Proceedings of the 2001 IEEE
International Conference on Robotics & Automation
Seoul, Korea • May 21-26, 2001

Planning of Graspless Manipulation by Multiple Robot Fingers

Yusuke MAEDA Hirokazu KIJIMOTO† Yasumichi AIYAMA†† *and* Tamio ARAI

Department of Precision Engineering, School of Engineering, The University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, JAPAN
maeda@prince.pe.u-tokyo.ac.jp

†NTT Communications Corporation
††Institute of Engineering Mechanics and Systems, University of Tsukuba

Abstract

Graspless manipulation is to manipulate objects without grasping by just pushing, tumbling, and so on. One of the main difficulties in graspless manipulation is planning. It is very time-consuming to plan a general graspless manipulation problem, because it requires complicated mechanical analysis including friction. To reduce the load of computation, we adopt a two-step approach: 1) construction and simplification of contact state graph at geometry level, and 2) planning of manipulation at mechanics level. In this paper, we focus the latter, and propose an algorithm to plan mechanically feasible manipulation. It generates digraphs that represent C-Subspaces for all the contact states, and unites them into one big graph, which we call "manipulation-feasibility graph." Manipulation plan can be obtained by searching the graph. This algorithm is implemented for planar graspless manipulation by multiple robot fingers, and planned results are shown.

1 Introduction

Ways to manipulate objects without grasping, such as pushing and tumbling (Figure 1), are generally referred to as "graspless manipulation" [1]. It is favorable for handling of heavy objects, because robots do not have to support all the weight of objects.

One of the biggest difficulty in graspless manipulation is planning of total motion sequence from a start to a goal. In conventional pick-and-place operation, once the object is grasped, manipulation planning is reduced to a geometrical collision avoidance problem. That is because the configuration of the manipulator hand and that of the object have a one-to-one correspondence. In graspless manipulation, however, that is not the case. So the planning of graspless manipulation requires consideration to not only geometrical but also mechanical effects such as friction. Moreover, graspless manipulation may be irreversible; for example, a manipulator may be able to push the object but may not be able to pull it back. Therefore, planning before execution is very important for graspless manipulation.

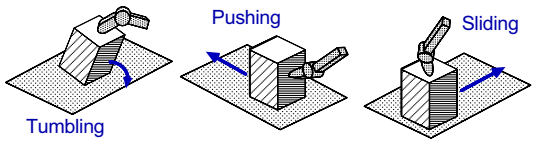


Figure 1: Graspless Manipulation

Planning of graspless manipulation is still a tough problem, especially when considering friction. Not only is it time-consuming, but we have no adequate scheme to implement the planning problem. Therefore, each of most related researches deals with a planning problem with a specific operation (e.g., pushing) [2, 3, 4, 5].

There exist few researches on manipulation planning that is not restricted to specific operations. Trinkle et al. defined "First-Order Stability Cell" for whole arm manipulation, and suggested the possibility of manipulation planning with the cell [6]. However, the details of the planning method is unclear and it is expected to require massive computation. Erdmann dealt with manipulation including sliding and tumbling, referred to as "Two-Palm Manipulation" [7], which is still under a restriction that contacts must be so-called "Type-A."

In this paper, in order to plan graspless manipulation, we adopt the following two-step approach:

1. Construction of a contact state graph [8] and simplification of it with rough estimation of manipulation cost.
2. Detailed planning of manipulation with mechanics in the circumscribed domain by the step 1.

In the step 1, we ignore robots and consider contact-state transition only between the object and the environment. Under rough cost estimation for contact state graph, we can obtain a sequence of contact-state transition by graph searching [9, 10]. The result is, however, just a candidate

