Signal Processing 120 (2016) 721–730

Contents lists available at ScienceDirect

Signal Processing

journal homepage: www.elsevier.com/locate/sigpro

# Error-tolerant manipulation by caging

Weiwei Wan, Feng Lu*, Rui Fukui

*The University of Tokyo, Tokyo, Japan*

ARTICLE INFO

*Article history:*
Received 4 September 2014
Received in revised form
19 November 2014
Accepted 6 December 2014
Available online 15 December 2014

*Keywords:*
Caging
Grasping
Robot finger

ABSTRACT

This paper delivers a preliminary attempt to find the optimized caging positions for a three-finger robotic hand designed for home-use logistical environments. The main idea behind optimizing caging positions falls in that optimal caging can afford largest margins to stop target objects from escaping into infinity. By employing the advantages of largest margins, optimal caging grasp can be robust enough to endure dramatic perception noises or errors and low sensing resolutions. This paper optimizes object grasping towards caging. Specifically, our algorithm utilizes Genetic Algorithm (GA) to accelerate the searching procedure and evaluate a fitness of the GA population by examining a combination of max–min, which corresponds to intersections of neighbour fingers' $\mathcal{CC}$ space margins, and least inter-finger distance for optimization. Simulation results show that the manipulation strategy proposed in this paper could in the worse case coordinate with sensors whose resolution are less than one pixel per centimeter.



## 1. Introduction

Many work has been devoted to robot manipulation in the past decades, including finger synthesizing [1], multiple robot cooperation [2], etc. Despite the various projects and publications, the research in manipulation can be divided into two groups where one group studies the manipulation problem from the viewpoint of actuation while the other group studies the problem from the viewpoint of perception.

In actuation, contemporary interests mainly lie in closures like form/force closure [3] and object closure/caging [4]. There also exists many practical applications of nonprehensile manipulation. Previous researches in closure tend to optimize force synthesizing for external force sets [5] or specific tasks [6], sometimes taking into kinematic constraints of hardware structures [7]. Theoretically, these optimization are complete and interesting. However, seldom some pragmatic systems consider closures due to toughness of perception. Traditional closure research [8] assumes exact positions, orientations and shapes of target objects. They suffer from crashing in real applications where perfect perception information cannot be obtained. Caging [9], as an extension to force closures, offers a passive and force-less way of manipulation. Although caging does not require exact force, it is far from error-tolerant synthesizing. Even the preliminary step, caging test, has been demonstrated challenging.

In perception, researchers try to build target information into boundary/surface clouds, curves, polygons or polyhedrons by using image segmentation [10,11]. Some works, such as curve fitting [12,13], have achieved certain success in cooperation with actuation. The cooperation involves two procedures. A geometric shape is modeled in the first procedure and acts as the medium of synthesizing in the second procedure. However, it is not always satisfying and may collapse with novel or unusual target shapes [14]. Some other works try to match target objects with database models, or namely, to recognize target objects. Modeling and retrieving indicate an excellent solution to

* Corresponding author.
*E-mail addresses:* wanweiwei07@gmail.com (W. Wan), lufeng@ut-vision.org (F. Lu).

http://dx.doi.org/10.1016/j.sigpro.2014.12.006
0165-1684/

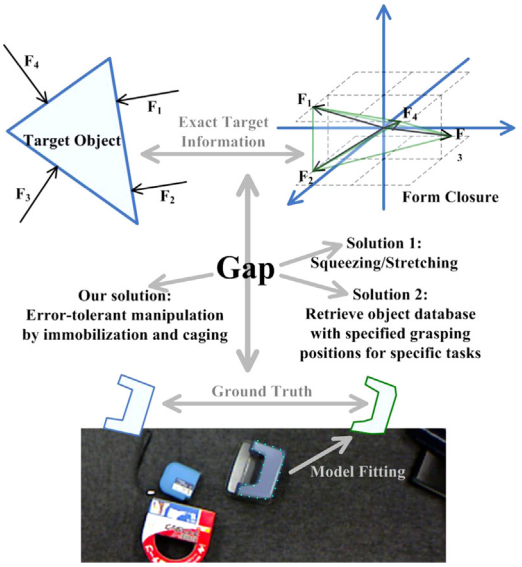


**Fig. 1.** The gap between perception and actuation.

offer precise shape information, even in the presence of overlaps. Nevertheless, errors arise from scale, translation and orientation [15]. Ambiguous configurations usually lead to major failure of actuation. Recently, researchers begin to seek for remedies of configuration errors by matching more precisely or modeling with larger compliance. But it remains hard to evaluate the flexibility.

Both research in actuation and perception have made conspicuous achievements. However, there is a gap between them. Researches in actuation take little consideration of perception information while the synthesizing approaches employed in perception researches guarantee no robust manipulation. The gap motivates us to explore synthesizing approaches that best tolerate perception errors. Fig. 1 demonstrates the gap and some popular solutions towards it.

In order to fill up the gap, we propose a robust finger synthesizing approach in this paper by considering model errors caused from the perception phase. Intrinsically, the approach is immobilization synthesizing that ensures (1) in the best case when perception is perfect, a target object is immobilized by fingers (2) even though certain perception errors occur, the target object is still constrained in a compact configuration region. If each finger is taken as a single-parameter link, our proposal finally converges into another immobilization by shrinking individual parameters.

Our approach is based on the relationship between finger objects in $\mathcal{C}$ space and $\mathcal{CC}$ space. It seeks maximum error-tolerance by simultaneously maximizing the breaking margins formed by intersections of adjacent finger neighbours and minimizing rotation moments caused by finger scatter. Major contributions of our proposal are as follows: (1) The proposal in this paper not only works with 2D planar objects but also works with 3D targets. (2) It can choose finger numbers automatically according to actual requirements or limitations of robustness. (3) A redundant solution is proposed to deal with concave objects, rather than 2-finger squeezing or stretching.

Our work is like [16,17] where correlative models or hierarchical models were developed for multimedia recognition. To our best knowledge, this paper is the initial work which applies such ideas to robotic manipulation. Using a max–min constraint plus a least inter-finger distance constraint for optimization essentially shares the same idea with the multi-modal based technique in signal processing. We use the optimization to process the vision information collected from KINECT sensors. We believe our work is of great interest to both researchers in robotics and signal processing.

The organization of this paper is as follows: background works like immobilization, $\mathcal{C}$ space, $\mathcal{CC}$ space, caging test, etc., are presented in Section 2. Section 3 discusses our consideration in perception errors and the routine of robust synthesizing in planar environment, followed by an extension to 3D environment in Section 4. These two sections compose major contributions of our work. The strategy to deal with concave objects is presented in Section 5 to ensure completeness of our proposal. Section 6 shows some experiments and analysis with both real sensor data and virtual 3D models. Finally, conclusions are drawn in the last section.

## 2. Background works

Form closure theoretically presents how to synthesize contacts that can resist certain external forces [18]. It requires at least $n_{dof}+1$ force contacts to grasp a target object with $n_{dof}$ degree of freedom. Formally, form closure can be achieved by fulfilling $0 \in \text{int}(\text{conv}(\{\mathbf{f_i}\}))$. Although form closure indicates an important conclusion, it is not as practical with real robotic hands where friction and finger constraints always exist and finger number seldom satisfies requirements.

