Fast Ellipse Detection via Gradient Information for Robotic Manipulation of Cylindrical Objects

Huixu Dong*, Guangbin Sun,Wee-Ching Pang,Ehsan Asadi,Dilip K.Prasad,I-Ming Chen,*Fellow,IEEE*

***Abstract*—Robotic manipulation of objects requires a fast recognition from image stream. For many cylindrical object (e.g. cans, cups, pipes, bottles, etc.) this is possible through detection of ellipse depicting the circular top of the cylinder. Growing industrial and warehouse applications of robots drive the demand for fast and reliable detection of ellipses, while state-of-the-art methods are lacking in either speed or accuracy strength. We present a novel algorithm to perform fast and robust ellipse detection. First, the method utilizes the information of edge curvature to split curves into arcs. Next, the arc convexity-concavity is used to classify arcs into different quadrants of ellipses. Then based on multiple geometric constraints the arcs can be grouped at low computational cost. Our method is compared with six state-of-the-art methods using three public image datasets. The comparison results show that the proposed algorithm outperforms other methods with high detection accuracy and fast detection speed. Lastly, the algorithm is applied to identifying cylindrical objects in real-time for arranging and tracking purposes.**

***Index Terms*—Grasping, Perception for Grasping and Manipulation, RGB-D Perception, Elliptic tracking, Gradient information.**

I. INTRODUCTION

ROBOTS are used for grasping objects in the everyday life [1, 2] via the perception system. Cylindrical objects are common man-made geometric volumes. Thus, perceiving ellipses (circles from different views) is needed for a robot. Ellipse detection has been applied in robotics to obtain the coordinates of cylindrical objects for object picking [3-6] and manipulation, as shown in Fig. 1. While the possibility of detecting cylindrical objects through ellipse detection in images is obvious, it is important to detect ellipses accurately and fast for this purpose. Further, although the problem of detecting ellipses seems simple and obvious, in practice, it is challenging to perform ellipse detection in real complex scenes under severe time constraints. In real scenes, there may be multiple elliptical objects, and other objects can partially occlude these elliptical objects.

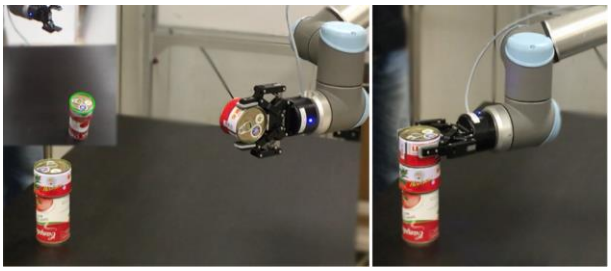

Fig. 1. Robotic manipulation (picking and stacking) of multiple cans through deploying ellipse detection in a real-time scenario.

Hough Transform (HT) and its variants [7, 8] were initially used for detecting shapes including ellipses by planar sets of points. However, the voting process of variants of HT makes them prohibitively slow. Some fast ways are possible by decomposition of parameter space, i.e., using methods for estimating some parameters (often the center coordinates) of ellipses and using HT for the remaining parameters only. The previously mentioned papers on robotics applications [1, 6] use direct fitting techniques to obtain the ellipse position, however with high computational costs and slow execution. Liu *et al*. [3] use a Quasi-Random Sample Consensus ellipse detection algorithm to recognize the cylindrical objects. However, this method is just effective for ellipses with a simple background. In fact, in robotic applications, the most likely condition is that many cylindrical objects of the same shape are stacked randomly in a container or a bin. In such scenario, a target occluded by other mechanical parts is not recognized.

The computational cost associated with ellipse detection can be reduced by using a selected subset of curves for ellipse fitting, irrespective of whether HT or other methods are used for determining ellipse parameters. Thus, arc selection is the most common approach used by modern ellipse detection methods. Arc selection is often performed through geometric criteria satisfied by ellipses' boundaries. For example, in several ellipse detection methods [9-14], short straight lines are detected to approximate arc segments, and these arcs segments are accumulated and merged into ellipses. Some

Manuscript received: February 3, 2018; Revised April 11,2018; Accepted May 4, 2018.

This paper was recommended for publication by Editor Francois Chaumette upon evaluation of the Associate Editor and Reviewers' comments.

Huixu Dong* (The corresponding author) is with Robotics Research Centre, Nanyang Technological University, 639798 Singapore (e-mail:dong0076@e.ntu.edu.sg).

Guangbin Sun is with Robotics Research Centre, Nanyang Technological University, 639798 Singapore (e-mail:guangbin.sun@gmail.com).

Wee-Ching Pang is with Robotics Research Centre, Nanyang Technological University, 639798 Singapore (e-mail: weeching@ntu.edu.sg).

Ehsan Asidi is with Robotics Research Centre, Nanyang Technological University, 639798 Singapore (e-mail: asadi.ehsan@yahoo.com).

Dilip K.Prasad is with School of Computer Engineering, Nanyang Technological University,639798 Singapore (e-mail:dilipprasad@gmail.com).

I-Ming Chen is with Robotics Research Centre, Nanyang Technological University, 639798 Singapore (e-mail: michen@ntu.edu.sg).

