbol{R}_{rand}$ to $\boldsymbol{R}_{new}$, assuming the configuration of the manipulation system, $\boldsymbol{C}_{rand} \in \mathcal{C}$, corresponding to $\boldsymbol{R}_{rand}$, as an initial condition. If this is not feasible, we revert to step 2, and select a new candidate, $\boldsymbol{R}_{rand}$, at random.

(Step 5) If the connection between $\boldsymbol{R}_{rand}$ and $\boldsymbol{R}_{new}$ is feasible, the ADD_NEW_SUBGOAL($\boldsymbol{C}_{new}$) function adds the configuration of the manipulation system, $\boldsymbol{C}_{new} \in \mathcal{C}$, corresponding to $\boldsymbol{R}_{new}$ obtained in step 4, to the current directed graph as a new subgoal (see dashed line in Fig.3).

(Step 6) The CONNECT($\boldsymbol{C}_{new}$, $\boldsymbol{R}_{goal}$) function checks if a local planner can connect $\boldsymbol{R}_{new}$ to $\boldsymbol{R}_{goal}$, assuming the configuration of the manipulation system, $\boldsymbol{C}_{new}$, corresponding to $\boldsymbol{R}_{new}$, as an initial condition. If this is feasible, the directed

graph is expanded from $\boldsymbol{R}_{new}$ to $\boldsymbol{R}_{goal}$ (see dashed line in Fig.3), and the exploration is terminated successfully. Otherwise, we revert to step 2, and select a new candidate, $\boldsymbol{R}_{rand}$, at random. The GENERATE_SUBGOAL function is terminated if the iteration of this function exceeds a specified upper limit of iteration.

As shown in Fig.3, this algorithm allows each subgoal to have only one incoming tree from a parent subgoal. In contrast, each subgoal is allowed to have several outgoing trees to several other child subgoals. Therefore, each subgoal can be reached from the initial orientation, $\boldsymbol{R}_{init}$, by a unique sequence of subgoals.

B. Connection between subgoals: CONNECT

The CONNECT function checks whether a local planner can find a trajectory for the multi-fingered hand in order to move the object from a specified object orientation, $\boldsymbol{R}_A$, to $\boldsymbol{R}_B$. That is, we solve an inverse problem for a given trajectory for the object orientation.

1) Model of the manipulation system: In order to explain the CONNECT function algorithm, basic equations for the model of the manipulation system as shown in Fig.8 were developed. We assume that the object and the multi-fingered hand are rigid bodies, and that each finger has one contact point with the object. We define a set of coordinate frames as follows: The base frame, $\{U\}$, is fixed to the hand palm; the object frame, $\{O\}$, is fixed to the mass center of the object; the finger frame, $\{ij\}$, is fixed to the link of finger j with the ith contact point. The contact frames, $\{Oi\}$ and $\{Fi\}$ are defined at the ith contact point on the object and fingers, respectively. The local frames, $\{LOi\}$ and $\{LFi\}$ are defined at the ith contact point as the coordinate frames fixed relative to $\{O\}$ and $\{ij\}$, respectively, that coincide with $\{Oi\}$ and $\{Fi\}$ at a time, t, when the object makes contact with the hand. The contact configuration at the ith contact point is given by $\boldsymbol{\eta}_i = [\boldsymbol{\xi}_{Oi}^T,\ \boldsymbol{\xi}_{Fi}^T,\ \psi_i]^T \in \Re^5$, where $\boldsymbol{\xi}_{Oi}$ and $\boldsymbol{\xi}_{Fi} \in \Re^2$ are the local coordinates that show the position of the ith contact point on the surface of the object and the link with respect to $\{O\}$ and $\{ij\}$, respectively, and ψ_i is the angle between the x-axes of $\{Oi\}$ and $\{F_i\}$ (see [8], [9] for further details).