In actual application, it is pragmatic to simply take into account immobilization grasp [19]. Fingers in immobilization grasp can be considered as fixed obstacles once they are placed to certain contact configurations. In that case, fingers do not exert forces explicitly but bear wrenches caused by target weights and external forces passively. Immobilization grasp owns reasonable benefits compared with the loose equilibrium grasp which depends too much on target properties and the strict form closure grasp. An immobilization grasp requires at least 2 or $n_{dim}+1$ [1] to $2\times n_{dim}$ finger contacts to immobilize an object in $n_{dim}$ dimensional space.

Given a finger $\mathcal{A}_i$, we employ $\mathcal{C}_{oi}$ to indicate its correspondent configuration space ($\mathcal{C}$ space) presence. Immobilization grasp means that a target object, denoted by a configuration point $q$ in $\mathcal{C}$ space, is isolated as a single point. Consequently, $\mathcal{C}$ space can be divided into three parts when immobilization grasp is formed, namely $q$, $\cup_{i=0}^{n}\mathcal{C}_{oi}$ and $\mathcal{C}_f$ ($\mathcal{C}_{free}$). According to this definition, a circle can never be immobilized since its 2 order derivative [20] is equal to 0 and no independent $q$ can be isolated. At least four fingers are required to immobilize a semi-circle.

[1] Two fingers are sometimes enough for certain concave targets. However, we prefer $n_{dim}+1$ in this paper for safety and convenience.

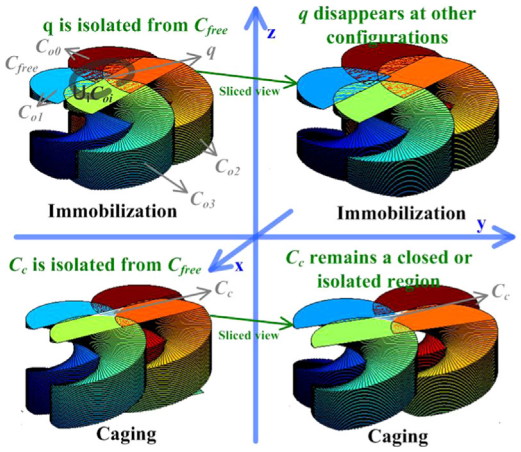


**Fig. 2.** The relationship between immobilization and caging.

Caging is different from form/force closures and immobilization. The key idea of caging is to constraint target objects in a certain area so that the objects cannot escape into infinity or to constrain the target object in a certain region [21,22]. Essentially, caging is a looser definition of immobilization where the isolated $q$ point expands into an isolated $\mathcal{C}$ space region $\mathcal{C}_c$. Correspondingly, $\mathcal{C}$ space is divided into $\mathcal{C}_c$, $\cup_{i=0}^{n}\mathcal{C}_{oi}$ and $\mathcal{C}_f$ where $\mathcal{C}_c \neq \varnothing \wedge (\mathcal{C}_c \cap \mathcal{C}_f = \varnothing)$. Fig. 2 demonstrates the relationship between immobilization and caging by using a semi-circle target object and four point fingers. The four rotating semi-circles in Fig. 2 indicate $\mathcal{C}_{oi}$ s of the four point fingers.

Caging gains much interest from the realm of multiple robot cooperation since it requires no contact between agent and target. Despite its advantages, caging test is a tough problem due to the infinite combinations of agent or finger configurations. Many approaches have been proposed to deal with the dimension curse. $\mathcal{CC}_{oi}$ of a $\mathcal{C}$ space finger $\mathcal{C}_{oi}$ is the most popular tool [23]. When fingers are identical, a $\mathcal{C}_{oi}$ can be mapped into $\mathcal{CC}_{oi}$ with respect to another $\mathcal{C}_{oj}$ following the definition of $\mathcal{C}$ space. The mapping made it easier to check connectivity of $\cup_{i=0}^{n}\mathcal{C}_{oi}$ and independence of $\mathcal{C}_c$ and $\mathcal{C}_f$.

In this paper, we further utilize the connectivity of $\cup_{i=0}^{n}\mathcal{C}_{oi}$ to optimize caging. The optimization aims at grasps that are most tolerant to perception errors. It simultaneously maximize the breaking margins formed by connected adjacent finger neighbours and minimize rotation moments caused by finger scatter. In the best case when perception is perfect, a target object is immobilized by fingers. When certain perception errors occur, the initial finger configuration deteriorates into caging, ensuring maximum error-tolerance.

## 3. 2D error-tolerant manipulation

With two fingers given, Erickson proposed a $z$-buffer based approach to accumulate the caging region of a third finger [24]. Vahedi proved that as the third finger shrinks, it will finally stop at an immobilizing grasp configuration [25]. Caging is actually the extension of immobilization while immobilization is a minimum caging.

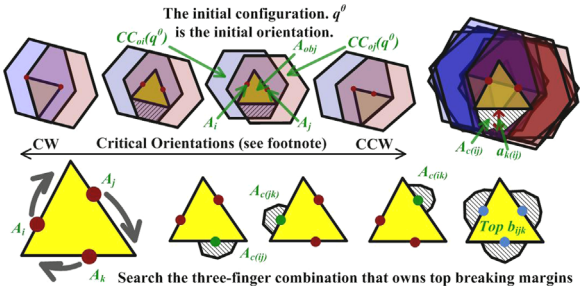


**Fig. 3.** Breaking margin and the optimal error-tolerant caging position.

The caging region of a third finger can be obtained when the other two are fixed. If we employ $\mathcal{A}_c$ to denote this region, the optimal error-tolerant position to pose the third finger should be the position that owns largest offset from $\partial\mathcal{A}_c \backslash \partial\mathcal{A}_{obj}$. Here $\partial\mathcal{A}_{obj}$ indicates boundary of the target object. This region is defined as the breaking margin in this paper. Suppose the other two fingers are named $\mathcal{A}_i$ and $\mathcal{A}_j$, then the breaking margin formed by $\mathcal{A}_i$ and $\mathcal{A}_j$ is $\mathcal{A}_{c(ij)}$ and the optimal position to pose the third finger $\mathcal{A}_k$ should fulfill $\mathcal{A}_k \in \mathcal{A}_{c(ij)}$ and owns maximum caging escaping distance $a_{k(ij)}$ where $a_{k(ij)} = \max(\mathrm{dist}(\mathcal{A}_k, \partial\mathcal{A}_{c(ij)} \backslash \partial\mathcal{A}_{obj}))$.

When none of the fingers are fixed, the optimal error-tolerant position for each finger can be obtained by virtually fix the other fingers. In this way, we can generate the following steps: (1) suppose $\mathcal{A}_i$ and $\mathcal{A}_j$ are fixed, generate $a_{k(ij)}$. (2) Suppose $\mathcal{A}_i$ and $\mathcal{A}_k$ are fixed, generate $a_{j(ik)}$. (3) Generate $a_{i(jk)}$ accordingly. (4) Find the minimum breaking offset of the three offsets, namely find $b_{ijk} = \min(a_{k(ij)}, a_{j(ik)}, a_{i(jk)})$. (5) For each $\mathcal{A}_i$, $\mathcal{A}_j$ and $\mathcal{A}_k$ combination, seek the combination that maximize $b_{ijk}$ as the optimal error-tolerant manipulation configuration. Fig. 3[2] demonstrates this idea.