0-7803-6475-9/01/$10.00© 2001 IEEE

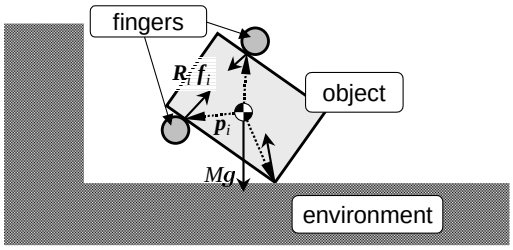
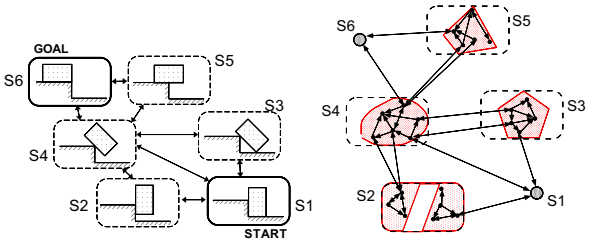


Figure 2: Model of Manipulation



(a) Contact State Graph (b) Manipulation-Feasibility Graph

Figure 3: Generation of Manipulation-Feasibility Graph

for lack of considering mechanics. So we generate simplified contact state graph that contains promising paths for manipulation. The simplification process can be formulated as a *k* shortest paths problem [11].

In the step 2, considering robots and mechanics, we elaborate this simplified contact state graph into a detailed graph, which we call "manipulation-feasibility graph." By searching this graph, we obtain a solution for graspless manipulation, if any.

We can find researches on the generation of contact state graphs and object motion planning (without robots) using the graphs in [8, 9, 10, 12]. In this paper, we exploit their results and concentrate on the detailed planning in the step 2.

2 Problem Settlement

In this paper, supposing planar graspless manipulation by multi-fingered robot hands, we make the following assumptions (Figure 2):

1. Manipulation is quasi-static.
2. Manipulated object and environment are planar polygonal rigid bodies.
3. Robot fingers are modeled as circular rigid bodies.
4. Distances between robot fingers have an upper limit.
5. Friction follows Coulomb's law. Friction coefficient on the same side of each body is uniform. Static and kinetic friction coefficients are equal.
6. Normal component of each contact force has an upper limit to avoid destruction of the object.
7. Slipping and rolling of robot fingers on the object surface is neglected. Regrasping is required to change the position of fingers on the object.
8. A (simplified) contact-state graph for the object is given.

Planning problem to be solved is:

> to find a sequence of positions and contact force of robot fingers to manipulate an object from an initial configuration to a goal configuration in the given contact state graph (Figure 3).

3 Outline of Planning Method

Our planning algorithm represents the C-Space of the object and robot fingers as a digraph, and reduces the planning problem to graph searching. Our method consists of the following procedures:

1. Determination of finger forces based on the evaluation of manipulation stability (Section 4). We make a test on each sampled point in C-Space for the feasibility of manipulation, and decide finger forces.
2. Generation of a manipulation-feasibility graph that represents graspless manipulation in a single contact state (Section 5). We generate nodes of the graph by discretizing the C-Subspace for each contact state, and connect the nodes by arcs.
3. Connection of manipulation-feasibility graphs for the representation of contact state transition (Section 6). We connect graphs generated in the procedure 2 into a single big graph.
4. Planning of graspless manipulation by searching the constructed graph (Section 7).

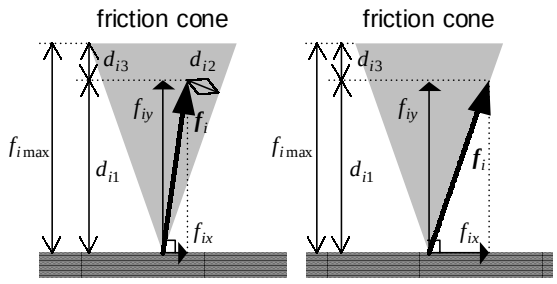
Figure 3 is a schematic view of generation of a manipulation-feasibility graph.

4 Determination of Finger Forces

4.1 Evaluation of Manipulation Stability

Graspless manipulation is inferior to pick-and-place in terms of the stability of manipulation. Therefore, we determine finger forces for given finger positions so that an index for the stability of manipulation is maximized.

Kerr and Roth determined finger forces for grasping in order to make grasp forces the furthest from the boundaries of constraints [13]. We modify their method to adapt



i) non-sliding contact ii) sliding contact

Figure 4: Model of Contacts

to graspless manipulation so that it can deal with sliding contacts. Although we formulate the problem only for planar cases in the following, spatial cases can be also formulated in the same manner by polygonal approximation of friction cones [14].

