# Efficient Collision Detection and Detach Control for Convex Prisms in Precision Manipulation

Dengpeng Xing, Fangfang Liu, Song Liu, and De Xu, *Senior Member, IEEE*

***Abstract*—This paper proposes an efficient and accurate method for collision detection between convex prisms in precision alignment and a detach controller based on the contact location for the collision occurrence. We project the objects onto an appropriate plane and detect the collision status considering the relationship between their planar contours. Efficient methods are presented for the overlap checking of several elementary contours, and the way to obtain the vertices of the projected shapes is also introduced. The detection is then accelerated by classifying into the corresponding case based on the relative configuration and using the efficient planar checking to replace the cubic calculation. A detach controller is presented to immediately separate the objects according to the contact location once collision occurs. The computational efficiency and comparison are demonstrated in simulations, and experiments are carried out to validate the detection and the controller.**

***Index Terms*—Collision detection, convex prism, detach controller.**

## I. Introduction

As COMMONLY used in physically based simulation, robotics, etc., collision detection determines whether the two or more shapes collide and returns a contact manifold if collision occurs. With different requirements in various applications, collision detection methods are still one of the research focuses in recent years [1], and they need to be fast, accurate, and robust in complex physics engines. These methods basically apply in virtual environment and experiment.

In virtual simulation, the object can be rigid or deformable. A progressive refinement framework for collision detection is provided in [2], and the package SWIFT is released to accelerate the status query between moving convex rigid polyhedra. To handle deformable models, an interactive algorithm is presented to detect intersections between severely deforming polygonal objects using a set of axis aligned bounding boxes [3]. The object's shape can also be nonconvex, and in [4], a sampling algorithm is proposed to utilize the offline precomputation and to perform the nearest-neighbor query of the penetration depth between nonconvex models. The collision status can also be detected along a trajectory, given two model configurations, and in [5], continuous collision detection queries are performed between triangulated models using Bernstein basis and Bézier curves to evaluate signs of polynomials. Most works use triangles or higher order primitives to represent the complex geometric models and some bounding volume hierarchies are also employed in order to increase the query efficiency. These raise a tradeoff between efficiency and accuracy, and the performance is dependent on the model complexity, materials, and shapes.

As an essential part for trajectory or path planning and a good way of dealing with disturbance, collision detection, and avoidance is often used for keeping marginal distance between objects. One of its core problems is the planning of the collision-free paths/trajectories in dynamic environments, in which the object employs a continuous sensing-control cycle to navigate in an environment with obstacles or other entities [6], [7]. In this domain, convex objects are usually concerned or convex pieces are segmented from complex models. Many existing methods employ potential fields to guide an object toward its goal, with a repulsive potential to ensure collision avoidance [8]. To address avoidance in the dynamic environment, a safety cylindrical region is established around the center, and the avoidance algorithm activates only when this region is invaded [9]. By assuming the velocities of the obstacles can be calculated online, some approaches control the object depending on the relative speed of each obstacle [10]. Collision avoidance has been applied in many areas, such as formation [11], haptic guidance [12], autonomous vehicles [13], etc. To online predict the collision, a geometry-based method [14] is presented in path planning among adversarial obstacles. To handle noisy sensor or partial observations, the prior collision queries are stored and the collision probability of a new sample is predicted, using approximate $k$-nearest neighbor to find the closest configuration [15]. In [16], collision avoidance is considered in path planning for cooperative dual-crane lifting in complex 3-D environments.

Precision manipulation or assembly aims at achieving high precision in object positioning and alignment, with microscopes. It raises worldwide attention because of its capability of manipulating small objects in high accuracy, and substantial works have been done in recent years [17]. For efficient operation in precision assembly, an insertion strategy based on

Manuscript received October 7, 2017; revised January 31, 2018; accepted March 13, 2018. Date of publication March 15, 2018; date of current version December 3, 2018. This work was supported by the Program for the National Nature Science Foundation of China under Grant 61673382, Grant 51405486, Grant 61421004, and Grant 61733004. Paper no. TII-17-2366. *(Corresponding author: Dengpeng Xing.)*

D. Xing, F. Liu, and D. Xu are with the Institute of Automation, Chinese Academy of Sciences and University of Chinese Academy of Sciences, Beijing 101408, China (e-mail: dengpeng.xing@ia.ac.cn; fangfang.liu@ia.ac.cn; de.xu@ia.ac.cn).

S. Liu is with the City University of Hong Kong, Hong Kong (e-mail: sdliusong@outlook.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TII.2018.2816260

probabilistic model is used in [18], which greatly decreases the insertion steps. Microscope helps to obtain high-precision visual feedback, while it also has the property of small view depth and small view field, which requires at least two microscopes to obtain 3-D observation [19]. Such a problem may cause the mutual block of objects in the view of cameras, and collision detection becomes a necessity to manipulate small objects in such a narrow view field. To improve the detection efficiency, projection methods are proposed in [20] for collision detection between two cylindrical objects and in [21] for efficient distance computation. This algorithm provides quick calculation but only applies for cylindrical shapes, limiting its usage to complex shapes.

In precision manipulation, objects move in the micro operational space, whose size is limited by the small view depth and view field of microscopes. Collision is possible due to disturbances, measurement errors, or misoperation, and collision detection is a necessary topic to safely navigate in precision manipulation. Furthermore, since the objects in precision manipulation are small sized and easy broken, the accuracy and efficiency of collision detection are both required, as well as the immediate reaction once collision appears for object protection. Based on these requirements, this paper intends to efficiently detect collision without losing accuracy and detach objects based on the contact location. Convex prism is a common type of objects to be operated, e.g., even cylinder can be seen as an $n$-gonal prism where $n$ approaches infinity. This paper focuses on the collision detection of convex prisms by projecting onto an appropriate plane and determining the status based on the projected contours. This dimensionality reduction greatly improves computational efficiency despite leading to nine cases according to the objects' configurations. We propose efficient methods to consider the intersection between a polygon and five elementary contours, and employ a policy of detecting while projecting to use these planar detection for each case to accelerate the cubic calculation. For collision settings, a detach controller is introduced based on the contact location for proper reaction. We have implemented simulations and experiments for verification and comparisons.

The main contribution of this paper is the efficient and accurate algorithm for collision detection of convex prisms in precision manipulation. We project the objects onto a plane and use lower dimensional contours to detect the higher dimensional collision. Compared with other methods, it can greatly improve the detection efficiency: 4% average computation cost of the SWIFT method. The difference from [20] lies in the object shape: the cylinder has a circular-end surface and this regularity simplifies the detection; while the prism has an irregular end surface: random number of vertices and random distance and angle of each vertex to its center subjected to the convex constraint. This property renders different ways of projection, contour analysis, and shape overlap. Moreover, the method in [20] can be viewed as a special case of this paper, where the prism is an infinity-gonal shape with an equal distance between vertex and its center. The drawback of the proposed method is the implementation complexity in some cases where contour violation occurs.

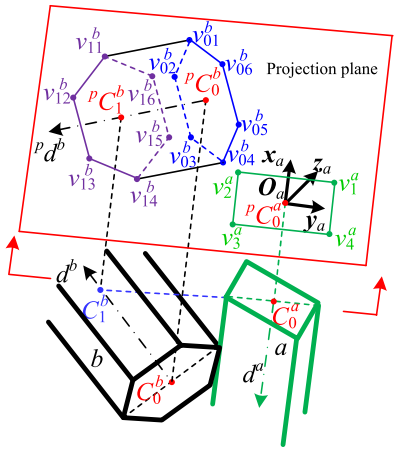


Fig. 1. Object projection.

## II. Description of the Proposed Algorithm

The characteristics of precision manipulation include the following.

1) The objects are small-sized, in micron to millimeter dimension, and commonly thin-walled, within dozens of microns.
2) The material is highly related to the application and the objects can be brittle, deformable, or even with surface coating.
3) Microscopes are used for state observation, whose properties are small view depth and view field.
4) For even smaller objects, e.g., micro-objects, the molecular forces, such as adhesive force, van der waals force, etc., are necessary concerns in dealing with the object interaction [22].

These characteristics determine the peculiarity of collision detection in precision manipulation: the requirement of accuracy and efficiency. Those small objects are manipulated in the micro operational space, which is very small since all microscopes need to clearly view the objects simultaneously. This results in demanding of careful navigation as the objects have to stay closely and they even block each other in all views. Due to disturbances, measurement errors, or misoperation, collision is not neglectable. Once this unexpected contact happens, fast detection and detachment is the last effort for object protection.

Consider two convex prisms $a$ and $b$, as shown in Fig. 1, which are held by the lower and upper grippers. The surface of object $a$ consists of $n^a$ rectangles and an $n^a$-sided polygonal base and the object $b$ has the same shape replacing $n^a$ with $n^b$. The vertex sets are labeled as $\boldsymbol{v}^a$ and $\boldsymbol{v}^b$ and the sets of the edges between vertices are labeled as $\boldsymbol{e}^a$ and $\boldsymbol{e}^b$. The axis directions of the objects are labeled as $\boldsymbol{d}^a$ and $\boldsymbol{d}^b$.

To improve the detection efficiency, we present a projection method to detect 3-D collision with planar contours and the idea derivation of this approach is listed as below. In any microscopic view, if the objects' contours have no overlap, noncollision is ensured and otherwise we need to further check given relative position and posture. In common, for noncollision objects, there always exists a view plane on which the objects' projections

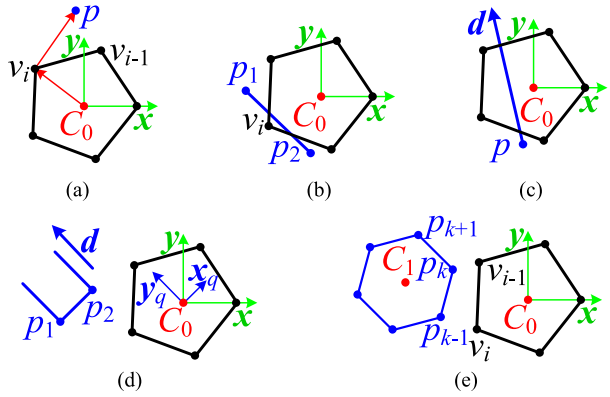


Fig. 2. Typical examples between a polygon and a point (a), or a line segment (b), or a half-line (c), or a very long rectangle (d), or a polygon (e).

do no overlap, but it is inefficient to find this view plane. For generality, we project the objects onto the plane that includes the end surface of one object $a$, for example, and the projected shape of the object $a$ is an $n^a$-sided polygonal. Viewing from the direction $\boldsymbol{d}^a$, we do not project the object $b$s part that is below the projection plane. According to the two objects' relative position and posture, the projection of the object $b$ can be null, a point, a line segment, a half-line, a long rectangle, and a polygon. Therefore, the 3-D detection changes to the overlap problem between two planar shapes. Dimensionality reduction leads to multiple cases in which appropriate planar overlap methods are used. Furthermore, we employ a projecting-while-detecting policy to further speed up calculation time. When collision is detected, the detach controller is activated based on the contact location to separate the objects along an appropriate direction.