These five steps constitute the core of our error-tolerant manipulation planning approach. However, we are still confronting a lot of challenging problems where the two major blocks are (1) computational complexity and (2) arbitrary finger numbers. The following part of this section will go deep into these two aspects.

### 3.1. Computational complexity

It has been proved that the computational complexity to generate $\mathcal{A}_{k(ij)}$ is asymptotic to $O(n_{bdry}^2 n_{co}^2)$. Here, $n_{co}$ denotes the number of critical orientations while $n_{bdry}$ denotes the number of boundary edges when the target shape is fitted to a certain polygon. $O(n_{bdry}^2 n_{co}^2)$ costs too much and indicates an impractical algorithm. What's worse, we need extra $O(n_{bdry}^3)$ to traverse the combinations of $\mathcal{A}_i$, $\mathcal{A}_j$ and $\mathcal{A}_k$. Formally, $n_{co} = O(n_{bdry}^2)$ and the total cost of global optimization could be $O(n_{bdry}^9)$ if the searching procedure is performed without any heuristics. Generally, a conclusion can be drawn that the actual cost should be between $\omega(n_{bdry}^6)$ and $o(n_{bdry}^9)$.

[2] The $\mathcal{A}_{c(ij)}$ is drawn with the concept of critical orientations, actually the contact-maintaining motion could be tracked by following intersections of $\mathcal{C}$ fingers. Tracking the intersection indicates a more complete solution to breaking margin.

As has been illustrated, the total cost is composed of $O(n_{bdry}^2 \times n_{co}^2 \times n_{bdry}^3)$ where the major consumer of computational resources is $n_{co}^2 = O(n_{bdry}^4)$. Note that although the last item $n_{bdry}^3$ implies another important consumer, it is not considered in this paper as we can refer to many ready-made algorithms such as genetic evolution to expedite its efficiency. In order to deal with the major consumer $n_{co}^2$, we dispose the concept of critical orientations and decompose canonical escaping motions into translation and rotation. Then the procedure to generate optimal error-tolerant positions changes into the following form. (1) Suppose $\mathcal{A}_i$ and $\mathcal{A}_j$ are fixed, generate the translational caging escaping distance of $\mathcal{A}_k$, namely $a_{k(ij)}^t$. (2) Generate $a_{i(jk)}^t$ and $a_{j(ik)}^t$ following the same procedure. (3) Find the minimum $b_{ijk}^t$. (4) For each $\mathcal{A}_i$, $\mathcal{A}_j$ and $\mathcal{A}_k$ combination, accumulate a potential set $\mathcal{D} = \{\{\mathcal{A}_i, \mathcal{A}_j, \mathcal{A}_k\} | b_{ijk}^t > \tau\}$. (5) For each element in $\mathcal{D}$, calculate the scatter $s_{ijk}$ of fingers. The combination from $\mathcal{D}$ that owns minimum $s_{ijk}$ will act as the configuration of best error-tolerant manipulation.

The $\tau$ in step (4) is a threshold determined with respect to user requirements. Generally, it should not deviate too much from $\max(b_{ijk}^t)$, according to specific target shapes. A formal way to evaluate the bound of $\tau$ is to check boundary curve of the target object. Interested readers may refer to REF for details. Besides set accumulation, $\tau$ plays an important role in deciding finger numbers. Details regarding this role are shown at the end of this section.

By accumulating a set of minimal combinations and seeking the element with least scatter in this set can simultaneously guarantee optimal translational and rotational error-tolerance, and hence ensure an error-tolerant manipulation. Since there is only linear operation in generating $a_{k_{ij}}^t$, its cost decreases into $O(n_{bdry})$. Consequently, the overall time efficiency of our error-tolerant optimization becomes $o(n_{bdry}^4)$, depending on the specific trick algorithm employed in combination searching. Actually, it is already a promising algorithm on modern computers even if we employ exhaustive searching directly.

The decomposed approach is not guaranteed to be exactly the same as the original optimization. In a certain sense, the original approach can neither be considered as a best solution. Here two assumptions are introduced. For the original approach, we assume the fixture of two given fingers. For the decomposed alternative, we assume higher importance on translational margins. Indeed, it depends on the measurement between translation and rotation, or formally $\omega_b b_{ijk}^t$ and $\omega_r s_{ijk}$, to evaluate the tolerance. The strategy in this section offers an effective way to collaborate $b_{ijk}^t$ and $s_{ijk}$ to approximate best tolerance as translational error does play a more important role in real manipulation applications.

### 3.2. Arbitrary finger numbers

In the preceding contents we have demonstrated that the optimal error-tolerant manipulation configurations can be generated in $O(n_{bdry} \times n_{bdry}^3)$ time. Here number three indicates that three fingers are considered in the planning procedure. However, more fingers are required in many cases. Caging is the extension of immobilization and thus it requires the same least finger numbers. Note that in this section target objects are limited to convex shapes, strategies to deal with concave targets will be discussed in Section 5. For planar targets, it requires at least three to four fingers to immobilize a target. For instance, we may need three fingers to cage a triangle while may require four fingers to cage a rectangle. Consequently, it is necessary to consider the case of multiple finger numbers.

We can nearly follow the same procedure as three-finger caging to perform $n$ finger optimization. That is to (1) calculate $a_i^t$ of each finger, (2) accumulate the set with maximal $b_{1,2,\ldots,n}^t$ and (3) seek the combination with least $s_{1,2,\ldots,n}$ as the optimal results. Here toughness comes from the calculation of $a_i^t$. We propose the following lemma to generate translational caging region.

**Lemma 1.** *In the case of planar convex caging, the translational caging region of a finger $\mathcal{A}_i$ is determined by its two adjacent neighbours $\mathcal{A}_{i-1}$ and $\mathcal{A}_{i+1}$, namely $\mathcal{A}_{c((i-1)(i+1))}^t$ where $\mathcal{A}_{c(i-1)(i+1)}^t$ is generated by intersection of $\mathcal{CC}_{o(i+1)}(q^\theta)$, $\mathcal{CC}_{o(i-1)}(q^\theta)$ and $\mathcal{A}_{obj}$.*

Note that the $\mathcal{CC}_{o(x)}$ s here are specific to certain orientation $q^\theta$. It corresponds a slice of Fig. 2 In order to prove Lemma 1, we introduce the following proposition. Note that the involved fingers in planar case should own at least two adjacent fingers, or else it is invalid.

**Proposition 1.** *When a planar object is translationally caged, all $\mathcal{C}$ fingers at a given orientation $q^\theta$ form a chain or any necessary finger $\mathcal{A}_i$ simultaneously lies in its adjacent $\mathcal{CC}_{o(i-1)}(q^\theta)$ and $\mathcal{CC}_{o(i+1)}(q^\theta)$ and $q$ of $\mathcal{A}_{obj}$ lies in $\mathcal{C}_c(q^\theta)$.*

According to Proposition 2, the translational caging region can be evaluated with respect to its adjacent neighbours. A key point here is adjacency, the adjacent neighbours on right hand, $i-1$, and left hand, $i+1$, of $i$. Adjacency requires convex finger formations. When the formation of fingers is concave, adjacent $\mathcal{CC}_{o(x)}(q^\theta)$ may fail to offer maximum tolerance and Lemma 1 fails. We require convex targets in this section as our planning aims at immobilization and the searching is performed according to combinations of object boundary clouds. The convexity of targets ensures convex formation of fingers. Fig. 4 demonstrates the idea of adjacency and Lemma 1. Following Lemma 1, $a_i^t$ can be generated and the optimization procedure can be performed smoothly.