Let us consider n contact points of the object with robot fingers or environment. Figure 4 illustrates a friction cone at the i-th contact. Friction coefficient at the contact is denoted by μ_i. We set the local reference frame so that its y-axis coincides to the direction of the contact normal. The contact force represented in this reference frame, $\boldsymbol{f}_i = [f_{ix}, f_{iy}]^T$, must conform to the following conditions:
(for non-sliding contacts)

$$d_{i1} = f_{iy} \geq 0 \quad (1)$$

$$d_{i2} = \frac{\mu f_{iy} - |f_{ix}|}{\sqrt{1+\mu_i^2}} \geq 0 \quad (2)$$

$$d_{i3} = f_{i\max} - f_{iy} \geq 0 \quad (3)$$

(for sliding contacts)

$$d_{i1} = f_{iy} \geq 0 \quad (4)$$

$$d_{i2} = \frac{\mu f_{iy} - \sigma_i f_{ix}}{\sqrt{1+\mu_i^2}} = 0 \quad (5)$$

$$d_{i3} = f_{i\max} - f_{iy} \geq 0 \quad (6)$$

where we denote the upper limit of finger force in the direction of contact normal by $f_{i\max}$. σ_i is 1 or -1 according to the sliding direction.

Here we define an index d_i that evaluates the margin of each contact force as:

$$d_i = \begin{cases} \min\limits_{j=1,2,3}(d_{ij}) & \text{(for non-sliding contacts)} \\ \min\limits_{j=1,3}(d_{ij}) & \text{(for sliding contacts).} \end{cases} \quad (7)$$

Then we define an index of the manipulation stability d as:

$$d = \min_i(d_i). \quad (8)$$

d corresponds to the minimum error of contact forces that permitted for the manipulation. $d < 0$ indicates that the manipulation is mechanically infeasible.

4.2 Calculation of Finger Forces

We can obtain contact forces which maximize our index d by solving a following linear programming problem [13, 14]. We rewrite the equations (1)-(6) as

$$\boldsymbol{A}_i \boldsymbol{f}_i - \boldsymbol{c}_i \geq \boldsymbol{0} \quad (9)$$

$$\boldsymbol{B}_i \boldsymbol{f}_i = 0, \quad (10)$$

where

$$\boldsymbol{A}_i = \begin{cases} \begin{bmatrix} 0 & 1 \\ 0 & -1 \\ \frac{-1}{\sqrt{1+\mu_i^2}} & \frac{\mu_i}{\sqrt{1+\mu_i^2}} \\ \frac{1}{\sqrt{1+\mu_i^2}} & \frac{\mu_i}{\sqrt{1+\mu_i^2}} \end{bmatrix} & \text{(for non-sliding contacts)} \\ \begin{bmatrix} 0 & 1 \\ 0 & -1 \end{bmatrix} & \text{(for sliding contacts)} \end{cases}$$

$$\boldsymbol{B}_i = \begin{cases} \begin{bmatrix} 0 & 0 \end{bmatrix} & \text{(for non-sliding contacts)} \\ \begin{bmatrix} \frac{-\sigma_i}{\sqrt{1+\mu_i^2}} & \frac{\mu_i}{\sqrt{1+\mu_i^2}} \end{bmatrix} & \text{(for sliding contacts)} \end{cases}$$

$$\boldsymbol{c}_i = \begin{cases} [0, -f_{i\max}, 0, 0]^T & \text{(for non-sliding contacts)} \\ [0, -f_{i\max}]^T & \text{(for sliding contacts)} \end{cases}$$

The linear programming problem to be solved is:

$$\text{maximize } d = \lambda^T \boldsymbol{d}$$

$$\text{subject to} \begin{cases} \boldsymbol{W}\boldsymbol{R}\boldsymbol{f} = M\boldsymbol{g} \\ \boldsymbol{A}\boldsymbol{f} - \boldsymbol{c} \geq \boldsymbol{d} \\ \boldsymbol{B}\boldsymbol{f} = \boldsymbol{0} \\ \boldsymbol{d} \geq \boldsymbol{0} \end{cases}$$

where

$$\boldsymbol{W} = \begin{bmatrix} \boldsymbol{I} & \cdots & \boldsymbol{I} \\ \boldsymbol{p}_1 \times \boldsymbol{I} & \cdots & \boldsymbol{p}_n \times \boldsymbol{I} \end{bmatrix}, \boldsymbol{R} = \begin{bmatrix} \boldsymbol{R}_1 & & \boldsymbol{O} \\ & \ddots & \\ \boldsymbol{O} & & \boldsymbol{R}_n \end{bmatrix}$$