Digital Object Identifier (DOI) : see top of this page.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2018.2836428, IEEE Robotics and Automation Letters

works [13, 15-17] have attempted to improve the detection accuracy with iterative approaches. Fornaciari *et al*. [18], Bai *et al*. [19] and Jia *et al*. [20] include the property of elliptical concavity to group arcs for detecting ellipses. However, the aforementioned methods are incapable of detecting ellipses accurately in real-time scenarios with complex backgrounds.

In this paper, we present a novel algorithm to perform fast and accurate ellipse detection. The algorithm is suitable for processing real-time real-scenario imagery in robotic applications and real-world images with complex background. The algorithm involves three steps. First, the arcs that may consist of elliptic candidates are segmented based on the change of arc curvatures. Subsequently, the arcs are classified according to the pixel gradient and the arc convexity-concavity. Finally, further geometric approaches with low computational costs are applied to validating the detected ellipses. Two experiments are conducted as well to assess the method performance in robotic manipulation.

The layout of the paper is as follows. Arc segmentation and classification are discussed in Section II. Detection and validation of ellipses are explored in Section III. Section IV presents the threshold determination and performance comparisons. Experiments of robotic manipulations are described in Section V. The paper is concluded in Section VI.

II. Arc Segmentation and Classification

A. Pre-processing Stage

Canny edge detector with auto-thresholding is applied to obtaining the edge image from an input image, as shown in Fig. 2(B). The coordinates and gradient at an edge pixel p_i are expressed as $\{x_i, y_i, \eta_i\}$, respectively. Since η_i cannot be calculated accurately in digital images, we use only the orientation of the gradient, denoted by its sign, rather than the actual value of η_i. The gradient sign function $\mathrm{X}(p_i)$ at the pixel p_i is defined as follows,

$$\mathrm{X}(p_i) = \begin{cases} +, if\ \ tan(\eta_i) > 0 \\ -, if\ \ tan(\eta_i) < 0 \end{cases}. \quad (1)$$

The edge pixels corresponding to horizontal and vertical gradients, whose gradient signs are undefined, are discarded. We define $Q(e_k)$ as the direction of the arc that lies in a quadrant. Consequently, the arcs of the positive gradient direction rest on the first or third quadrants ($Q(e_k) \in \{\mathrm{I} \cup \mathrm{III}\}$) while the arcs of the negative gradient direction belong to the second or fourth quadrants ($Q(e_k) \in \{\mathrm{II} \cup \mathrm{IV}\}$) as illustrated in Fig. 2(C, D). I, II, III and IV represent the number of four quadrants. We note here that this property shall be used in the section II(C) again for arc classification.

B. Arc Segmentation Process

Since the curvature change over an elliptic arc is smooth, we obtain smooth arcs by splitting at critical pixels including turning corners and inflexion points, as described below. A turning corner is a point where there is a large change of curvature, and an inflexion point is a point where the direction of curvature changes, as shown in Fig. 3. An edge curve c is fitted by a series of the line segments $\{l_1, l_2, \dots, l_N\}$ by using

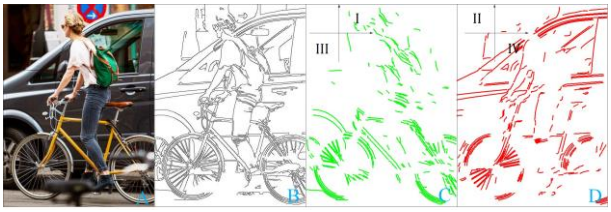

Fig. 2. The initial image(A); The edge image(B); Curves resting on the first or third quadrants (C) and the second or fourth quadrants(D).

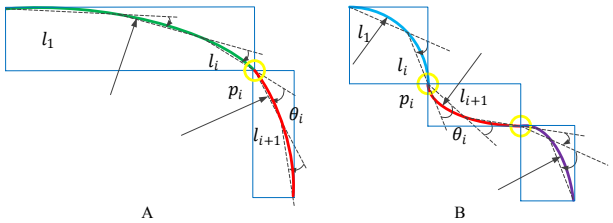

Fig. 3. The curves with the large change of curvature (A) and the change of direction of curvature (B). Specifically, l_i represents the line segment fitting a curve($i = 1,2,3,\cdots$). θ_i denotes the intersection angle between the line segment l_i and l_{i+1}. p_i is a turning corner(A) or an inflexion point (B).

Algorithm I. The Arc Segmentation Algorithm

Arc segmented at turning corners and inflexion points

Input: T_θ , curves: A
Output: the arc vector: C
For $i = 1$ to the size of A (i.e. number of curves), **do**
  Fit the i^{th} curve ($A[i]$) by line segments, store in L;
  Calculate angles θ formed by consecutive line segments in L;
  For $j = 1$ to the number of line segments in L, **do**
    If ($abs(\theta_1\text{-}\theta_j) > T_\theta$)
      For $m = 1$ to the length of $A[i]$ (i.e. size of the i^{th} curve) **do**
        If ($L[j].x == A[i][m].x$ *AND* $L[j].y == A[i][m].y$)
          Reserve $A[i][m]$ in B;
  For $k = 1$ to the size of B **do**
    Create a new empty array $C[k]$, corresponding to a new arc
    For $n = B[k-1]$ to $B[k]$ **do**
      Reserve $A[i][n]$ in $C[k]$.

Note: B contains the critical points, i.e. the points at which the curve $A[i]$ should be split into arcs.