## III. Collision Detection of Elementary Contours

Fig. 2 shows five contour types of projecting a convex prism onto a plane, including a point, a half-line, a line segment, a rectangle with unlimited edges in one direction, and a polygon. The following considers whether overlap occurs between the projected shapes.

Fig. 2(a) shows the position relationship between a point and a polygon. Use polar coordinates to represent the position of the point $\boldsymbol{p}$ and obtain its polar angle $\theta_p$. Search in the polygon and find the neighbor vertices $\boldsymbol{v}_i$ and $\boldsymbol{v}_{i-1}$ with polar angle $\theta_{i-1}$ and $\theta_i$, so that $\theta_p \in [\theta_{i-1}, \theta_i)$. The parameter to determine whether the point $\boldsymbol{p}$ lies inside of the polygon yields

$$\Delta_p(\boldsymbol{p}, \boldsymbol{v}) = [\boldsymbol{v}_i \times (\boldsymbol{v}_{i-1} - \boldsymbol{v}_i)] \cdot [(\boldsymbol{v}_i - \boldsymbol{p}) \times (\boldsymbol{v}_{i-1} - \boldsymbol{v}_i)] \tag{1}$$

where $\Delta_p(\boldsymbol{p}, \boldsymbol{v})$ demonstrates whether the point $\boldsymbol{p}$ locates inside the polygon determined by the vertex set $\boldsymbol{v}$ or not, "$\times$" is the cross product, and "$\cdot$" is the dot product. $\Delta_p < 0$ means the point is outside the polygon. If collide, the contact point locates

$$\boldsymbol{s}_p(\boldsymbol{p}) = \boldsymbol{p} \tag{2}$$

where $\boldsymbol{s}_p(\boldsymbol{p})$ is the vector pointing from the convex center to the contact point.

As to the position relationship between a line segment and a polygon, there are two instances in which they collide. One is that at least one of the two endpoints $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ is not outside the polygon, which can be distinguished using (1), and the other is that the line segment has intersections with the polygon, as shown in Fig. 2(b). Label the points' polar angles as $\theta_{p_1}$ and $\theta_{p_2}$ and define the set $\boldsymbol{S}_1$ in which the polygonal vertices satisfy $\theta_i \in [\theta_{p_1}, \theta_{p_2}]$. The intersection between the line segment and the polygon is determined by

$$\zeta_1 = \min_{\boldsymbol{v}_i \in \boldsymbol{S}_1} [\boldsymbol{p}_1 \times (\boldsymbol{p}_2 - \boldsymbol{p}_1)] \cdot [(\boldsymbol{p}_1 - \boldsymbol{v}_i) \times (\boldsymbol{p}_2 - \boldsymbol{p}_1)] \tag{3}$$

where $\zeta_1 \le 0$ means the line segment passes through the polygon. Adding the two instances together leads to

$$\Delta_l(\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{v}) = \begin{cases} -1, & \Delta_p(\boldsymbol{p}_1, \boldsymbol{v}) \ge 0 \cup \Delta_p(\boldsymbol{p}_2, \boldsymbol{v}) \ge 0 \\ -1, & \zeta_1 \le 0 \\ 1, & \text{otherwise} \end{cases} \tag{4}$$

where $\Delta_l(\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{v})$ is the parameter detecting the collision between a line segment and a polygon. $\Delta_l > 0$ means no intersection between the line segment and the polygon. If collide, the contact point locates

$$\boldsymbol{s}_l(\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{v}) = \begin{cases} \boldsymbol{s}_p(\boldsymbol{p}_1), & \Delta_p(\boldsymbol{p}_1, \boldsymbol{v}) \ge 0 \\ \boldsymbol{s}_p(\boldsymbol{p}_2), & \Delta_p(\boldsymbol{p}_2, \boldsymbol{v}) \ge 0 \\ \boldsymbol{p}_c, & \text{otherwise} \end{cases} \tag{5}$$

where $\boldsymbol{s}_l(\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{v})$ represents the contact point location and $\boldsymbol{p}_c$ is the cross point between the line $\overline{\boldsymbol{p}_1\boldsymbol{p}_2}$ and $\overline{C_0\boldsymbol{v}_i}$.

It is similar to investigate the intersection between a half-line and a polygon, as shown in Fig. 2(c). Label $\theta_p$ as the point's polar angle and $\theta_d$ to represent the direction $\boldsymbol{d}$. Define the set $\boldsymbol{S}_2$ as the vertex set in which $\theta_i \in [\theta_d, \theta_p]$ holds. The intersection between these two shapes is determined by

$$\zeta_2 = \min_{\boldsymbol{v}_i \in \boldsymbol{S}_2} (\boldsymbol{p} \times \boldsymbol{d}) \cdot [(\boldsymbol{p} - \boldsymbol{v}_i) \times \boldsymbol{d}] \tag{6}$$

where $\zeta_2 \le 0$ means there exists an intersection between the half-line and the polygon. The parameter indicating the collision status between them yields

$$\Delta_h(\boldsymbol{p}, \boldsymbol{d}, \boldsymbol{v}) = \begin{cases} -1, & \Delta_p(\boldsymbol{p}, \boldsymbol{v}) \ge 0 \\ -1, & \zeta_2 \le 0 \\ 1, & \text{otherwise} \end{cases} \tag{7}$$

where $\Delta_h(\boldsymbol{p}, \boldsymbol{d}, \boldsymbol{v}) < 0$ represents that the half-line starting from the point $\boldsymbol{p}$ and directing in $\boldsymbol{d}$ intersects with the polygon $\boldsymbol{v}$. When collision occurs, the contact location yields

$$\boldsymbol{s}_h(\boldsymbol{p}, \boldsymbol{v}) = \begin{cases} \boldsymbol{s}_p(\boldsymbol{p}), & \Delta_p(\boldsymbol{p}, \boldsymbol{v}) \ge 0 \\ \boldsymbol{p}_c & \text{otherwise} \end{cases} \tag{8}$$

where $\boldsymbol{s}_h(\boldsymbol{p}, \boldsymbol{v})$ is the contact point location and $\boldsymbol{p}_c$ is the cross point between the line $\overline{C_0\boldsymbol{v}_i}$ and the half-line.

Fig. 2(d) considers the relationship between a polygon and a rectangle whose edges are of unlimited length along the direction $\boldsymbol{d}$. No intersection can be asserted between the two shapes if the line segment $\overline{p_1p_2}$ lies outside the polygon and the edges

do not cross the polygon. To check the latter, transform the two shapes into the coordinates of $\boldsymbol{O}_q - \boldsymbol{x}_q\boldsymbol{y}_q\boldsymbol{z}_q$, where the origin sets at $\boldsymbol{C}_0$, $\boldsymbol{y}_q$ is parallel with $\boldsymbol{d}$, $\boldsymbol{z}_q$ is parallel with $\boldsymbol{z}$, and the rotational matrix is ${}^q\boldsymbol{R} = [\boldsymbol{d} \times \boldsymbol{z}, \boldsymbol{d}, \boldsymbol{z}]^{-1}$. Describing each point and vertex in the new coordinates yields

$$ {}^q\boldsymbol{v}_i = {}^q\boldsymbol{R}\boldsymbol{v}_i, {}^q\boldsymbol{p}_j = {}^q\boldsymbol{R}\boldsymbol{p}_j \tag{9} $$

where ${}^q\boldsymbol{v}_i$ and ${}^q\boldsymbol{p}_j$ are the expression of the polygon vertices and the point in $\boldsymbol{O}_q$. The polar angle of each vertex is recomputed as ${}^q\theta_i$, and the sets

$$ \begin{cases} \boldsymbol{S}_3 = \{{}^q\boldsymbol{v}_i | {}^q\theta_i \in ({}^q\theta_{p_1}, \frac{\pi}{2})\} \\ \boldsymbol{S}_4 = \{{}^q\boldsymbol{v}_i | {}^q\theta_i \in (\frac{\pi}{2}, {}^q\theta_{p_2})\} \\ \boldsymbol{S}_5 = \{{}^q\boldsymbol{v}_i | {}^q\theta_i \in ({}^q\theta_{p_2}, {}^q\theta_{p_1})\} \end{cases} \tag{10} $$

where $\boldsymbol{S}_3$ is the set whose polar angle is within ${}^q\theta_{p_1}$ and $\frac{\pi}{2}$ if ${}^q\boldsymbol{p}_1$ locates on the right $\boldsymbol{x}_q$ axis, $\boldsymbol{S}_4$ is the vertex set in which the polar angle is within ${}^q\theta_{p_2}$ and $\frac{\pi}{2}$ if ${}^q\boldsymbol{p}_2$ locates on the left $\boldsymbol{x}_q$ axis, and $\boldsymbol{S}_5$ is the set including the vertices that are within the sector from ${}^q\theta_{p_2}$ to ${}^q\theta_{p_1}$. Use the following parameter

$$ \eta({}^q\boldsymbol{v}_i, {}^q\boldsymbol{p}_j) = \left(-{}^q\boldsymbol{p}_j \times \boldsymbol{y}_q\right) \cdot \left[\left({}^q\boldsymbol{v}_i - {}^q\boldsymbol{p}_j\right) \times \boldsymbol{y}_q\right] \tag{11} $$

where $\eta({}^q\boldsymbol{v}_i, {}^q\boldsymbol{p}_j)$ determines if the vertex ${}^q\boldsymbol{v}_i$ and the geometry center ${}^q\boldsymbol{C}_0$ locate on the same side of the vertical line passing the point ${}^q\boldsymbol{p}_j$. Considering the above-mentioned two equations leads to