The pseudo code to generate error-tolerant manipulation configurations of a convex target are shown in Algorithm 1. $\tau$, which is employed to accumulate the potential set $\mathcal{D}$, also acts as a filter to decide finger numbers. Some objects, for instance the object in Fig. 4, owns small $\max(b_{1,2,\ldots,n}^t)$ so that $\mathcal{D}$ accumulates few elements. In that case, the system may require more fingers for safety. Generally, four fingers are practical safe upper bound of planar manipulation and planar error-tolerant manipulation of convex targets requires three to four fingers.

**Algorithm 1.** etPlanning.

**Data**: $\mathcal{A}_{obj}$, $\mathcal{CC}_{fgr}(q^{\theta})$, $n_{fgr}$, $\tau$
**Result**: $\mathcal{L}_{fgr}$, $b_{fgr}$
$\mathcal{D} \leftarrow \varnothing$
**global** *boundary clouds* $\leftarrow$ `getBoundry` $\mathcal{A}_{obj}$
**foreach** $n_{fgr}$ *combination* $\mathcal{N}_{fgr}$ *of boundary clouds* **do**
  $b_{tmp} \leftarrow$ `getMinMargin` $\mathcal{N}_{fgr}$
  **if** $b_{tmp} > \tau$ **then**
    $|\mathcal{D} \leftarrow \mathcal{D} \oplus [\mathcal{N}_{fgr}, b_{tmp}]$
  **end**
**end**
**If** $|\mathcal{D}| \equiv 0$ **then**
  `etPlanning` $(\mathcal{A}_{obj},\ \mathcal{CC}_{fgr}(q^{\theta}), n_{fgr}+1, \tau)$
**else**
  $s_{min} \leftarrow +\infty$
  **foreach** $\mathcal{N}_i$ *in* $\mathcal{D}$ **do**
    $s_{tmp} \leftarrow$ `getScatter` $(\mathcal{N}_i)$
    **if** $s_{tmp} < s_{min}$ **then**
      $s_{min} \leftarrow s_{tmp}$
      $\mathcal{L}_{fgr} \leftarrow \mathcal{N}_i$
      $b_{fgr} \leftarrow b_i$
    **end**
  **end**
**end**
**return** $\mathcal{L}_{fgr}, b_{fgr}$

## 4. Extending to 3d targets

Lemma 1 shows the method to generate translational caging region of planar convex targets. It can be extended into 3D case without much modification. Lemma 2 expounds the translational caging region of spatial convex targets.

**Lemma 2.** *In the case of spatial convex caging, the translational caging region of a finger $\mathcal{A}_i$ is determined by its three adjacent neighbours $\mathcal{A}_{i_0}$, $\mathcal{A}_{i_1}$ and $\mathcal{A}_{i_2}$, namely $\mathcal{A}^t_{c(i_0 i_1 i_2)}$ where $\mathcal{A}^t_{c(i_0 i_1 i_2)}$ is generated by intersection of $\mathcal{CC}_{oi_0}(q^{\alpha\beta\gamma})$, $\mathcal{CC}_{oi_1}(q^{\alpha\beta\gamma})$, $\mathcal{CC}_{oi_2}(q^{\alpha\beta\gamma})$ and $\mathcal{A}_{obj}$.*

We have a same supporting proposition to prove Lemma 2. Note that each finger would surely owns three adjacent

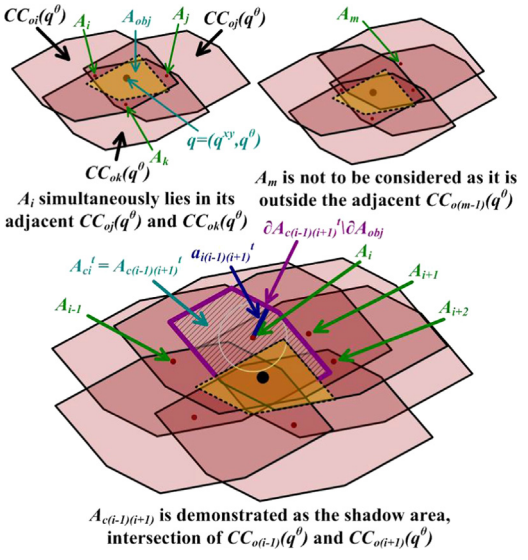


**Fig. 4.** Translational breaking margin formed by adjacent $\mathcal{CC}(q^{\theta})$.

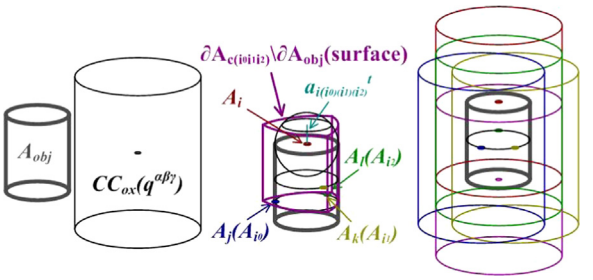


**Fig. 5.** Error-tolerant manipulation planning of a 3D cylinder. From left to right: (1) the original $\mathcal{A}_{obj}$ and a $\mathcal{CC}$ finger with respect to initial $q^{\alpha\beta\gamma}$. (2) The various 3D correspondent concepts. (3) One optimal finger configuration that owns maximum error-tolerance.

neighbours as the minimum finger number to cage a 3D target is four. According to the immobilization theory, we need at least four to six fingers to cage a general spacial target.

**Proposition 2.** *When a spatial object is translationally caged, all $\mathcal{C}$ fingers at a given orientation $q^{\alpha\beta\gamma}$ form a enfolding layer or any necessary finger $\mathcal{A}_i$ simultaneously lies in its adjacent $\mathcal{CC}_{oi_0}(q^{\alpha\beta\gamma})$, $\mathcal{CC}_{oi_1}(q^{\alpha\beta\gamma})$ and $\mathcal{CC}_{oi_2}(q^{\alpha\beta\gamma})$ and $q$ of $\mathcal{A}_{obj}$ lies in $\mathcal{C}_c(q^{\alpha\beta\gamma})$.*

Like planar caging, adjacent neighbours are essential point of spatial caging. The three adjacent neighbours can be obtained by selecting neighbours with the shortest distance in space. This procedure can be performed with a simple path planner or a general searching algorithm such as A*. Here, target objects are taken as obstacles and the convexity of finger formation is assumed for convenience. Since our global optimization is based on surface contacts, the assumption of formation convexity is guaranteed with respect to the requirements of convex spatial shapes.