Let $\boldsymbol{V}_O = [\boldsymbol{v}_O^T \quad \boldsymbol{\omega}_O^T]^T \in \Re^6$ be the linear and angular velocity vector of the object, and let $\dot{\boldsymbol{\theta}}_i \in \Re^{n_{\theta i}}$ be the joint angular velocity vector of the finger which has the ith contact point. $n_{\theta i}$ indicates the number of joints of the finger which has the ith contact point. The relative linear and angular velocity, $[\boldsymbol{v}_i^T,\ \boldsymbol{\omega}_i^T]^T \in \Re^6$, of $\{LOi\}$ with respect to $\{LFi\}$ expressed in $\{LOi\}$ can be written as

$$\begin{bmatrix} \boldsymbol{v}_i \\ \boldsymbol{\omega}_i \end{bmatrix} = \begin{bmatrix} \boldsymbol{G}_{vi}^T \\ \boldsymbol{G}_{\omega i}^T \end{bmatrix} \boldsymbol{V}_O - \begin{bmatrix} \boldsymbol{J}_{vi} \\ \boldsymbol{J}_{\omega i} \end{bmatrix} \dot{\boldsymbol{\theta}}_i \tag{3}$$

where $\boldsymbol{G}_{vi},\ \boldsymbol{G}_{\omega i} \in \Re^{6\times 3}$ are the wrench matrices, and $\boldsymbol{J}_{vi},\ \boldsymbol{J}_{\omega i} \in \Re^{3\times n_{\theta i}}$ are the hand Jacobian matrices.

The relationship between the evolutions of a contact configuration, $\boldsymbol{\eta}_i = [\boldsymbol{\xi}_{Oi}^T,\ \boldsymbol{\xi}_{Fi}^T,\ \psi_i]^T$, and the relative linear and angular velocity, $[\boldsymbol{v}_i^T,\ \boldsymbol{\omega}_i^T]^T = [v_{xi},\ v_{yi},\ v_{zi},\ \omega_{xi},\ \omega_{yi},\ \omega_{zi}]^T$ can be written using Montana's equations as

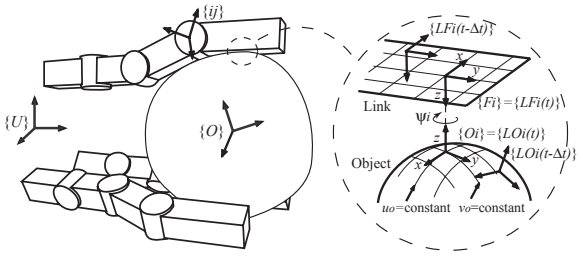

Fig. 8. Model of the manipulation system.

$$\begin{aligned}
\dot{\boldsymbol{\xi}}_{Oi} &= \mathcal{M}_{Oi}^{-1}\mathcal{K}_i^{-1}\left(\begin{bmatrix} -\omega_{yi} \\ \omega_{xi} \end{bmatrix} - \widetilde{\mathcal{K}}_{Fi}\begin{bmatrix} v_{xi} \\ v_{yi} \end{bmatrix}\right) \\
\dot{\boldsymbol{\xi}}_{Fi} &= \mathcal{M}_{Fi}^{-1}\boldsymbol{R}_{\psi i}\mathcal{K}_i^{-1}\left(\begin{bmatrix} -\omega_{yi} \\ \omega_{xi} \end{bmatrix} + \mathcal{K}_{Oi}\begin{bmatrix} v_{xi} \\ v_{yi} \end{bmatrix}\right) \\
\dot{\psi}_i &= \omega_{zi} + \mathcal{T}_{Oi}\mathcal{M}_{Oi}\dot{\boldsymbol{\xi}}_{Oi} + \mathcal{T}_{Fi}\mathcal{M}_{Fi}\dot{\boldsymbol{\xi}}_{Fi} \\
v_{zi} &= 0
\end{aligned} \tag{4}$$

where $\mathcal{M} \in \Re^{2\times 2}$, $\mathcal{K} \in \Re^{2\times 2}$, and $\mathcal{T} \in \Re^{1\times 2}$ are the metric, curvature, and torsion tensor forms, respectively. $\boldsymbol{R}_{\psi_i}$ is a rotational matrix corresponding to ψ_i, $\widetilde{\mathcal{K}}_{Fi} = \boldsymbol{R}_{\psi_i}\mathcal{K}_{Fi}\boldsymbol{R}_{\psi_i}$ and $\mathcal{K}_i = (\mathcal{K}_{Oi} + \widetilde{\mathcal{K}}_{Fi})$.