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_1 & & \boldsymbol{O} \\ & \ddots & \\ \boldsymbol{O} & & \boldsymbol{A}_n \end{bmatrix}, \boldsymbol{B} = \begin{bmatrix} \boldsymbol{B}_1 & & \boldsymbol{O} \\ & \ddots & \\ \boldsymbol{O} & & \boldsymbol{B}_n \end{bmatrix}$$

$$\boldsymbol{c} = [\boldsymbol{c}_1^T, \ldots, \boldsymbol{c}_n^T]^T, \boldsymbol{d} = [d, \ldots, d]^T$$

$$\boldsymbol{0} = [0, \ldots, 0]^T, \lambda = [1, 0, \ldots, 0]^T.$$

$\boldsymbol{I}$ is 2×2 unit matrix, $\boldsymbol{p}_i$ is the relative position of the i-th contact from the centroid of the object, and $\boldsymbol{R}_i$ is the matrix that converts $\boldsymbol{f}_i$ in the local reference frame of the contact to the force represented in the world frame. M is the mass of the object, and $\boldsymbol{g}$ is the gravity acceleration vector. For simplicity, we formulated the linear programming problem straightforward. We can reduce it to a smaller problem using pseudoinverse matrix [13, 14].

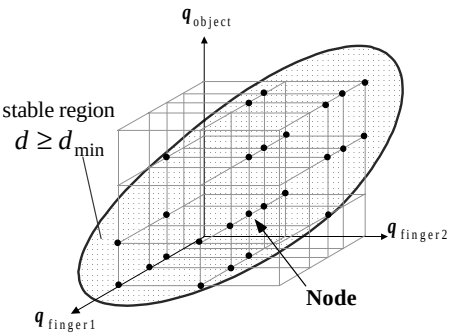


Figure 5: Generation of Nodes

Given the positions of the contacts, we determine the finger forces so that d is maximized. Note that the maximized value means an ideal value of d when the contact forces with environment are indeterminate.

5 Manipulation-Feasibility Graph in a Single Contact State

5.1 Representation of C-Space

Here we try to define the C-Space that represents the degrees of freedom for both the object and the robot fingers. We represent the positions of the robot fingers in manipulation as the positions on the surface of the polygonal object. Therefore, each robot finger has one degree-of-freedom. The planar object has three degree-of-freedom, so the dimension of the C-Space is $(3+N)$, where N is the number of the robot fingers. The manipulation planning problem results in a searching problem in this C-Space.

However, we cannot search the C-Space as conventional obstacle-avoidance problems, because:

- Even if the movement of a point to the other point in the C-Space is feasible, the reverse movement may be infeasible.
- Regrasping causes discontinuous movement of robot fingers in the C-Space.

To overcome these problems, we approximately represent the C-Space by a digraph. That is, we construct nodes of a graph by discretizing the C-Space, and connect between the nodes by arcs.

5.2 Generation of Nodes

In a single contact state, the degree of freedom of the object is one or two, so we only have to consider the C-Subspace whose dimension is $(1+N)$ or $(2+N)$. Here we denote the coordinates of the object in this C-Subspace by $\boldsymbol{q}_{\text{object}}$ and those of robot fingers by $\boldsymbol{q}_{\text{finger1}}, \ldots, \boldsymbol{q}_{\text{finger}N}$.

Lattice points in this C-Subspace are sampled as the candidates of the nodes. We adopt each sampled point as a valid node if geometrical constraints (see Section 2) are satisfied and the manipulation stability d is greater than zero at the point (Figure 5).

