This is an electronic version of an article published
in ADVANCED ROBOTICS, Volume 19, No. 5, pp. 501-521, 2005.
ADVANCED ROBOTICS is available online at:
www.tandfonline.com/10.1163/156855305323383776

Planning of Graspless Manipulation by Multifingered Robot Hand

Yusuke MAEDA Tamio ARAI

maeda@ynu.ac.jp

Abstract

Graspless (or nonprehensile) manipulation is a method to manipulate objects without grasping. It includes pushing, tumbling and pivoting. Graspless manipulation is important as a complement of conventional pick-and-place for the dexterity of robots.

In this paper, we propose a method of planning of general graspless manipulation by multifingered robot hands; the method can automatically generate various graspless operations by robot fingers including regrasping. In this method, a graph that represents feasible manipulation is constructed based on mechanical analysis, and a manipulation plan is obtained by A* search of the graph. By considering whether each robot finger should be position- or force-controlled, we can obtain robust manipulation plans against external disturbances.

We present some examples of planned graspless manipulation of a cuboid by two robot fingers; our planner successfully generates sliding, pushing and tumbling operations. We also show an experimental result of execution of planned graspless manipulation by a robot with a multifingered hand.

keywords: graspless manipulation, manipulation planning, multifingered robot hand

1 Introduction

[Figure 1 about here.]

Manipulation without grasping is referred to as graspless manipulation [1] or nonprehensile manipulation [2]. In this paper, we study graspless manipulation where the manipulated object is supported not only by robot fingers but also by the environment; it includes pushing, sliding and tumbling (Figure 1). Graspless manipulation brings the following advantages to robots:

- Manipulation without supporting all the weight of the object.
- Manipulation with simple mechanisms.
- Manipulation when grasping is impossible.

Thus graspless manipulation is important as a complement of conventional pick-and-place to enhance the dexterity of robots.

Planning of robot motion to move an object from an initial configuration to a goal is a fundamental problem in robotic manipulation. Robot motion planning for graspless manipulation is, however, much more difficult than that for pick-and-place. In pick-and-place operation, once an object is grasped, the correspondence of its motion to the robot motion is trivial; therefore manipulation planning is reduced to a geometrical collision avoidance. On the other hand, the correspondence in graspless manipulation is nontrivial because it highly depends on mechanical effect such as gravity and friction; thus manipulation planning requires not only geometrical but also mechanical analysis. Moreover, graspless manipulation may be irreversible: For example, a robot may be able to push an object but may not be able to pull it back.

Because of these difficulties, most of studies related to planning of graspless manipulation fall into the following categories:

Motion planning of manipulated objects only. A method for planning object motion in contact with the environment is presented, but robot motion required to achieve the object motion is not explicitly considered [3–6].

Planning of manipulation by a specific operation. A planning scheme specialized for a specific graspless operation is presented. For example, pushing [7–9], tumbling [10–12], and others [13–15].

By contrast, in this paper, we present a planning method for general graspless manipulation for multi-fingered robot hands; the method can generate motion of robot fingertips to achieve various graspless operations such as pushing and tumbling.

This paper is organized as follows: Section 2 introduces a model of graspless manipulation. Section 3 describes a method to determine appropriate finger control modes (force-control mode or position-control mode) at an instant in graspless manipulation. This method is used to check the local feasibility of manipulation. Section 4 proposes a method of motion planning of robot fingertips for graspless manipulation. In Section 5, some examples of planned graspless manipulation including pushing and tumbling are presented. We also show an experimental result of execution of planned manipulation by a robot with a multifingered hand. Finally, this paper is concluded in Section 6.

2 Model of Graspless Manipulation

2.1 Assumptions and Problem Statement

In this paper, for graspless manipulation by multifingered robot hands, we make the following assumptions:

1. The manipulated object, robot fingertips, and the environment are rigid.
2. Manipulation is quasi-static.

3. Coulomb friction exists between the object and the environment (or the robot fingertips). The friction coefficient on a contact surface is uniform.

4. Static and kinetic friction coefficients are equal.

5. All the contacts can be approximated by finite point contacts.

6. Each of friction cones can be approximated by a polyhedral convex cone [16].

7. Each robot finger is modeled as a rigid sphere and is in one-point non-sliding contact with the object; we consider only fingertips.

8. The normal component of each finger force has an upper limit.

9. Each robot finger is either in "position-control mode" or in "force-control mode":

   (a) Each of robot fingers in position-control mode can apply arbitrary force *passively* within its friction cone and the upper limit of its normal component.

   (b) Each of robot fingers in force-control mode is in hybrid position/force control [17, 18]; the finger can apply commanded normal force *actively* within its upper limit and arbitrary tangential force *passively* within its friction cone.

10. Sliding and rolling of robot fingers on object surfaces is not allowed. Regrasping is required to change locations of fingertips on the object.

The problem to be solved is to determine a sequence of fingertip positions and finger control modes to move an object from a given initial configuration to a given goal configuration by graspless manipulation. A sequence of desired normal forces is also to be obtained for fingers in force-control mode.

2.2 Mechanical Model

[Figure 2 about here.]

Consider graspless manipulation of an object as in Figure 2. We set an object reference frame whose origin coincides with the center of mass of the object. Let $\boldsymbol{p}_{\mathrm{env}\,1}, \ldots, \boldsymbol{p}_{\mathrm{env}\,m} \in \Re^3$ be positions of contact points between the object and the environment. Similarly, let $\boldsymbol{p}_{\mathrm{rob}\,1}, \ldots, \boldsymbol{p}_{\mathrm{rob}\,n} \in \Re^3$ be positions of contact points between the object and the robot finger $1, \ldots, n$. We denote unit normal vectors at contact point $\boldsymbol{p}$ inward to the object by $\boldsymbol{n}(\boldsymbol{p}) \in \Re^3$.

Let us denote the sets of positions of sliding and non-sliding contacts by $\boldsymbol{\mathcal{C}}_{\mathrm{slide}}$ and $\boldsymbol{\mathcal{C}}_{\mathrm{stat}}$, respectively. We can identify whether $\boldsymbol{p}_{\mathrm{env}\,i} \in \boldsymbol{\mathcal{C}}_{\mathrm{slide}}$ or $\boldsymbol{p}_{\mathrm{env}\,i} \in \boldsymbol{\mathcal{C}}_{\mathrm{stat}}$ when the object motion is specified. We approximate each friction cone at contact point $\boldsymbol{p}$ by a polyhedral convex cone with unit edge vectors, $\boldsymbol{c}_1(\boldsymbol{p}), \ldots, \boldsymbol{c}_s(\boldsymbol{p}) \in \Re^3$. For $\boldsymbol{p}_{\mathrm{env}\,i} \in \boldsymbol{\mathcal{C}}_{\mathrm{slide}}$, let $\boldsymbol{c}'(\boldsymbol{p}_{\mathrm{env}\,i}) \in \Re^3$ be a unit edge vector of the friction cone at contact point $\boldsymbol{p}_{\mathrm{env}\,i}$ opposite to its sliding direction.

Then the set of possible contact force $\boldsymbol{f} \in \Re^3$ at $\boldsymbol{p}_{\mathrm{env}\,i}$ can be written as follows:

$$\begin{cases} \{\boldsymbol{f} | \boldsymbol{f} \in \mathrm{span}\{\boldsymbol{c}_1(\boldsymbol{p}_{\mathrm{env}\,i}), \ldots, \boldsymbol{c}_s(\boldsymbol{p}_{\mathrm{env}\,i})\}\} & \text{if } \boldsymbol{p}_{\mathrm{env}\,i} \in \boldsymbol{\mathcal{C}}_{\mathrm{stat}}, \\ \{\boldsymbol{f} | \boldsymbol{f} \in \mathrm{span}\{\boldsymbol{c}'(\boldsymbol{p}_{\mathrm{env}\,i})\}\} & \text{if } \boldsymbol{p}_{\mathrm{env}\,i} \in \boldsymbol{\mathcal{C}}_{\mathrm{slide}}, \end{cases} \tag{1}$$

where $\mathrm{span}\{\ldots\}$ is a polyhedral convex cone spanned by its element vectors [16]. On the other hand, the set of possible finger force $\boldsymbol{f}$ at $\boldsymbol{p}_{\mathrm{rob}\,i}$ is:

$$\begin{cases} \{\boldsymbol{f} | \boldsymbol{f} \in \mathrm{span}\{\boldsymbol{c}_1(\boldsymbol{p}_{\mathrm{rob}\,i}), \ldots, \boldsymbol{c}_s(\boldsymbol{p}_{\mathrm{rob}\,i})\},\ \boldsymbol{n}(\boldsymbol{p}_{\mathrm{rob}\,i})^T \boldsymbol{f} \leq f_{\max i}\} \\ \qquad\qquad \text{if robot finger } i \text{ is in position-control mode,} \\ \{\boldsymbol{f} | \boldsymbol{f} \in \mathrm{span}\{\boldsymbol{c}_1(\boldsymbol{p}_{\mathrm{rob}\,i}), \ldots, \boldsymbol{c}_s(\boldsymbol{p}_{\mathrm{rob}\,i})\},\ \boldsymbol{n}(\boldsymbol{p}_{\mathrm{rob}\,i})^T \boldsymbol{f} = f_{\mathrm{com}\,i} \leq f_{\max i}\} \\ \qquad\qquad \text{if robot finger } i \text{ is in force-control mode,} \end{cases} \tag{2}$$

where $f_{\max i}$ is the upper limit of the normal component of the finger force and $f_{\mathrm{com}\,i}$ is the commanded normal force for robot finger i.