Relative motions at a contact point are different according to the present contact mode. The relative motions are constrained by $v_{xi} = v_{yi} = v_{zi} = 0$ for a rolling contact, and $v_{zi} = 0$ for a sliding contact. Introducing a selection matrix, $\boldsymbol{B}_i$, into (3) based on the contact mode yields the kinematic constraint equation,

$$\boldsymbol{G}_{Ai}^T\boldsymbol{V}_O = \boldsymbol{J}_{Ai}\dot{\boldsymbol{\theta}}_i \tag{5}$$

where $\boldsymbol{G}_{Ai} = [\boldsymbol{G}_{vi}\ \boldsymbol{G}_{\omega i}]^T\boldsymbol{B}_i^T$ and $\boldsymbol{J}_{Ai} = \boldsymbol{B}_i[\boldsymbol{J}_{vi}^T\ \boldsymbol{J}_{\omega i}^T]^T$. The selection matrix is given by $\boldsymbol{B}_i = [\boldsymbol{E}_3 \quad \boldsymbol{0}_{3\times 3}]$ for a rolling contact and $\boldsymbol{B}_i = [0,\ 0,\ 1,\ \boldsymbol{0}_{1\times 3}]$ for a sliding contact. $\boldsymbol{E}_3 \in \Re^{3\times 3}$ is a unit matrix, and $\boldsymbol{0}_{3\times 3}$ and $\boldsymbol{0}_{1\times 3}$ are zero matrices whose dimension are expressed in subscripts. Summing (5) for all contacts yields

$$\boldsymbol{G}_A^T\boldsymbol{V}_O = \boldsymbol{J}_A\dot{\boldsymbol{\theta}} \tag{6}$$

where $\boldsymbol{G}_A \in \Re^{6\times(3n_R+n_S)}$, $\boldsymbol{J}_A \in \Re^{(3n_R+n_S)\times n_\theta}$, and $\boldsymbol{\theta} \in \Re^{n_\theta}$ when there are n_C contacts, consisting of n_R rolling contacts and n_S sliding contacts, and the robotic hand has n_θ joints.

Assuming a Coulomb friction model, contact forces, $f_{xi},\ f_{yi}$, and $f_{zi} \in \Re^1$, expressed in $\{LOi\}$ applied to the object at a rolling contact are constrained by

$$\sqrt{f_{xi}^2 + f_{yi}^2} \le -\mu f_{zi}, \quad f_{zi} \le 0 \tag{7}$$

where μ is the coefficient of friction at the contact point. In the case of a sliding contact, since the contact forces lie on the boundary of the corresponding friction cone and its tangential forces are directed opposite to the direction of motion, we can obtain

$$f_{xi} = \widetilde{\mu}_{xi}f_{zi}, \quad f_{yi} = \widetilde{\mu}_{yi}f_{zi}, \quad f_{zi} \le 0 \tag{8}$$

where $\widetilde{\mu}_{\bullet i} = \mu v_{\bullet i}/\sqrt{v_{xi}^2 + v_{yi}^2}, \quad \bullet \in \{x, y\}$.