RDP algorithm [21] with a tolerance threshold T_t so that the curve c are denoted as $c: \{l_1, l_2, \dots, l_N\}$. Approximation of a curve by line segments reduces the computational cost since the subsequent calculation is performed on the endpoints of the line segments only instead of all the pixels. The vector angles between the pairs of the consecutive line segments from l_i to l_{i+1} are defined as $\{\theta_1, \theta_2, \dots, \theta_{N-1}\}$, which lie in the range of $-\pi$ and π. We set a threshold (T_θ) to evaluate the amount of the change between the first angle θ_1 and the i^{th} angle θ_i (Fig. 3). Thus, the critical pixels are determined via the following constraint,

$$|\theta_1 - \theta_i| > T_\theta.$$

Big difference between θ_1 and θ_i when the first angle θ_1 and the i^{th} angle θ_i have the same signs implies that the pixels are turning corners. The change of the sign of θ_i relative to θ_1 indicates the change of the direction of the curvature. In this case, such points P_i are the inflexion points. The arc segmentation algorithm, via determining the turning corners

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2018.2836428, IEEE Robotics and Automation Letters

and inflexion points, is provided in Algorithm I. Notes that curves A represents a set of the curves that are chosen after the pre-processing stage, point vector B stores critical points (turning corners and inflexion points), and the output arc vector C reserves the arcs from the broken curves that are broken at the critical points.

Next, we use an oriented bounding box (OBB) to enclose a smooth arc derived as above. If the ratio of the long side to the short side of OBB is bigger than a threshold (T_o), the curvature of the arc is considered as a line segment to be discarded. The remaining curves are deemed as smooth arcs that may represent ellipses and used in the subsequent steps.

C. Arc Classification

Each smooth arc is further classified into two classes based on the convexity-concavity, as we define below. If the midpoint position of the arc is higher than the midpoint position of a line segment formed by the endpoints of an arc, the arc is convex, otherwise it is assigned to a concave arc, as shown in Fig. 4. The convexity-concavity of the arc e is represented by the function $\Omega(e)$ as follows

$$\Omega(e) = \begin{cases} +, & convex; \\ -, & concave. \end{cases} \tag{2}$$

When the positions of the midpoints G and Q are the same as each other, the arc e cannot be considered as a convex or concave arc, thus is rejected.

Based on the functions $\mathrm{X}(P)$ and $\Omega(e)$, each arc e can be classified to the specific quadrant as follows,

$$\Psi = \begin{cases} \mathrm{I} & \langle \mathrm{X}(P), \Omega(e) \rangle = \langle +, + \rangle \\ \mathrm{II} & \langle \mathrm{X}(P), \Omega(e) \rangle = \langle -, + \rangle \\ \mathrm{III} & \langle \mathrm{X}(P), \Omega(e) \rangle = \langle -, - \rangle \\ \mathrm{IV} & \langle \mathrm{X}(P), \Omega(e) \rangle = \langle +, - \rangle \end{cases} \tag{3}$$

where Ψ represents the quadrants where arcs rest. For example, consider an arc that belongs to $\mathrm{I} \cup \mathrm{III}$ according to the property of $\mathrm{X}(P)$and P represents the set of pixels(e.t.,p_i), it can be classed to the first quadrant (I) if $\Omega(e)$ is positive otherwise ($\Omega(e) < 0$) falls in the third quadrant (III). Thus, all the arcs are classed to the four quadrants depending on Eq.(3) (see Fig. 5). The quadrant assigned after this classification simply means that if an arc would belong to an ellipse, it would lie in the quadrant as identified above.

D. Arc Grouping Based on Geometric Constraints

After classing arcs, we choose a triplet of arcs from the four different quadrants to consider if the triplet can constitute an ellipse. There are four combinations for a triplet of arcs lying in the four quadrants,(e_a, e_b, e_c) in $(\mathrm{I}, \mathrm{II}, \mathrm{III})$or $(\mathrm{I}, \mathrm{II}, \mathrm{IV})$ or $(\mathrm{II}, \mathrm{III}, \mathrm{IV})$ or$(\mathrm{I}, \mathrm{III}, \mathrm{IV})$, where a, b, c are the indexes of different arcs, as shown in Fig. 5. A triplet $\tau_{abc} = (e_a, e_b, e_c)$ is considered eligible for ellipse detection if it satisfies the below grouping constraints, which indicate that the arcs may belong to the same ellipse.

First, we use the grouping approach based on the relative positions among arcs [20]. With reference to Fig. 5, the following equation is used to describe the constraint on relative positions between two eligible arcs as follows,

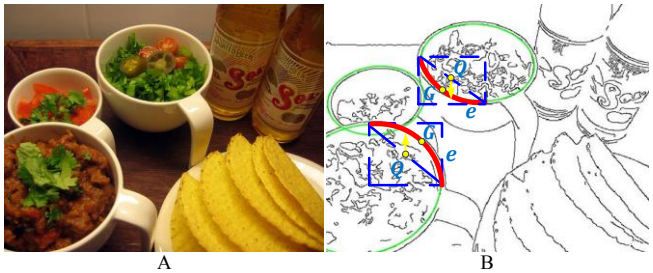

Fig. 4. A convex arc (B-top) and a concave arc (B-bottom).Q represents the midpoint of the line segment formed by two endpoints of the red arc e and G denotes the midpoint of the red arc e. The blue frame represents the bounding box.