$$ \zeta_3 = \begin{cases} \min_{{}^q\boldsymbol{v}_i \in \boldsymbol{S}_3} \eta({}^q\boldsymbol{v}_i, {}^q\boldsymbol{p}_1), & {}^q\boldsymbol{p}_1 \cdot \boldsymbol{x}_q > 0 \\ \min_{{}^q\boldsymbol{v}_i \in \boldsymbol{S}_4} \eta({}^q\boldsymbol{v}_i, {}^q\boldsymbol{p}_2), & {}^q\boldsymbol{p}_2 \cdot \boldsymbol{x}_q < 0 \\ \min_{{}^q\boldsymbol{v}_i \in \boldsymbol{S}_5} ({}^q\boldsymbol{p}_1 - {}^q\boldsymbol{v}_i) \cdot \boldsymbol{y}_q, & \text{otherwise} \end{cases} \tag{12} $$

where $\zeta_3$ is the parameter to detect if the edges parallel with $\boldsymbol{d}$ pass through the polygon. Combining the above-mentioned parameter and (3) results in

$$ \Delta_r(\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{d}, \boldsymbol{v}) = \begin{cases} -1, & \Delta_l(\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{v}) < 0 \\ -1, & \zeta_3 \le 0 \\ 1, & \text{otherwise} \end{cases} \tag{13} $$

where $\Delta_r(\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{d}, \boldsymbol{v}) < 0$ means collision between the polygon $\boldsymbol{v}$ and the rectangle determined by two points $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ and the direction $\boldsymbol{d}$. When collision occurs, the contact location yields

$$ \boldsymbol{s}_r(\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{v}) = \begin{cases} \boldsymbol{s}_p(\boldsymbol{p}_1), & \Delta_p(\boldsymbol{p}_1, \boldsymbol{v}) \ge 0 \\ \boldsymbol{s}_p(\boldsymbol{p}_2), & \Delta_p(\boldsymbol{p}_2, \boldsymbol{v}) \ge 0 \\ \boldsymbol{p}_c & \text{otherwise} \end{cases} \tag{14} $$

where $\boldsymbol{s}_r(\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{v})$ represents the contact point location and $\boldsymbol{p}_c$ is the cross point between the lines $\overline{C_0\boldsymbol{v}_i}$ and the rectangle.

An iterative method is proposed to detect the collision status between two polygons, which is shown in Fig. 2(e). Draw a line connecting the geometry centers $C_0$ and $C_1$, and it intersects the polygon's edges. The vertices on the edges can be obtained by searching around the polar angle of $\overline{C_0C_1}$: label $\boldsymbol{p}_{k-1}$ and $\boldsymbol{p}_k$ as the edge vertices of the polygon $C_1$ and $\boldsymbol{v}_{i-1}$ and $\boldsymbol{v}_i$ as the edge vertices of the polygon $C_0$. We first use $\Delta_l$ to check if the edge $\overline{\boldsymbol{p}_{k-1}\boldsymbol{p}_k}$ intersects the polygon $C_0$, and if not, determine which point, between $\boldsymbol{p}_{k-1}$ and $\boldsymbol{p}_k$, is closer to the polygon $C_0$ using

$$ \begin{aligned} \zeta_4 = &\|(\boldsymbol{v}_{i-1} - \boldsymbol{v}_i) \times (\boldsymbol{p}_k - \boldsymbol{v}_{i-1})\|_2 \\ &- \|(\boldsymbol{v}_{i-1} - \boldsymbol{v}_i) \times (\boldsymbol{p}_{k-1} - \boldsymbol{v}_{i-1})\|_2 \end{aligned} \tag{15} $$

where $\zeta_4 < 0$ means $\boldsymbol{p}_k$ is closer to the other polygon than $\boldsymbol{p}_{k-1}$. The result of which point is closer decides later iteration direction, and the following two parameters are used to represent the closer point and the iteration direction

$$ \begin{cases} k_0 = k, k_1 = 1, & \zeta_4 \le 0 \\ k_0 = k-1, k_1 = -1, & \text{otherwise} \end{cases} \tag{16} $$

where $k_0$ means the vertex number of the closer one and $k_1$ means the iteration direction. Project the closer point $\boldsymbol{p}_{k_0}$ onto the line $\overline{\boldsymbol{v}_{i-1}\boldsymbol{v}_i}$, and check whether the projection point locates on the iteration extension of the line, which is

$$ \zeta_5 = \begin{cases} \left(\boldsymbol{p}_{k_0} - \boldsymbol{v}_{i-1}\right) \cdot \left(\boldsymbol{v}_{i-1} - \boldsymbol{v}_i\right), & \zeta_4 \le 0 \\ \left(\boldsymbol{p}_{k_0} - \boldsymbol{v}_i\right) \cdot \left(\boldsymbol{v}_i - \boldsymbol{v}_{i-1}\right), & \text{otherwise} \end{cases} \tag{17} $$

where $\zeta_5 > 0$ means $\boldsymbol{p}_{k_0}$ locates on the line's iteration extension. If the projection point of $\boldsymbol{p}_{k_0}$ lies outside the line $\overline{\boldsymbol{v}_{i-1}\boldsymbol{v}_i}$, use $\boldsymbol{v}_{i-k_1-1}$ and $\boldsymbol{v}_{i-k_1}$ to replace $\boldsymbol{v}_{i-1}$ and $\boldsymbol{v}_i$. The next step is to see if the direction $\overrightarrow{\boldsymbol{p}_{k_0}\boldsymbol{p}_{k_0+k_1}}$ points toward $\overrightarrow{\boldsymbol{v}_{i-1}\boldsymbol{v}_i}$

$$ \zeta_6 = \left[\left(\boldsymbol{p}_{k_0+k_1} - \boldsymbol{p}_{k_0}\right) \times \left(\boldsymbol{v}_i - \boldsymbol{v}_{i-1}\right)\right] \cdot \boldsymbol{z} \tag{18} $$

where $\zeta_6 \ge 0$ confirms noncollision. If $\zeta_6 < 0$, we turn to use $\Delta_p$ to check if $\boldsymbol{p}_{k_0+k_1}$ is inside the polygon $C_0$. If inside, collision is ensured, and otherwise replace the point $\boldsymbol{p}_{k_0}$ with $\boldsymbol{p}_{k_0+k_1}$ and repeat the above-mentioned detection until a contact status is determined. For collision, the contact location yields

$$ \boldsymbol{s}_c(\boldsymbol{p}, \boldsymbol{v}) = \begin{cases} \boldsymbol{s}_l(\boldsymbol{p}_i, \boldsymbol{p}_{i-1}, \boldsymbol{v}), & \Delta_l(\boldsymbol{p}_i, \boldsymbol{p}_{i-1}, \boldsymbol{v}) \le 0 \\ \boldsymbol{s}_p(\boldsymbol{p}_{k_0+k_1}), & \text{otherwise} \end{cases} \tag{19} $$

where $\boldsymbol{s}_c(\boldsymbol{p}, \boldsymbol{v})$ is the position vector of the contact point between the polygons $\boldsymbol{p}$ and $\boldsymbol{v}$.

## IV. Collision Detection of Convex Prisms

### A. Projection

On the geometry center of object $a$ is set the origin of the coordinates $O_a - x_a y_a z_a$, where $\boldsymbol{z}_a$ is coincident with $\boldsymbol{d}^a$, and the rotational matrix ${}^a\boldsymbol{R}_w$ transforms from the world coordinates to $O_a$. Select the object $a$s end surface as the projection plane, and project the object $b$s part, which is no upper than $a$s end surface, onto this plane. In the coordinates of $O_a$, the projection of $b$s main axis results in

$$ {}^p\boldsymbol{d}^b = {}^a\boldsymbol{R}_w \frac{\boldsymbol{d}^a \times \boldsymbol{d}^b \times \boldsymbol{d}^a}{\|\boldsymbol{d}^a \times \boldsymbol{d}^b \times \boldsymbol{d}^a\|_2} \tag{20} $$

where ${}^p\boldsymbol{d}^b$ is the projection of $\boldsymbol{d}^b$ onto the plane. The projection of the geometry center of $b$s end surface yields

$$ \Delta^p\boldsymbol{C}_0^{b-a} = {}^a\boldsymbol{R}_w \left(\boldsymbol{d}^a \times \Delta\boldsymbol{C}_0^{b-a} \times \boldsymbol{d}^a\right) \tag{21} $$

where $\Delta \boldsymbol{C}_0^{b-a}$ is the error vector from $b$s geometry center to $a$s and $\Delta^p \boldsymbol{C}_0^{b-a}$ is the projection location of $C_0^b$ expressed in the coordinates of $O_a$.

Another coordinates $O_b - x_b y_b z_b$ are set at $C_0^b$ with $\boldsymbol{z}_b$ parallel with $\boldsymbol{d}^b$, and ${}^w\boldsymbol{R}_b$ is the transformation matrix from the world coordinates to $O_b$. The coordinates of $O_{b'}$: the origin sets at the projection of $b$s geometry center, $\boldsymbol{y}_{b'}$ is along the projection of $\boldsymbol{d}^b$, and $\boldsymbol{z}_{b'}$ is opposite to $\boldsymbol{d}^a$. The matrix ${}^{b'}\boldsymbol{R}_w$ transforming from the world coordinates to $O_{b'}$ yields

$$ {}^{b'}\boldsymbol{R}_w = \left[ -\frac{\boldsymbol{d}^b \times \boldsymbol{d}^a}{\|\boldsymbol{d}^b \times \boldsymbol{d}^a\|_2} \quad \frac{\boldsymbol{d}^a \times \boldsymbol{d}^b \times \boldsymbol{d}^a}{\|\boldsymbol{d}^a \times \boldsymbol{d}^b \times \boldsymbol{d}^a\|_2} \quad -\boldsymbol{d}^a \right]^{-1}. \tag{22} $$