Applying the principle of virtual work to (5) and (8) yields the resultant force/moment at the object center of mass as shown below:

$$\begin{aligned} \boldsymbol{f}_O &= \sum_{i=1}^{n_S}(\boldsymbol{G}_{vxi}\widetilde{\mu}_{xi}f_{zi} + \boldsymbol{G}_{vyi}\widetilde{\mu}_{yi}f_{zi} + \boldsymbol{G}_{vzi}f_{zi}) \\ &\quad + \sum_{i=1}^{n_R}(\boldsymbol{G}_{vxi}f_{xi} + \boldsymbol{G}_{vyi}f_{yi} + \boldsymbol{G}_{vzi}f_{zi}) \end{aligned} \tag{9}$$

where $\boldsymbol{G}_{vxi}$, $\boldsymbol{G}_{vyi}$, and $\boldsymbol{G}_{vzi} \in \Re^6$ correspond to each column vector of $\boldsymbol{G}_{vi} \in \Re^{6\times 3}$, respectively. Similarly, the relationship between the contact forces and the joint torques, $\boldsymbol{\tau}_H \in \Re^{n_\theta}$ is given by

$$\begin{aligned} \boldsymbol{\tau}_H &= -\sum_{i=1}^{n_S}(\boldsymbol{J}_{vxi}^T\widetilde{\mu}_{xi}f_{zi} + \boldsymbol{J}_{vyi}^T\widetilde{\mu}_{yi}f_{zi} + \boldsymbol{J}_{vzi}^T f_{zi}) \\ &\quad - \sum_{i=1}^{n_R}(\boldsymbol{J}_{vxi}^T f_{xi} + \boldsymbol{J}_{vyi}^T f_{yi} + \boldsymbol{J}_{vzi}^T f_{zi}) \end{aligned} \tag{10}$$

where $\boldsymbol{J}_{vxi}^T$, $\boldsymbol{J}_{vyi}^T$, and $\boldsymbol{J}_{vzi}^T \in \Re^{n_{\theta i}}$ correspond to each column vector of $\boldsymbol{J}_{vi}^T \in \Re^{n_{\theta i}\times 3}$, respectively.

Therefore, using $\boldsymbol{f}_O$ in (9), the equation of motion for the object is given by

$$\boldsymbol{M}_O\dot{\boldsymbol{V}}_O = \boldsymbol{f}_O - \boldsymbol{h}_O \tag{11}$$

where $\boldsymbol{M}_O \in \Re^{6\times 6}$ is the mass matrix of the object, and $\boldsymbol{h}_O$ is a vector of Coriolis and gravity terms. Similarly, the equation of motion for the robotic hand using $\boldsymbol{\tau}_H$ in (10) can be written as given below:

$$\boldsymbol{M}_H\,\ddot{\boldsymbol{\theta}} = \boldsymbol{\tau} + \boldsymbol{\tau}_H - \boldsymbol{h}_H \tag{12}$$

where $\boldsymbol{M}_H \in \Re^{n_\theta\times n_\theta}$ is the moment inertia matrix, $\boldsymbol{\tau}$ is a vector of the joint driving torque, and $\boldsymbol{h}_H$ is a vector of Coriolis and gravity terms.

2) Algorithm: An algorithm of the CONNECT function is presented using Fig.9. Refer to [12] for further details on the algorithm of the local planner.

(Step 1) The GENERATE_TRAJ($\boldsymbol{R}_\mathrm{A}$, $\boldsymbol{R}_\mathrm{B}$) function generates the trajectory for the object orientation, $\boldsymbol{R}(t)$, which connects the object orientation, $\boldsymbol{R}_A$, of the subgoal A to the object orientation, $\boldsymbol{R}_B$, of the subgoal, B, during the specified manipulation time, ΔT. Motions of the manipulation system are made to stop at each subgoal in order to avoid discontinuity of velocity at each subgoal. Using exponential coordinates for rotation, the trajectory for object orientation can be given by

$$\boldsymbol{R}(t) = \boldsymbol{R}_A\exp(\mathrm{skew}(\widehat{\boldsymbol{k}}_E)\phi_E d(t)), \quad t\in[0,\ \Delta T] \tag{13}$$