Then we define the following matrices:

$$\begin{aligned} \boldsymbol{W}_{\mathrm{env}} &:= \begin{bmatrix} \boldsymbol{I}_3 & \ldots & \boldsymbol{I}_3 \\ \boldsymbol{p}_{\mathrm{env}\,1} \times \boldsymbol{I}_3 & \ldots & \boldsymbol{p}_{\mathrm{env}\,m} \times \boldsymbol{I}_3 \end{bmatrix} \in \Re^{6 \times 3m} \\ \boldsymbol{C}_{\mathrm{env}} &:= \mathrm{blockdiag}(\boldsymbol{C}_{\mathrm{env}\,1}, \ldots, \boldsymbol{C}_{\mathrm{env}\,m}) \\ \boldsymbol{C}_{\mathrm{env}\,i} &:= \begin{cases} [\boldsymbol{c}_1(\boldsymbol{p}_{\mathrm{env}\,i}) \ldots \boldsymbol{c}_s(\boldsymbol{p}_{\mathrm{env}\,i})] \in \Re^{3 \times s} & \text{if } \boldsymbol{p}_{\mathrm{env}\,i} \in \boldsymbol{\mathcal{C}}_{\mathrm{stat}}, \\ [\boldsymbol{c}'(\boldsymbol{p}_{\mathrm{env}\,i})] \in \Re^{3 \times 1} & \text{if } \boldsymbol{p}_{\mathrm{env}\,i} \in \boldsymbol{\mathcal{C}}_{\mathrm{slide}}. \end{cases} \\ \boldsymbol{W}_{\mathrm{rob}} &:= \begin{bmatrix} \boldsymbol{I}_3 & \ldots & \boldsymbol{I}_3 \\ \boldsymbol{p}_{\mathrm{rob}\,1} \times \boldsymbol{I}_3 & \ldots & \boldsymbol{p}_{\mathrm{rob}\,n} \times \boldsymbol{I}_3 \end{bmatrix} \in \Re^{6 \times 3n} \\ \boldsymbol{C}_{\mathrm{rob}} &:= \mathrm{blockdiag}(\boldsymbol{C}_{\mathrm{rob}\,1}, \ldots, \boldsymbol{C}_{\mathrm{rob}\,n}) \in \Re^{3n \times ns} \\ \boldsymbol{C}_{\mathrm{rob}\,i} &:= [\boldsymbol{c}_1(\boldsymbol{p}_{\mathrm{rob}\,i}) \ldots \boldsymbol{c}_s(\boldsymbol{p}_{\mathrm{rob}\,i})] \in \Re^{3 \times s} \\ \boldsymbol{N}_{\mathrm{rob}} &:= \mathrm{blockdiag}(\boldsymbol{n}(\boldsymbol{p}_{\mathrm{rob}\,1}), \ldots, \boldsymbol{n}(\boldsymbol{p}_{\mathrm{rob}\,n})) \in \Re^{3n \times n}, \end{aligned}$$

where $\boldsymbol{I}_3$ is the 3×3 identity matrix; $\boldsymbol{p} \times \boldsymbol{I}_3 \in \Re^{3 \times 3}$ is a skew-symmetric matrix defined such that $(\boldsymbol{p} \times \boldsymbol{I}_3)\boldsymbol{x} \equiv \boldsymbol{p}_i \times \boldsymbol{x}$; $\mathrm{blockdiag}\{\ldots\}$ is a block diagonal matrix composed of its elements. $\boldsymbol{W}_{\mathrm{env}}$ and $\boldsymbol{W}_{\mathrm{rob}}$ are wrench matrices.

The equilibrium equation of the object in quasi-static graspless manipulation can be expressed as:

$$\boldsymbol{Q}_{\mathrm{unknown}} + \boldsymbol{Q}_{\mathrm{known}} + \boldsymbol{W}_{\mathrm{env}} \boldsymbol{C}_{\mathrm{env}} \boldsymbol{k}_{\mathrm{env}} + \boldsymbol{W}_{\mathrm{rob}} \boldsymbol{C}_{\mathrm{rob}} \boldsymbol{k}_{\mathrm{rob}} = \boldsymbol{0}, \tag{3}$$

where $\boldsymbol{Q}_{\mathrm{unknown}} \in \Re^6$ is an unknown external (generalized) force applied to the object by disturbances; $\boldsymbol{Q}_{\mathrm{known}} \in \Re^6$ is a known external (generalized) force such as gravity; $\boldsymbol{k}_{\mathrm{env}} (\geq \boldsymbol{0})$ and $\boldsymbol{k}_{\mathrm{rob}} (\geq \boldsymbol{0})$ are coefficient vectors to represent contact forces. $\boldsymbol{W}_{\mathrm{env}} \boldsymbol{C}_{\mathrm{env}} \boldsymbol{k}_{\mathrm{env}}$ and $\boldsymbol{W}_{\mathrm{rob}} \boldsymbol{C}_{\mathrm{rob}} \boldsymbol{k}_{\mathrm{rob}}$ are total generalized forces exerted by contact points with the environment and the robot fingertips, respectively.

The upper limitation on the magnitude of normal finger forces can be written as:

$$\boldsymbol{N}_{\rm rob}^T \boldsymbol{C}_{\rm rob} \boldsymbol{k}_{\rm rob} \leq \boldsymbol{f}_{\rm max}, \tag{4}$$

where $\boldsymbol{f}_{\rm max} := [f_{\rm max\,1}, \ldots, f_{\rm max\,n}]^T \in \Re^n$.

3 Determination of Finger Control Modes

3.1 Overview

Robotic contact tasks such as graspless manipulation are usually performed by force-controlled robots to avoid excessive internal force. In some cases, however, force control is inappropriate in terms of manipulation stability; even minute disturbance could perturb the path of the manipulated object. Pushing operation on a plane is a typical example and therefore usually performed by a position-controlled pusher (for example, "stable push" [9]).

Thus, in planning of general graspless manipulation, we should consider control modes of robot fingers for appropriate use of both position control and force control according to situation to achieve robust manipulation. We incorporate a method of determination of finger control modes at each instant [19] into our graspless manipulation planner. This method is used to test whether manipulation at an instant is feasible or not in planning.

The procedure of the control mode determination is to find the "optimal" combination of finger control modes (and desired normal forces for force-controlled fingers). We search a combination of control modes for all the robot fingers that maximizes a performance index of manipulation stability [20], as far as excessive internal force could not be generated. A brief description of the method is given below. See [19] for more details.

3.2 Possibility of Excessive Internal Force

The possibility of excessive internal force applied to the object can be judged by the following linear programming problem [21]:

$$\underset{\boldsymbol{k}_{\rm env},\ \boldsymbol{k}_{\rm rob}}{\text{maximize}}\ \boldsymbol{b}_{\rm env}^T \boldsymbol{k}_{\rm env} + \boldsymbol{b}_{\rm rob}^T \boldsymbol{k}_{\rm rob} \tag{5}$$

$$\text{subject to} \begin{cases} \boldsymbol{W}_{\rm env} \boldsymbol{C}_{\rm env} \boldsymbol{k}_{\rm env} + \boldsymbol{W}_{\rm rob} \boldsymbol{A}_{\rm pos} \boldsymbol{C}_{\rm rob} \boldsymbol{k}_{\rm rob} = \boldsymbol{0} \\ \boldsymbol{k}_{\rm env} \geq \boldsymbol{0},\ \boldsymbol{k}_{\rm rob} \geq \boldsymbol{0}, \end{cases}$$

where

$$\boldsymbol{b}_{\rm env} := [1, \ldots, 1]^T$$

$$\boldsymbol{b}_{\rm rob} := [\boldsymbol{b}_{\rm rob\,1}^T, \ldots, \boldsymbol{b}_{\rm rob\,n}^T]^T \in \Re^{ns}$$

$$\boldsymbol{b}_{\rm rob\,i} := \begin{cases} [1, \ldots, 1]^T \in \Re^s & \text{if finger } i \text{ is in position-control mode,} \\ [0, \ldots, 0]^T \in \Re^s & \text{if finger } i \text{ is in force-control mode;} \end{cases}$$

$\boldsymbol{A}_{\rm pos}$ is a selection matrix defined as follows:

$$\boldsymbol{A}_{\rm pos} := {\rm blockdiag}(a_1\boldsymbol{I}_3, \ldots, a_n\boldsymbol{I}_3) \in \Re^{3n\times 3n},$$

where

$$a_i := \begin{cases} 1 & \text{if finger } i \text{ is in position-control mode,} \\ 0 & \text{if finger } i \text{ is in force-control mode.} \end{cases}$$

When this linear programming problem is bounded, then the magnitude of internal force is also bounded; that is, excessive internal force could not be generated. Otherwise, excessive internal force might be generated because contact forces of arbitrary magnitude can cancel each other and be increased unlimitedly.

Note that Problem (5) does not include constraints on the upper limit of finger forces. When Problem (5) says that excessive internal force might be generated, forces of position-controlled fingers can be increased up to or beyond the limit unexpectedly. That may cause serious damage to the fingers and therefore we should avoid such situations.