In summary, we can follow the following steps to obtain the best error-tolerant manipulation contacts of a spatial target. (1) For each finger, find its three adjacent neighbours and calculate $a^t_i$. (2) Accumulate the $\mathcal{D}$ that owns relative large $b^t_{1,2,\ldots,n} > \tau$. (3) Seek the combination with least $s_{1,2,\ldots,n}$ as the optimal results. In order to guarantee robustness, we perform this procedure increasingly with respect to 4, 5 and 6 fingers. The optimization process terminates when fingers and their configurations fulfill specified error thresholds $\tau$. Fig. 5 illustrates this idea with a cylinder object.

Although the idea of best error-tolerance by using immobilization and caging can be extended to 3D spatial environment. We prefer perceiving horizontal 2D surface and employ extra strategies to deal with vertical information. This is because it is hard to design a manipulator that can fulfill the dexterity requirements of 3D caging. The extension here could be more theoretical than practical.[3]

## 5. Concave targets

Researchers in the filed of manipulation tend to deal with concave targets by using two-finger stretching and squeezing. Intuitively, we can directly combine stretching and squeezing into Algorithm 1 to deal with concave

[3] Anyway, we look forward to certain application of 3D cases.

targets. This procedure can be divided into two parts. (1) Perform stretching and squeezing as well as evaluating their error-tolerance. (2) If the tolerance is satisfying, adopt two-finger manipulation, else perform Algorithm 1.

However, the procedure is vulnerable in two points. On one hand, it fails with some concave targets. For instance, if the target owns an "L" shape, it may not be manipulated by two-finger stretching or squeezing. Nevertheless, the boundary finger formation is sometimes concave. Such an object can be processed by the intuitive procedure. On the other hand, the stability of two-finger manipulation is not as reliable as multiple finger cases. It depends too much on friction, inertia matrix and various other physical properties of target objects. Usually, these properties are unknown or hard to obtain through simple perception. Therefore, we prefer an extension approach based on convex error-tolerance planning.

Fig. 6 demonstrates the idea to deal with concave targets. In the first place, the convex hull of target boundary clouds is generated. If the convex hull is no larger than original $\mathcal{A}_{obj}$, or if it fits to a given convexity criteria. The target object is considered as a convex object and dispatched to Algorithm 1 for further planning. If is recognized as a concave target. We dispatch the convex to Algorithm 1 for manipulation planning. The upper middle figure of Fig. 6 shows the result of convexilization. Here the green boundary indicates original target boundary while the red boundary belongs to convex hull. The boundary of this convex target is dispatched to Algorithm 1 for further planning. Finally, we have four finger positions, denoted by green and red in the upper right figure of Fig. 6, generated. The green finger positions in Fig. 6 denote finger positions on original $\partial\mathcal{A}_{obj}$ while the red finger positions denote the additional convex finger contacts. These red contacts are re-searched on their correspondent $\partial\mathcal{A}_{obj}$ region for refinement. The lower left figure in Fig. 6 illustrates this idea. When $\sum_{i=i_0,i_1,\ldots,i_n}\omega_i\hat{n}_i=0$, the finger configuration $\mathcal{L}_{fgr}=\{i_0,i_1,\ \ldots,i_n|\mathrm{argmin}(s_{i_0,i_1,\ldots,i_n})\}$ is selected as the final result of concave manipulation planning.

The strategy to deal with concave targets is an extension to convex error-tolerant manipulation planning. It does not guarantee optimal finger numbers but tries to maintain error-tolerance with multiple fingers. The strategy can inherit merits from convex caging and requires the same finger number as their convex counterparts.

## 6. Experiments and analysis

Three groups of experiments are carried out to validate and demonstrate the ideas introduced in this paper. The first group of experiments aims at testing error-tolerance of the configuration generated by our algorithm. This group of experiments is performed in WEBOTS simulation environment. Dynamics from ODE in WEBOTS offers a powerful way to check the error-tolerant ability of our planner. he second group of experiments is carried out in pure 3D environment. Its aim is to invalidate our algorithm with 3D convex targets. The last group of experiments is based on data perceived from KINECT. Depth map in the view of KINECT is processed for manipulation. This group of experiments aims to test our algorithm against planar convex and planar concave targets and show its promising future in real applications.

### 6.1. Experiments to invalidate tolerance

The first group of experiments aims to invalidate the tolerance of our algorithm. It is performed with WEBOTS by using ODE to simulate the physical environment. Our experiment platform is based on the Neuronics katana manipulator. End-effector of the katana manipulator is substituted by a 0.2 m by 0.2 m board with four cylinder poles to emulate kinematic-free palm and fingers. The left image in Fig. 7 shows an overview of our experiment platform.

Three different targets are introduced to test the tolerance of our proposal. None of these targets are concave as we assume that convex targets are more challenging than concave ones. Our targets include a rectangle, a cylinder and a random polygon. The aim of this experiment focuses on testing the tolerance but emulating the results of various targets. More planning results are shown in the third experiment group. Lower part of Fig. 7 presents the planning results of our algorithm. The white segment in these images indicate the shortest path to "break" the caging. Minimum value of these white segments from the surrounding $\mathcal{CC}_{o(x)}$ s equals the best tolerance. *We can manipulate a target as following. In the first step, an initial finger configuration is chosen by considering best tolerance. Then, fingers converge along the white segments until contacts between them and the target surface are formed.* Fig. 8 demonstrates the initial configuration towards the third target. In this figure, $\frac{1}{8}$ of the best tolerance is chosen to set the initial finger configuration. The initial configuration is indeed a robust caging with least finger number.

With the initial configuration, the robot can contract its fingers to form the final immobilization manipulation. Fig. 9[4] demonstrates this procedure. Note that actual motion may depend on robot properties. For instance, it may depend on whether the robot is an one-parameter or multi-parameter agent.

Since $b^t_{ijk}$ is used to measure the tolerance, it is a larger evaluation compared with real value where both translation and rotation play important roles. The initial configuration should only own an offset of $\omega b^t_{ijk}, \omega\in[0,1)$, or a small portion of $b^t_{ijk}$. Choosing a proper $\omega$ depends on target shapes and the precision of perception devices. In the best case where $\omega=0$, the initial configuration is the final immobilization. Table 2 shows the performance of different $\omega$ in this experiment group. Geometric parameters of these target objects are as following. Rectangle, [(0.05, 0.05), (0.05, −0.05), (−0.05, −0.05), (−0.05, 0.05), (0.05, 0.05)], *height*=0.05 m. Cylinder, *radius*=0.05 m, *height*=0.05 m. Polygon, [(0.05, 0.05), (0.06, −0.025), (0, −0.05), (−0.05, −0.025), (0.025, 0.05), (0.05, 0.05)], *height*=0.05 m. Empirically, it is safe to have the initial $\omega$=0.1 0.2 for daily utilities.

[4] For concave targets, we suggest initializing and contracting along surface normals.

### 6.2. Experiments with virtual 3D models

The second group of experiments aims to invalidate the algorithm with spatial shapes. As described previously, although we expect applications of 3D error-tolerant algorithm, it is indeed a theoretical work. It is hard to design a manipulator that can fulfill the kinematic requirements of 3D error-tolerant manipulation. The experiment in this subsection is performed with a shape composed with two cones on the same bottom. Strictly speaking, they are not cones but pyramids. Exact shape depends on resolution of voxels in Constructive Solid Geometry (CSG). In this experiment, our resolutions are set to 42 surface voxels and 114 surface voxels so that we have the 3D shape demonstrated in Fig. 10.

