3D Two-Fingered Caging for Two Types of Objects: Sufficient Conditions and Planning

Satoshi Makita*

Department of Control Engineering,
Sasebo National College of Technology,
1-1 Okishin-cho, Sasebo, Japan
Fax: 81-956-34-8493, E-mail: makita@sasebo.ac.jp
*Corresponding author

Kensuke Okita

Canon Inc.

Yusuke Maeda

Div. of Systems Research, Faculty of Engineering,
Yokohama National University,
79-5 Tokiwadai, Hodogaya-ku, Yokohama, Japan
Fax: 81-45-339-3918, E-mail: maeda@ynu.ac.jp

Abstract:
Caging is a method to capture an object geometrically by robots. Position-controlled robots can make the object inescapable from the robot formation. Moreover less number of mobile robots or a robot hand with low degree of freedom can constrain the object with considering concavity of the object. In this paper, we propose two types of caging: ring-type and waist-type, which both can be accomplished by a two-fingered hand. We derive sufficient conditions for caging of the two types and construct RRT-based motion planner for caging by a robotic arm/hand system. In motion planning, we find one of final configurations that satisfy the sufficient conditions and produce a path of robot configuration to the goal. With object recognition using AR picture markers, we can acquire geometrical information of objects and plan robot motion for caging. We show some experimental results of planning and execution of planned caging motion for four objects.

Keywords: Caging, Multifingered hand, Manipulation

Reference to this paper should be made as follows: Makita, S., Okita, K and Maeda, Y. "3D Two-Fingered Caging for Two Types of Objects: Sufficient Conditions and Planning" *Int. J. of Mechatronics and Automation*,

1 Introduction

Robotic manipulation by a multifingered hand has been studied in a long term (Bicchi and Kumar; 2000; Shimoga; 1996). The basic method of it is grasping, and one of the simplest hands is a parallel jaw gripper (Smith et al.; 1999; Yamanobe and Nagata; 2010). A parallel jaw gripper has generally two fingers with one degree of freedom (DOF) and pinches an object. In addition, there are also studies on pinching grasping by higher DOF fingers (Ozawa et al.; 2005) and more fingers (Borst et al.; 2002). Pinching by a multifingered hand, especially a two-fingered hand (or dual-finger hand) is one of basic issues of robotic manipulation, because controlling lower DOF hands as end effectors of robotic arm (or manipulator) can be comparatively easy. A common approach for pinching an object by a multifingered hand is to address force equilibrium of the object and force control of the fingers.

On the other hand, caging, in which an object is just surrounded by robots and is geometrically inescapable from the robot formation, has been studied as a geometrical confinement and preshaping of the robots for grasping (Rimon and Blake; 1996). Rimon and Blake studied caging a concave object by two circle fingertips in a 2D scene (Rimon and Blake; 1999). This method can be regarded as pinching an object on the plane by a parallel jaw gripper. Caging by two or more fingertips approximated by circles or points in the plane has been variously studied (Pipattanasomporn and Sudsang; 2006; Erickson et al.; 2007; Vahedi and van der Stappen; 2008; Rodriguez and Mason; 2008). Caging in a 3D scene, however, has not been investigated enough. Pipattanasomporn and Sudsang presented an algorithm to obtain caging sets for a polyhedron by two fingers (Pipattanasomporn and Sudsang; 2011). The algorithm is applied only to caging by point fingers since practical robot hands were not taken into account. Some 3D caging by a practical multifingered hand were independently studied. Makita and Maeda derived sufficient conditions for caging of some simple objects by an articulated symmetric hand (Makita and Maeda; 2008). They proposed a method to produce final configuration of the robot hand for 3D caging, but approaching to the final configuration was not considered. Diankov et al. planned caging grasps and manipulation by a robotic arm/hand system and a humanoid robot, respectively (Diankov et al.; 2008). In (Diankov et al.; 2008), caging grasp set, which determines location of the hand for the object, was obtained by an experimental approach. Maeda et al. proposed caging-based grasping, in which rigid limbs of robots confine an object as caging and soft parts around the rigid bodies grip the object (Maeda et al.; 2012). They derived a sufficient condition for caging of a sphere with a symmetric hand.

In this paper, we focus on caging by a two-fingered hand in a 3D scene. We derive sufficient conditions for caging of two types of objects, and with the conditions, we plan motions of a robotic arm/hand system for caging. Geometrical information of objects such as shape, size and position and orientation is obtained by using AR picture markers attached to the objects.

In some previous works, motion planning of robots for caging has been mainly studied on final configuration of robots that satisfies the conditions for caging (Rimon and Blake; 1999; Erickson et al.; 2007; Vahedi and van der Stappen; 2008). Wang and Kumar studied planar caging manipulation with multiple mobile robots, in which the robots approach to the object and achieve caging formation (Wang and Kumar; 2002). Diankov et al. planned caging grasps and manipulation by an arm/hand robot and a humanoid robot, with an RRT-based algorithm (Diankov et al.; 2008). RRT (Rapidly-exploring Random Trees) (Lavalle; 1998) is a path planning algorithm with using random sampling in a configuration space.

Caging can be regarded as preshaping motion of grasping (Rimon and Blake; 1999). Motion planners of graping have been variously proposed. Goal configuration of robots for grasping can be determined one of possible candidates with some kind of objective functions. Miller et al. developed a grasping planning software “GraspIt!” (Miller et al.; 2003; Miller and Allen; 2004). They defined set of grasping strategies for some primitive shapes and tested the grasping patterns with evaluating grasp quality. Yamanobe and Nagata proposed grasp planning with a parallel jaw gripper (Yamanobe and Nagata; 2010). They also used shape primitives of objects, which have proper basic grasping configurations.

Our proposed motion planning is similar to the above methods in point of using shape primitives. Since we do not have any objective functions to evaluate *appropriate* caging configurations, we just obtain one of candidate configurations that satisfies sufficient conditions for caging. The sufficient conditions for caging are derived for two shape primitives (Sec. 2). Moreover we also produce a path in the robot configuration from an initial state to one of goal configuration. It means the planned motion includes hand approaching to the object and accomplishing caging. Considering that the robot configuration is in high dimension, we adopt RRT (Lavalle; 1998), which uses random sampling efficiently in the high dimensional configuration (Sec. 3). Required geometrical specification of objects can be obtained by using AR picture markers attached to the objects (Sec. 4.2). Finally, we verify our proposed caging procedure with a practical robotic arm/hand system for four objects (Sec. 4). To avoid any collisions due to errors between the computational planning environment and the actual one, we also test planning algorithm with considering margins.

This paper is a revised and expanded version of (Makita et al.; 2012).