where

$$\begin{aligned} d(t) &= -2(t/\Delta T)^3 + 3(t/\Delta T)^2 \\ \phi_E &= \cos^{-1}\left(\frac{\mathrm{trace}(\boldsymbol{R}_A^T\boldsymbol{R}_B) - 1}{2}\right) \in [0,\ \pi] \\ \mathrm{skew}(\widehat{\boldsymbol{k}}_E) &= \frac{1}{2\sin\phi_E}(\boldsymbol{R}_A^T\boldsymbol{R}_B - \boldsymbol{R}_B^T\boldsymbol{R}_A) \in so(3) \end{aligned}$$

CONNECT(C_A, R_B)
1 GENERATE_TRAJ(R_A, R_B)
2 Flag ⟵ INVERSE_PROBLEM
3 **if** (Flag = CONNECTABLE) **Return** TRUE
4 **else Return** FALSE

Fig. 9. The CONNECT algorithm.

(Step 2) The INVERSE_PROBLEM function solves an inverse problem for a given object trajectory in step 1, using the model of manipulation system obtained in the previous section. We derived $\dot{\boldsymbol{\theta}}(t)$ from (6) using the pseudo-inverse, $\boldsymbol{J}_A^\sharp$, of $\boldsymbol{J}_A$. The trajectory for object angular velocity, $\boldsymbol{\omega}_O(t)$, is given by $\mathrm{skew}(\boldsymbol{\omega}_O(t)) = \dot{\boldsymbol{R}}(t)\boldsymbol{R}^T(t)$. In case of the simultaneous translation of the object, a linear velocity trajectory for the object, $\boldsymbol{v}_O(t)$, is also specified. Substituting $\boldsymbol{V}_O(t) = [\boldsymbol{v}_O^T(t)\ \boldsymbol{\omega}_O^T(t)]^T$ and $\dot{\boldsymbol{\theta}}(t)$ in (3) yields the relative linear and angular velocity, $(\boldsymbol{v}_i,\ \boldsymbol{\omega}_i)$. Then, substituting $(\boldsymbol{v}_i,\ \boldsymbol{\omega}_i)$ into (4) yields $\dot{\boldsymbol{\eta}}_i(t)$. Differentiation of (6) yields the joint angular acceleration, $\ddot{\boldsymbol{\theta}}(t)$. The contact forces needed to generate the object acceleration, $\dot{\boldsymbol{V}}_O(t)$, are determined by solving the quadratic program shown in [10]. Finally, substituting the contact forces and the joint angular accelerations in (12) yields the joint driving torques, $\boldsymbol{\tau}(t)$. The kinematical and dynamical validity of solutions obtained in the above-mentioned manner is checked, based on the conditions for instantaneous motion planning defined in [12].

(Steps 3 and 4) The CONNECT function returns TRUE when the inverse problem has solutions. Otherwise, it returns FALSE.

V. Computer simulations

This section verifies the effectiveness of the proposed manipulation planning by a computer simulation. The algorithm was implemented in MATLAB on a 1.8 GHz Intel Pentium IV PC with 1 GB of memory. We detail the simulation model for the manipulation system shown in Fig.1. As shown in Fig.10, the first joint of the three fingers is located at the same position, and is actuated, independent of the others. Each third flat link makes contact with the object. The radius of the spherical object is 0.1 m and its mass is 1.0 kg. The first and second links are 0.14 and 0.2 m long, respectively. The dimensions of the flat link are 0.15 m by 0.07 m. The mass of each link is 0.5 kg. Each contact is assumed to be a rolling contact with a friction coefficient of 0.8.