Let $n^a$ and $n^b$ represent the vertex number of the objects $a$ and $b$, and $C_0^a$ and $C_0^b$ be their geometry centers. The vertex position is expressed in different coordinates as

$$ \begin{aligned} {}^b\boldsymbol{v}_{0,k}^b &= \left[ \rho_k^b \cos\left({}^b\theta_{0,k}^b\right) \quad \rho_k^b \sin\left({}^b\theta_{0,k}^b\right) \quad 0 \right]^{\mathrm{T}} \\ \boldsymbol{v}_{0,k}^{b-a} &= \Delta \boldsymbol{C}_0^{b-a} + {}^w\boldsymbol{R}_b\, {}^b\boldsymbol{v}_{0,k}^b, \\ {}^p\boldsymbol{v}_{0,k}^{b-a} &= {}^a\boldsymbol{R}_w \left( \boldsymbol{d}^a \times \boldsymbol{v}_{0,k}^{b-a} \times \boldsymbol{d}^a \right) \\ {}^p\boldsymbol{v}_{0,k}^{b} &= {}^p\boldsymbol{v}_{0,k}^{b-a} - \Delta^p \boldsymbol{C}_0^{b-a}, \\ {}^{b'}\boldsymbol{v}_{0,k}^{b} &= \boldsymbol{z}_{b'} \times {}^{b'}\boldsymbol{R}_w\, {}^w\boldsymbol{R}_b\, {}^b\boldsymbol{v}_{0,k}^b \times \boldsymbol{z}_{b'}. \end{aligned} \tag{23} $$

where ${}^b\boldsymbol{v}_{0,k}^b$, ${}^p\boldsymbol{v}_{0,k}^b$, and ${}^{b'}\boldsymbol{v}_{0,k}^b$ are the $k$ vertex of the object $b$s end surface expressed in the coordinates of $O_b$, $O_a$, and $O_{b'}$, $\rho_k^b$ and ${}^b\theta_{0,k}^b$ are the polar parameter of the vertex, $\boldsymbol{v}_{0,k}^{b-a}$ and ${}^p\boldsymbol{v}_{0,k}^{b-a}$ are the vector from $C_0^a$ to the vertex expressed in the world coordinates and $O_a$, and $\boldsymbol{z}_{b'}$ is the vertical axis of the coordinates of $O_{b'}$. The object $a$s vertices locate at

$$ {}^p\boldsymbol{v}_i^a = {}^a\boldsymbol{R}_w \boldsymbol{v}_i^a, \quad {}^{b'}\boldsymbol{v}_i^a = {}^{b'}\boldsymbol{R}_w \left( \boldsymbol{v}_i^a - \Delta \boldsymbol{C}_0^{b-a} \right) \tag{24} $$

where ${}^p\boldsymbol{v}_i^a$ and ${}^{b'}\boldsymbol{v}_i^a$ are the expression of $a$s vertex in the coordinates of $O_a$ and $O_{b'}$.

The main axes of the two objects may be perpendicular, reflected by a parameter

$$ \Delta_1 = \boldsymbol{d}^a \cdot \boldsymbol{d}^b \tag{25} $$

where $\Delta_1 \neq 0$ indicates nonperpendicularity. In this scenario, $b$s main axis intersects with the projection plane at $C_1^b$, whose projection is

$$ {}^{b'}\boldsymbol{C}_1^b = \tan\alpha \left( \Delta \boldsymbol{C}_0^{b-a} \cdot \boldsymbol{d}^a \right) \boldsymbol{y}_{b'} \tag{26} $$

where ${}^{b'}\boldsymbol{C}_1^b$ is the project of $C_1^b$ expressed in the coordinates of $O_{b'}$, and $\alpha$ is the acute angle between the main axes. The cross section between the object $b$ and the projection plane is also a polygon with its vertex position at

$$ {}^{b'}\boldsymbol{v}_{1,k}^b = {}^{b'}\boldsymbol{v}_{0,k}^b \odot \left[ 1 \ \sec^2\alpha \ 1 \right]^{\mathrm{T}} + {}^{b'}\boldsymbol{C}_1^b \tag{27} $$

where $\odot$ is the Hadamard product and ${}^{b'}\boldsymbol{v}_{1,k}^b$ is the vertex position of the cross section expressed in $O_{b'}$. The above-mentioned equation means that compared with the polygon ${}^{b'}\boldsymbol{v}_0^b$, the vertex of $\boldsymbol{v}_1^b$ has the same $\boldsymbol{x}_{b'}$ value and expands in $\boldsymbol{y}_{b'}$.

The object $b$s projection is then a convex polygon combined by the outer vertices of the polygons ${}^{b'}\boldsymbol{v}_0^b$ and ${}^{b'}\boldsymbol{v}_1^b$ as well as the line connecting these two polygons. Label $r$ and $l$ to represent the rightmost and leftmost points in the viewpoint of $\boldsymbol{x}_{b'}$

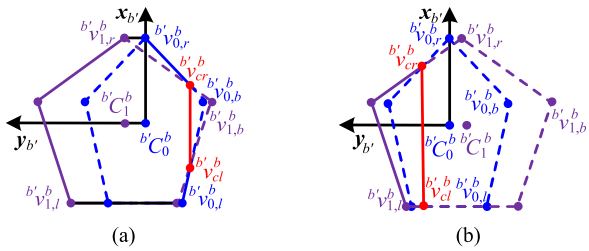


Fig. 3. Violation examples. (a) $\Delta \boldsymbol{C}_0^{b-a} \cdot \boldsymbol{d}^a < 0$. (b) $\Delta \boldsymbol{C}_0^{b-a} \cdot \boldsymbol{d}^a \geq 0$.

$$ {}^{b'}\boldsymbol{v}_{0,r}^b \cdot \boldsymbol{x}_{b'} = \max_{1 \leq k \leq n^b} {}^{b'}\boldsymbol{v}_{0,k}^b \cdot \boldsymbol{x}_{b'}, {}^{b'}\boldsymbol{v}_{0,l}^b \cdot \boldsymbol{x}_{b'} = \min_{1 \leq k \leq n^b} {}^{b'}\boldsymbol{v}_{0,k}^b \cdot \boldsymbol{x}_{b'} \tag{28} $$

where ${}^{b'}\boldsymbol{v}_{0,r}^b$ is the point with maximum projection onto $\boldsymbol{x}_{b'}$, ${}^{b'}\boldsymbol{v}_{0,l}^b$ has the minimum value along $\boldsymbol{x}_{b'}$, and they correspond to the polar angle ${}^{b'}\theta_{0,r}^b$ and ${}^{b'}\theta_{0,l}^b$. Sort the vertices in an ascending sequence and the following two sets are made

$$ \begin{cases} \boldsymbol{S}_6 = \{ {}^q\boldsymbol{v}_{1,r}^b, \ldots, {}^q\boldsymbol{v}_{1,l}^b | {}^q\theta_{1,r}^b < \cdots < {}^q\theta_{1,l}^b \} \\ \boldsymbol{S}_7 = \{ {}^q\boldsymbol{v}_{0,l}^b, \ldots, {}^q\boldsymbol{v}_{0,r}^b | {}^q\theta_{0,l}^b < \cdots < {}^q\theta_{0,r}^b \} \end{cases} \tag{29} $$

where $\boldsymbol{S}_6$ is the vertex set of the polygon ${}^pC_1^b$ above the connecting lines and $\boldsymbol{S}_7$ is the vertex set of the polygon ${}^pC_0^b$ below the connecting lines. The object $b$s projected polygon is ${}^{b'}\boldsymbol{v}^b = \{\boldsymbol{S}_6, \boldsymbol{S}_7\}$ and a new center of this contour is set as ${}^{b'}C^b$ by averaging the coordinate values of vertices.

The violation between the polygons ${}^{b'}\boldsymbol{v}_0^b$ and ${}^{b'}\boldsymbol{v}_1^b$ may happen. Since the polygons are convex, the bottom point is important in figuring out the violation

$$ {}^{b'}\boldsymbol{v}_{0,b}^b \cdot \boldsymbol{y}_{b'} = \min_{{}^{b'}\boldsymbol{v}_{0,k}^b \in \boldsymbol{S}_7} {}^{b'}\boldsymbol{v}_{0,k}^b \cdot \boldsymbol{y}_{b'} \tag{30} $$

where ${}^{b'}\boldsymbol{v}_{0,b}^b$ is the bottom point. The parameter to distinguish violation is

$$ \Delta_2 = \left( {}^{b'}\boldsymbol{v}_{1,b}^b - {}^{b'}\boldsymbol{v}_{0,b}^b \right) \cdot \boldsymbol{y}_{b'} \tag{31} $$

where $\Delta_2 < 0$ means the above-mentioned violation occurs. If they violate, $b$s vertex set needs to be rearranged. Fig. 3 shows two examples in which violation exists in the two polygons. The intersection (red lines in the figure) between the two polygons satisfies

$$ {}^{b'}\boldsymbol{v}_{cr,b}^b \cdot \boldsymbol{y}_{b'} = -\cot^2\alpha\, {}^{b'}\boldsymbol{C}_1^b \cdot \boldsymbol{y}_{b'} \tag{32} $$

where ${}^{b'}\boldsymbol{v}_{cr,b}^b$ is the cross point between polygons with a positive value along $\boldsymbol{x}_{b'}$. Then ${}^{b'}\boldsymbol{v}_{cr,b}^b \cdot \boldsymbol{x}_{b'}$ is acquired after finding which edge it belongs to and correspondingly ${}^{b'}\boldsymbol{v}_{cl,b}^b$ with a negative value on the $\boldsymbol{x}_{b'}$ axis is obtained. The vertex set for $b$s projection is changed by adding the points ${}^{b'}\boldsymbol{v}_{cr,b}^b$ and ${}^{b'}\boldsymbol{v}_{cl,b}^b$ and discarding the vertices below the cross points.

### B. Case Classification

Based on the above-mentioned projection and the detection methods for elementary contours, multiple cases are classified according to the relative position and posture between the objects. The parameter used to distinguish whether the object $b$ has an intersection with the projection plane yields

$$\Delta_3 = \max_{1\le k\le n^b} \boldsymbol{d}^a \cdot \boldsymbol{v}_{0,k}^{b-a} \tag{33}$$

where $\Delta_3$ is the maximum value of the dot product between $\boldsymbol{d}^a$ and all the vertices on $b$s end surface.

*Case 1:* $\Delta_3 < 0$, the object $b$ is totally above $a$ and no collision occurs.

As to $\Delta_3 = 0$, the object $b$ is no upper than $a$, where one or two vertices locate on the projection plane or the two end surfaces are on the same plane. Further investigation is needed between the vertices on the projection plane and the object $a$s projected contours. Cases 2–6 consider $\Delta_1 \neq 0$.

*Case 2:* $\Delta_3 = 0$ with only one vertex satisfying the equality. This changes to investigate the relationship between a point and a polygon, and label the point as ${}^p\boldsymbol{v}_k^{b-a}$. $\Delta_p({}^p\boldsymbol{v}_k^{b-a}, {}^p\boldsymbol{v}^a) < 0$ means no contact between the two objects, and otherwise collision happens, with the contact locating at $\boldsymbol{s} = {}^a\boldsymbol{R}_w^{-1}\boldsymbol{s}_p({}^p\boldsymbol{v}_k^{b-a}, {}^p\boldsymbol{v}^a)$.