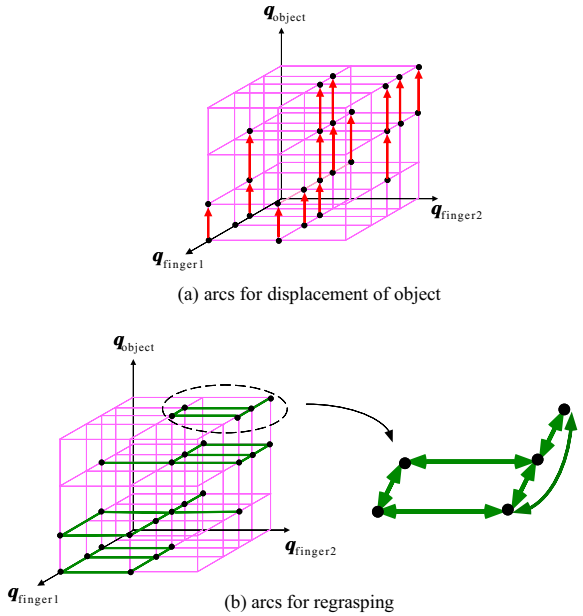


Figure 6: Generation of Arcs

5.3 Generation of Arcs

Manipulation-feasibility graphs have two kinds of arcs: for displacement of the object and for regrasping. We describe how to generate these arcs below.

Arcs for displacement of the object correspond to moving the object without changing the positions of the robot fingers on the surface of the object. These arcs connect adjacent nodes in the C-Subspace for the displacement of the object (Figure 6 (a)). For reliable quasi-static manipulation, we set a lower limit of the manipulation stability, $d_{\min}$. We sample several points on each arc and calculate d at the points. Unless $d > d_{\min}$ at all the points, the arc is discarded. If there exist no sliding contacts, arcs are bidirectional. Otherwise, we have to make individual tests for the directions of the sliding and generate a directed arc for each.

Arcs for regrasping correspond to changing a position of a finger on the object with neither moving the object nor changing positions of the other fingers. Note that the arcs for regrasping may be generated between non-adjacent nodes, because the position of the regrasping finger on the object changes discontinuously. At each node in the C-Subspace, we calculate d when a regrasping finger is removed. If $d > d_{\min}$, the finger can freely change its position on the object. We generate bidirectional arcs between all the nodes whose coordinates are the same except for that of the regrasping finger (Figure 6 (b)).

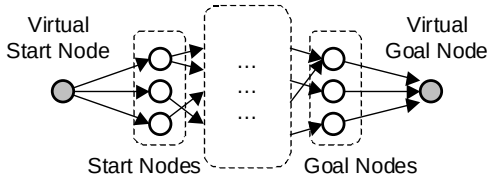


Figure 7: Virtual Start/Goal Node

After the generation of all the arcs, we have a manipulation-feasibility graph for a single contact state.

6 Representation of Contact-State Transition

For planning of graspless manipulation over several contact states, we have to generate a big manipulation-feasibility graph by connecting graphs for all the possible contact states. We can connect two nodes in different manipulation-feasibility graphs when both nodes represent an identical configuration, that is, an instant of contact state transition. To avoid complication, C-Subspaces for different contact states should be discretized in the same manner. By connecting nodes for all the contact state transitions, we obtain a big manipulation-feasibility graph over multiple contact states.

7 Planning of Graspless Manipulation

7.1 Search of Manipulation-Feasibility Graph

In this section, we do planning of graspless manipulation by searching manipulation-feasibility graphs. Our planning problem does specify the initial and goal configurations of the object, and not those of the robot fingers. Therefore, we have multiple start nodes and goal nodes in a manipulation-feasibility graph. To unite the start (or goal) nodes into one, we add a virtual start (or goal) node that is connected to all the start (or goal) nodes (Figure 7).

Planning of graspless manipulation can be carried out by assigning cost to each arc on the manipulation-feasibility graph, and searching the minimum-cost path from the (virtual) start node to the (virtual) goal node by Dijkstra's method. In this paper, we decide the assignment of costs according to the following policies:

- Avoid manipulation with low stability.
- Minimize the number of regrasping.
- Minimize the load of the robot fingers if the number of regrasping is equal.