3.3 Stability Measure for Graspless Manipulation

[Figure 3 about here.]

The performance index of manipulation stability defined in [20] evaluates how much the object can resist external disturbances without changing its motion. The value of the index, z, can be interpreted as the radius of the inscribed six-dimensional hypersphere in the set of $\boldsymbol{Q}_{\rm unknown}$ that can satisfy the quasi-static equilibrium condition, (3).

Approximating the hypersphere with a convex hyperpolyhedron (see Figure 3 for a three-dimensional schematic sketch), we can calculate the approximate value of z by solving the following linear programming problem:

$$\underset{z,\ \boldsymbol{k}_{{\rm env}\,1},\ldots,\boldsymbol{k}_{{\rm env}\,N},\ \boldsymbol{k}_{{\rm rob}\,1},\ldots,\boldsymbol{k}_{{\rm rob}\,N},\ \boldsymbol{f}_{\rm com}}{\text{maximize}} \quad z \tag{6}$$

$$\text{subject to} \begin{cases} z\boldsymbol{l}_1 = \boldsymbol{R}^{1/2}\left(\boldsymbol{Q}_{\rm known} + \boldsymbol{W}_{\rm env}\boldsymbol{C}_{\rm env}\boldsymbol{k}_{{\rm env}\,1} + \boldsymbol{W}_{\rm rob}\boldsymbol{C}_{\rm rob}\boldsymbol{k}_{{\rm rob}\,1}\right) \\ \quad\vdots \\ z\boldsymbol{l}_N = \boldsymbol{R}^{1/2}\left(\boldsymbol{Q}_{\rm known} + \boldsymbol{W}_{\rm env}\boldsymbol{C}_{\rm env}\boldsymbol{k}_{{\rm env}\,N} + \boldsymbol{W}_{\rm rob}\boldsymbol{C}_{\rm rob}\boldsymbol{k}_{{\rm rob}\,N}\right) \\ \boldsymbol{N}_{\rm rob}^T\boldsymbol{C}_{\rm rob}\boldsymbol{k}_{{\rm rob}\,1} \le \boldsymbol{f}_{\max} \\ \quad\vdots \\ \boldsymbol{N}_{\rm rob}^T\boldsymbol{C}_{\rm rob}\boldsymbol{k}_{{\rm rob}\,N} \le \boldsymbol{f}_{\max} \\ \boldsymbol{N}_{\rm rob}^T\boldsymbol{A}_{\rm force}\boldsymbol{C}_{\rm rob}\boldsymbol{k}_{{\rm rob}\,1} = \boldsymbol{f}_{\rm com} \\ \quad\vdots \\ \boldsymbol{N}_{\rm rob}^T\boldsymbol{A}_{\rm force}\boldsymbol{C}_{\rm rob}\boldsymbol{k}_{{\rm rob}\,N} = \boldsymbol{f}_{\rm com} \\ \boldsymbol{k}_{{\rm env}\,1},\ldots,\boldsymbol{k}_{{\rm env}\,N} \ge \boldsymbol{0},\ \boldsymbol{k}_{{\rm rob}\,1},\ldots,\boldsymbol{k}_{{\rm rob}\,N} \ge \boldsymbol{0}, \end{cases}$$

where $\boldsymbol{f}_{\mathrm{com}}$ is a commanded normal force vector defined as $\boldsymbol{f}_{\mathrm{com}} := [f_{\mathrm{com}\,1}, \ldots, f_{\mathrm{com}\,n}]^T \in \Re^n$; let $f_{\mathrm{com}\,i} = 0$ if finger i is in position-control mode; $\boldsymbol{k}_{\mathrm{env}\,i}$ and $\boldsymbol{k}_{\mathrm{rob}\,i}$ are coefficient vectors to represent contact forces; $\boldsymbol{A}_{\mathrm{force}}$ is a selection matrix defined as:

$$\boldsymbol{A}_{\mathrm{force}} := \boldsymbol{I}_{3n} - \boldsymbol{A}_{\mathrm{pos}} \ \in \Re^{3n \times 3n};$$

$\boldsymbol{l}_1, \ldots, \boldsymbol{l}_N \in \Re^6$ are position vectors of vertices of a six-dimensional hyperpolyhedron that circumscribes the six-dimensional unit hypersphere; $\boldsymbol{R} \in \Re^{6\times 6}$ is a positive definite matrix for scaling of force and moment, which is used to introduce the following norm for generalized forces, $\boldsymbol{Q} \in \Re^6$:

$$\|\boldsymbol{Q}\|_{\boldsymbol{R}} := \sqrt{\boldsymbol{Q}^T \boldsymbol{R} \boldsymbol{Q}} = \sqrt{(\boldsymbol{R}^{1/2}\boldsymbol{Q})^T (\boldsymbol{R}^{1/2}\boldsymbol{Q})}, \tag{7}$$

where $\boldsymbol{R}^{1/2} \in \Re^{6\times 6}$ is the Cholesky decomposition of $\boldsymbol{R}$. We can have a coordinate-invariant norm by using the following scaling matrix:

$$\boldsymbol{R} := \begin{bmatrix} \boldsymbol{I}_3 & \boldsymbol{O} \\ \boldsymbol{O} & M\boldsymbol{J}^{-1} \end{bmatrix} \in \Re^{6\times 6}, \tag{8}$$

where M is the mass of the object and $\boldsymbol{J} \in \Re^{3\times 3}$ is the inertia tensor of the object. By solving Problem (6) we can find $\boldsymbol{f}_{\mathrm{com}}$ that achieves the maximum manipulation stability for a specified selection matrix, $\boldsymbol{A}_{\mathrm{force}}$.

3.4 Procedure for Determining Finger Control Modes

The procedure to determine control modes of robot fingers (and desired normal forces for force-controlled fingers) is as follows:

1. Assume a combination of control modes (position control or force control) for each robot finger (i.e. Specify $\boldsymbol{A}_{\mathrm{force}}$).

2. Test the possibility of excessive internal force (Problem (5)). If excessive internal force may be generated, give up this combination and go to step 4.

3. Calculate desired normal finger forces ($\boldsymbol{f}_{\mathrm{com}}$) so that the value of manipulation stability, z, will be maximized (Problem (6)). If the maximized z is larger than the current maximum value, replace it.

4. If all the combinations of control modes have been already tested, stop. Otherwise, go back to step 1.

When all the procedure is completed, we have a combination of control modes that achieves the maximum manipulation stability, if any. If there exist no combinations of control modes with a positive value of manipulation stability, the robot fingers cannot perform the specified object motion stably even against infinitesimal external disturbances. Thus this procedure can be used as a test of the local feasibility of graspless manipulation.

The procedure presented above is a naive one and can be more effective by brach-and-bound techniques [19]. In that case, the computation time of the procedure heavily depends on the complexity of the situation, but it typically takes 0.01 to 1.0 CPU seconds on a PC with Pentium4–2.8GHz.

4 Planning of Graspless Manipulation

4.1 Configuration Space

Here we define a configuration space (C-Space) that represents the degrees of freedom for both the object and the robot fingertips. Note that the C-Space is used only for problem formulation and we do not explicitly compute the free space in the C-Space. A configuration of the manipulation system is a composition of a configuration of the object and that of each robot fingertip. In order to reduce the dimension of the C-Space, we represent configurations of the robot fingertips as their locations on the surfaces of the object, instead of those in three-dimensional space. Planning of graspless manipulation is transformed into finding a path in this C-Space.

However, we cannot search this C-Space in the same manner with conventional obstacle avoidance problems because graspless manipulation may be irreversible and regrasping causes discontinuous "jump" in this C-Space. Accordingly, we approximately represent this C-Space by a directed graph referred to as "manipulation-feasibility graph"; we construct nodes of the graph by discretizing the C-Space, and connect the nodes with directed arcs. As a result, planning of graspless manipulation is reduced to searching this graph.

4.2 Generation of Manipulation-Feasibility Graph

Lattice points in the C-Space are sampled as potential nodes of the manipulation-feasibility graph. We accept each of the sampled points as a valid node if geometrical constraints are satisfied at the configuration (Figure 4). We connect two nodes with a directed arc if the corresponding manipulation is "feasible"—that is, manipulation stability z calculated by the method described in Section 3 is larger than a threshold, $z_{\min}$.

Manipulation-feasibility graphs have two kinds of arcs: for displacement of the object and for regrasping. Arcs for displacement of the object correspond to moving the object without changing the locations of the robot fingertips on the surfaces of the object. These arcs connect adjacent nodes in the C-Space for the displacement of the object (Figure 5(a)). We sample several points on each arc and calculate z at the points. Unless $z \geq z_{\min}$ at all the points, the arc is discarded.

Arcs for regrasping correspond to changing a location of one fingertip on the object with neither moving the object nor changing locations of the other fingertips. Note that the arcs for regrasping may be generated between "non-adjacent" nodes, because a location of a fingertip on the object changes discontinuously by regrasping. At each node in the manipulation-feasibility graph, we calculate z when a finger is removed. If $z \geq z_{\min}$, the finger can freely change its location on the object. In this case, we

can generate bidirectional arcs for corresponding regrasping (Figure 5(b)).

[Figure 4 about here.]

[Figure 5 about here.]