We consider the re-orientation of an object to the final goal expressed by rotation about the equivalent axis vector, $\widehat{\boldsymbol{k}} = [-0.5,\ 0.5,\ \sqrt{0.5}]^T$ by the equivalent angle, $\phi = 30$ deg from the initial grasp, as shown in Fig.1. The maximal equivalent angle that determines the search space for orientation is $\phi_{max} = 30.0$ deg (see step 3 in section IV-A). In addition to the generation of subgoals for an object orientation, the generation of the subgoals for object position is allowed in the neighborhood at an initial object position in order to facilitate

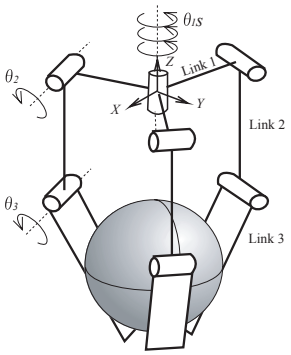

Fig. 10. Simulation model.

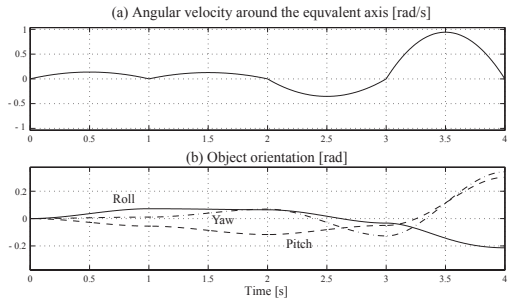

Fig. 11. Planned trajectories of object orientation and angular velocity around the equivalent axis.

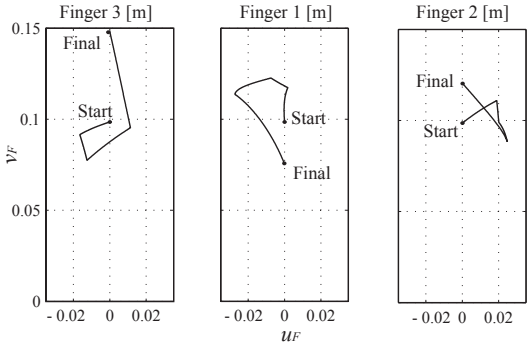

Fig. 12. Paths of contact points on the surface of the third links for a large search space, ($r_{max} = 0.03$[m], $\phi_{max} = 30.0$[deg]).

the finding of solutions. The final object position, $\boldsymbol{x}_{goal} \in \Re^3$, is chosen so that it is equal to the initial object position, $\boldsymbol{x}_{init} \in \Re^3$. A basic idea of the algorithm concerning the generation of positional subgoals is similar to the case of orientation. That is, a positional subgoal, $\boldsymbol{x}_{rand} \in \Re^3$, is randomly selected from the existing positional subgoals. The candidate for a positional subgoal, $\boldsymbol{x}_{new} \in \Re^3$, is randomly generated in the search space defined around $\boldsymbol{x}_{rand}$. A sphere with a radius $r_{max} = 0.03$ m is assumed as the positional search space. Similarly, solving the inverse problem, the feasibility of connection between $\boldsymbol{x}_{rand}$ and $\boldsymbol{x}_{new}$, and $\boldsymbol{x}_{new}$ and $\boldsymbol{x}_{goal}$ is checked.

Averaging over 20 trials, the rate of finding solutions was 95.0% despite difficult manipulation tasks such as the re-orientation of the object without changing contact modes. The reason that the solution finding ratio is not 100% for the same final goal orientation is that the randomized approach used in the proposed algorithm does not always guarantee to find feasible solutions. On an average, the computational time for finding solutions was 41.5 min. The planner generated 111.4 candidates for subgoals, and the final trajectory goes through 4.5 intermediate subgoals. Among these simulation results, there is no solution of trajectory that can directly connect the initial orientation to the final orientation.

Fig.2 shows the example of snapshots of re-orienting an object for the initial grasp and the final object orientation as shown in Fig.1. The obtained trajectory goes through three subgoals. Fig.11 (a) shows the angular velocity around the equivalent axis. Fig.11 (b) shows the roll, pitch, and yaw angle with respect to the base frame for an object rotational trajectory corresponding to (13). Shifting the object orientation toward the subgoals generated at each time, $\Delta T = 1.0$ sec, the object was manipulated to the final orientation. The paths of the contact points on the surface of the third links are shown in Fig.12.