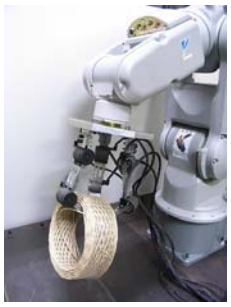
Figure 1 Caging a ring-like object (Ring-type caging)

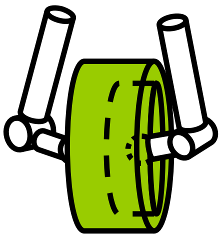
Figure 2 Caging a concave object (Ring-type caging)

2 Caging Conditions

2.1 *Classification*

We classify patterns of caging by a multifingered hand with depending on both the shapes of objects and the methods of constraint. In this paper, we focus on caging that can be achieved by two-fingered hand, and name the patterns as follows:

Ring-type Caging In cases of caging a ring-like object such as a torus, a robot hand constrains the object inserting its fingers into a hollow of the object (Figure1). In addition, ring-type caging can be also achieved even when the object do not have any through holes but any dimples (Figure2). Ring-type caging can be applied to objects that has ring-like parts such as mugs. An advantage of this caging is that even a low DOF robot hand can constrain objects without any force control.

Waist-type Caging In cases of caging an object that has a constricted part such as Figure3, 4, a robot hand constrains the object winding its fingers around the constricted part of the object.

2.2 *Sufficient Conditions*

It is difficult to derive necessary and sufficient conditions of multifingered caging because there are various patterns

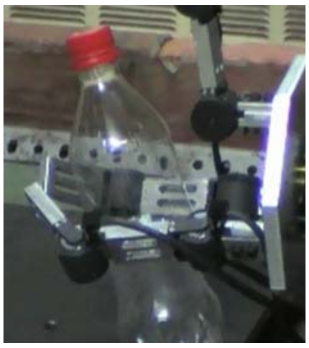
Figure 3 Caging a bottle (Waist-type caging)

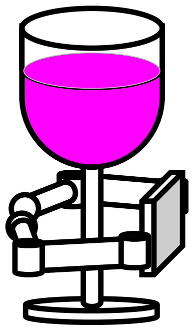
Figure 4 Caging a wineglass (Waist-type caging)

of caging and objects, and the conditions are probably complicated. Thus, for the facilitation of both derivation and application of caging conditions, we derive sufficient conditions for the above two types of caging in this paper. First, we present a general series of sufficient conditions for multifingered caging. Next we derive each detailed conditions for the patterns of caging.

If all the following conditions are satisfied, caging an object by a robot hand can be accomplished.

1. **Cage-formed conditions:** The robot hand constructs the caged region from which the object cannot escape.
2. **In-cage conditions:** The object is present in the caged region formed by the robot.
3. **Collision-free conditions:** The robot hand has no collision with the object and any obstacles.

2.3 Assumptions and Notations

We assume a symmetric robot hand as follows:

- The hand has 2 fingers, and each finger has $\bar{L}$ joints.
- All the joints are revolute, and their bodies have no volume, that is, they can be approximated by points.
- The j-th body of the i-th finger can be approximated by a line segments with length l_j.
- The palm of the robot hand is a line segment with length l_{palm}
- Each finger is attached to each end of the palm in opposite and moves with same vector of joint variables:

$$\bar{\boldsymbol{\theta}} := [\theta_1, \theta_2, \ldots, \theta_{\bar{L}}]^T, \tag{1}$$

  where θ_j is a joint variable of the j-th joint of each finger.

As mentioned above, we assume every limb of the robot hand has no volume, and then derive sufficient conditions for multifingered caging. Note that the derived conditions can be applied to the hand whose limbs have volume, with taking collisions into account.

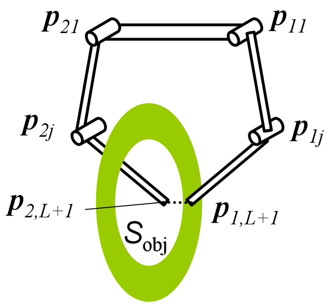

Figure 5 Caging a ring-like object (Ring-type caging)

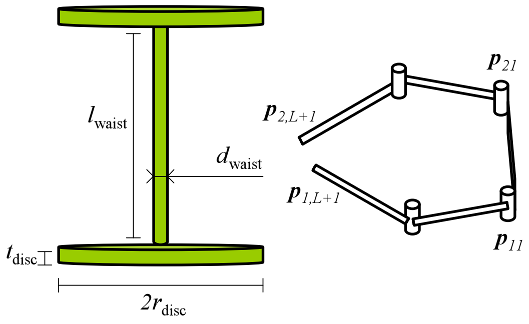

Figure 6 Caging a dumbbell-like object (Waist-type caging)

2.4 Cage-formed Conditions

2.4.1 Ring-type Caging

Let us consider caging a ring-like object by a robot hand with two fingers (Figure5) as (Makita and Maeda; 2008). For an example of ring-like objects, we define a swept volume when a circle moves along a closed curve while keeping itself vertical to the closed curve. The diameter of the circle is d_{ring}. Thus it is a torus when the closed curve is a circle with a larger diameter than d_{ring}.

When the two fingertips of the hand approach to each other in the hollow part of the object, and the distance between both fingertips: $d_{L+1}(\bar{\boldsymbol{\theta}}) \left(= \|\boldsymbol{p}_{2,\bar{L}+1}(\bar{\boldsymbol{\theta}}) - \boldsymbol{p}_{1,\bar{L}+1}(\bar{\boldsymbol{\theta}})\|\right)$ is shorter than d_{ring}, the hand can capture the object. In other words, caging a ring-like object is achieved when $d_{L+1}(\bar{\boldsymbol{\theta}})$ satisfies the following condition:

$$d_{L+1}(\bar{\boldsymbol{\theta}}) < d_{\text{ring}}. \tag{2}$$

2.4.2 Waist-type Caging

Let us consider caging a dumbbell-like object by a robot hand with two fingers as Figure6. The object is composed of a cylinder with two identical circular plates (disks) attached on each base, and their parameters are denoted as follows: d_{waist}: the diameter of the cylinder; l_{waist}: the length of the cylinder; r_{disk}: the radius of the disk; t_{disk}: the thickness of the disk.

We derive a sufficient condition to constrain each part of the object: the cylinder and the disk, respectively. The cylinder part cannot escape from the gap between both fingertips when its distance, $d_{\bar{L}+1}(\bar{\boldsymbol{\theta}})$, is less than the diameter of the cylinder:

$$d_{\bar{L}+1}(\bar{\boldsymbol{\theta}}) < d_{\text{waist}}. \tag{3}$$

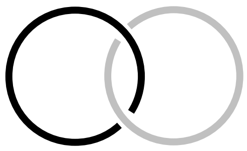
Figure 7 Hopf link

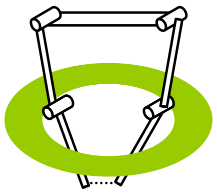
Figure 8 Hopf link is not constructed

The disk part cannot escape from the gap between both fingertips when $d_{\bar{L}+1}(\bar{\boldsymbol{\theta}})$ is less than the thickness of the disk:

$$d_{\bar{L}+1}(\bar{\boldsymbol{\theta}}) < t_{\text{disk}}. \tag{4}$$

In addition, the disk cannot escape between finger bodies when every distance between joints or the fingertips is less than the diameter of the disk:

$$d_{ijik}(\bar{\boldsymbol{\theta}}) < 2r_{\text{disk}} \quad (k \neq j) \tag{5}$$
$$d_{ij(i+1)k}(\bar{\boldsymbol{\theta}}) < 2r_{\text{disk}}, \tag{6}$$
$$(i = 1, \ldots, N) \quad (j = 1, \ldots, \bar{L}) \quad (k = 1, \ldots, \bar{L}+1)$$

where $d_{ijik} = \|\boldsymbol{p}_{ij}(\bar{\boldsymbol{\theta}}) - \boldsymbol{p}_{ik}(\bar{\boldsymbol{\theta}})\|$.