At first, we calculate manipulation stability d at both ends of each arc. If $d \leq d_{\min}$ at either end, the cost of arc shall be ∞. When $d > d_{\min}$, the cost assigned to arcs for the

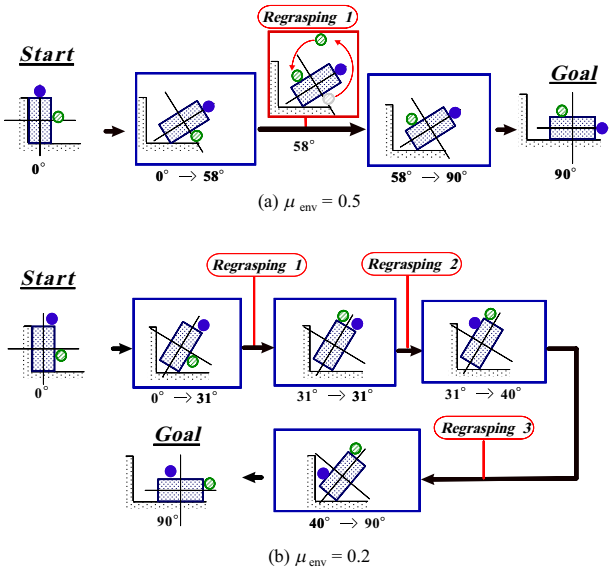


Figure 8: Planned Tumbling Operation

displacement of the object, c_{disp} is:

$$c_{\text{disp}} = \sum_j |\boldsymbol{f}_{\text{finger}j}||\Delta\boldsymbol{q}_{\text{finger}j}| \tag{11}$$

where $\boldsymbol{f}_{\text{finger}j}$ is the force vector of j-th finger, and $\Delta\boldsymbol{q}_{\text{finger}j}$ is the displacement of the finger for the arc. c_{disp} is a quantitative measure for the load of the robot fingers.

The cost for arcs for regrasping, c_{regr} is:

$$c_{\text{regr}} = X_{\text{regr}} \tag{12}$$

where X_{regr} is a much greater constant than c_{disp}. By the above cost assignment, we place a first priority on minimizing the number of regrasping, and a second on minimizing the load of the robot fingers (in the sense of eq. (11)).

7.2 Examples of Planned Manipulation

Our planning algorithm is implemented on UNIX workstations using C language.

Let us consider a rectangular object whose size is 4×2 and weight is 1. Two robot fingers manipulate this object. Each of the fingers can be represented as a circle whose radius is 1. The distance between the fingers must be smaller than 2. Friction coefficient between the object and the fingers is 0.5, $f_{i\max} = 10$, $X_{\text{regr}} = 10^7$, and $d_{\min} = 0.05$.

As a manipulation in a single contact state, tumbling operations are planned when friction coefficient between the object and environment, μ_{env}, is 0.5 and 0.2 (Figure 8). When $\mu_{\text{env}} = 0.2$, more "careful" operation is generated with more times of regrasping because the object is slippery. In the case that $\mu_{\text{env}} = 0.5$, our program generated 4,166 valid nodes among 33,201 node candidates,