Our planning problem does specify the initial and goal configurations of the object, but not those of the robot fingertips. This means that we have multiple start nodes and goal nodes in a manipulation-feasibility graph. To unite the start (and goal) nodes into one, we add a virtual start (and goal) node that is connected to all the start (and goal) nodes (Figure 6). Searching starts at the virtual start node and ends at the virtual goal node.

[Figure 6 about here.]

Thus we can construct a manipulation-feasibility graph for manipulation planning. Note that we do not have to generate all the nodes and the arcs *before* graph search; tests for generating them can be carried out *during* graph search.

4.3 Cost Assignment for Graph Searching

We have to assign a cost, C, to each arc in the manipulation-feasibility graph to find the minimum-cost path from the virtual start node to the virtual goal node by graph searching. In this paper, we determine cost assignment considering the following demands in this order of priority:

1. Minimize the number of regrasping.
2. Minimize the displacement of the fingertips.
3. Maximize the manipulation stability.

The cost assigned to arcs for displacement of the object is:

$$C = \max_i \sum_{j=1}^{P} \left(1 + \frac{X_{\mathrm{stab}}}{z_j}\right) \|\Delta \boldsymbol{q}_{\mathrm{finger}\,i,j}\|, \tag{9}$$

where $\|\Delta \boldsymbol{q}_{\mathrm{finger}\,i,j}\|$ is absolute displacement of the i-th fingertip in the j-th segment of the arc that is divided into P segments; z_j is the manipulation stability in a representative point sampled from the j-th segment; X_{stab} is a constant defined so that $X_{\mathrm{stab}}/z_{\min} \ll 1$.

On the other hand, the cost for arcs for regrasping is:

$$C = X_{\mathrm{regr}}, \tag{10}$$

where X_{regr} is a constant that is typically much larger than the value of (9).

For the arcs that are connected to the virtual start/goal node, $C = 0$.

4.4 Heuristic Function for A* Algorithm

We adopt A* algorithm [22] for fast graph searching. A* algorithm with an admissible heuristic function can find the minimum-cost path in the manipulation-feasibility graph efficiently. In this paper, an admissible heuristic function for manipulation planning, H, is designed as follows:

$$H = \begin{cases} \max_i \|\Delta \boldsymbol{q}^*_{\text{finger}\,i}\| & \text{if current fingertip locations are geometrically feasible even in the goal configuration,} \\ n_{\text{regr}} X_{\text{regr}} & \text{otherwise,} \end{cases} \tag{11}$$

where $\|\Delta \boldsymbol{q}^*_{\text{finger}\,i}\|$ is estimated displacement of i-th fingertip from the current object configuration to the goal object configuration without changing fingertip locations on the object; n_{regr} is the number of fingertips whose locations will be geometrically infeasible in the goal configuration; in other words, at least n_{regr} fingertips must change their location on the object by regrasping to achieve the goal configuration.

We assign very large cost to regrasping in our formulation (10). Therefore good estimation of the necessity of regrasping is important for the efficiency of graph searching. In order to accelerate graph searching, the heuristic function (11) utilizes the fact that regrasping is necessary when a fingertip will collide with obstacles without changing its location.

5 Planned Results

In this section, we present some examples of planned manipulation by our proposed algorithm. They are graspless manipulation of a cuboid by two robot fingertips. The computation times for the examples below are measured on a Linux PC with Pentium 4 at 2.8 GHz.

Suppose a cuboid whose dimension is $5 \times 5 \times 10$. The mass of the object is 1 ($M = 1$) and its distribution is uniform; each robot finger is modeled as a sphere whose radius is 1; the gravitational acceleration is 9.8; the friction coefficient between the object and the environment is 0.5, and that between the object and each finger is 0.7; each friction cone is represented as a polyhedral convex cone with six unit edge vectors ($s = 6$); $\boldsymbol{f}_{\max} = [10, 10]^T$. Other parameters of the planning algorithm are as follows:

$$z_{\min} = 0.5;\ X_{\text{regr}} = 10^2;\ X_{\text{stab}} = 10^{-2};\ P = 3;\ N = 76;$$

$$\{\boldsymbol{l}_i\} := \Big\{ k[\pm 1, 0, 0, 0, 0, 0]^T,\ k[0, \pm 1, 0, 0, 0, 0]^T,\ k[0, 0, \pm 1, 0, 0, 0]^T,\ k[0, 0, 0, \pm 1, 0, 0]^T,$$
$$k[0, 0, 0, 0, \pm 1, 0]^T,\ k[0, 0, 0, 0, 0, \pm 1]^T,\ \frac{k}{\sqrt{6}}[\pm 1, \pm 1, \pm 1, \pm 1, \pm 1, \pm 1]^T \Big\},$$

where $k = 2\sqrt{3 - \sqrt{6}}$ (≈ 1.48).

We replace each of surface/edge contacts with its equivalent point contacts, vertices of the convex hull of the surface/edge contact. Note that this replacement is valid as far as we deal with manipulation without surface/edge contacts in rotation [20].

In this section we consider only one degree of freedom for the manipulated object. The degree of freedom is discretized into 30 segments. Fingertip locations are limited to 7×7 grid points on each face of the object. Thus, the maximum number of nodes in the manipulation-feasibility graph is ${}_{7\times7\times6}C_2 \times (30 + 1) = 1,335,201$.

5.1 Planning of Sliding Operations

Let us consider one-dimensional sliding of the cuboid on a horizontal plane. In this case, graspless manipulation as shown in Figure 7 is generated; one force-controlled finger is located on the top of the object to press it down and another position-controlled finger is located behind the object to push it. It takes 533 CPU minutes for this planning with A*. When we do not use any heuristic functions, 1007 CPU minutes are required.

When we use "weaker" robot fingers by setting $\boldsymbol{f}_{\max} = [5, 5]^T$ instead of $\boldsymbol{f}_{\max} = [10, 10]^T$, different graspless manipulation is generated (Figure 8); both fingers are position-controlled and push the object from behind. This corresponds to "stable push" [9]. The computation requires 203 CPU minutes (or 378 CPU minutes without heuristics).

These results indicate that our algorithm tends to generate graspless manipulation with large internal force like grasping for "strong" robot fingers as in Figure 7; on the other hand, graspless manipulation without internal force tends to be generated for "weak" robot fingers as in Figure 8.

[Figure 7 about here.]

[Figure 8 about here.]

5.2 Planning of Tumbling Operations

Let us consider tumbling of the object. When an obstacle exists behind the object, graspless manipulation as shown in Figure 9 is generated; the robot fingers pinch the object to tumble it down. In this case, one finger is in position-control mode and another finger is in force-control mode. The computation time is 84 CPU minutes (or 1294 CPU minutes without heuristics).

When an additional obstacle exists by the side of the object, pinching the object is impossible. In this case, tumbling with regrasping is generated (Figure 10). The computation time is 990 CPU minutes (or 1296 CPU minutes without heuristics).

[Figure 9 about here.]

[Figure 10 about here.]

5.3 Execution of Planned Manipulation

We conducted experiments with a multifingered robot hand to validate that our planner can generate "decent" manipulation. The experimental system consists of a robot arm with five degrees of freedom

and a two-fingered hand attached at the end of the arm. Each finger has three degrees of freedom and a six-axis force sensor at its fingertip.

We used a cork cuboid of 0.092 [kg] as a manipulated object. Its dimension is 60[mm] × 60[mm] × 100[mm]. The friction coefficient between the object and fingertips and that between the object and the environment was identified as 0.15 and 1.2, respectively, from preliminary experiments. We planned graspless manipulation of the object using these values and played back the planned result with the robot.

Here we show a result of tumbling of the object on a plane. In this case, our algorithm generated pinching of the object by the fingers to tumble it like the case of Figure 9; one finger is in force-control mode and another finger is in position-control mode throughout the tumbling operation. The planned operation is executed successfully by the experimental system (Figure 11). Figure 12 shows finger normal forces in the tumbling operation. The force-controlled finger ("finger1") succeeded in applying normal force roughly as planned.

[Figure 11 about here.]

[Figure 12 about here.]

6 Conclusion

We developed a method of planning of graspless manipulation by multifingered robot hand. The method can generate a sequence of desired fingertip locations and control modes to achieve robust graspless manipulation from an initial configuration to a goal configuration. Some planned results by this method including pushing and tumbling operations were presented; those indicate our method can plan various graspless operations. An experimental result of successful execution of planned manipulation by a robot with a multifingered hand was also shown.

A major problem in our algorithm is that it still requires much computation for planning in spite of the improvement made by A* search. Because the computation time heavily depends on the number of nodes in a manipulation-feasibility graph, currently we are trying to reduce it by incorporating randomized motion planning techniques into our algorithm.

Acknowledgments

The authors would like to thank Yasumichi AIYAMA and Tsukasa KYOUSOU at University of Tsukuba for their great assistance in carrying out the experiments.

REFERENCES

[1] Y. Aiyama, M. Inaba, and H. Inoue, "Pivoting: A new method of graspless manipulation of object by robot fingers," in *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, (Yokohama, Japan), pp. 136–143, 1993.

[2] M. T. Mason, "Progress in nonprehensile manipulation," *Int. J. of Robotics Research*, vol. 18, no. 1, pp. 1129–1141, 1999.

[3] S. Hirai, *Analysis and Planning of Manipulation Using the Theory of Polyhedral Convex Cones*. PhD thesis, Kyoto University, 1991.