Consequently, a series of cage-formed conditions for caging a dumbbell-like object is that (3), (4), (5) and (6) are satisfied.

2.5 *In-cage Conditions*

2.5.1 *Ring-type Caging*

A ring-like object is present in the region caged by a robot hand in cases of ring-type caging when a hopf link (Figure7) is constructed by two closed curves: one goes through inside the object, and another is composed of a curve inside the hand and a line segment connecting both fingertips.

We assume that a closed curve inside an object is on a plane, and the closed region and the closed planer region inside the curve is denoted by $\mathcal{S}_{\text{obj}}$ (Figure5). When a hopf link is constructed, a robot hand or a line segment connecting both fingertips passes through $\mathcal{S}_{\text{obj}}$ odd times. If they pass through even times, a hopf link cannot be constructed such as Figure8.

Let us consider a case of caging a torus whose $\mathcal{S}_{\text{obj}}$ is a circle with r_{obj} radius. In the case, a closed curve of a robot hand goes through $\mathcal{S}_{\text{obj}}$ when the intersection of the closed curve and a plane including $\mathcal{S}_{\text{obj}}$ is within r_{obj} from the center of $\mathcal{S}_{\text{obj}}$.

2.5.2 *Waist-type Caging*

A dumbbell-like object is present in the region caged by a robot hand in cases of waist-type caging when the cylinder part of the object passes through a polygon whose vertices are on the joints and the fingertips of the hand. We assume that $\mathcal{S}_{\text{rob}}$ denotes the polygon inside the robot hand, and $\boldsymbol{t}_{\text{waist}}$ denotes the unit direction vector of the center line of the cylinder part. Then we can expressed the center line of the cylinder part, $\boldsymbol{p}_{\text{cline\_waist}}$ as follows:

$$\boldsymbol{p}_{\text{cline\_waist}} = u_{\text{cline\_waist}} \boldsymbol{t}_{\text{waist}} + \boldsymbol{p}_{\text{cm\_waist}}, \tag{7}$$
$$-l_{\text{waist}}/2 < u_{\text{cline\_waist}} < l_{\text{waist}}/2, \tag{8}$$

where $u_{\text{cline\_waist}}$ is an arbitrary value, and $\boldsymbol{p}_{\text{cm\_waist}}$ denotes the center of mass of the cylinder part. When an intersection between $\mathcal{S}_{\text{rob}}$ and the center line of the cylinder part exist,

$$\boldsymbol{p}_{\text{cline\_waist}} \in \mathcal{S}_{\text{rob}}. \tag{9}$$

2.6 *Collision-free Conditions*

The sufficient conditions for caging addressed in Sec. 2.2 is that two conditions presented in Sec. 2.4 and Sec. 2.5 are satisfied, and the robot hand must collide with neither the target objects nor any obstacles. Then we can adopt extensively-used programming libraries of collision detection such as “A Proximity Query Package” (PQP (Larsen et al.; 1999)). In PQP, tested objects are approximated by polygon models composed of triangles, and each triangle is examined whether it occupies any other triangles of other objects.

When we approximate tested objects by larger polygon models, we can set margins to avoid collisions in experiments with actual equipments. The collisions may arise due to positioning errors of robots, errors in modeling and errors in posture measurement of objects.

2.7 *Application to More Complex Objects*

We derived the sufficient conditions of caging for two simple objects. Additionally, we show the applications of these conditions to more complex objects with using the simple objects as shape primitives.

When an object includes either of the above two objects in its body, the corresponding conditions presented in Sec. 2.4 and Sec. 2.5 and the collision-free condition for whole body of the target object are a series of sufficient condition for caging the object. For example, let us consider a case such as Figure3, and the bottle includes an approximated dumbbell-like object (Figure6) in its body. When the robot hand satisfies the sufficient condition for the dumbbell-like object, and does not collide with the whole body of the bottle, the hand can achieve caging the bottle.

The above application can be considered in cases that a target object is not enough approximated by the simple objects but includes them in the part of the object. Let us consider a case such as Figure9, and the handle of the cup includes a closed curve in its body. When the sufficient condition for caging the included ring is satisfied and the robot hand does not collide with the whole body of the cup, the hand can achieve caging the cup.

3 Motion Planning

3.1 *Assumptions and Notations*

In this paper, we study motion planning for caging to produce a path from an initial state to a goal in configuration space

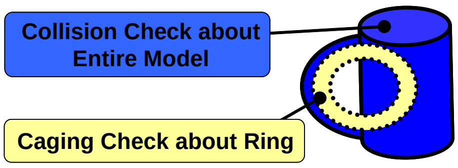

Figure 9 An example of caging conditions for more complex objects

of robots. The goal configuration satisfies the sufficient condition derived in Sec. 2. We use a symmetric hand mentioned in 2.3 and a manipulator with L_{man} joints. We define a vector of joint variables of the manipulator as follows:

$$\boldsymbol{\Theta} := [\Theta_1, \ldots, \Theta_{L_{\mathrm{man}}}]^T, \in \mathbb{R}^{L_{\mathrm{man}}}, \tag{10}$$

where Θ_i denotes the joint variable of the i-th joint. With (1) and (10), we define a vector of robot configuration, $\boldsymbol{z}$, as follows:

$$\boldsymbol{z} := [\bar{\boldsymbol{\theta}}^T, \boldsymbol{\Theta}^T]^T \in \mathbb{R}^{\bar{L}+L_{\mathrm{man}}}. \tag{11}$$

3.2 *RRT-based Planning*

The robot configurations in our caging problems, $\boldsymbol{z}$ has relatively high DOF. In addition, there are multiple goal configurations because the sufficient conditions are described by some inequalities with the joint variables. Thus, we develop our planning algorithm based on RRT (Rapidly exploring Random Trees) (Lavalle; 1998), which is a path planning method using random sampling. RRT is specially designed to handle high DOF. The algorithm in (Lavalle; 1998) is a very basic of RRT, and we can easily adopt it to our planning problem, although more developed algorithms have been presented (Yershova and LaValle; 2007, 2009).