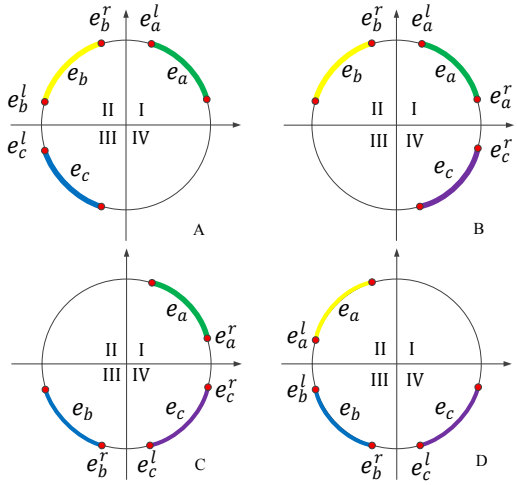

Fig. 5. The four configurations for a triple of arcs. Arcs with different colors are classed into the four quadrants. ${e_i}^l$ and ${e_i}^r$ indicates the left and right endpoints of an arc and i represents a, b,or c.

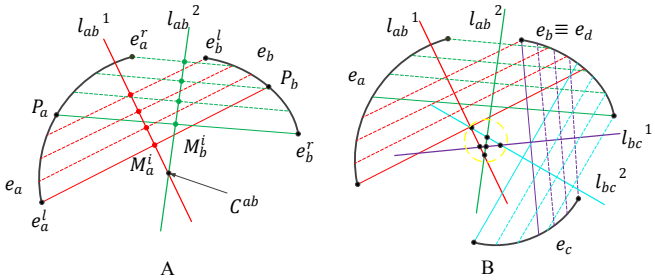

Fig. 6. Estimations of centers of a pair of arcs (A) and a triplet of arcs (B). C^{ab} donates the intersection of the lines ${l_{ab}}^1$and ${l_{ab}}^2$. ${l_{ab}}^1$, ${l_{ab}}^2$ and ${l_{bc}}^1$, ${l_{bc}}^2$ represent the lines through the midpoints of parallel chords formed by the arc pairs (a_a, a_b) and (a_b, a_c), respectively; the black solid points surrounded by a yellow circle are the intersections of lines through the midpoints of parallel chords.

$$\begin{cases} e_a^l(x) > e_b^r(x), e_b^l(y) > e_c^r(y), if\ (e_a, e_b, e_c) \text{in } (\mathrm{I}, \mathrm{II}, \mathrm{III}); \\ e_a^l(x) > e_b^r(x), e_a^r(y) > e_c^r(y), if\ (e_a, e_b, e_c) \text{in } (\mathrm{I}, \mathrm{II}, \mathrm{III}); \\ e_a^r(y) > e_c^r(y), e_c^l(x) > e_b^r(x), if\ (e_a, e_b, e_c) \text{in } (\mathrm{I}, \mathrm{II}, \mathrm{III}); \\ e_a^l(y) > e_b^l(y), e_c^l(x) > e_b^r(x), if\ (e_a, e_b, e_c) \text{in } (\mathrm{I}, \mathrm{II}, \mathrm{III}), \end{cases} \tag{4}$$

where ${e_i}^l(x)$ and ${e_i}^l(y)$ represent the x-coordinate and y-coordinate of the left endpoint of an arc e_i; ${e_i}^r(x)$ and ${e_i}^r(y)$ indicate the x-coordinate and y-coordinate of the right endpoint of an arc e_i; i represents a, b,or c. As shown in Fig. 5(A), along the horizontal axis, the x-coordinate of ${e_a}^l$ must be bigger than that of ${e_b}^r$. Similarly, along the vertical axis,

the y-coordinate of ${e_b}^l$ must be bigger than that of ${e_c}^l$. We define an eligible pair of arcs $\mathcal{E}_{ab} = (e_a, e_b)$. Thus, an eligible triplet is defined as two pairs sharing a common arc such as $\tau_{abc} = \{(\mathcal{E}_{ab}, \mathcal{E}_{cd}) | \, e_b \equiv e_d \}$.

Second, the locations of centers of arcs in a triplet are also considered as an additional constraint for selecting potential arcs. If the centers of arcs in a triplet rest on the same area within a range, it is highly likely that they fit the same ellipse.

III. Detection and Validation of Ellipses

A. Determination of Ellipse Center

The ellipse centre for a pair of given arcs can be calculated by a geometric property of ellipses: the midpoints of parallel chords are collinear and the intersection of lines through midpoints of parallel chords is the ellipse centre [22]. We use the average value of all the intersections of lines through midpoints of parallel chords as the ellipse center.