[4] Y. Yu, Y. Yokokohji, and T. Yoshikawa, "Two kinds of degree of freedom in constraint state and their application to assembly planning," in *Proc. of IEEE Int. Conf. on Robotics and Automation*, (Minneapolis, MN, U.S.A.), pp. 1993–1999, 1996.

[5] X. Ji and J. Xiao, "Planning motion compliant to complex contact states," *Int. J. of Robotics Research*, vol. 20, no. 6, pp. 446–465, 2001.

[6] Y. Aiyama, T. Yasui, and T. Arai, "Planning of object motion by graspless manipulation using contact state transition graph," in *Proc. of IEEE Int. Symp. on Assembly and Task Planning*, (Fukuoka, Japan), pp. 184–189, 2001.

[7] M. T. Mason, "Mechanics and planning of manipulator pushing operations," *Int. J. of Robotics Research*, vol. 5, no. 3, pp. 53–71, 1986.

[8] M. Kurisu and T. Yoshikawa, "Trajectory planning of an object in pushing operation," in *Proc. of Japan-USA Symp. on Flexible Automation*, (Kobe, Japan), pp. 1009–1016, 1994.

[9] K. M. Lynch and M. T. Mason, "Stable pushing: Mechanics, controllability, and planning," *Int. J. of Robotics Research*, vol. 15, no. 6, pp. 533–556, 1996.

[10] N. Sawasaki, M. Inaba, and H. Inoue, "Tumbling objects using a multi-fingered robot," in *Proc. of Int. Symp. on Industrial Robots*, (Tokyo, Japan), pp. 609–616, 1989.

[11] A. Marigo, M. Ceccarellio, S. Piccinocchi, and A. Bicchi, "Planning motions of polyhedral parts by rolling," *Algorithmica*, vol. 26, no. 3-4, pp. 560–576, 2000.

[12] A. Yamashita, T. Arai, J. Ota, and H. Asama, "Motion planning of multiple mobile robots for cooperative manipulation and transportation," *IEEE Trans. on Robotics and Automation*, vol. 19, no. 2, pp. 223–237, 2003.

[13] J. C. Trinkle, R. C. Ram, A. O. Farahat, and P. F. Stiller, "Dexterous manipulation planning and execution of an enveloped slippery workpiece," in *Proc. of IEEE Int. Conf. on Robotics and Automation*, (Atlanta, GA, U.S.A.), pp. 442–448, 1993.

[14] M. Erdmann, "An exploration of nonprehensile two-palm manipulation," *Int. J. of Robotics Research*, vol. 17, no. 5, pp. 485–503, 1998.

[15] H. Terasaki and T. Hasegawa, "Motion planning of intelligent manipulation by a parallel two-fingered gripper equipped with a simple rotating mechanism," *IEEE Trans. on Robotics and Automation*, vol. 14, no. 2, pp. 207–219, 1998.

[16] S. Hirai and H. Asada, "Kinematics and statics of manipulation using the theory of polyhedral convex cones," *Int. J. of Robotics Research*, vol. 12, no. 5, pp. 434–447, 1993.

[17] M. H. Raibert and J. J. Craig, "Hybrid position/force control of manipulators," *ASME J. of Dynamic Systems, Measurement, and Control*, vol. 102, no. 2, pp. 126–133, 1981.

[18] G. Liu and Z. Li, "A unified geometric approach to modeling and control of constrained mechanical systems," *IEEE Trans. on Robotics and Automation*, vol. 18, no. 4, pp. 575–587, 2002.

[19] Y. Maeda and T. Arai, "Automatic determination of finger control modes for graspless manipulation," in *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, (Las Vegas, NV, U.S.A.), pp. 2660–2665, 2003.

[20] Y. Maeda and T. Arai, "A quantitative stability measure for graspless manipulation," in *Proc. of IEEE Int. Conf. on Robotics and Automation*, (Washington D.C., U.S.A.), pp. 2473–2478, 2002.

[21] Y. Maeda, Y. Aiyama, T. Arai, and T. Ozawa, "Analysis of object-stability and internal force in robotic contact tasks," in *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, (Osaka, Japan), pp. 751–756, 1996.

[22] J. Pearl, *Heuristics: Intelligent Search Strategies for Computer Problem Solving*. Addison-Wesley, 1984.