Our planning procedure is almost based on the original RRT and described as follows:

Step 1. Set an initial configuration, $\boldsymbol{z}_{\mathrm{ini}}$, as a seed of a configuration path branches.
Step 2. Generate a random configuration, $\boldsymbol{z}_{\mathrm{rand}}$.
Step 3. Find the nearest configuration, $\boldsymbol{z}_{\mathrm{near}}$ in the current configuration path branches.
Step 4. Put a candidate of new configuration, $\boldsymbol{z}_{\mathrm{cand}}$, between $\boldsymbol{z}_{\mathrm{rand}}$ and $\boldsymbol{z}_{\mathrm{near}}$.
Step 5. Examine whether the robot collides with the target object or any obstacles at $\boldsymbol{z}_{\mathrm{cand}}$. When no collisions are detected, the candidate configuration is added to the branches as a new configuration, $\boldsymbol{z}_{\mathrm{new}}$.
Step 6. Repeat from Step 2 to Step 5 until $\boldsymbol{z}_{\mathrm{new}}$ satisfies both cage-formed conditions and in-cage conditions.

In Step 3, in order to determine the nearest configuration in the current path branches from a randomly-sampled configuration, $\boldsymbol{z}_{\mathrm{rand}}$, we calculate a norm between a configuration in the branches, $\boldsymbol{z}$ and $\boldsymbol{z}_{\mathrm{rand}}$, d_{cfg} defined with the following equation:

$$d_{\mathrm{cfg}}(\boldsymbol{z}, \boldsymbol{z}_{\mathrm{rand}}) := \sum_{j=1}^{\bar{L}} \left(\frac{(\bar{\theta}_j - \bar{\theta}_{\mathrm{rand}})^2}{\dot{\theta}^2_{\mathrm{max},j}} \right) + \sum_{j=1}^{L_{\mathrm{man}}} \left(\frac{(\Theta_j - \Theta_{\mathrm{rand}})^2}{\dot{\Theta}^2_{\mathrm{max},j}} \right), \tag{12}$$

where, $\dot{\theta}_{\mathrm{max},j}$ and $\dot{\Theta}_{\mathrm{max},j}$ is maximum joint velocity of j-th joint of the finger and the manipulator, respectively. This normalization depends on the potential of each actuators, and the norm, $d_{\mathrm{cfg}}(\boldsymbol{z}, \boldsymbol{z}_{\mathrm{rand}})$, means the square sum of the time to move from $\boldsymbol{z}$ to $\boldsymbol{z}_{\mathrm{rand}}$ with each maximum joint velocity. When a configuration in the current branches, $\boldsymbol{z}$, gives the minimum d_{cfg}, the $\boldsymbol{z}$ becomes $\boldsymbol{z}_{\mathrm{near}}$.

In Step 4, a candidate configuration of the branches, $\boldsymbol{z}_{\mathrm{cand}}$, is arranged between $\boldsymbol{z}_{\mathrm{rand}}$ and $\boldsymbol{z}_{\mathrm{near}}$ with Linear Interpolation Method as follows:

$$\boldsymbol{z}_{\mathrm{cand}} := \boldsymbol{z}_{\mathrm{near}} + a_{\mathrm{br}} \frac{\boldsymbol{z}_{\mathrm{rand}} - \boldsymbol{z}_{\mathrm{near}}}{||\boldsymbol{z}_{\mathrm{rand}} - \boldsymbol{z}_{\mathrm{near}}||}, \tag{13}$$

where a_{br} is a positive number restricted in the following condition:

$$0 < a_{\mathrm{br}} \leq ||\boldsymbol{z}_{\mathrm{rand}} - \boldsymbol{z}_{\mathrm{near}}||. \tag{14}$$

In Step 5, we use PQP (Larsen et al.; 1999) to inspect presence or absence of collision among the robot, the object and any obstacles at $\boldsymbol{z}_{\mathrm{cand}}$. Unless any collisions are detected, we increase a_{br} in (13) and examine collision tests about new $\boldsymbol{z}_{\mathrm{cand}}$. A certain $\boldsymbol{z}_{\mathrm{cand}}$ in which no collision exist can be added as a new configuration path branches, $\boldsymbol{z}_{\mathrm{new}}$.

Finally we can find a goal configuration that satisfies the sufficient condition of caging, and obtain a path of the robot motion.

3.3 *Biased Sampling with Solving Inverse Kinematics*

Random sampling is useful for path planning in a complex environment and high-dimensional configuration space, particularly cases of difficulty to find a goal configuration. However, it requires large computational cost to search wide area in configuration space, which includes obviously-distant points from a goal configuration. Thus, we adopt a method of biased sampling to our planner. The objective of biased sampling is to move the robot hand toward a target object more efficiently, and to reduce wasteful sampling of configuration.

Our biased sampling is based on solving inverse kinematics of manipulator. A standard inverse kinematics problems are solved with a determined posture, but in caging, a particular posture is not set because of multiple solutions in sufficient conditions of caging. Thus, we give a desired position of the tool center point of the manipulator that is enough close to a target object to capture it. Next we give a desired orientation by using an algorithm to generate uniformly-distributed random unit quaternions (Kuffner; 2004). If we cannot acquire a solution for the position and orientation of the manipulator, we have to repeat the above trial until a solution is found.

Once we obtain a set of joint variables of the arm, $\boldsymbol{\Theta}$, solving an inverse kinematics problem, we utilize the joint variables (IKV) instead of a randomly-sampled values (RSV) with a fixed probability, p_{biased}, in Step 2 of our planning procedure (See in Sec. 3.2). We note that the joint variables of wrist rolling joint of the arm, $\Theta_{\bar{L}_{\mathrm{man}}}$, and those of the fingers, $\bar{\boldsymbol{\theta}}$ are always randomly-sampled (Table1).

Table 1 Probabilistic utilization of random sampling and biased sampling

	$\Theta_1, \ldots, \Theta_{\bar{L}_{\rm man}-1}$	$\Theta_{\bar{L}_{\rm man}}$	$\bar{\theta}$
$p_{\rm biased}$	IKV	RSV	RSV
$1 - p_{\rm biased}$	RSV	RSV	RSV

4 Experiments

We show some experimental results of our caging planning and execution. The intended objects for caging in this paper are four.

4.1 Experimental Settings

Our robotic arm/hand system to perform caging experiments consists of a 6-DOF manipulator (MOTOMAN-HP3J) and a robot hand that comprises two fingers with two joints each. In motion planning, the arm/hand robot are represented as follows (Figure10): .

- Each limb of the fingers is represented by a cuboid with width: 0.03 [m], thickness: 0.01 [m], length: 0.05 [m].
- Each actuator is represented by a cylinder with radius: 0.015 [m], length: 0.03 [m].
- The palm is represented by a plate with $0.07 \times 0.07 \times 0.01$[m].