We obtain two sets of chords that are parallel to the lines $e_a^l P_b$ and $e_b^r P_a$ (P_a and P_b being the midpoints of arcs e_a and e_b, respectively), as shown in Fig. 6(A). The number T_n of parallel chords in each group is set as 16 based on our experiments, which are discussed later in section IV. $M_a^i$and M_b^i are the midpoints of the two sets of parallel chords, respectively. We calculate the slopes (t_1, t_2) of the lines $(l_{ab}^1,\ l_{ab}^2)$ through $M_a^i$and M_b^i. $i = 1,2,\ldots,16$. The intersection of two lines $(l_{ab}^1,\ l_{ab}^2)$ is the centre $(C.x, C.y)$. Thus, the centre coordinates are estimated as follows,

$$C.x = \frac{M_b.y - t_2 \times M_b.x - M_a.y + t_1 \times M_a.x}{t_2 - t_1},$$
$$C.y = \frac{t_1 \times M_b.y - t_2 \times M_a.y + t_1 t_2 (M_a.x - M_b.x)}{t_2 - t_1}. \quad (5)$$

The average values of 16 groups of $\left(M_a^i, M_b^i\right)$ are (M_a, M_b) for a pair of arcs. To improve its robustness to noises, we use four lines through the midpoints to yield six intersections for a triplet of arcs in Fig. 6(B). The average values of the coordinates of all the intersections is adopted as the ellipse centre that are further used for estimating other parameters. We consider that a triplet of arcs $\tau_{abc} = \{(\mathcal{E}_{ab}, \mathcal{E}_{cd}) | \, e_b \equiv e_d\}$ can consist of the same ellipse if and only if the distance of the centres of two pairs (e_a, e_b) and (e_c, e_b) lie within a given threshold (T_c) . Here we use 4 sets of parallel chords to calculate the elliptic centre for $\mathcal{E}_{ab}$ and $\mathcal{E}_{cd}$. Theoretically speaking, 6 sets of parallel chords are available for computing the ellipse centres. In experiments, we found the methods of determining the centre of an ellipse with 6 and 4 sets of chords have almost the same effects on the detection accuracy; however, the method with 6 sets of chords costs more time than that of 4 sets of chords. This is illustrated in Table I, which reports experimental results.

B. Parameter Estimations

In order to obtain the remaining parameters, we decompose the parameter space of an ellipse for the ratio N of the ellipse minor semi-axes length B to major semi-axes length A and other defined parameter K ($K = \tan\rho$, ρ is the orientation of an ellipse), as described in [18, 23-25]. The equation that expresses N in terms of K is provided as

$$N^2 = -\frac{(q_1 - K)(q_2 - K)}{(1 + q_1 K)(1 + q_2 K)}, \quad (6)$$

with $\alpha = q_1 q_2 - q_3 q_4$ and $\beta = q_2 q_4 (q_3 - q_1) + q_1 q_3 (q_4 - q_2) + (q_1 + q_2 - q_3 - q_4)$. q_1 and q_3 represent the slopes of the parallel chords of a triple of arcs respectively. q_2 and q_4 denote the slopes of the lines fitting the midpoints of parallel chords. q_1, q_2, q_3 and q_4 include two slopes for each pair of arcs, as shown in Fig. 6(B). Thus,

$$K = \pm\sqrt{1 - \frac{\beta}{\alpha}}. \quad (7)$$

For a pair of points found, the $N - K$ accumulator is continuously updated according to Eq. (6), we can get the highest peaks of the values of N and ρ are in two 1D accumulators. Here the ellipse polar equations and coordinate transformation relations are applied to transforming the coordinate (x_i, y_i) of a point on an arc in the world coordinate system to the coordinate (x_o, y_o) in the ellipse coordinate system. In the two dimensions, the coordinate transformation is done by the following way,

$$\begin{bmatrix} x_o \\ y_o \end{bmatrix} = \begin{bmatrix} \cos\rho & \sin\rho \\ -\sin\rho & \cos\rho \end{bmatrix} \begin{bmatrix} x_i - x_c \\ y_i - y_c \end{bmatrix}. \quad (8)$$

According to Eq. (7-8), we obtain the following equations,

$$x_o = \frac{(x_i - x_c) + (y_i - y_c)K}{\sqrt{K^2 + 1}}, y_o = \frac{(y_i - y_c) - (x_i - x_c)K}{\sqrt{K^2 + 1}}.$$

A_x is estimated as follows [23],

$$A_x = \sqrt{\frac{x_o^2 N^2 + y_o^2}{N^2 (1 + K^2)}}. \quad (9)$$

Finally, combining $A_x = A\cos\rho$ and $N = \frac{B}{A}$, we can identify an ellipse candidate with a parameter set (x_c, y_c, N, K, A).

C. Ellipse Validation

The arcs that satisfy the three constraints above still may consist of an invalid ellipse, such as false positives or duplicated ones. Therefore, an ellipse validation is essential to discard false detections, such as false positives, by identification of invalid ellipses detected by our method.

1) Validation by the ratio of the length

The length ratio of the sum of the long and width lengths of bounding box enclosing an arc to the sum of the major and minor semi-axes lengths $(A, B$ respectively) can be regarded as a measurement of circumference, as shown in Fig. 7. Such scheme is less sensitive to the quantization problem in the pixel count feature. Assuming that an ellipse is fit by the triplet $\tau_{123} = (e_1, e_2, e_3)$ enclosed by bounding boxes with the size of $m_i \times n_i (i = 1,2,3)$, we define a function $\mathrm{K}(\tau)$ as follows,

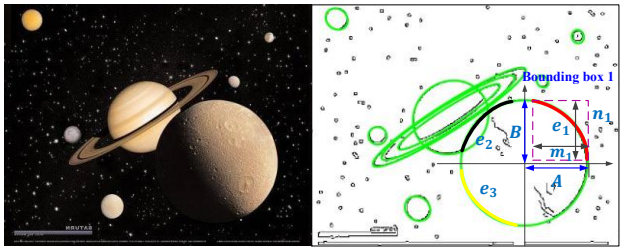

Fig. 7. Validation by the ratio of the length is illustrated here. The purple frame represents the bounding box.

$$\mathrm{K}(\tau) = \frac{\sum_{i=1}^{3}(m_i+n_i)}{3(A+B)}. \quad (10)$$

A higher value of $\mathrm{K}(\tau)$ implies a larger probability of a candidate ellipse being a real one. If $\mathrm{K}(\tau)$ exceeds a threshold (T_l), the ellipse candidate can be verified further by the below constraint, otherwise it is discarded. We define a score $s(\mathrm{K}(\tau))$, which is equivalent to $\mathrm{K}(\tau)$, to represent the probability of a set of arcs consisting of a real ellipse.