List of Figures

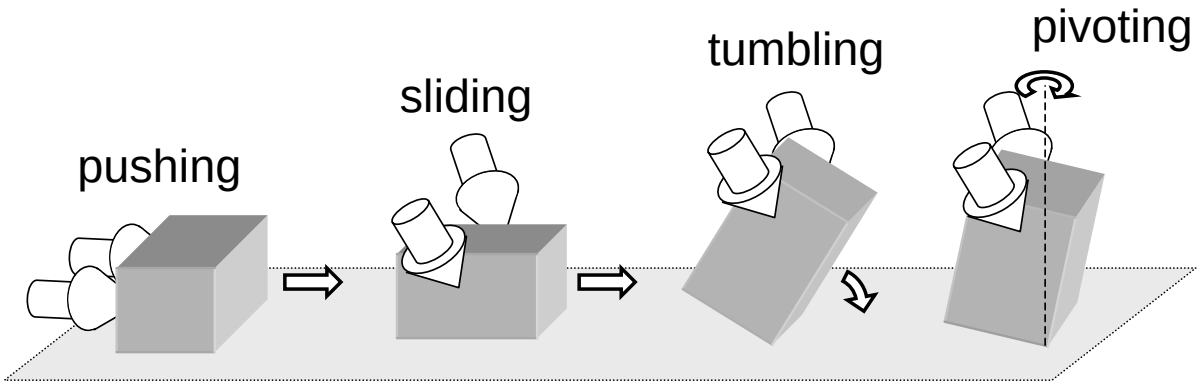


Figure 1: Graspless Manipulation

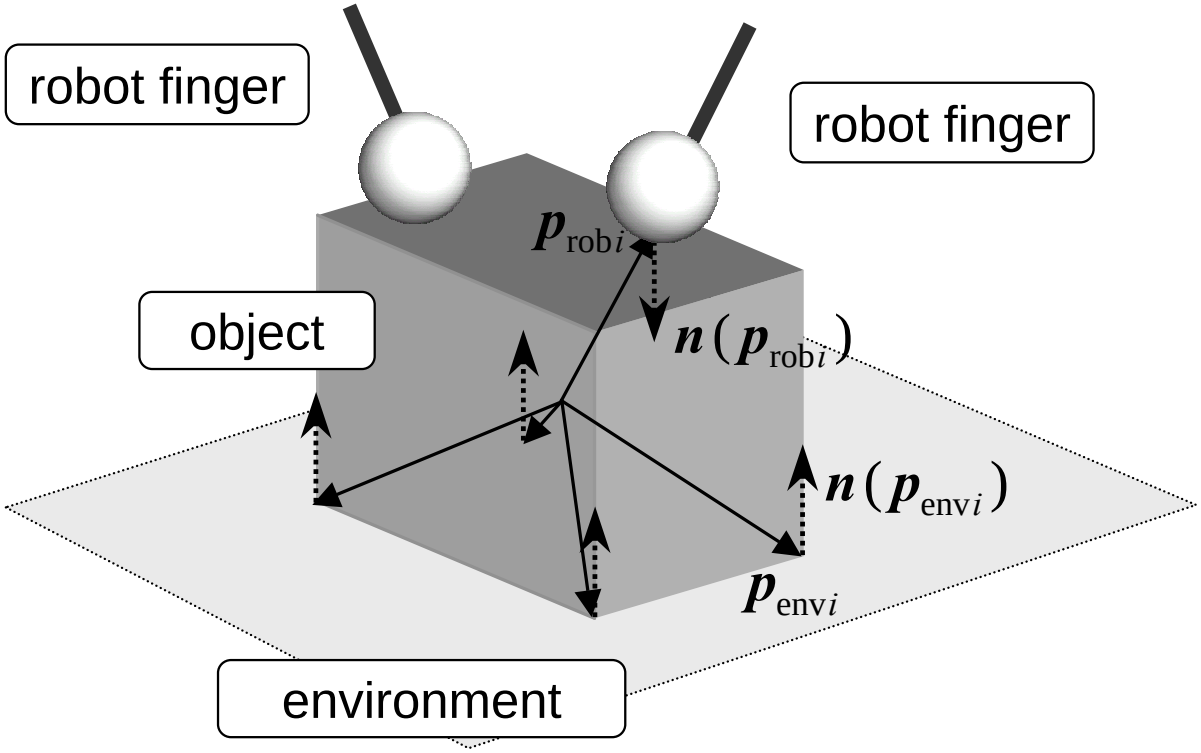


Figure 2: Object in Graspless Manipulation

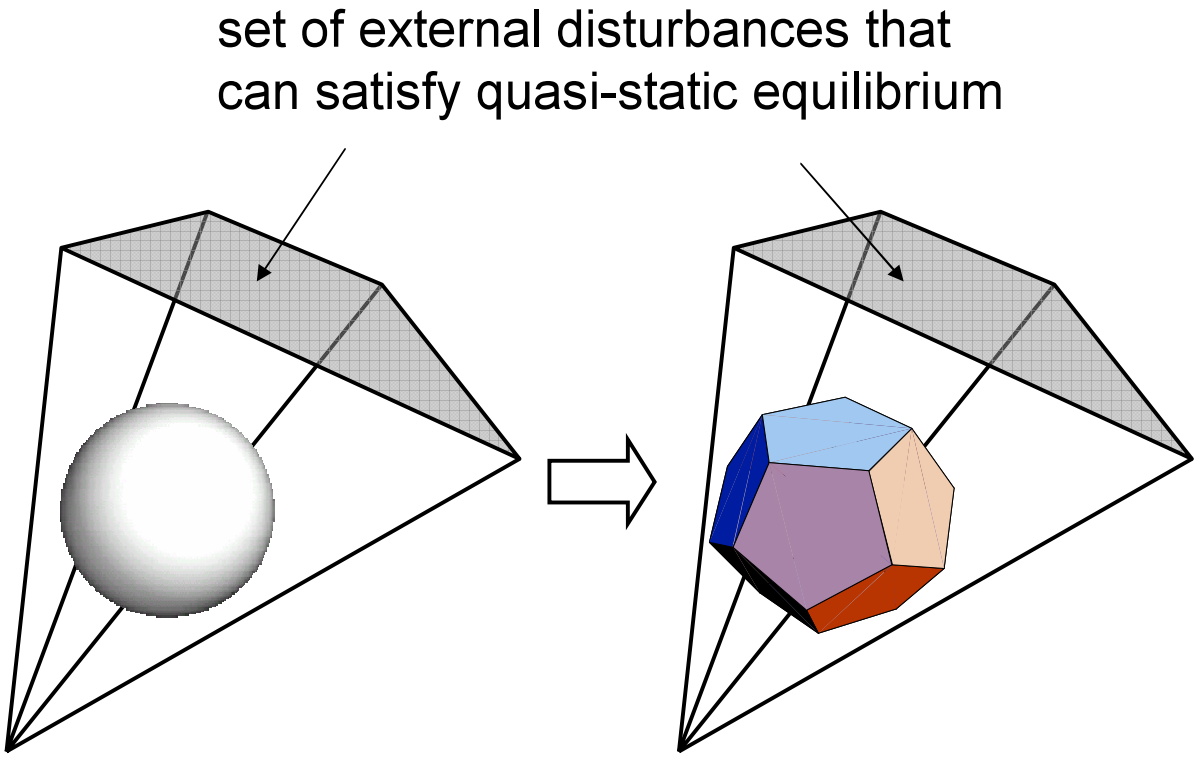


Figure 3: Approximate Calculation of Stability Measure

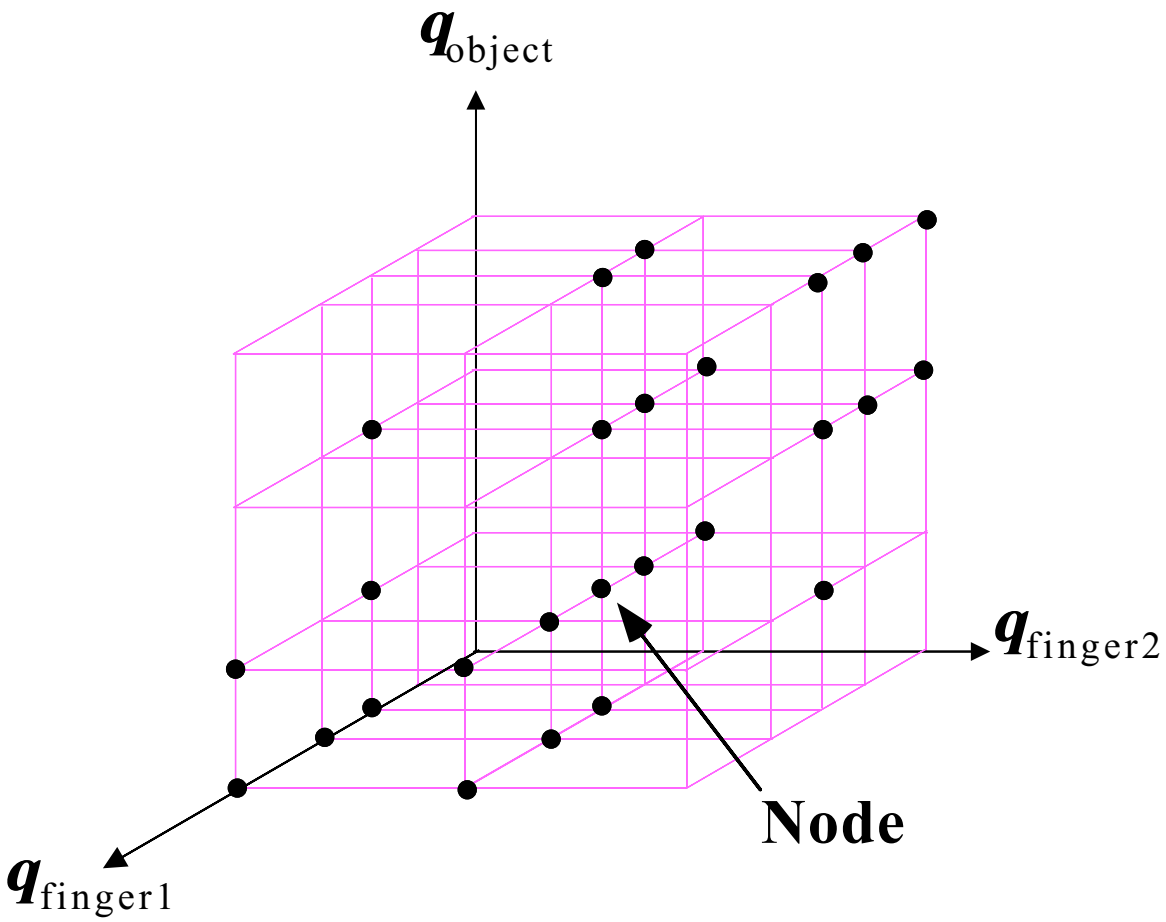
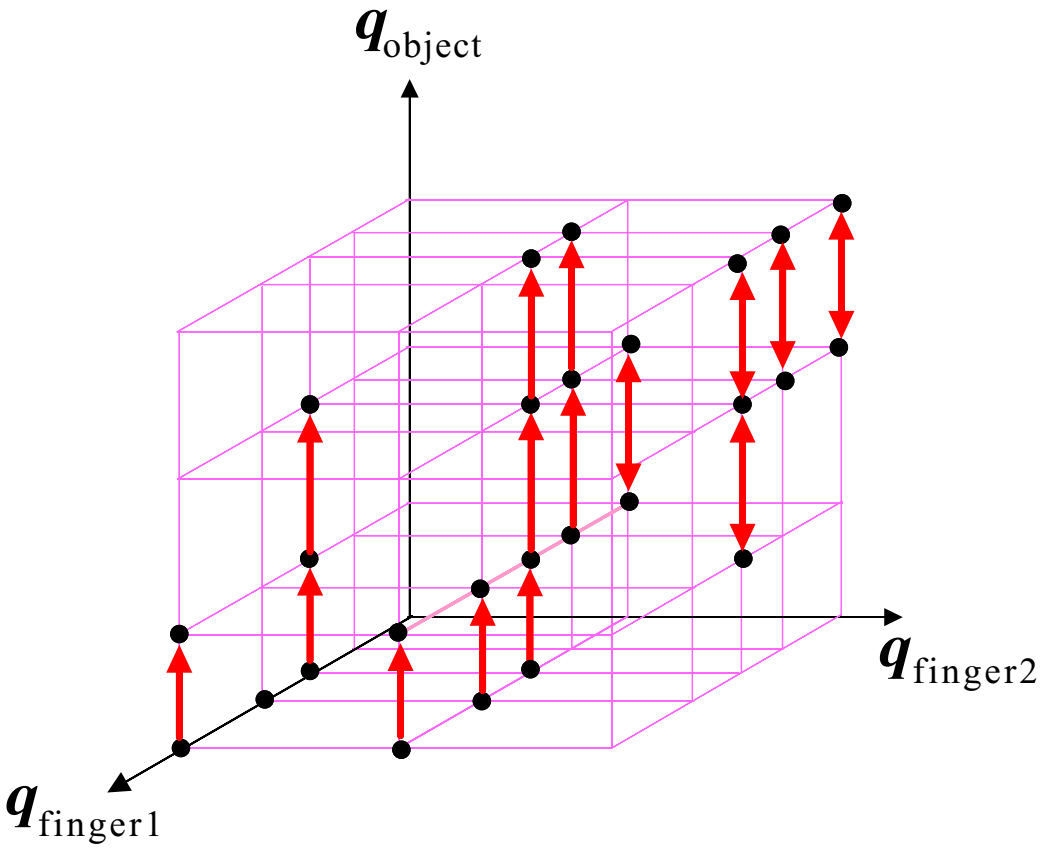
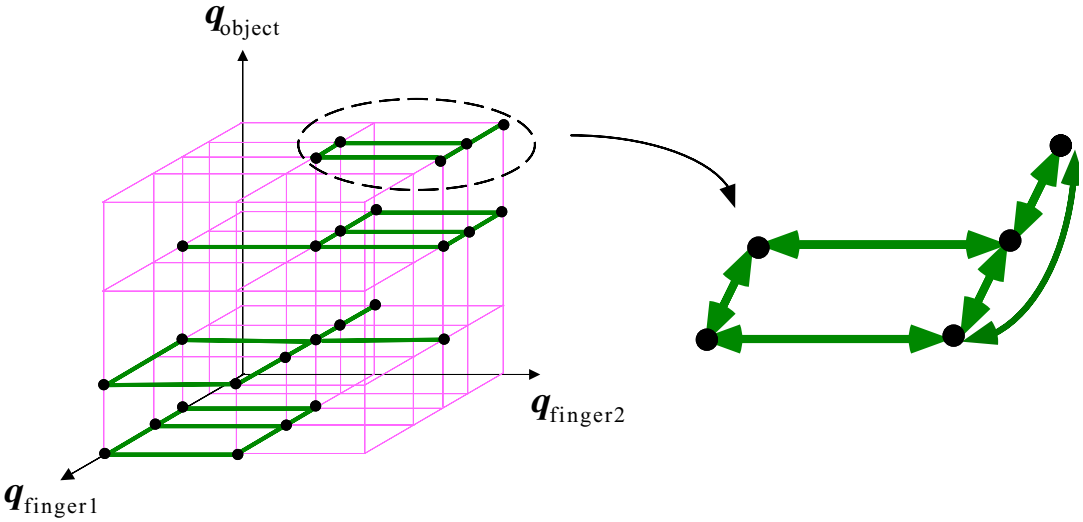