Each range of the joints are

$$\begin{gathered}
-\Theta_{l,\max} < \Theta_l < \Theta_{l,\max},\ -\bar{\theta}_{l,\max} < \bar{\theta}_l < \bar{\theta}_{l,\max}, \\
\text{where,} \quad \bar{\theta}_{2,\max} = 0.5\pi,\ \bar{\theta}_{3,\max} = \pi, \\
\Theta_{1,\max} = 0.5\pi,\ \Theta_{2,\max} = 0.472\pi,\ \Theta_{3,\max} = 0.806\pi \\
\Theta_{4,\max} = 0.944\pi,\ \Theta_{5,\max} = 0.667\pi,\ \Theta_{6,\max} = 0.5\pi,
\end{gathered}$$

where the angle of $\bar{\theta}_1$ is fixed to 0.5π (every unit of the above parameters is [rad]). Parameters of the manipulator are described in Table2 and 3. The joints of both the arm and the hand are position-controlled. In addition, there are no force sensors and/or tactile sensors to execute caging constraint.

There are four target objects: a cylinder (with a hollow), a thin ring, a mug-like object (Figure11) for ring-type caging, and a dumbbell for waist-type caging, which is with $r_{\rm disk} = 45$, $t_{\rm disk} = 1$, $d_{\rm waist} = 26$ and $r_{\rm waist} = 222$ ([mm]). In collision tests, curved surfaces are not dealt with, and every model must be a polyhedron. Thus we approximate every curve by line segments, for example, a circle is approximated by a polygon.

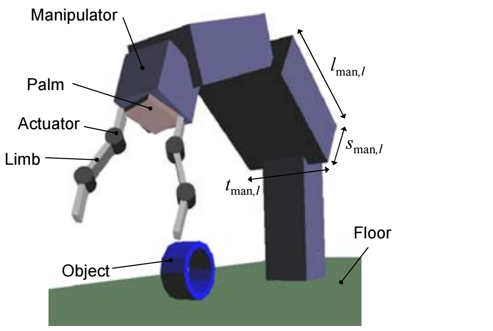

Figure 10 Models of a robot and an object in motion planning

Table 2 Link parameters of MOTOMAN-HP3J

i	α_{i-1}[rad]	a_{i-1}[m]	d_i[m]	$\bar{\theta}_i$[rad]
1	0	0	0	$\bar{\theta}_1$
2	$-\frac{\pi}{2}$	0.26	0	$\bar{\theta}_2$
3	0	0	0	$\bar{\theta}_3$
4	$-\frac{\pi}{2}$	0.03	0.27	$\bar{\theta}_4$
5	$\frac{\pi}{2}$	0	0	$\bar{\theta}_5$
6	$-\frac{\pi}{2}$	0	0	$\bar{\theta}_6$

Table 3 Link size of the manipulator in collision check and visualization

i	$l_{\rm man,l}$[m]	$s_{\rm man,l}$[m]	$t_{\rm man,l}$[m]
1	0.29	0.104	0.104
2	0.26	0.0995	0.199
3	0.00	0.094	0.136
4	0.27	0.094	0.136
5	0.09	0.095	0.095
6	0.01	0.095	0.095

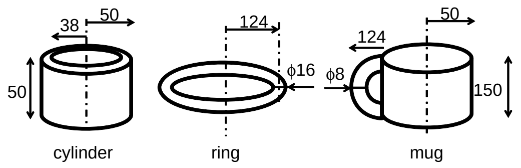

Figure 11 Objects for caging experiments

We test two changed postures of each object. We change only its position for the cylinder, the thin ring and the mug-like object, and both position and orientation for the dumbbell-like object.

Motion planning is performed on a Linux PC whose CPU is Intel Core i7 running at 2.8 GHz with multithread processing.

4.2 Object Recognition

To plan a robot path for caging a target object, the specifications of the object: shape, size and posture are required. In this paper, we use AR picture markers to recognize the object on which a marker is attached, and retrieve the information of the object from a database with an identification number embedded in the marker (Figure12). AR picture markers can be recognized with ARToolKitPlus (Schmalstieg; n.d.; Wagner and Schmalstieg; 2007), which is a software library for building AR applications. The information on the database includes the ID number embedded in the marker, the possible patterns to cage the object, the size of the object and the relative posture of the marker on the object. Thus, measuring position and orientation of the marker, we can estimate those of the object. Additionally, the obtained possible pattern of caging for the object helps us to determine a particular strategy of caging to plan a path of robot motion.

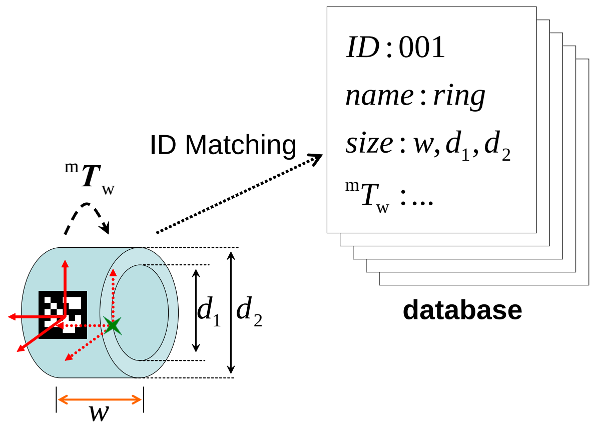
ID Matching
$^{\mathrm{m}}\boldsymbol{T}_{\mathrm{w}}$
$ID:001$
$name:ring$
$size:w,d_1,d_2$
$^{\mathrm{m}}T_{\mathrm{w}}:...$
database
d_1
d_2
w

Figure 12 Linking an AR picture marker on the object and information of the object on the database

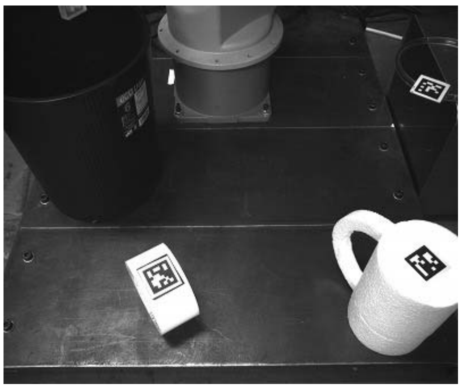

Figure 13 AR picture markers on the objects

Our procedure of object recognition with AR markers is as following steps:

Step 1. Recognize the AR marker attached to the target object, and read the embedded ID number. At the same time, measure posture of the marker.

Step 2. Load the object properties from the database using the ID number (Figure12). They include the attachment location of the marker on the object, the shape and the size of the object and the objective pattern of caging. The posture of the object can be easily calculated.

Using AR picture markers to recognize objects facilitates measurement of the shape and the size of the objects. We tested the accuracy of our position measurement with AR picture markers. The markers are 30×30 mm and located in several points to face to the camera almost oppositely such as Figure13. In our experiments, estimated errors in position measurement using ARToolKitPlus are within 5 mm.

Note that the information of obstacles around the objects are given in this paper.