*Case 3:* $\Delta_3 = 0$ and the equality holds in two vertices. It turns out to be the relationship judgement between a line segment and a polygon, and label the two points as ${}^p\boldsymbol{v}_{k-1}^{b-a}$ and ${}^p\boldsymbol{v}_k^{b-a}$. $\Delta_l({}^p\boldsymbol{v}_{k-1}^{b-a}, {}^p\boldsymbol{v}_k^{b-a}, {}^p\boldsymbol{v}^a) < 0$ leads to a collision status and otherwise safety is ensured. If collide, the contact point locates at $\boldsymbol{s} = {}^a\boldsymbol{R}_w^{-1}\boldsymbol{s}_l({}^p\boldsymbol{v}_{k-1}^{b-a}, {}^p\boldsymbol{v}_k^{b-a}, {}^p\boldsymbol{v}^a)$.

*Case 4:* $\Delta_3 = 0$ and at least three vertices subject to this condition. This is the case where the two end surfaces are in the same plane, and it becomes a problem of the relationship investigation between two polygons. Apply the methods for two polygons to check if these two shapes intersect, and if they do, $\boldsymbol{s} = {}^{b'}\boldsymbol{R}_w^{-1}\boldsymbol{s}_c({}^{b'}\boldsymbol{v}^b, {}^{b'}\boldsymbol{v}^a)$ is their contact position.

Following contents explore $\Delta_3 > 0$. Fine detection is time-consuming and we first propose a method to coarsely detect the two contours. The following three parameters are used

$$L_0 = \max_{1\le k\le n^b} \rho_k^b, \quad L_1 = \frac{L_0}{\cos^2\alpha}, \quad L_a = \max_{1\le i\le n^a} \rho_i^a \tag{34}$$

where $L_0$ is the possible maximum radius of the projected polygon corresponding to the object $b$s end surface, $L_1$ is the possible maximum radius of the cross section between the object $b$ and the projection plane, and $L_a$ is the largest polar radius of the object $a$. In the $O_{b'}$ coordinates, use a minimum rectangle to envelop all kinds of the object $b$s projection shapes, and non-collision is promised if the object $a$s contours is outside this envelope rectangle

$$\Delta_4 = \begin{cases} 1, & \|\Delta^{b'}\boldsymbol{C}_0^{b-a}\cdot\boldsymbol{x}_{b'}\| - L_0 - L_a > 0 \\ 1, & \Delta^{b'}\boldsymbol{C}_0^{b-a}\cdot\boldsymbol{y}_{b'} - L_0 - L_a > 0 \\ 1, & -\left({}^{b'}\boldsymbol{C}_1^b + \Delta^{b'}\boldsymbol{C}_0^{b-a}\right)\cdot\boldsymbol{y}_{b'} - L_1 - L_a > 0 \\ -1, & \text{otherwise} \end{cases} \tag{35}$$

where $\Delta_4$ is a coarse detection parameter.

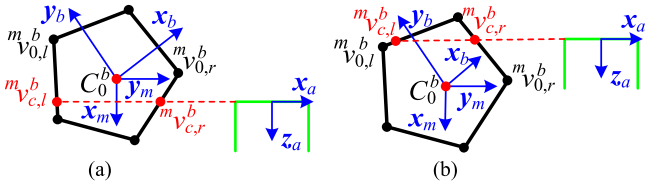


Fig. 4. Two instances for Case 9.

*Case 5:* $\Delta_4 > 0$, sufficient margin exists between the two polygons and no collision asserts.

*Case 6:* $\Delta_4 \le 0$, and the method for two convex polygons is applied. To determine the contact position if collision occurs, define a function

$$\chi(x) = \begin{cases} x, & x > 0 \\ 0, & \text{otherwise} \end{cases} \tag{36}$$

where $\chi(x)$ is a step function. Label the contact point found in the coordinates $O_{b'}$ as ${}^{b'}\boldsymbol{v}_c^b$, and passing through this point draw a vertical line to intersect the bottom of the polygon ${}^{b'}\boldsymbol{v}^b$ with the cross point labeled as ${}^{b'}\boldsymbol{v}_{1,c}^b$. The contact point locates

$$\begin{aligned}\boldsymbol{s} = {}^{b'}\boldsymbol{R}_w^{-1}\boldsymbol{s}_c\left({}^{b'}\boldsymbol{v}^b, {}^{b'}\boldsymbol{v}^a\right) \\ + \cot\alpha\chi\left({}^{b'}\boldsymbol{v}_{1,c}^b\cdot\boldsymbol{y}_{b'} - {}^{b'}\boldsymbol{v}_c^b\cdot\boldsymbol{y}_{b'}\right)\boldsymbol{d}^a.\end{aligned} \tag{37}$$

The perpendicular cases, i.e., $\Delta_1 = 0$ are discussed below.

*Case 7:* Only one vertex satisfies $\Delta_3 = 0$. The object $b$s projection is a half-line, starting from this vertex (label it as ${}^p\boldsymbol{v}_{0,k}^{b-a}$) and along ${}^p\boldsymbol{d}^b$. $\Delta_h({}^p\boldsymbol{v}_{0,k}^{b-a}, {}^p\boldsymbol{d}^b, {}^p\boldsymbol{v}^a) > 0$ represents noncollision, and contact otherwise with its location at $\boldsymbol{s} = {}^a\boldsymbol{R}_w^{-1}\boldsymbol{s}_h({}^p\boldsymbol{v}_{0,k}^{b-a}, {}^p\boldsymbol{v}^a)$.

*Case 8:* $\Delta_3 = 0$ with two vertices satisfying the equality. The object $b$s projection is a rectangle with unlimited edges parallel with the direction ${}^p\boldsymbol{d}^b$. Label the two points as ${}^p\boldsymbol{v}_{k-1}^{b-a}$ and ${}^p\boldsymbol{v}_k^{b-a}$, and $\Delta_r({}^p\boldsymbol{v}_{k-1}^{b-a}, {}^p\boldsymbol{v}_k^{b-a}, {}^p\boldsymbol{d}^b, {}^p\boldsymbol{v}^a) < 0$ corresponds to a collision status and contacts otherwise. The contact position is $\boldsymbol{s} = {}^a\boldsymbol{R}_w^{-1}\boldsymbol{s}_r({}^p\boldsymbol{v}_{k-1}^{b-a}, {}^p\boldsymbol{v}_k^{b-a}, {}^p\boldsymbol{v}^a)$.

*Case 9:* $\Delta_3 < 0$. The object $b$s projected contour is a rectangle, and we need to determine its width. Set a coordinate frame $O_m - x_m y_m z_m$, whose origin is at $C_0^b$, $\boldsymbol{x}_m$ is parallel with $\boldsymbol{d}^a$, and $\boldsymbol{z}_m$ is coincident with $\boldsymbol{d}^b$. The rotational matrix transforming from $O_m$ to the world frame is ${}^w\boldsymbol{R}_m = \left[\boldsymbol{d}^a, \boldsymbol{d}^b\times\boldsymbol{d}^a, \boldsymbol{d}^b\right]$. Label ${}^m\boldsymbol{v}_{0,l}^b$ and ${}^m\boldsymbol{v}_{0,r}^b$ as the leftmost and rightmost points on $b$s end surface in the coordinates $O_m$, and their positions can be computed resorting to (28). Fig. 4 shows two instances the difference between which lies in whether the projection plane surpasses the leftmost and rightmost points. The projection plane may intersect with the object $b$'s end surface at

$${}^m\boldsymbol{v}_{c,r}^b\cdot\boldsymbol{x}_m = -\Delta\boldsymbol{C}_0^{b-a}\cdot\boldsymbol{d}^a \tag{38}$$

where ${}^m\boldsymbol{v}_{c,r}^b$ is the rightmost point of the intersection line between the object $b$s end surface and the projection plane. The rectangle width is determined by the point ${}^m\boldsymbol{v}_{c,r}^b$ if this point is lower than the rightmost point, and by ${}^m\boldsymbol{v}_{0,r}^b$ otherwise, which

yields

$$ {}^{m}\boldsymbol{v}_r^b = \begin{cases} {}^{m}\boldsymbol{v}_{c,r}^b, & \left({}^{m}\boldsymbol{v}_{0,r}^b - {}^{m}\boldsymbol{v}_{c,r}^b\right)\cdot \boldsymbol{x}_m < 0 \\ {}^{m}\boldsymbol{v}_{0,r}^b, & \text{otherwise} \end{cases} \tag{39} $$

where ${}^{m}\boldsymbol{v}_r^b$ is the point whose projection corresponds to the rightmost of the projected contour. In the same way, the leftmost point ${}^{m}\boldsymbol{v}_l^b$ can be obtained. Transforming these two points to the coordinates of $O_a$ results in

$$ {}^{p}\boldsymbol{v}_r^b = {}^{a}\boldsymbol{R}_m\, {}^{m}\boldsymbol{v}_r^b + \Delta^{p}\boldsymbol{C}_0^{b-a}, {}^{p}\boldsymbol{v}_l^b = {}^{a}\boldsymbol{R}_m\, {}^{m}\boldsymbol{v}_l^b + \Delta^{p}\boldsymbol{C}_0^{b-a} \tag{40} $$

where ${}^{p}\boldsymbol{v}_r^b$ and ${}^{p}\boldsymbol{v}_l^b$ are the expression in the coordinates of $O_a$. The parameter used to distinguish intersection between a rectangle and a polygon is applied. $\Delta_r({}^{p}\boldsymbol{v}_l^b, {}^{p}\boldsymbol{v}_r^b, {}^{p}\boldsymbol{d}^b, {}^{p}\boldsymbol{v}^a) > 0$ means no collision, and contact otherwise. The projection of the contact point locates $\boldsymbol{s}_h = {}^{a}\boldsymbol{R}_w^{-1}\boldsymbol{s}_r({}^{p}\boldsymbol{v}_l^b, {}^{p}\boldsymbol{v}_r^b, {}^{p}\boldsymbol{v}^a)$ on the projection plane, and its distance to the projection plane yields

$$ \boldsymbol{s}_v = \begin{cases} \chi\left({}^{m}\boldsymbol{v}_l^b\cdot\boldsymbol{x}_m - {}^{m}\boldsymbol{v}_{c,l}^b\cdot\boldsymbol{x}_m\right)\boldsymbol{d}^a, & {}^{q}\boldsymbol{R}\left(\boldsymbol{s}_h - {}^{w}\boldsymbol{R}_m\, {}^{m}\boldsymbol{v}_l^b\right)\times\boldsymbol{x}_q = 0 \\ \chi\left({}^{m}\boldsymbol{v}_r^b\cdot\boldsymbol{x}_m - {}^{m}\boldsymbol{v}_{c,r}^b\cdot\boldsymbol{x}_m\right)\boldsymbol{d}^a, & {}^{q}\boldsymbol{R}\left(\boldsymbol{s}_h - {}^{w}\boldsymbol{R}_m\, {}^{m}\boldsymbol{v}_r^b\right)\times\boldsymbol{x}_q = 0 \\ \left[\chi\left({}^{m}\boldsymbol{v}_r^b\cdot\boldsymbol{x}_m\right) + \Delta\boldsymbol{C}_0^{b-a}\cdot\boldsymbol{d}^a\right]\boldsymbol{d}^a, & \text{otherwise} \end{cases} \tag{41} $$

where ${}^{w}\boldsymbol{R}_m = {}^{a}\boldsymbol{R}_w^{-1}{}^{a}\boldsymbol{R}_m$. The first two rows of the above-mentioned equation correspond to the distances from ${}^{m}\boldsymbol{v}_l^b$ and ${}^{m}\boldsymbol{v}_r^b$ to the projection plane, and the last row means the contact lies on the end surface of the object $b$. The contact position yields $\boldsymbol{s} = \boldsymbol{s}_h + \boldsymbol{s}_v$.