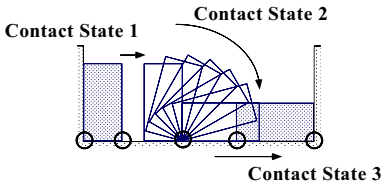


Figure 9: Manipulation over Three Contact States

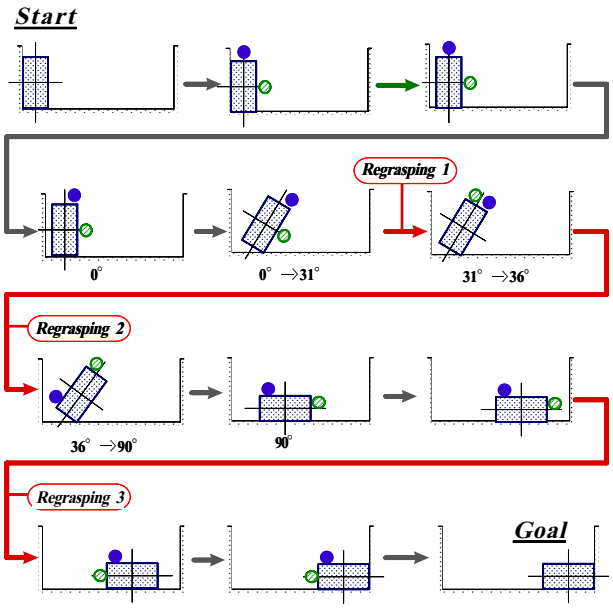


Figure 10: Planned Composite Manipulation

and spent time for the planning was 20 CPU seconds on a 334MHz UltraSPARC-IIi workstation.

Next we planned a graspless manipulation over multiple contact states shown as Figure 9. Figure 10 is the result when $\mu_{\text{env}} = 0.2$. Generated manipulation is a combination of sliding-tumbling-sliding with four-time regrasping. It took about 330 CPU seconds on the same workstation to obtain the above result. In that case, the number of node candidates was 506,368 and the number of valid nodes was 78,172.

8 Conclusion

We presented a planning method for graspless manipulation by multiple robot fingers. Based on mechanical analysis, our algorithm successfully generated tumbling and sliding operations with regrasping in planar cases.

A drawback of current implementation is that it does not allow the sliding and rolling contacts between the object and the fingers. Rolling contacts can enhance the ability of manipulation of the fingers, therefore we are trying to incorporate rolling contacts of the fingers into our method as a variation of regrasping.

References

[1] Y. Aiyama et al.: "Pivoting: A New Method of Graspless Manipulation of Object by Robot Fingers," Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp.136–143, 1993.

[2] N. Sawasaki et al.: "Tumbling Objects Using a Multi-Fingered Robot," Proc. of 20th Int. Symp. on Industrial Robots, pp.609–616, 1989.

[3] H. Terasaki and T. Hasegawa: "Intelligent Manipulation of Sliding Operations with Parallel Two-Fingered Grippers," Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp.1590–1597, 1993.

[4] M. Kurisu and T. Yoshikawa: "Trajectory Planning of an Object in Pushing Operation," Proc. of Japan-USA Symp. on Flexible Automation, pp.1009–1016, 1994.

[5] K. Lynch and M. Mason: "Stable Pushing: Mechanics, Controllability, and Planning," Int. J. of Robotics Research, Vol.15, No.6, pp.533–556, 1996.

[6] J. Trinkle et al.: "First-Order Stability Cells of Active Multi-Rigid-Body Systems," IEEE Trans. on Robotics and Automation, Vol.11, No.4, pp.545–557, 1995.

[7] M. Erdmann: "An Exploration of Nonprehensile Two-Palm Manipulation," Int. J. of Robotics Research, Vol.17, No.5, pp.485–503, 1998.

[8] S. Hirai: "Analysis and Planning of Manipulation Using the Theory of Polyhedral Convex Cones," Ph.D. Dissertation, Kyoto University, 1991.

[9] Y. Yu et al.: "Two Kinds of Degree of Freedom in Constraint State and Their Application to Assembly Planning," Proc. of IEEE Int. Conf. on Robotics and Automation, pp.1993–1999, 1996.

[10] Y. Aiyama et al.: "Contact-State Transition Graph for Graspless Manipulation Planning," Trans. of Japan Soc. of Mechanical Engineers, Series C, Vol.65, No.636, pp.3239–3244, 1999 (in Japanese).

[11] D. Eppstein: "Finding the k Shortest Paths," SIAM J. on Computing, Vol.28, No.2, pp.652–673, 1998.

[12] J. Xiao and X. Ji: "A Divide-and-Merge Approach to Automatic Generation of Contact States and Planning of Contact Motion," Proc. of IEEE Int. Conf. on Robotics & Automation, pp.750–756, 2000.

[13] J. Kerr and B. Roth: "Analysis of Multifingered Hands," Int. J. of Robotics Research, Vol.4, No.4, pp.3–17, 1986.

[14] H. Kijimoto et al.: "Performance Analysis and Planning of Graspless Manipulation," Proc. of IEEE Int. Symp. on Assembly and Task Planning, pp.238–243, 1999.