2) Clustering by similarities among detected ellipses

Multiple candidate ellipses may be fit to the same 'real ellipse' as multiple arc sets may belong to the same 'real ellipse'. In order to ensure that a "real ellipse" is just made up of one arc set, such candidate ellipses are clustered depending on the order of the scores [12]. All valid ellipses with the same centre are ranked according to the decreasing score $\left(s(\mathrm{K}(\tau))\right)$. Then we use the method proposed by Bascca *et al.*[26] to assess the similarity of two ellipses by comparing the differences of ellipse parameters. Specifically, a feature vector is defined as $V(x_c, y_c, A, B, \rho)$ where (x_c, y_c) represents the coordinates of an ellipse center, A and B denote the lengths of the semi-major and semi-minor axes, respectively, and ρ is the orientation of ellipse. The Euclidean distance between two feature vectors is considered as the distinctiveness measure:

$$D(V,W) = \sqrt{\sum_{i=1}^{5}\left(E_{V,i} - E_{W,i}\right)^2}, \quad (11)$$

where $E_{V,i}$ and $E_{W,i}$ denotes the i^{th} parameters in the vectors V and W,respectively. If $D(V,W)$ is less than a threshold (T_s),the ellipses V and W are concluded to belong to the same ellipse cluster. Otherwise, the corresponding ellipse is regarded similar to the reference one and discarded. Thus, when an ellipse cannot be assigned to any cluster, it becomes a reference for a new cluster. We adopt $T_s = 20$ given in [26].

IV. Threshold Determination and Performance Comparison

We have used three publicly available datasets for evaluating the performances of the ellipse detection approaches, namely, Dataset Prasad[12] , Dataset #1 and dataset #2[18]. Dataset Prasad is composed of 198 images with the number of small ellipses and still available online. Dataset#1 consists of 400 high-definition image with a cylindrical object and lower-resolution images containing many objects with complex scenes. Dataset #2 is made up of several videos taken by a cell phone in scenes with many lighting conditions. We demonstrate the performance of our method, comparing against six state-of-the-art ellipse detectors proposed by Jia *et al.*[20], Fornaciari *et al.*[18], Prasad *et al.*[12], Mai *et al.*[13], Bai *et al.*[19], Liu *et al.*[27]. Source codes of these methods in C++ or MATLAB are available online. Default parameters are used for these methods. All the experiments are performed on a PC with 8GB RAM and an Intel Core i7 processor.

A. Evaluation Metrics Regarding Algorithm Performance

Performance metrics, such as F-measure and the execution time t proposed in [12], are used for quantitative comparisons. For a detected ellipse ε_d and the ground-truth ellipse ε_g, the overlap ratio Φ is the Jaccard index of similarity between the ellipses ε_d and ε_g. If the overlap ratio Φ satisfies $\Phi > \Phi_0$ with $\Phi_0 = 0.8$ for three public datasets, the detected ellipse is considered a correct detection, otherwise, it is counted as a miss. As a result, the $Precision$ and $Recall$ values are computed as follows:

$$Precision = \frac{\Omega}{\mathrm{N}},\ Recall = \frac{\Omega}{\mathrm{M}}, \quad (12)$$

where Ω denotes the number of correctly detected ellipses, N is the number of detected ellipses, and M is the number of