## V. Detach Controller

When objects collide in their motion, detach control is needed to immediately separate them for safety concern. This detachment relates with the position of the contact point, choosing the direction to maximally push the objects away. Meanwhile, this controller is designed in terms of the motion state error: a small force for the almost-arrived state and a large detachment if the desired state is far away. Regarding these issues, the detach controller is designed as

$$ \boldsymbol{u}_d = \begin{cases} \varepsilon\|\Delta\boldsymbol{x}_d\|_2\overrightarrow{\boldsymbol{c}_v}, & \boldsymbol{s}\in\boldsymbol{v}^a \\ \varepsilon\|\Delta\boldsymbol{x}_d\|_2\overrightarrow{\boldsymbol{c}_e}, & \boldsymbol{s}\in\boldsymbol{e}^a \\ \varepsilon\|\Delta\boldsymbol{x}_d\|_2\frac{\boldsymbol{d}^a\times\left(\boldsymbol{e}_i^a+\boldsymbol{e}_{i+1}^a\right)}{\|\boldsymbol{d}^a\times\left(\boldsymbol{e}_i^a+\boldsymbol{e}_{i+1}^a\right)\|_2}, & \boldsymbol{d}^a\times\boldsymbol{s}\times\boldsymbol{d}^a\in\boldsymbol{v}^a \\ \varepsilon\|\Delta\boldsymbol{x}_d\|_2\frac{\boldsymbol{d}^a\times\boldsymbol{e}_i^a}{\|\boldsymbol{d}^a\times\boldsymbol{e}_i^a\|_2}, & \boldsymbol{d}^a\times\boldsymbol{s}\times\boldsymbol{d}^a\in\boldsymbol{e}^a \\ -\varepsilon\|\Delta\boldsymbol{x}_d\|_2\boldsymbol{d}^a, & \boldsymbol{s}\cdot\boldsymbol{d}^a = 0 \\ \varepsilon\|\Delta\boldsymbol{x}_d\|_2\boldsymbol{c}_c, & \text{otherwise} \end{cases} \tag{42} $$

where $\boldsymbol{u}_d$ is the detach force when collision occurs, $\varepsilon$ is a positive parameter, $\Delta\boldsymbol{x}_d = \boldsymbol{x} - \boldsymbol{x}_d$ is the state error, $\boldsymbol{e}_i^a$ is the edge if the contact point lies on it or one of the edges beside the contact point if it is a vertex, $\overrightarrow{\boldsymbol{c}_v}$ and $\overrightarrow{\boldsymbol{c}_e}$ are the detach directions, and $\boldsymbol{c}_c$ is the shortest distance vector for detachment. In the above-mentioned equation, the first row stands for the cases in which the contact point is one of the vertices and the detach direction $\overrightarrow{\boldsymbol{c}_v}$ is set to push the object $b$ away the contact point, considering $b$s main axis. To do that, the direction $\boldsymbol{d}_v$ is introduced, which is the projection of $\boldsymbol{d}^b$ onto the plane that is perpendicular to the sum of adjacent edges

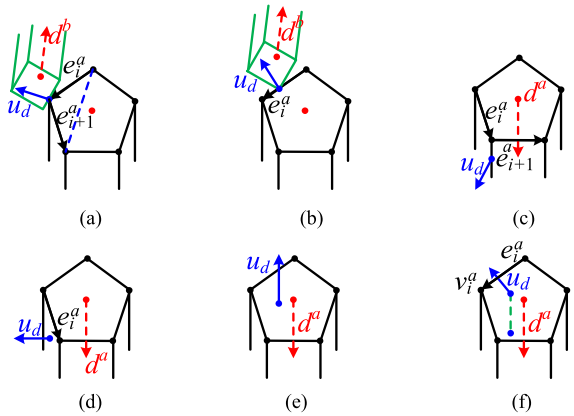


Fig. 5. Detach directions for different contact locations.

$$ \boldsymbol{d}_v = \frac{\left(\boldsymbol{e}_i^a+\boldsymbol{e}_{i+1}^a\right)\times\boldsymbol{d}^b\times\left(\boldsymbol{e}_i^a+\boldsymbol{e}_{i+1}^a\right)}{\left\|\left(\boldsymbol{e}_i^a+\boldsymbol{e}_{i+1}^a\right)\times\boldsymbol{d}^b\times\left(\boldsymbol{e}_i^a+\boldsymbol{e}_{i+1}^a\right)\right\|_2}. \tag{43} $$

$\overrightarrow{\boldsymbol{c}_v}$ is then determined, as shown in Fig. 5(a), based on the condition whether $\boldsymbol{d}_v$ points away from $\boldsymbol{d}^a$

$$ \overrightarrow{\boldsymbol{c}_v} = \begin{cases} \boldsymbol{d}_v, & \boldsymbol{d}_v\cdot\left[\boldsymbol{d}^a\times\left(\boldsymbol{e}_i^a+\boldsymbol{e}_{i+1}^a\right)\right]\geq 0 \\ \left(\boldsymbol{e}_i^a+\boldsymbol{e}_{i+1}^a\right)\times\boldsymbol{d}_v. & \text{otherwise.} \end{cases} \tag{44} $$

In (42), the second row deals with the cases in which the contact points belong to the edges and the detach direction is $\overrightarrow{\boldsymbol{c}_e}$. Similarly, the direction $\boldsymbol{d}_e$ is introduced, which is the projection of $\boldsymbol{d}^b$ onto the plane that is perpendicular to the edge

$$ \boldsymbol{d}_e = \frac{\boldsymbol{e}_i^a\times\boldsymbol{d}^b\times\boldsymbol{e}_i^a}{\|\boldsymbol{e}_i^a\times\boldsymbol{d}^b\times\boldsymbol{e}_i^a\|_2}. \tag{45} $$

$\overrightarrow{\boldsymbol{c}_e}$ is then computed to point outwards $\boldsymbol{d}^a$, as shown in Fig. 5(b)

$$ \overrightarrow{\boldsymbol{c}_e} = \begin{cases} \boldsymbol{d}_e, & \boldsymbol{d}_e\cdot\left(\boldsymbol{d}^a\times\boldsymbol{e}_i^a\right)\geq 0 \\ \boldsymbol{e}_i^a\times\boldsymbol{d}_e. & \text{otherwise.} \end{cases} \tag{46} $$

In (42), the third row considers the objects contacting at the lateral edge, as shown in Fig. 5(c), and the detach direction is perpendicular to $\boldsymbol{d}^a$ and the sum of adjacent edges. The fourth row in (42) and Fig. 5(d) discuss the objects contacting at the side face, and the detach direction is normal to the side face. The fifth row in (42), also shown in Fig. 5(e), handles the cases that the contact point locates on the end surface and the detach direction is opposite to $\boldsymbol{d}^a$. The last row in (42) shows the cases where the contact point locates inside the object, as shown in Fig. 5(f). This may happen to flexible objects or in simulation, and the detach force is along the direction with the shortest distance

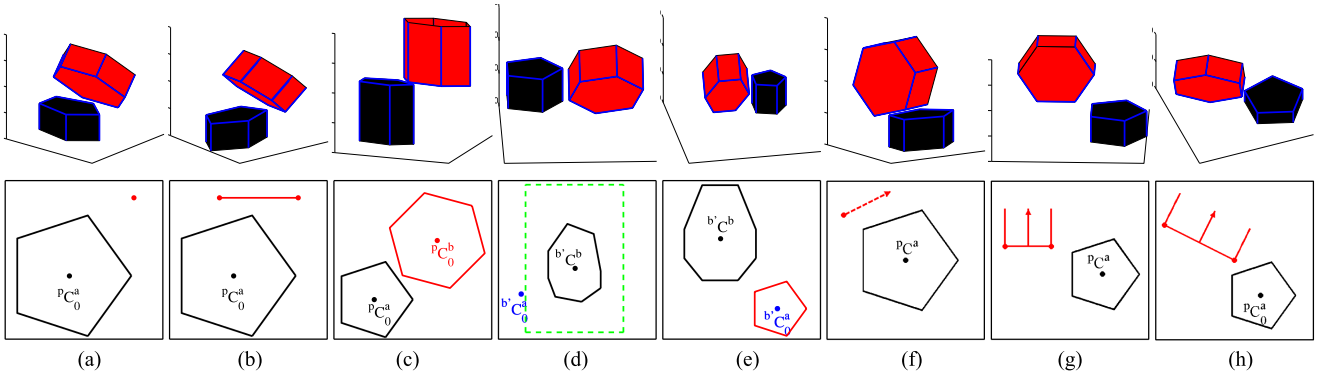


Fig. 6. Projections of eight typical instances for contact detection, where the upper row is the 3-D models, and the lower half is the projection contours. (a) Case 2. (b) Case 3. (c) Case 4. (d) Case 5. (e) Case 6. (f) Case 7. (g) Case 8. (h) Case 9.

to separate the objects. Label $g \in [1, n^b]$ as the index with the minimum $\|\boldsymbol{e}_i \times (\boldsymbol{s} - \boldsymbol{v}_i) \times \boldsymbol{e}_i\|_2$, and comparing between the distances from the contact point to the side plane and to the end surface the vector $\boldsymbol{c}_c$ yields

$$\boldsymbol{c}_c = \begin{cases} \boldsymbol{e}_g \times (\boldsymbol{s} - \boldsymbol{v}_g) \times \boldsymbol{e}_g, & \|\boldsymbol{e}_g \times (\boldsymbol{s} - \boldsymbol{v}_g) \times \boldsymbol{e}_g\|_2 \le \boldsymbol{s} \cdot \boldsymbol{d}^a \\ -(\boldsymbol{s} \cdot \boldsymbol{d}^a)\, \boldsymbol{d}^a. & \text{otherwise.} \end{cases} \tag{47}$$

Since the environment and the object number may be various, long lasting of this detach force may not be appropriate. Therefore, in this paper, the detach force only functions at the current step. It can also be regulated according to the strategy requirement.