Figure 4: Generation of Nodes



(a) arcs for displacement of object



(b) arcs for regrasping

Figure 5: Generation of Arcs

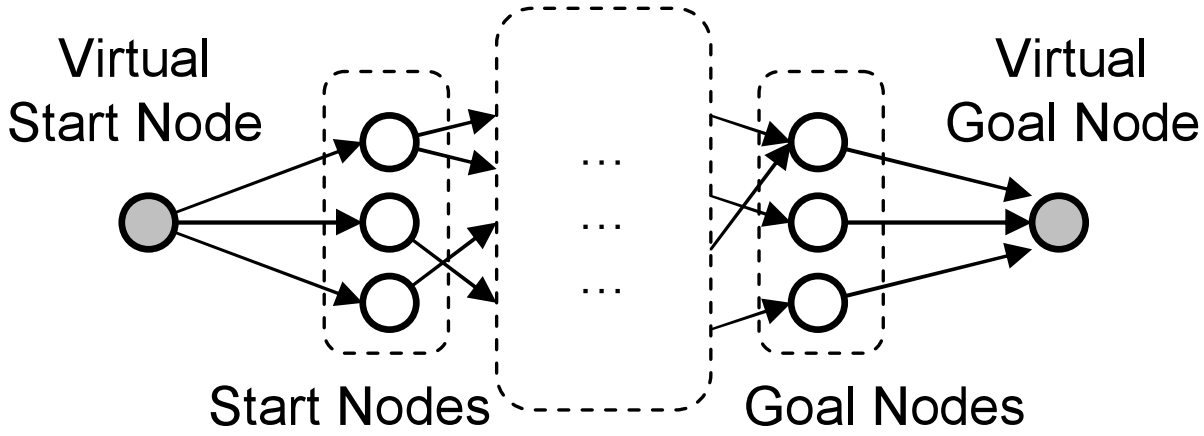
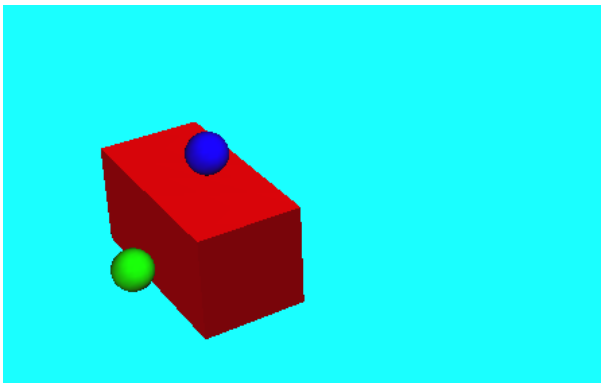
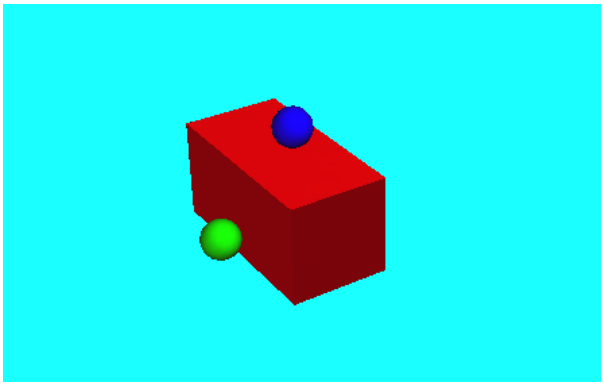