Upper-left and lower-left images in Fig. 10 show the original $\mathcal{A}_{obj}$. They own slight difference due to voxel resolution. The middle column of Fig. 10 demonstrates the $\mathcal{CC}_{o(x)}$ s that correspond to objects in the left. Here the inner red models illustrate $\mathcal{C}_{o(x)}$ while the outer cyan models illustrate $\mathcal{CC}_{o(x)}$. As the algorithm searches maximum $b^t_{1,2,\ldots,n}$ with minimum $s_{1,2,\ldots,n}$, the adjacent three neighbours $\mathcal{CC}_{o(i_0)}$, $\mathcal{CC}_{o(i_1)}$ and $\mathcal{CC}_{o(i_2)}$ intersect with each other for tolerance evaluation. The bounding cyan blocks in the third column of Fig. 10 demonstrate these intersections. Finally, error-tolerant finger configurations are shown with CMYK color spots in the third column with their normal directions rendered in red segments.

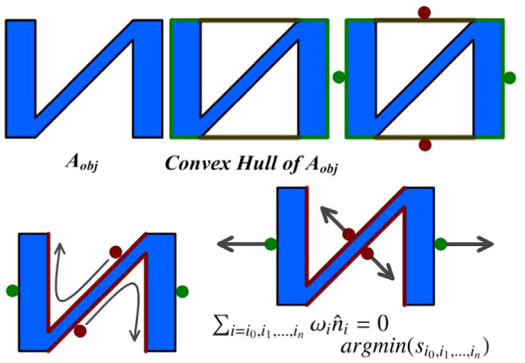


**Fig. 6.** The procedure to deal with concave targets. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

Finger configuration of the cone object conforms our expectation and four fingers are necessary to manipulate the target. In extreme case when target becomes a regular octahedron, we can expect four finger contacts with orthogonal normals to obtain maximum tolerance. It is the same with two general cones on the same bottom. We have four contacts with orthogonal normals as the error-tolerant manipulation configuration as shown in Fig. 10.

### 6.3. Experiments with KINECT

The last group of experiments with KINECT data is performed to invalidate the computational efficiency and the ability of our planner to deal with both planar convex and concave targets. KINECT data in this experiment is obtained by manual operation. Note that we do not fix KINECT to a certain position but operate it by hand. The manual operation may introduce more noises than fixture operation. Despite the noisy target clouds, our planner can generate manipulation position and translational tolerance effectively.

Fig. 11 shows the results with various daily objects. In left-right and top-bottom order, they are (1) biscuit package, (2) wallet, (3) mouse, (4) folded towel, (5) coffee can, (6) plate, (7) tape, (8) tape box, (9) tape measure, (10) paper box, (11) bent towel and (12) kettlebell. The items (2), (3), (4), (7), (8), (9) and (11) are overlapped with some other objects, they are named by the top object in view. To conquer the overlapping problem, we pick up the peak of depth histogram and utilize region growing for top target. It is assumed that all overlapped objects are dealt step by step from top to bottom and in each step, the top object is extracted for manipulation. Various strategies can be employed to grow regions from the starting peak. In this paper, we simultaneously record the changes of pixel depth and surface normal directions to extract target region. The second and fifth rows of Fig. 11 demonstrate the extracted region boundaries of top targets.

The third and sixth rows of Fig. 11 show the error-tolerant manipulation positions of each target boundary. The green boundaries in these two rows correspond to boundaries in second and fifth rows, namely $\partial\mathcal{A}_{obj}$. The cyan, magenta, yellow and black boundaries corresponds to the $\partial\mathcal{CC}_{o(x)}$ of each finger $x$. The red segments are introduced to render surface normals along boundary $\partial\mathcal{A}_{obj}$. These normals are generated by using $K$-curvature

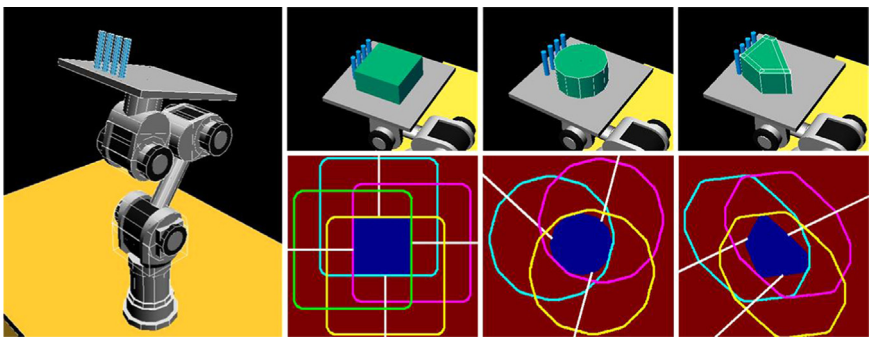

**Fig. 7.** Experiment platform, target models and planning results of the first experiment group.

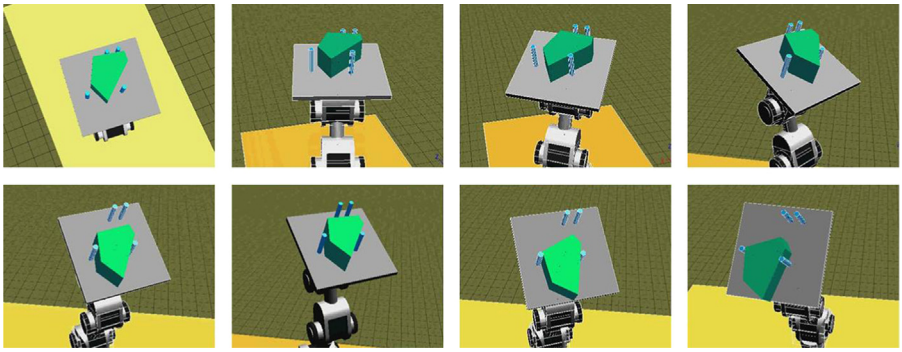

**Fig. 8.** Initial finger configuration is actually a robust caging with least finger number.

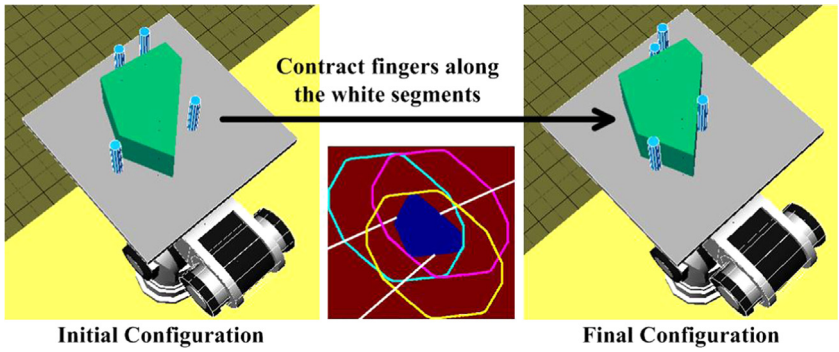


**Fig. 9.** Contract initial finger configuration to immobilize target.

where two parameters $k_1$ and $k_2$ with values tuned towards boundary length are involved to conquer noises. The twelve scenes are processed by our algorithm without any prior information. Table 2 shows specific numerical results of the process. Our algorithm can decide finger numbers according to requirements and discern concave targets from convex targets automatically. Note that in this process, $\tau$ is set to the larger value of $0.25 \times globalmax$ and $20 \times step$ where *globalmax* means the maximum $b^t_{1,2,\ldots,n}$ while *step* denotes the sampling intervals of boundary clouds.