4.3 Ring-type Caging

In each scene, we could plan the motion of caging that is composed of approaching the hand to the object from an initial configuration (Figure14) and capturing it. The planned

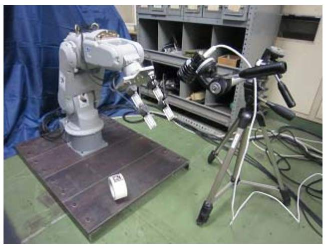

Figure 14 Initial position and orientation of the robot for every trial

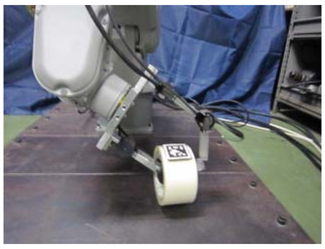

Figure 15 A planned goal configuration of caging the cylinder

motion of the robot was not smoothed in this paper, and applied to practical execution directly.

The cylinder can be self-standing so that the robot fingers access to the hollow region easily (Figure15). Thus, in the caging a cylinder, the planner simply produces a caging motion with the sufficient condition mentioned in Sec. 2.2. The variation of computation time for planning in ten trials was from 1 to 142 CPU seconds.

In caging a thin ring, we require some pedestals to put the target object on there because the ring cannot be self-standing and the robot finger cannot go through the hollow of the ring. Thus we put the ring on the box so that half of the ring are outside of the box in the experiments (Figure16). The box is dealt with as an obstacle in motion planning and checked collision with the robot. The variation of computation time for planning in ten trials was from 1 to 12 CPU seconds.

In caging a mug-like object, the sufficient condition for ring-like object addressed in Sec. 2.4.1 and 2.5.1 are used to examine caging achievement that the hand capture a handle part of the target object. In addition, collision tests are inspected between the whole body of the object and the robot as Figure9. The robot could track the planned motion and insert its fingers into the handle part of the mug-like object successfully (Figure17, 18). The variation of computation time for planning in ten trials was from 1 to 177 CPU seconds.

4.4 Waist-type Caging

We could also plan the robot motion for caging a dumbbell-like object in two patterns of posture (Figure19, 20). The variation of computation time for planning in ten trials is from 3 to 483 CPU seconds about caging of the lying dumbbell-like object; 2 to 73 CPU seconds about caging of the standing one.

4.5 Collision Avoidance

In the planned motion of simulation, the robot did not at all touch the object and any obstacles because the motion planning is performed with the collision-free condition (See Sec. 2.6). In the practical experiments, however, it sometimes collided with the objects. The collision is seems to be mainly due to some errors in position estimation with AR picture marker, which are 5mm in position at a maximum.

In some scenes, caging tasks could be accomplished even when the robot touched an object and moved it from its correct position and orientation (Figure21). In Figure21,

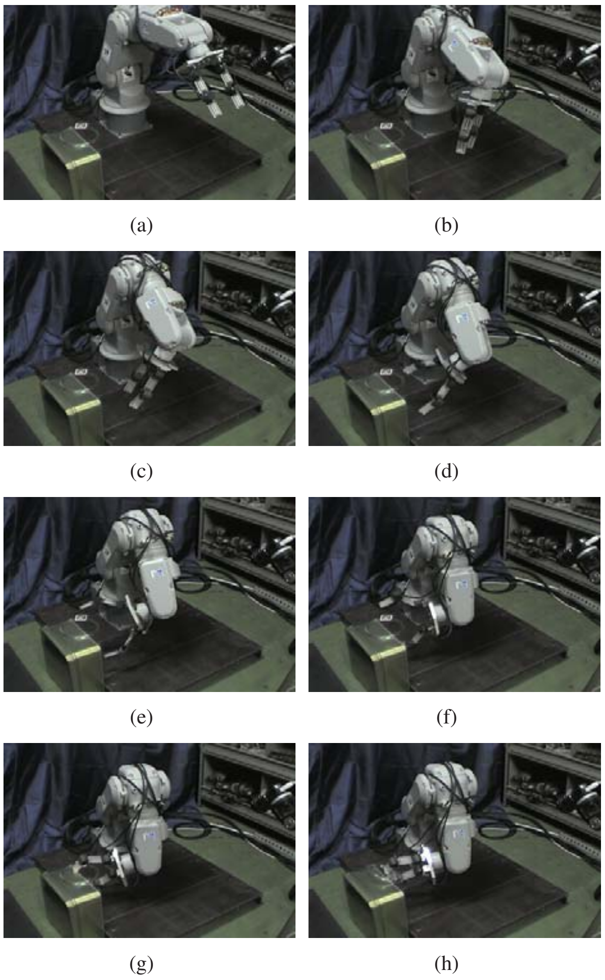
Figure 16 A planned motion for caging the thin ring

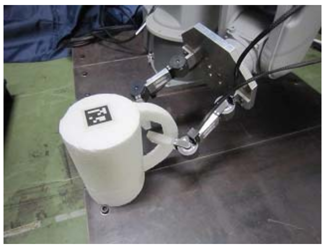
Figure 17 A planned goal configuration of caging the mug-like object

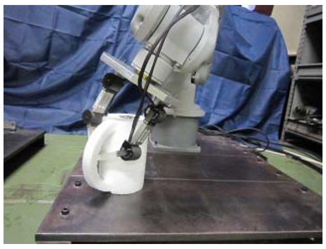
Figure 18 Another planned goal configuration of caging the mug-like object

although the posture of the dumbbell was changed between the initial state (Figure21(a)) and the goal state (Figure21(b)), The hand could capture the handle of the object. It is because some margins are included in the sufficient conditions expressed with a series of inequalities. In other words, even if some unexpected actual errors exist, a goal configuration planned without the errors can also satisfy all the sufficient conditions in successful cases.

In other cases, execution of planned motion failed because the object was moved by the robot, and the robot hand could not reach the object (Figure22). In Figure22, the robot touches the cylinder while moving along the planned motion

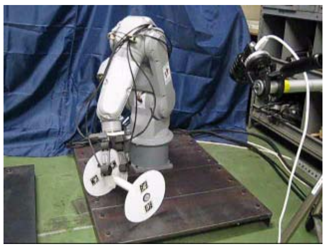
Figure 19 A planned goal configuration of caging the dumbbell-like object in lying position

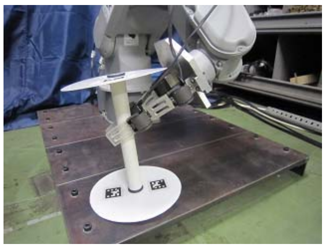
Figure 20 A planned goal configuration of caging the dumbbell-like object in standing position