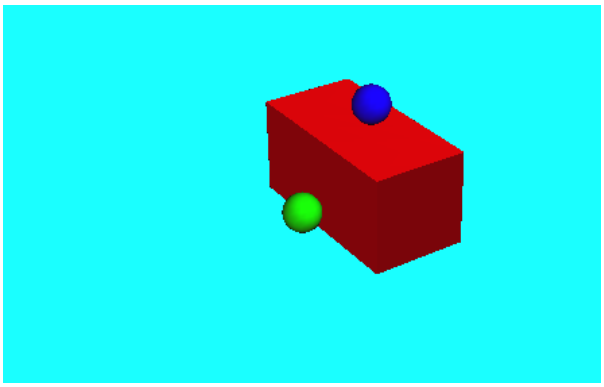
Figure 6: Virtual Start/Goal Node



(a) start

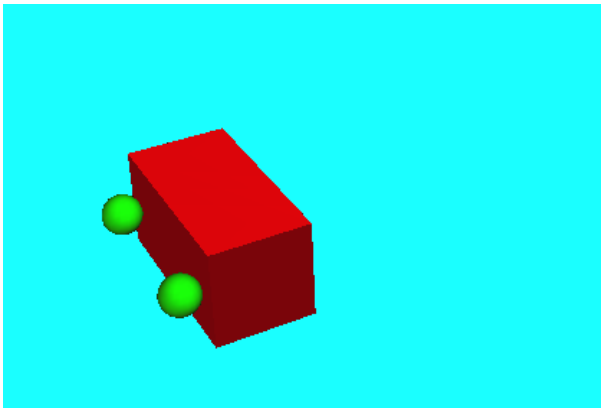


(b) intermediate

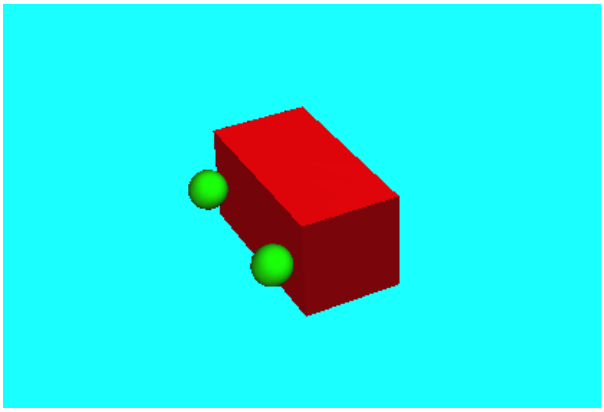


(c) goal

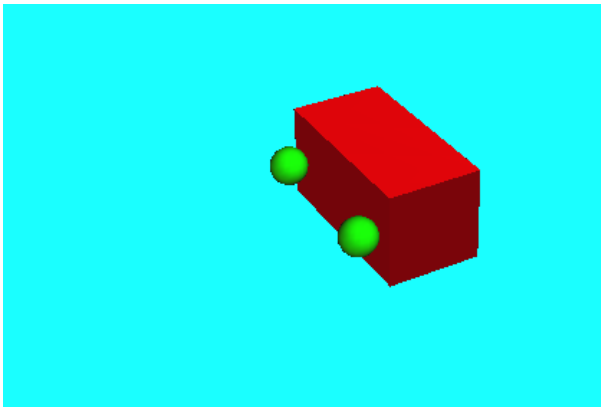
Figure 7: Planned Sliding Operation



(a) start

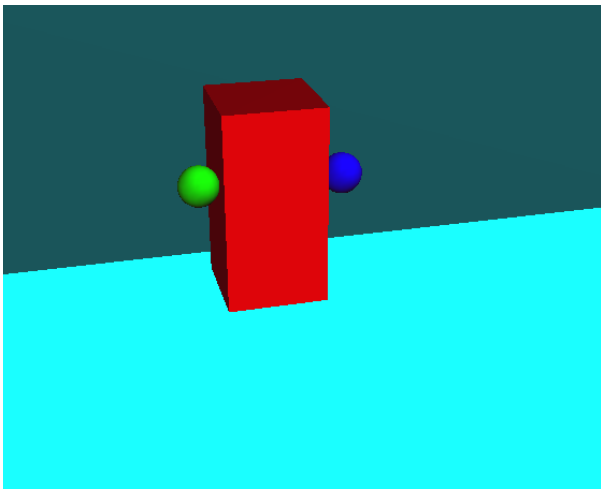


(b) intermediate

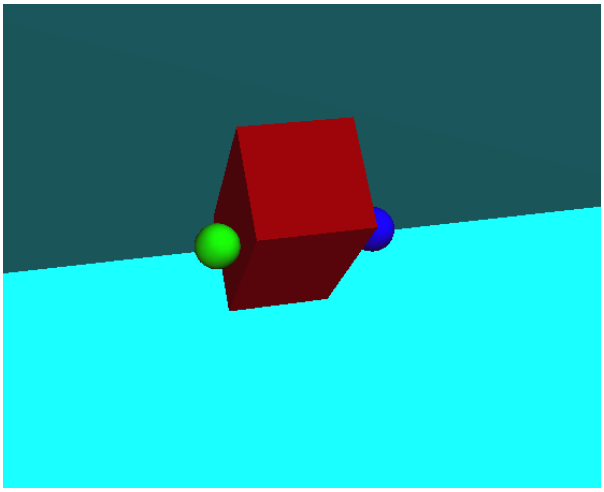


(c) goal

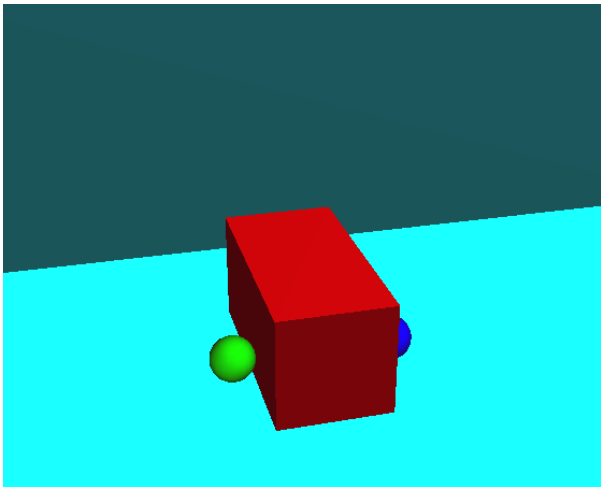
Figure 8: Planned Pushing Operation



(a) start (0[deg])

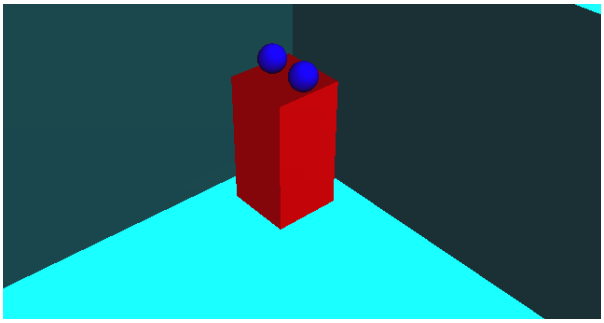


(b) intermediate (45[deg])

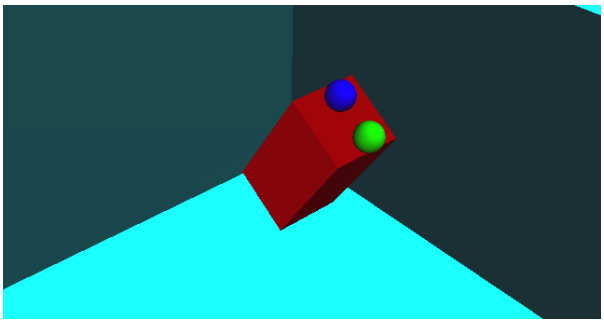


(c) goal (90[deg])

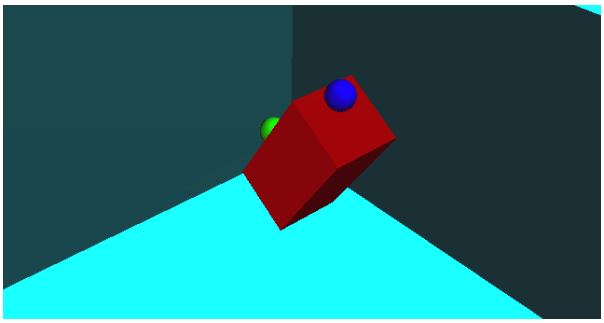
Figure 9: Planned Tumbling Operation



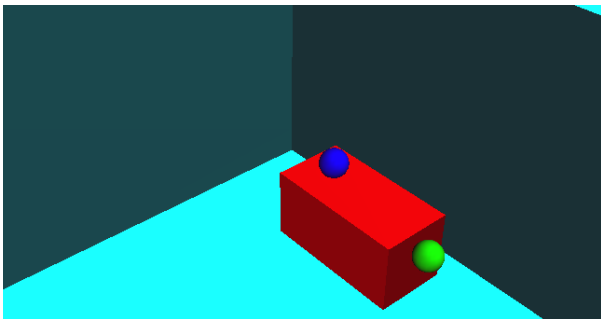
(a) start (0[deg])



(b) before regrasping (33[deg])

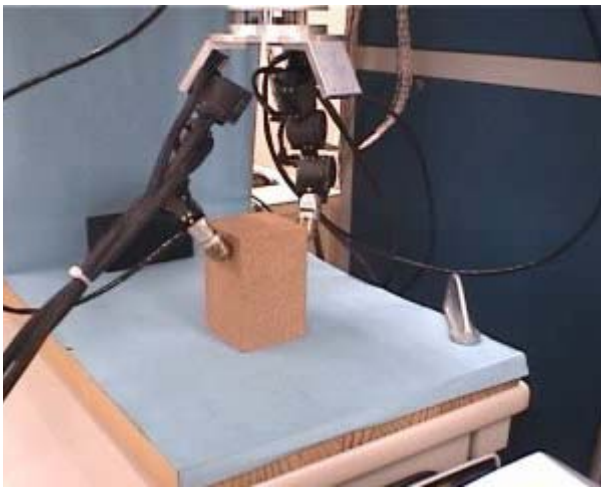


(c) after regrasping (33[deg])

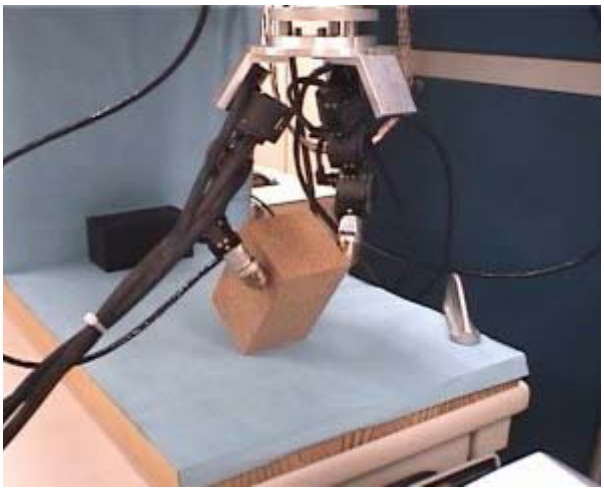


(d) goal (90[deg])

Figure 10: Planned Tumbling Operation with Regrasping



(a) start



(b) intermediate

(c) goal

Figure 11: Experiment of Tumbling Operation

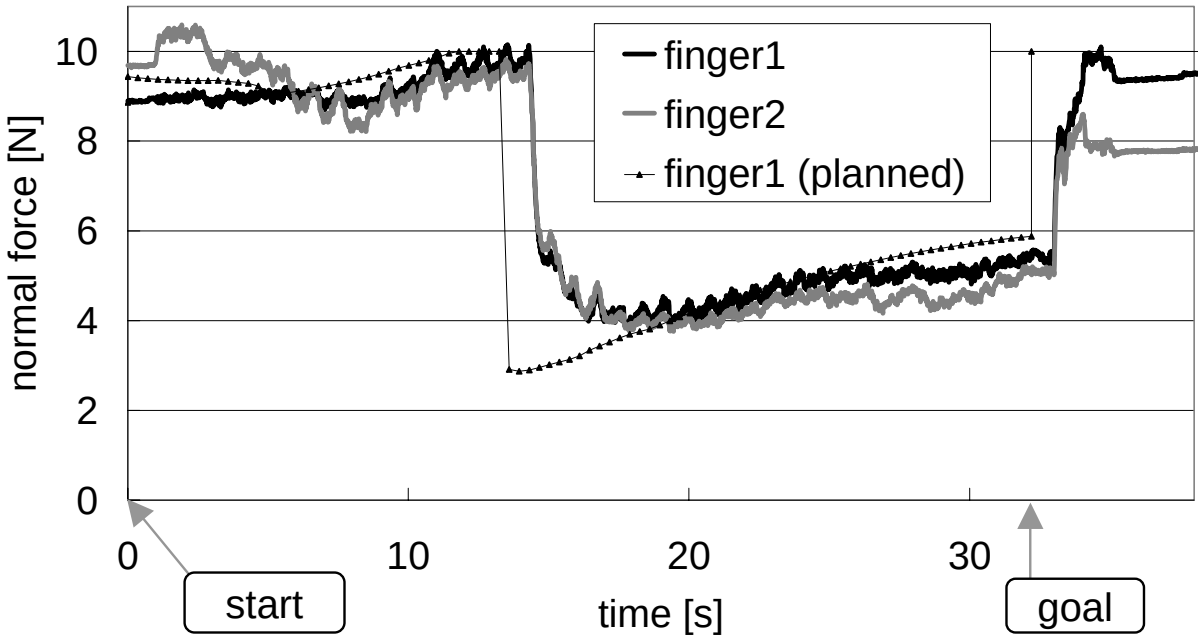


Figure 12: Finger Forces in Tumbling