The columns "Bdry" and "Fgrs" represent the boundary cloud points and the necessary finger number to perform a robust manipulation. *step* is set to 1/48 of total boundary length, meaning that we rasterize the boundary clouds into 48 pivots for planning. The column "cost" shows time efficiency of each planning. Due to surface normal errors caused by noises, the planner may first find a 3-finger immobilizing configuration with certain "tolerance" and then reject it according to $\tau$. In that case, the planner shall spend resources on 3-finger planning before achieving a more tolerant result. Note that the super script $t$ of "Tolerance" means translational robustness. Magnitude of this value is shown in square of distance pixels. The "+" symbol in "cost" column denotes individual cost. For instance, the final 4-finger planning of item (2) costs 6.84 s and we need extra 5.04 s on 3-finger planning before 4-finger calculation. The "→" in "Tolerance" column shows the changes in error-tolerance of 3-finger and 4-finger results. Since 3-finger result is not satisfying, the algorithm requires more fingers for robustness.

Items (11) and (12) are concave targets, they are discerned from the other objects in a convexilization step. This step employs Douglas–Peucker polyline simplification to ignore minor concavity. The "Bdry" column in Table 1 shows their convexilized boundary point number and original boundary point number without and with parenthesis. The parenthesized value in column "Cost" shows the cost of re-searching along concave boundary regions. In Fig. 11, the results of convexilized object are shown in CMYK color while final finger configurations are shown in blue spots. Since fingers on original $\partial A_{obj}$ are not re-searched, they convert into final blue spots directly. Hence only the blue finger configuration and un-researched spots can be seen in Fig. 11.

## 7. Conclusions and future works

In this paper, we presented an optimal manipulation approach which obtains ultimate robustness towards perception noises and errors. Based on $\mathcal{C}$ space and $\mathcal{CC}$ space, we propose an approach to locate the best translational caging that owns largest translational margins and introduce least inter-finger distance of $\mathcal{W}$ space to make up rotational constraints. Based on an improved GA, we propose an efficient way to locate pragmatic optimal caging positions by evaluating both translational caging and rotational inter-finger distance. Experimental results on various virtual objects demonstrate the robustness and efficiency of our proposal.

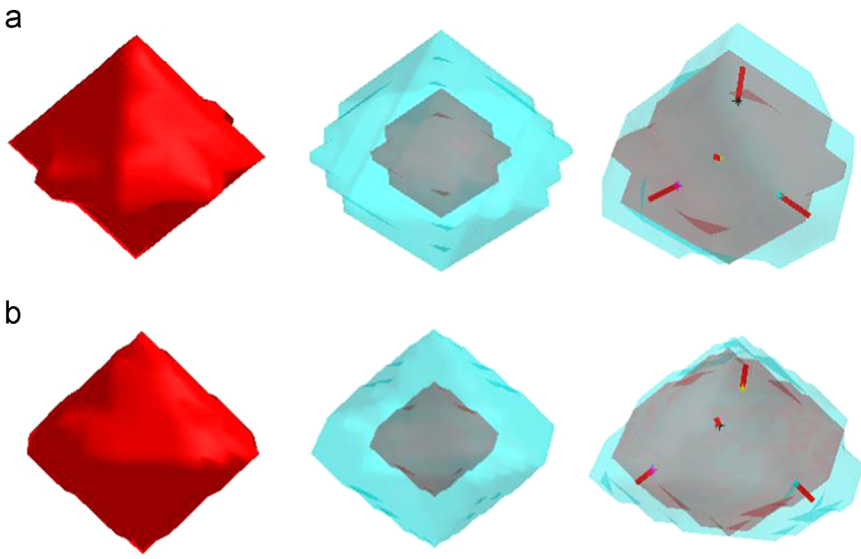


**Fig. 10.** The results of 3D manipulation planning. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

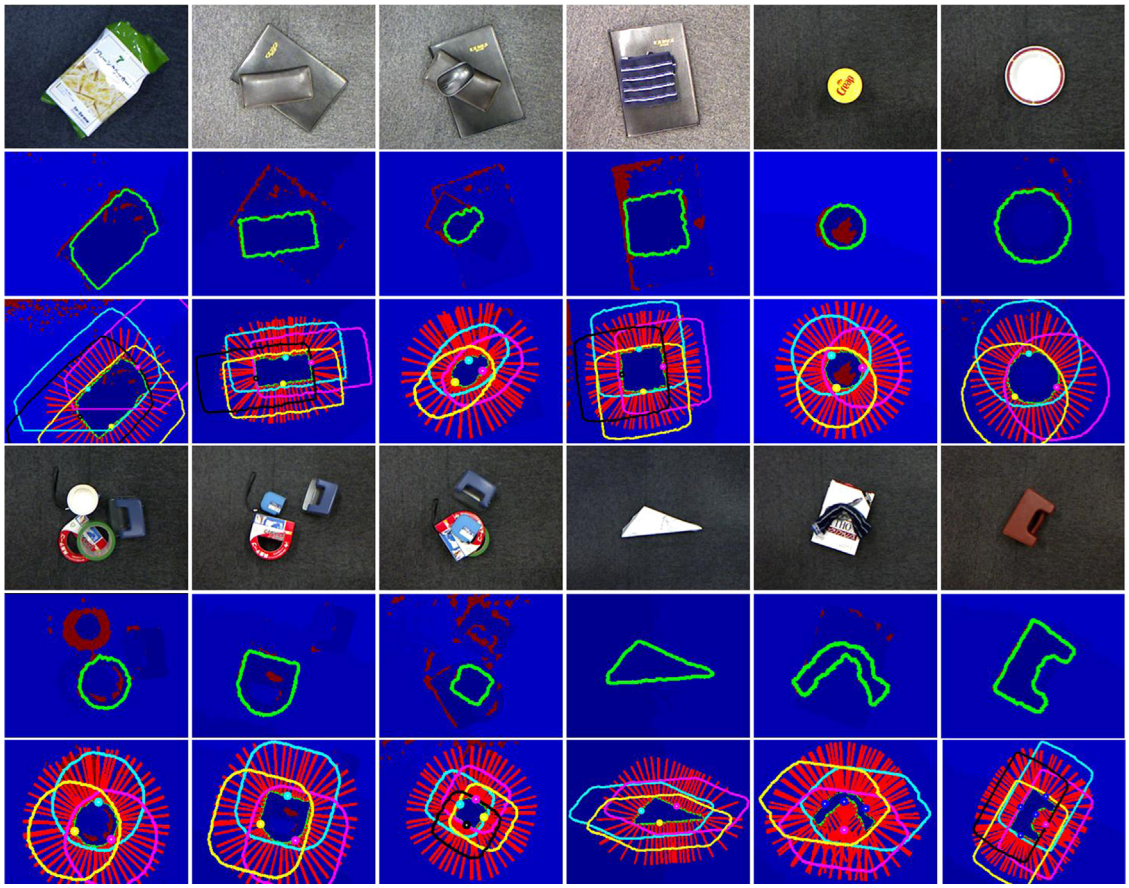

**Fig. 11.** The results with various daily objects with KINECT from top view. Refer to the contents in Section 4 for details. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