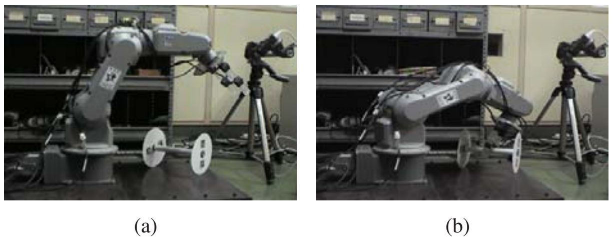
Figure 21 A successful case of planned caging motion even some collisions occur (the object moved)

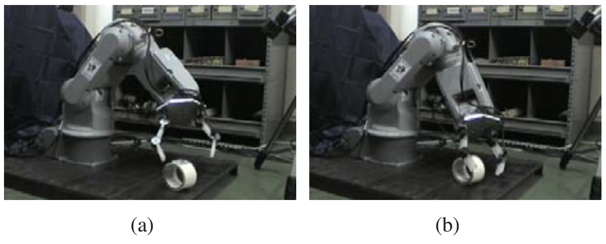
Figure 22 A failure case of planned caging motion caused by any collisions (the object moved)

path, and the object moves away toward the base of the manipulator. The failure tends to happen when the robot bodies are located very close to the object in planned motion. For example, when the planner make the robot bodies locate within 5mm from the object, the error in position estimation mentioned above often can cause any collisions.

To avoid collision in practical experiments, we tested collision checks with a certain amount of margins, that is, we adopted larger models of the objects for planning. Taking the maximum errors of position estimation into account, we added 7mm to each size of the objects defined in Sec. 4.1.

Our motion planner could also produce caging motion for each enlarged model of objects, and no physical collision between the actual robot and the objects occurred in practical experiments. Since the modeled object is larger than its real scale, the robot bodies were located farther point from the object than in previous cases without considering any margins. Every computation time for planning in the cases with considering margins is larger than those without margins. It is because enlarging the model of the object causes decreasing the space where the robot hand can reach, and required path becomes less found. The variation of

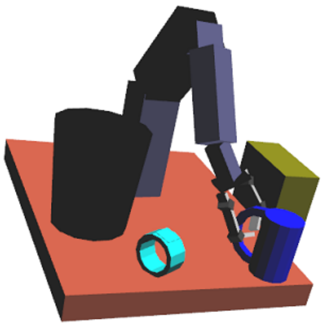
Figure 23 Simulation environment in caging manipulation

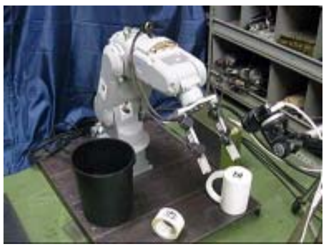
Figure 24 Experimental environment in caging manipulation

computation time for caging in ten trials was from 2 to 1684 CPU seconds about caging the cylinder; from 1 to 103 CPU seconds about caging the thin ring; from 3 to 298 CPU seconds about caging the mug-like object; from 5 to 1352 CPU seconds about caging the lying dumbbell-like object; from 1 to 579 CPU seconds about caging the standing dumbbell-like object, respectively.

4.6 Caging Manipulation

After the robot hand caged one of the object, the robot hand can transport the object without any object escaping from the hand. In this section, we manipulate the robot system by remote control after the robot accomplished a planned caging motion, to let the robot transport the caged object to a trash box. The experiment scheme is below.

Step 1. Capture an experimental scene by a camera and recognize an AR picture marker attached to each object. When some markers exist, the object with the marker at the nearest position from the camera is selected to plan a caging motion.
Step 2. Plan a caging (capturing) motion and perform the planned motion.
Step 3. After the caging, we remotely control the robot capturing the object to transport it to a trash box.
Step 4. Continue Step 1 to Step 3 until all the target objects are in the trash box.

The simulation environment and the experiment one is as Figure23 and Figure24 respectively. Margins of collision detection mentioned in Sec. 4.5 are considered that each is 7mm.

Each caging motion of the robot could be produced and performed by the actual robot system without any collisions between the robot and anything else. After each caging (or capturing) of the object, we remotely manipulate the robot to transport the caged object to the trash box. At that time, the object did not escape from the robot hand, and the transportation could be performed successfully (Figure25).