## VI. Simulations and Experiments

In simulations, we test the validity of the proposed method and compare its performance with a popular method used in animation. In experiments, we apply the detection and control methods to precision manipulation to demonstrate their feasibility.

### A. Simulations

To show the projected contours, the object $a$ is selected as a prism whose end surface is regular pentagonal with its circumcircle's radius as 2, and the end surface of the object $b$ is chosen as regular hexagonal with its circumcircle's radius as 2.5. Fig. 6 shows the projected contours of eight cases for collision detection, where the upper row is the relative posture in 3-D and the lower row displays the projected contours. For case 1, there's no appearance of object $b$s shape on the projection plane and as a result there's no case 1 in Fig. 6. In Fig. 6(d), the green dashed is the marginal rectangle for case 5, out of which noncollision is ensured. Comparing the 3-D configuration and its projection, we find that the planar contours can reflect the 3-D collision status and that the projection method decreases the detection complexity. Besides the examples shown in Fig. 6, a lot of cases have been simulated, in which the relative position and posture are randomly set, and the detected statuses of the proposed algorithm and the other methods turn out to be the same. This shows the validity of the proposed method. Given accurate position and posture of objects as well as their geometries, false alarm does not happen, but due to disturbances and measurement errors, the input information is not accurate and false alarm may possibly appear in practice.

TABLE I
Average Computation Results for Comparison

| Methods | case 1 | case 2 | case 3 | case 4 | case 5 | case 6 | case 7 | case 8 | case 9 |
|---|---|---|---|---|---|---|---|---|---|
| projection | 0.03 | 0.10 | 0.17 | 0.31 | 0.09 | 0.55 | 0.12 | 0.20 | 0.29 |
| SWIFT | 5.03 | 4.96 | 5.02 | 5.06 | 4.99 | 4.95 | 5.01 | 5.09 | 4.91 |

Computation time is chosen as the index to evaluate the performance of the proposed method, and as commonly used in animation, the SWIFT method is employed for comparison. A computer with Intel i3 3.30 GHz processor is used for the computation, and we first use the objects whose end surfaces are both the convex polygons with 3600 vertices. Plenty of simulations are carried out, and Table I shows the average time cost in each case. The computation efficiency of the projection method depends on the case: the minimum time cost is 0.03 ms in case 1 and the longest computation is 0.55 ms in case 6 where the relationship between two polygons are considered, with an average spend of 0.2 ms. The ratio between the fastest and slowest computation is 6%. From another aspect, the 3-D detection efficiency varies as considering different planar shapes: the time cost between a polygon and a point or a line segment or a half-line averages 0.13 ms; the detection between a polygon and a unlimited-length rectangle spends 0.25 ms; and the slowest lies in the consideration between two polygons, about 0.43 ms. The SWIFT costs are steady to each case and it spends around 5 ms for average detection. Compared with it, the proposed method is about 0.6%–11% of computation cost with 4% in average.

The time cost of a collision algorithm relates to the vertex number of objects and the computation prolongs as the surface becomes complicated. We select the objects with different vertices and analyze the methods' time complexity. A hundred of different positions and postures are initialized for each case and an average cost result is computed. Fig. 7(a) and (b) presents the computation time of each case using the projection method

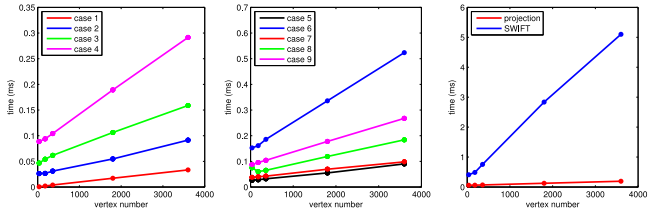


Fig. 7. Computational time versus vertex number for the two methods. (a) Cases 1–4. (b) Cases 5–9. (c) Average computation.

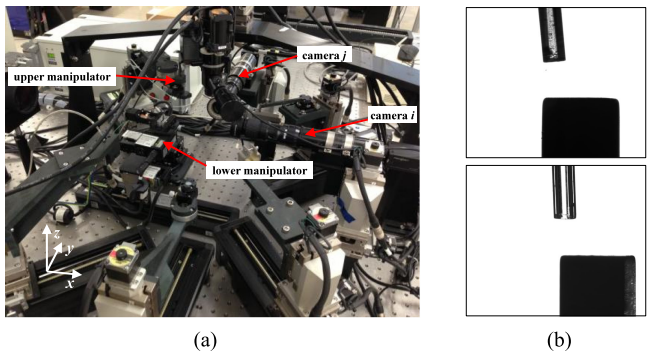


Fig. 8. Experimental platform and the objects observed in the horizontal microscopes. (a) Platform. (b) Objects.

with 36, 180, 360, 1800, and 3600 vertices. As can be seen, the computation time is linear with the complexity of the object's surfaces, and the ratio of each case is different: the largest ratio lies in case 6, where two polygons with many vertices are involved, and the smallest belongs to case 1 where only one condition is checked. The most time-consuming case costs about extra 0.1 ms as 1000 more vertices are added in the objects, while the most efficient case spends another 9 $\mu$s when the detection deals with 1000 more vertices, which shows approximately 9% in the ratio. Fig. 7(c) shows the time cost of each method versus the objects' vertex number. Although these two methods are linear with vertex number, their cost growths versus vertex number are different: as 1000 more vertices add into the object, the SWIFT costs 1.3 ms more and the projection method spends extra 0.037 ms. That is the same to conclude that the cost growths versus vertex number of the projection method is about 2.8% of the SWIFT method. These demonstrate that this projection method has great advantages in computational efficiency for prism collision detection.

## B. Experiments

Experiments are carried out on a platform with six robot arms and three microscopes, as shown in Fig. 8(a). In this experiment, only two robot arms and two microscopes are used. The cameras $i$ and $j$ are GC2450 and PointGrey 50S5M-C whose cell size is 3.45 $\mu$m. As to the upper manipulator, the translational DoFs are realized by three Suguar KWG06030-G, with resolution of $\pm 0.5\mu$m, and the manual rotation is executed by the tilt stage Sigma KKD-25C. The lower manipulator uses Micos ES-100 with movement errors within 0.1 $\mu$m for vertical elevation, and KGW06050-L, KGW06075-L, and SGSP-40yaw for rotation. We have experimented three types of objects: a triangular prism with side length of 2.8 mm, a quadrangular prism with side length of 3 mm, and a glass cylinder with diameter of 985 $\mu$m. Fig. 8(b) shows the last two objects observed in the two microscopes. The usage of the cylinder object intends to show the general application of the proposed method.

A PD controller is employed to generate the attraction force to drive the object approaching its desired state and the controller parameters are set as $k_p = 0.3\boldsymbol{I}_{6\times 6}$ and $k_d = 0.002\boldsymbol{I}_{6\times 6}$. The PD controller parameters are regulated with Ziegler–Nichols method. The detach scaling parameter is chosen as $\varepsilon = 10$, and 3600 vertices are used to fit the end surface of the cylinder. The object is expected to move from the initial state $\Delta\boldsymbol{C}_0^{b-a} = [2551, -2780, -1970]^{\mathrm{T}}\,\mu$m to the desired state $\Delta\boldsymbol{C}_0^{b-a} = [0, 0, 1000]^{\mathrm{T}}\,\mu$m, and the unchanged postures are $\boldsymbol{d}^b = [-0.0262, -0.0087, 0.9996]^{\mathrm{T}}$ and $\boldsymbol{d}^a = [0, 0, -1]^{\mathrm{T}}$. The initial positions and postures are selected randomly and the sampling time in experiment is 0.02 s, limited by the frequency of the cameras.

Fig. 9 shows the figures as the object $b$ moves from its initial state to the desired state, where the upper sequence is the views from the microscope $i$ and the lower is from the microscope $j$. The object $a$ is placed so that the diagonals of the square are parallel with the $x$ and $y$ axes. Therefore, in the second to fourth configurations, the two objects block each other in the two views but collision only occurs in the third figure. To show the detection results, Fig. 10 displays the objects' contours on the projection plane. These contours also reflect the relative movement between the two objects: the first three contours correspond to case 5 and the center projections of the objects move closer; the rest contours correspond to case 6, where objects are too close to be coarsely detected; and the last contour indicates the upper object ascends so that the shape violation happens. It can also be seen although the objects are blocked at 4 s (also applied at 3 s) in the view planes, they are not in collision concluded from the viewpoint of the projection plane. After 6 s, the upper object has no projected shape and then case 1 applies. The whole detection cost for this movement is about 121 ms while SWIFT spends about 3375 ms. For a better view of the object movement, Fig. 11(a) presents the object trajectory viewed from the axis that is perpendicular to $\boldsymbol{d}^a$ and the vector from the desired state to the initial state. The red is the object $a$, the black is the object $b$ in its initial state, and the blue is the object $b$ in its final state. The dotted contours represent the enlarged diameter, which is 1.5 mm, for the sake of protection, and the magenta is the trajectory of the objects $b$. The attraction controller pushes the object $b$ directly to its desired state and the two objects collide in the process, with the detach controller immediately activated to separate them. The contact point lies on the side face and the detach force pushes toward the direction perpendicular to $\boldsymbol{d}^a$ and the side face with substantial margin generated between objects. After the detachment's effect is counteracted, the attraction controller continues to move the object $b$ to its desired state. The rest of Fig. 11 are the states and forces versus time, and at about 3.5 s, the objects' collision is reflected in the state and force change.