Consider the scenario where the radius, r_{max}, of spherical search space for the object position is reduced from 0.03 m to 0.005 m. Figs.13 and 14 show snapshots of re-orienting the object and the paths of the contact points on the surface of the third links, respectively. Compared to the previous scenario, the number of generated subgoals increased from three to eleven, and it is found that the contact points made repeated turns on the surface of the link. The planner requires a high number of turns since the solution space for manipulation with the rolling contacts is limited according to the reduction in search space for the object position.

Fig.15 shows the generated subgoals of the object orientation for the scenario shown in Fig.2. The orientation is expressed by the Gauss spherical representation for the object frame of each subgoal. That is, the orientation is illustrated by the point of intersection of each unit vector in the coordinate directions of the object frame of each subgoal with a unit sphere whose center equals the origin of the base frame. The numbers at each point indicate the sequence of the generated subgoals. As shown in Fig.15, the planner explored the entire configuration space and subgoals were generated iteratively. The above simulation results confirm the validity of this algorithm of manipulation planning based on randomized techniques.

VI. Conclusion

This paper presented an algorithm of randomized manipulation planning for re-orientation of an object in order to achieve dexterous manipulation by a robotic multi-fingered hand. The results also showed an algorithm of generation of random orientation by taking into account the physical property of rotation of a three-dimensional rigid body. The validity of the proposed algorithms was shown by computer simulations. Previous work on manipulation planning, which deals with the controllability of non-holonomic planning with rolling contacts [7], needs strict mathematical formulations. On the other hand, the proposed planning does not require complicated formula-

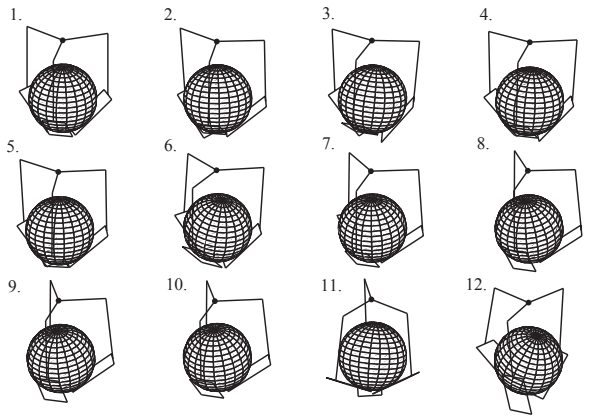

Fig. 13. Snapshots of manipulating object for a small search space, ($r_{max} = 0.005$[m], $\phi_{max} = 30.0$[deg]).

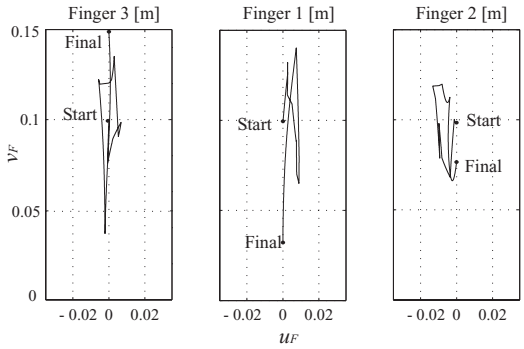

Fig. 14. Paths of contact points on the surface of the third links for a small search space.

tions, so it is practical and flexible for any kind of manipulation system. While the manipulation planning is not guaranteed to find feasible solutions and/or optimal solutions because of the randomized planning feature, it is possible to obtain optimal solutions, which satisfy a cost function defined according to a manipulation task, among the solutions obtained by multiple trials. Finally, rigid bodies and point contacts with Coulomb friction were assumed. Although they may not be valid in a real world sense, soft contacts can be incorporated by considering an appropriate compliance contact model.