**Table 1**
Experimental results with the targets listed in Fig. 11.

| Itm | Bdry | Type | Fgrs | Cost | Tolerance$^t$ | $\lvert\mathcal{D}\rvert$ |
|---|---|---|---|---|---|---|
| 1 | 1107 | Convex | 4 | $10.81+6.00s$ | $441\rightarrow2025$ | 1/7 |
| 2 | 689 | Convex | 4 | $6.84+5.04s$ | $196\rightarrow1170$ | 7/34 |
| 3 | 373 | Convex | 3 | $3.5s$ | 433 | 2/6 |
| 4 | 645 | Convex | 4 | $5.63+6.86s$ | $232\rightarrow3364$ | 1/1 |
| 5 | 347 | Convex | 3 | $4.93s$ | 145 | 3/24 |
| 6 | 599 | Convex | 3 | $5.34s$ | 484 | 2/17 |
| 7 | 403 | Convex | 3 | $5.04s$ | 225 | 3/18 |
| 8 | 465 | Convex | 3 | $3.83s$ | 353 | 4/25 |
| 9 | 285 | Convex | 4 | $4.54+3.63s$ | $80\rightarrow533$ | 1/8 |
| 10 | 541 | Convex | 3 | $5.14s$ | 1105 | 5/93 |
| 11 | 593(459) | Concave | 3 | $2.18(0.05)s$ | 970 | 3/20 |
| 12 | 595(482) | Concave | 4 | $7.42+2.76(0.04)s$ | 1424 | 5/12 |

**Table 2**
Experimental results with different $\omega$.

| *Targets* | $b^t_{ijk}$ | $\omega=\frac{1}{2}$ | $\omega=\frac{1}{4}$ | $\omega=\frac{1}{6}$ | $\omega=\frac{1}{8}$ | $\omega=\frac{1}{10}$ |
|---|---|---|---|---|---|---|
| Rectangle | 0.0155 | Fail | Pass | Pass | Pass | Pass |
| Cylinder | 0.0530 | Fail | Pass | Pass | Pass | Pass |
| Polygon | 0.0325 | Fail | Fail | Fail | Pass | Pass |

In the future, we would like to carry out the optimal caging approach with real sensor data and apply it on our self-designed caging manipulator.

## References


[1] R.C. Brost, K.Y. Goldenberg, A complete algorithm for synthesizing modular fixtures for polygonal parts, in: Proceedings of the IEEE International Conference on Robotics and Automation, 1994, pp. 535–542.

[2] P. Song, V. Kumar, A potential field based approach to multi-robot manipulation, in: Proceedings of the IEEE International Conference on Robotics and Automation, 2002, pp. 1217–1222.

[3] V.-D. Nguyen, Constructing force-closure grasps, Int. J. Robot. Res. 7 (1988) 3–16.

[4] G.A.S. Pereira, V. Kumar, M.F.M. Campos, Decentralized algorithms for multirobot manipulation via caging, Int. J. Robot. Res. 23 (2004) 783–795.

[5] T. Watanabe, T. Yoshikawa, Grasping optimization using a required external force set, IEEE Trans. Autom. Sci. Eng. 4 (2007) 52–66.

[6] Z. Li, S.S. Sastry, Task-oriented optimal grasping by multifingered robot hands, IEEE J. Robot. Autom. 4 (1988) 32–44.

[7] E. Chinellato, R.B. Fisher, A. Morales, A.P. del Pobil, Ranking planar grasp configurations for a three-finger hand, in: Proceedings of the IEEE International Conference on Robotics and Automation, 2003, pp. 1133–1138.

[8] Y.-H. Liu, M.-L. Lam, D. Ding, A complete and efficient algorithm for searching 3-d form-closure grasps in the discrete domain, IEEE Trans. Robot. 20 (2004) 805–816.

[9] E. Rimon, A. Blake, Caging planar bodies by one-parameter two-fingered gripping systems, Int. J. Robot. Res. 18 (1999) 299–318.

[10] L. Zhang, M. Song, X. Liu, J. Bu, C. Chen, Fast multi-view segment graph kernel for object classification, Signal Process. 93 (2013) 1597–1607.

[11] L. Zhang, Y. Yang, Y. Gao, Y. Yu, C. Wang, X. Li, A probabilistic associative model for segmenting weakly-supervised images, IEEE Trans. Image Process. (2014) 4150–4159.

[12] B. Zheng, J. Takamatsu, K. Ikeuchi, An adaptive and stable method for fitting implicit polynomial curves and surfaces, IEEE Trans. Pattern Anal. Mach. Intell. 32 (2010) 561–568.

[13] N.J. Redding, Implicit polynomials, orthogonal distance regression, and the closest point on a curve, IEEE Trans. Pattern Anal. Mach. Intell. 22 (2000) 191–199.

[14] M. Richtsfeld, M. Vincze, Robotic grasping of unknown object, in: Contemporary Robotics–Challenges and Solutions, InTech, 2009, pp. 19–32.

[15] D. Berenson, S. Srinivasa, J. Kuffner, Task space regions: a framework for pose-constrained manipulation planning, Int. J. Robot. Res.

[16] L. Zhang, M. Song, X. Liu, L. Sun, C. Chen, J. Bu, Recognizing architecture styles by hierarchical sparse coding of blocklets, Inf. Sci. 254 (2014) 141–154.

[17] L. Zhang, Y. Gao, C. Hong, Y. Feng, J. Zhu, D. Cai, Motion planning for disc-shaped robots pushing a polygonal object in the plane, IEEE Trans. Cybern. 44 (2014) 1408–1419.

[18] M.T. Mason, Mechanics of Robotic Manipulation, The MIT Press, Cambridge, MA, USA, 2001.

[19] J. Czyzowicz, I. Stojmenovic, J. Urrutia, Immobilizing a shape, Int. J. Comput. Geom. Appl. 9 (1999) 181–206.

[20] E. Rimon, J.W. Burdick, Mobility of bodies in contact. I. A new 2nd order mobility index for multiple-finger grasps, IEEE Trans. Robot. Autom. 14 (1998) 696–708.

[21] S. Makita, Y. Maeda, 3d multifingered caging: Basic formulation and planning, in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008, pp. 2697–2702.

[22] M. Vahedi, F. van der Stappen, On the complexity of the set of three-finger caging grasps of convex polygons, in: Proceedings of the Robotics, Science and Systems, 2009.

[23] Z. Wang, Y. Hirata, K. Kosuge, An algorithm for testing object caging condition by multiple mobile robots, in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, 2005, pp. 3022–3027.

[24] J. Erickson, S. Thite, F. Rothganger, J.P. Fellow, Capturing a convex object with three discs, IEEE Trans. Robot. 23 (2007) 1133–1140.

[25] M. Vahedi, A.F. van der Stappen, Caging convex polygons with three fingers, in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008, pp. 1777–1783.