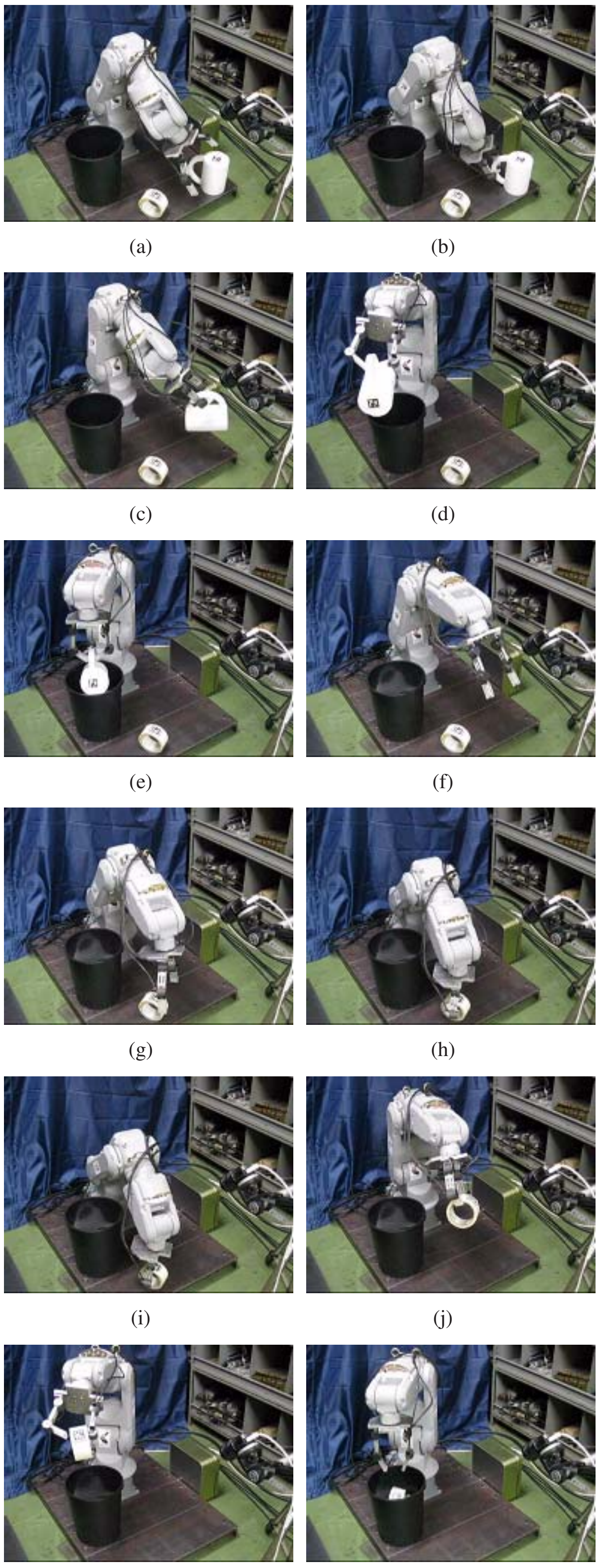

Figure 25 A case of planned caging motion and remote controlled transportation

5 Conclusions

In this paper, we proposed caging grasps by a two-fingered robot hand for two types of objects. First, we classified patterns of the caging into two types with depending on both shapes of objects and strategies of constraint. Next, we derived sufficient conditions for each pattern of caging. With the sufficient conditions, we constructed an RRT-based motion planner for caging, and produced caging motions of a robotic arm/hand system for four objects. We adopted a biased-sampling method with solving inverse kinematics problems of the robot arm for search efficiency. We attached AR picture markers to the objects to recognize them, and estimated the position and orientation of the marker with ARToolKit libraries. Then the position and orientation of the object can be easily calculated with those of the marker. The planned motion of caging could be successfully accomplished by the actual robotic arm/hand system. Since collisions in experiments sometimes occurred due to the errors in posture estimation of the objects, we introduced margins into collision checks in motion planning. After caging the object, we can transport it without its escaping from the hand.

In future works, more variety of caging should be achieved by a multifingered hand, and combinations of caging and other manipulation should be also addressed. Additionally, reduction of computation time has to be considered.

Acknowledgment

This work was partially supported by KAKENHI #22700200 to Y. Maeda.

References

Bicchi, A. and Kumar, V. (2000). Robotic grasping and contact: A review, *Proc. of IEEE Int. Conf. on Robotics and Automation*, San Francisco, CA, U.S.A., pp. 348–353.

Borst, C., Fischer, M. and Hirzinger, G. (2002). Calculating hand configurations for precision and pinch grasps, *Proc. of IEEE Int. Conf. on Intelligent Robots and Systems*, Lausanne, Switzerland, pp. 1553–1559.

Diankov, R., Srinivasa, S. S., Ferguson, D. and Kuffner, J. (2008). Manipulation planning with caging grasps, *Proc. of IEEE/RAS Int. Conf. on Humanoid Robots*, Daejeon, Korea, pp. 285–292.

Erickson, J., Thite, S., Rothganger, F. and Ponce, J. (2007). Capturing a convex object with three discs, *IEEE Trans. on Robotics* **23**(6): 1133–1140.

Kuffner, J. J. (2004). Effective sampling and distance metrics for 3d rigid body path planning, *Proc. of IEEE Int. Conf. on Robotics and Automation*, New Orleans, LA, U.S.A., pp. 3993–3998.

Larsen, E., Gottschalk, S., Lin, M. C. and Manocha, D. (1999). Fast proximity queries with swept sphere volumes, *Technical Report TR99-018*, Computer Science Dept., University of North Carolina, Chapel Hill.

Lavalle, S. M. (1998). Rapidly-exploring random trees: A new tool for path planning, *Technical Report TR98-11*, Computer Science Dept., Iowa State University.

Maeda, Y., Kodera, N. and Egawa, T. (2012). Caging-based grasping by a robot hand with rigid and soft parts, *Proc. of IEEE/RSJ Int. Conf. on Robotics and Automation*, Saint Paul, MI, USA, pp. 5150–5155.

Makita, S. and Maeda, Y. (2008). 3d multifingered caging: Basic formulation and planning, *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and System*, Nice, France, pp. 2697–2702.

Makita, S., Okita, K. and Maeda, Y. (2012). Motion planning for 3d multifingered caging with object recognition using ar picture markers, *Proc. of IEEE Int. Conf. on Mechatronics and Automation*, Chengdu, China, pp. 2158–2164.

Miller, A. T. and Allen, P. K. (2004). Graspit!: A versatile simulator for robotic grasping, *IEEE Robotics and Automation Magazine* **11**(4): 110–122.

Miller, A. T., Knoop, S., Christensen, H. I. and Allen, P. K. (2003). Automatic grasp planning using shape primitives, *Proc. of Int. Conf. on Robotics and Automation*, Taipei, Taiwan, pp. 1824–1829.

Ozawa, R., Arimoto, S., Nakamura, S. and Bae, J.-H. (2005). Control of an object with parallel surfaces by a pair of finger robots without object sensing, *IEEE Trans. on Robotics* **21**(5): 965–976.

Pipattanasomporn, P. and Sudsang, A. (2006). Two-finger caging of concave polygon, *Proc. of IEEE Int. Conf. on Robotics and Automation*, Orlando, FL, U.S.A., pp. 2137–2142.

Pipattanasomporn, P. and Sudsang, A. (2011). Two-finger caging of nonconvex polytopes, *IEEE Trans. on Robotics* **27**(2): 324–333.

Rimon, E. and Blake, A. (1996). Caging 2d bodies by 1-parameter two-fingered gripping systems, *Proc. of IEEE Int. Conf. on Robotics and Automation*, Minneapolis, MN, U.S.A., pp. 1458–1464.

Rimon, E. and Blake, A. (1999). Caging planar bodies by one-parameter two-fingered gripping systems, *Int. J. of Robotics Research* **18**(3): 299–318.

Rodriguez, A. and Mason, M. T. (2008). Two finger caging: Squeezing and stretching, *Proc. of Int. Workshop on the Algorithmic Foundation of Robotics*, Guanajuato, México.

Schmalstieg, D. (n.d.). ARToolKitPlus, `http://handheldar.icg.tugraz.at/artoolkitplus.php`.

Shimoga, K. B. (1996). Robot grasp synthesis algorithm: A survey, *Int. J. of Robotics Research* **15**(3): 230–266.

Smith, G., Lee, E., Goldberg, K., Böhringer, K. and Craig, J. (1999). Computing parallel-jaw grips, *Int. Conf. on Robotics and Automation*, Detroit, MI, USA, pp. 1897–1903.

Vahedi, M. and van der Stappen, A. F. (2008). Caging polygons with two and three fingers, *Int. J. of Robotics Research* **27**(11-12): 1308–1324.

Wagner, D. and Schmalstieg, D. (2007). Artoolkitplus for pose tracking on mobile devices, *Computer Vision Winter Workshop*.

Wang, Z. and Kumar, V. (2002). Object closure and manipulation by multiple cooperating mobile robots, *Proc. of IEEE Int. Conf. on Robotics and Automation*, Washington D.C., U.S.A., pp. 394–399.

Yamanobe, N. and Nagata, K. (2010). Grasp planning for everyday objects based on primitive shape representation for parallel jaw grippers, *Int. Conf. on Robotics and Biomimetics*, Tianjin, China, pp. 1565–1570.

Yershova, A. and LaValle, S. M. (2007). Improving motion planning algorithm by efficient nearest-neighbor searching, *Trans. on Robotics* **23**(1): 151–157.

Yershova, A. and LaValle, S. M. (2009). Motion planning for highly constrained spaces, *Proc. of IEEE Conference on Robot Motion and Control*, pp. 297–306.