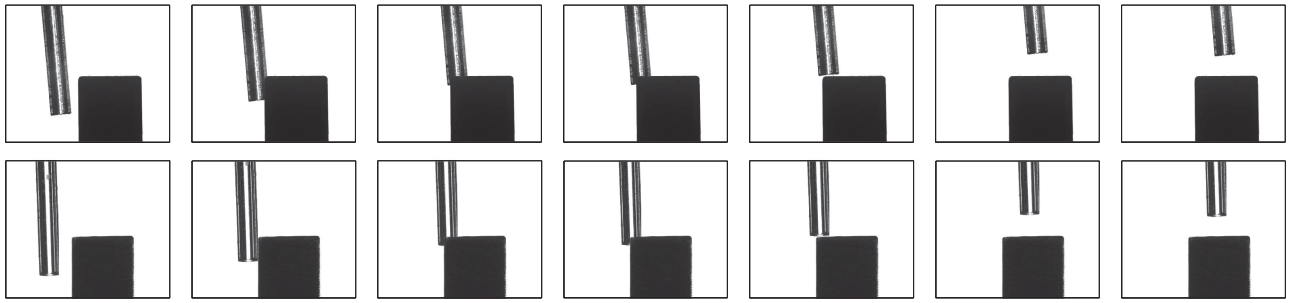

Fig. 9. Object configurations at 0, 2, 3.5, 4, 6, 10, and 20 s. The first row is the figures in the view of microscope $i$ and the second row is of microscope $j$.

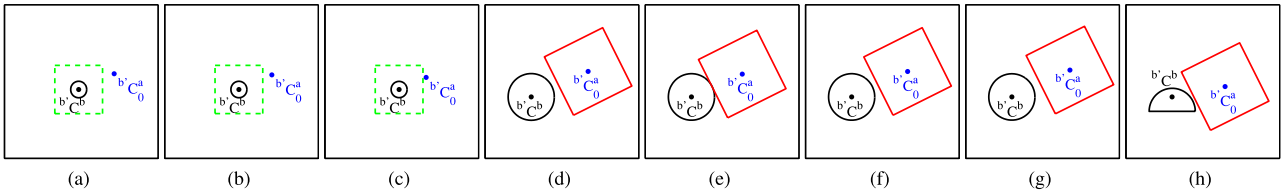


Fig. 10. Planar contours on the projection plane in movements. (a) initialized state; (b)–(h) are at 1, 2, 3, 3.5, 4, 5, and 5.9 s.

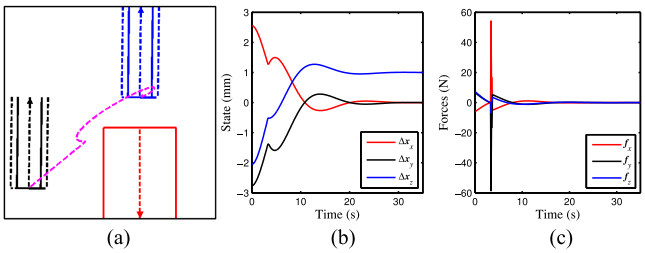


Fig. 11. Experiment results. (a) the trajectory observed from the axis that is parallel with $\boldsymbol{d}^a \times (\boldsymbol{x}_d - \boldsymbol{x}_0)$; (b) the object state; (c) the forces to drive the object.

## VII. Conclusion

To improve the computational efficiency of collision detection for convex prisms without lowering accuracy, this paper applies a projection method to detect 3-D collision by means of their planar contours. We investigate the fast overlap checking methods of five kinds of elementary shapes, classify the 3-D detection into nine cases according to their relative position and posture, and directly apply the efficient planar detection instead of 3-D computation using a detecting while projecting policy. For the collision occurrence, a detach controller is introduced based on which part of the object the contact point locates. In simulations, the proposed method is compared with SWIFT, showing great advantages in time complexity, and experiments carried out on the precision manipulation platform demonstrate the validity of the detection and controller.

Future research will consider collision prediction in terms of probability based on object speed, projected contours, and related cases, which also facilitates collision alarm and avoidance as well as the adaptation of the detach parameter.

## References

[1] P. Jimenez, F. Thomas, and C. Torras, "3-D collision detection: A survey," *Comput. Graph.*, vol. 25, no. 2, pp. 269–285, 2001.

[2] S. A. Ehmann and M. C. Lin, "Accelerated proximity queries between convex polyhedra by multilevel Voronoi marching," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2000, pp. 2101–2106.

[3] X. Zhang and Y. J. Kim, "Interactive collision detection for deformable models using streaming AABBs," *IEEE Trans. Vis. Comput. Graphics*, vol. 13, no. 2, pp. 318–329, Mar./Apr. 2007.

[4] L. He, J. Pan, D. Li, and D. Manocha, "Efficient penetration depth computation between rigid models using contact space propagation sampling," *IEEE Robot. Autom. Lett.*, vol. 1, no. 1, pp. 10–17, Jan. 2015.

[5] M. Tang *et al.*, "Fast and exact continuous collision detection with Bernstein sign classification," *ACM Trans. Graph.*, vol. 33, no. 6, pp. 186.1–8, 2014.

[6] M. Santos, C. D. Rosales, M. Sarcinelli-Filho, and R. Carelli, "A novel null-space-based UAV trajectory tracking controller with collision avoidance," *IEEE/ASME Trans. Mechatronics*, vol. 22, no. 6, pp. 2543–2553, Dec. 2017.

[7] E. Dean, K. R. Amaro, F. Bergner, I. Dianov, and G. Cheng, "Integration of robotic technologies for rapidly deployable robots," *IEEE Trans. Ind. Inf.*, to be published.

[8] S. Ge and Y. J. Cui, "Dynamic motion planning for mobile robots using potential field method," *Auton. Robots*, vol. 13, no. 3, pp. 207–222, 2002.

[9] L. Garcia-Delgado *et al.*, "Quad-Rotors formation based on potential functions with obstacle avoidance," *IET Control Theory Appl.*, vol. 6, no. 12, pp. 1787–1802, 2012.

[10] R. Schlanbusch and E. Oland, "Spacecraft formation reconfiguration with dynamic collision avoidance," in *Proc. IEEE Aerosp. Conf.*, 2013, pp. 1–12.

[11] J. Seo, Y. Kim, S. Kim, and A. Tsourdos, "Collision avoidance strategies for unmanned aerial vehicles in formation flight," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 53, no. 6, pp. 2718–2734, Dec. 2017.

[12] T. L. Baldi, S. Scheggi, M. Aggravi, and D. Prattichizzo, "Haptic guidance in dynamic environments using optimal reciprocal collision avoidance," *IEEE Robot. Autom. Lett.*, vol. 3, no. 1, pp. 265–272, Jan. 2018.

[13] J. Funke, M. Brown, S. M. Erlien, and J. C. Gerdes, "Collision avoidance and stabilization for autonomous vehicles in emergency scenarios," *IEEE Trans. Control Syst. Technol.*, vol. 25, no. 4, pp. 1204–1216, Jul. 2016.

[14] Y. Lu, Z. Xi, and J. Lien, "Online collision prediction among 2-D polygonal and articulated obstacles," *Int. J. Robot. Res.*, vol. 35, no. 5, pp. 476–500, 2016.

[15] J. Pan and D. Manocha, "Fast probabilistic collision checking for sampling-based motion planning using locality-sensitive hashing," *Int. J. Robot. Res.*, vol. 35, no. 12, pp. 1477–1496, 2016.
[16] P. Cai, I. Chandrasekaran, J. Zheng, and Y. Cai, "Automatic path planning for dual-crane lifting in complex environments using a prioritized multi-objective PGA," *IEEE Trans. Ind. Inf.*, vol. 14, no. 3, pp. 829–845, Mar. 2018.
[17] I. Tomasic, A. Andersson, and P. Funk, "Mixed-effect models for the analysis and optimization of sheet-metal assembly processes," *IEEE Trans. Ind. Inf.*, vol. 13, no. 5, pp. 2194–2202, Oct. 2017.
[18] S. Liu, Y. Li, D. Xing, D. Xu, and H. Su, "An efficient insertion control method for precision assembly of cylindrical components," *IEEE Trans. Ind. Electron.*, vol. 64, no. 12, pp. 9355–9365, Dec. 2017.
[19] S. Liu *et al.*, "Relative pose estimation for alignment of long cylindrical components based on microscopic vision," *IEEE/ASME Trans. Mechatronics*, vol. 21, no. 3, pp. 1388–1398, Jun. 2016.
[20] D. Xing, F. Liu, S. Liu, and D. Xu, "Motion control for cylindrical objects in microscope's view using a projection method: I. Collision detection and detach control," *IEEE Trans. Ind. Electron.*, vol. 64, no. 7, pp. 5524–5533, Jul. 2017.
[21] D. Xing, F. Liu, S. Liu, and D. Xu, "Motion control for cylindrical objects in microscope's view using a projection method: II. Collision avoidance with reduced dimensional guidance," *IEEE Trans. Ind. Electron.*, vol. 64, no. 7, pp. 5534–5544, Jul. 2017.
[22] M. Savia and H. N. Koivo, "Contact micromanipulation—Survey of strategies," *IEEE/ASME Trans. Mechatronics*, vol. 14, no. 4, pp. 504–514, Aug. 2009.

**Dengpeng Xing** received the B.S. degree in mechanical electronics and the M.S. degree in mechanical manufacturing and automation from the Tianjin University, Tianjin, China, in 2002 and 2006, respectively and the Ph.D. degree in control science and engineering from the Shanghai Jiao Tong University, Shanghai, China, in 2010.

He is currently an Associate Professor with the Research Center of Precision Sensing and Control, Institute of Automation, Chinese Academy of Sciences, Beijing, China. His research interests include robot control and learning, precision assembly, and optimization.

**Fangfang Liu** received the B.S. and Ph.D. degrees from the Zhejiang University, Hangzhou, China, in 2006 and 2012, respectively, both in mechanical electronics.

She is currently an Associate Professor with the Research Center of Precision Sensing and Control, Institute of Automation, Chinese Academy of Sciences, Beijing, China. Her research interests include precision mechanical design and mechatronic control.

**Song Liu** received the B.Sc. degree in sensing technology and instrumentation from the Shandong University, Jinan, China, in 2012, and the Ph.D. degree in control science and engineering from the Institute of Automation, Chinese Academy of Sciences, Beijing China and the Department of Mechanical and Biomedical Engineering, City University of Hong Kong, Hong Kong, in 2017.

He is currently a Postdoctoral Fellow with the Robot Vision Research Laboratory, City University of Hong Kong. His current research interests include visual measurement, visual servo control, microassembly, and micromanipulation.

**De Xu** (M'05–SM'09) received the B.S. and M.S. degrees from the Shandong University of Technology, Jinan, China, in 1985 and 1990, respectively, and the Ph.D. degree from the Zhejiang University, Hangzhou, China, in 2001, all in control science and engineering.

Since 2001, he has been with the Institute of Automation, Chinese Academy of Sciences, Beijing, China, where he is currently a Professor with the Research Center of Precision Sensing and Control. His research interests include robotics and automation, in particular, the control of robots, such as visual control and intelligent control.