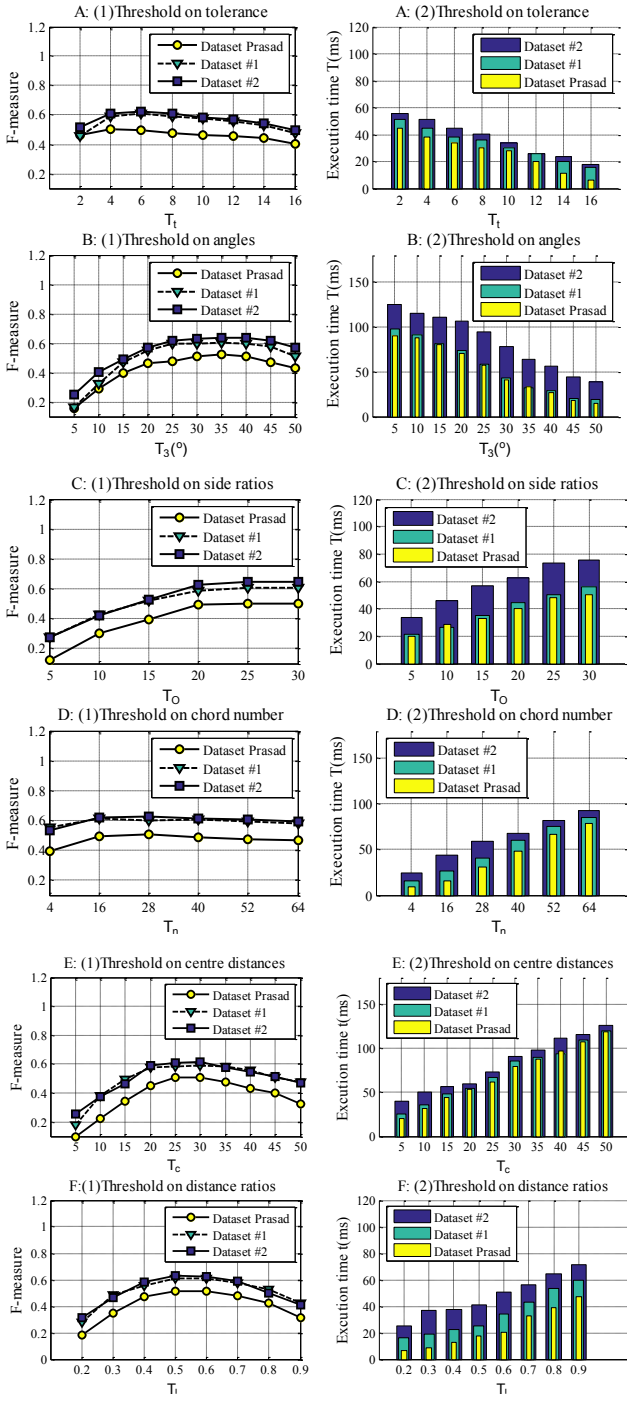

Fig. 8. Threshold determinations. A:(1-2), B:(1-2), C:(1-2), D:(1-2), E:(1-2) and F:(1-2) represent the fluctuations of F-measure and consuming time with the parameters T_t, T_θ, T_o, T_n, T_c, T_l changing, respectively.

ground-truth ellipses. Based on Eq. (10), F-measure is obtained as follows,

$$F\text{-measure} = \frac{2\times Precision\times Recall}{Precision+Recall}. \quad (13)$$

B. Threshold Determination

The performance of ellipse detector depends on the choices of threshold parameters, namely $T_t, T_\theta, T_o, T_n, T_c, T_l$. However, we cannot ensure suitability of a chosen threshold in all images for achieving the best performance of ellipse detection. The parameters are tested on three datasets detailed above. Thus, we choose a value of threshold that indicates the best detection performance in terms of $F\text{-}measure$ and the execution time (t) by tuning a threshold parameter while fixing other threshold parameters, as done in Fig. 8. Figure 8(A) shows the influence of T_t on the performance. The detection effectiveness decreases with T_t greater than 6. The consuming time increases with decreasing T_t. We set $T_t = 6$ $(T_\theta = 25^o, T_o = 25, T_n = 12, T_c = 20, T_l = 0.5)$ as a good trade-off. As observed in Fig. 8(B), the best value of T_θ is $35^o (T_t = 6, T_o = 25, T_n = 12, T_c = 20, T_l = 0.5)$ since the execution time monotonically decreases with increasing values of T_θ. The effect of T_o on the effectiveness and execution time are shown in Fig. 8(C), where it is obvious that the detection effectiveness saturates while the execution time consistently increases when T_o is larger than 20. Thus, we choose 20 as the value of $T_o (T_t = 6, T_\theta = 35^o, T_n = 12, T_c = 20, T_l = 0.5)$. Figure 8 (D) illustrates that the parameter T_n does not need to be larger than 16 for reaching the top performance. The best trade-off between accuracy and speed is obtained with $T_n = 16$ $(T_t = 6, T_\theta = 35^o, T_o = 20, T_c = 20, T_l = 0.5)$,which allows to avoid unnecessary computation. As clearly illustrated in Fig. 8(E), the best values of T_c are 25 and 30 for the detection effectiveness, but the execution time of the value 30 is more time demanding than the value 25. We choose $T_c = 25$ as a trade-off to gain good effectiveness and keep execution time at the minimum $(T_t = 6, T_\theta = 35^o, T_o = 20, T_n = 16, T_l = 0.5)$. For T_l larger than 0.6, there is a significant decline in the detection effectiveness and the computational time also decreases. We set $T_l = 0.5$ $(T_t = 6, T_\theta = 35^o, T_o = 20, T_n = 16, T_c = 25)$ because this value can guarantee the best effectiveness and an acceptable execution time compared with $T_l = 0.4$, as shown in Fig. 8(F).

C. Performance Evaluations

Our method is compared with six state-of-the-art methods using three public image datasets and the detection performances are depicted in Table I and some practical cases are shown in Fig. 9. For Dataset Prasad, Prasad's detector outperforms all other methods for only their own dataset in terms of Fmeasure (0.7548) and the proposed method is in the second place. The performance of our algorithm is the best in Dataset #1 and Dataset #2. Specifically, the presented ellipse detector is superior to these of Fornaciari *et al* and Jia *et al*, especially in detecting small, occluded and over-lapping ellipses. Specifically, the presented ellipse detector is superior to these of Fornaciari *et al* and Jia *et al*, especially in detecting small, occluded and over-lapping ellipses. Our approach requirement is around 0.1% computation time of Prasad's and is third only to Fornaciari's and Jia's approaches in terms of the average detection time. The execution time of our detector can work in real-time video imagery for practical scenarios.