REFERENCES

[1] M.J. Cherif and K.K. Gupta, "Planning quasi-static fingertip manipulations for reconfiguring objects," *IEEE Transactions on Robotics and Automation*, vol. 15, no. 5, pp.837–848, 1999.

[2] A.A. Cole, J.E. Hauser and S.S. Sastry, "Kinematics and control of multifingered hands with rolling contact," *IEEE Transactions on Automatical Control*, vol. 34, no. 4, pp.398–404, 1989.

[3] A.A. Cole, P. Hsu and S.S. Sastry, "Dynamic control of sliding by robot hands for regrasping," *IEEE Transactions on Robotics and Automation*, vol. 8, no. 1, pp.42–52, 1992.

[4] E. Frazzoli, M.A. Dahleh and E. Feron, "Real-time motion planning for agile autonomous vehicles," *Journal of Guidance, Control and Dynamics*, vol. 25, no. 1, pp.116–129, 2002.

[5] R. Kindel, D. Hsu, J.C. Latombe and S. Rock, "Kinodynamic motion planning amidst moving obstacles," *In Proc. of IEEE Int. Conf. on Robotics and Automation*, 2000.

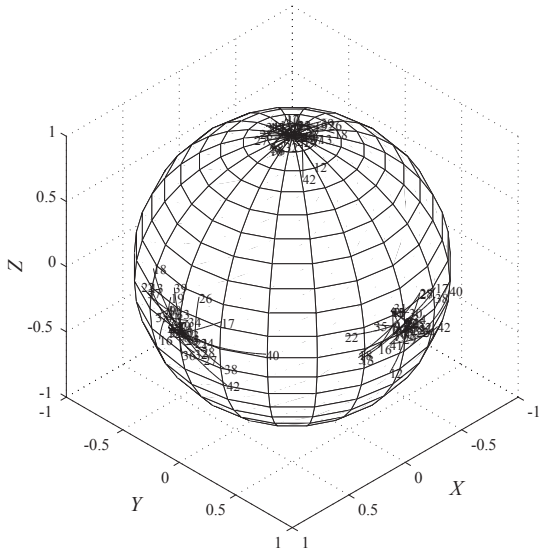

Fig. 15. Generation of subgoals for object orientation by randomized techniques.

[6] S.M. LaValle and J.J. Kuffner,Jr, "Randomized kinodynamic planning," *The International Journal of Robotics and Research*, vol. 15, no. 5, pp.378–400, 2001.

[7] A. Marigo and A. Bicchi, "Rolling bodies with regular surface: contrability theory and applications," *IEEE Transactions on Automatic Control*, vol. 45, no. 9, pp.1586–1599, 2000.

[8] D.J.Montana, "The kinematics of contact and grasp," *Int. Journal of Robotics Research*, vol. 7, no. 3, pp.17–32, 1988.

[9] R.M. Murray, Z. Li and S.S. Sastry, *A Mathematical Introduction to Robotic Manipulation.* CRC Press, 1993.

[10] M.A. Nahon and J. Angeles, "Real-time force optimization in parallel kinematic chains under inequality constraints," *IEEE Transaction on Robotics and Automation*, vol. 8, no. 4, pp.439–450, 1992.

[11] N. Sarker, X. Yun and V. Kumar, "Dynamic control of 3-D rolling contacts in two-Arm manipulation," *IEEE Transactions on Robotics and Automation*, vol. 13, no. 3, pp.364–376, 1997.

[12] M. Yashima and H. Yamaguchi, "Dynamic motion planning whole arm grasp systems based on switching contact modes," *In Proc. of IEEE Int. Conf. on Robotics and Automation*, pp.2492–2499, 2002.

[13] M. Yashima, Y. Shiina and H. Yamaguchi, "Randomized manipulation planning for a multi-fingered hand by switching contact modes," *In Proc. of IEEE Int. Conf. on Robotics and Automation*, pp.2689–2694, 2003.