TABLE I. PERFORMANCE RESULTS FOR THREE DATASETS

		F-measure	*Time(ms)*	*Precision*	*Recall*
Dataset Prasad	Mai 2008	0.2535	962.2	0.4135	0.2269
	Liu2009	0.0950	1049	0.0700	0.1505
	Bai 2009	0.2395	3470	0.2195	0.2890
	Prasad 2012	0.7428	8300	0.8512	0.6813
	Fornaciari 2014	0.3661	12.61	0.8541	0.2330
	Jia 2017	0.4332	8.42	0.7390	0.3064
	Ours	**0.5173**	**12.96**	**0.6142**	**0.5173**
	Ours[1]	**0.5154**	**20.32**	**0.6212**	**0.4404**
Dataset #1	Mai 2008	0.2604	1979.5	0.3299	0.2463
	Liu2009	0.1170	3960	0.1248	0.1415
	Bai 2009	0.2121	136085	0.2420	0.1909
	Prasad 2012	0.4512	17130	0.4425	0.4610
	Fornaciari 2014	0.5716	16.55	0.7117	0.4777
	Jia 2017	0.5733	12.61	0.6161	0.5361
	Ours	**0.6132**	**24.63**	**0.7647**	**0.5118**
	Ours[1]	**0.6175**	**31.56**	**0.7583**	**0.5208**
Dataset #2	Mai 2008	0.1003	3520	0.1159	0.0978
	Liu2009	0.0703	9720	0.0570	0.1324
	Bai 2009	0.0370	631930	0.0249	0.1086
	Prasad 2012	0.4478	14470	0.4792	0.4750
	Fornaciari 2014	0.5880	37.725	0.5900	0.5862
	Jia 2017	0.5423	30.608	0.4846	0.6155
	Ours	**0.6323**	**41.528**	**0.6547**	**0.6114**
	Ours[1]	**0.6296**	**50.864**	**0.6382**	**0.6213**

[1]Note: Ours[1] represents the elliptic detector with 6 sets of parallel chords for three combinations of pairs of arcs.

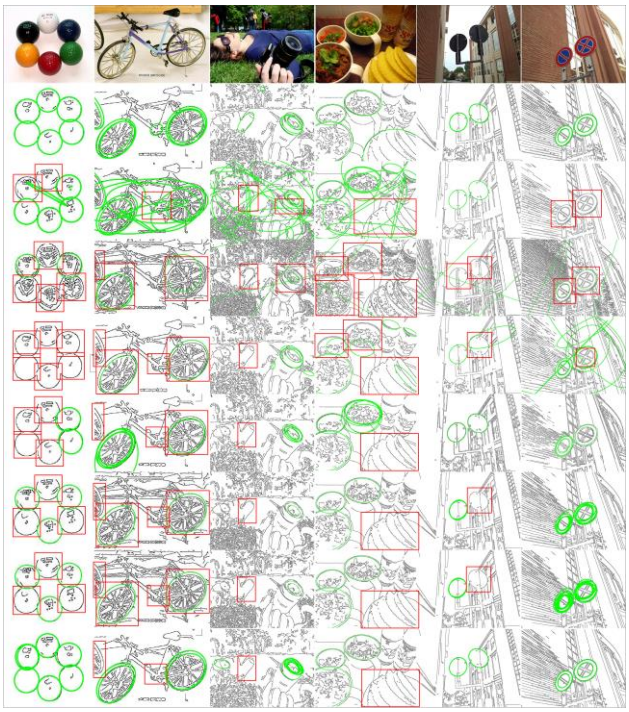

Fig. 9. Detection examples of the real images. From the first row to the last row, the results correspond to 1-ground truth, 2-Mai (2008), 3-Liu(2009), 4-Bai(2009), 5-Prasad(2012), 6-Fornaciari(2014), 7-Jia(2017) and 8-our detector, respectively. The original images are from Dataset Prasad (the first two images), Dataset #1(the middle two images) and Dataset #2(the last two images). The ellipse cases that cannot be detected are enclosed by the red rectangular frames.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2018.2836428, IEEE Robotics and Automation Letters

Here, we discuss the aspects that give our method a competitive advantage over other methods. Since, we split curve into arcs based on the change in the arc curvatures and the arc convexity-concavity, our method presents a better performance in detecting occluded or overlapping ellipses. Although arc extraction and arc grouping cost additional computations, they are effective in reducing computations in the subsequent steps. Consequently, the detection process can be real-time. Prasad's detector has an apparent disadvantage in the execution time since they explore matching all arcs within the search region of an arc to form a set of arcs that potentially belong to the same ellipse. In this sense, it takes an exhaustive approach, which results in high computational costs and limited applicability for real-time or video rate ellipse detections. The approach of Fornaciari's and Jia's methods to classify arcs into quadrants by the arc convexity-concavity based on sizes of areas is not suitable for small arcs. This results in poor performance of these methods in detecting overlapping or occluded ellipses.

D. Limitations of the proposed detector

We evaluate the limitations of the proposed detector by the dataset [15] of synthetic images (300×300) with occluded ellipses and small ellipses(Fig. 10). The proposed algorithm imposes the constraints in the scenarios where the server occlusions result in the lack of visibility. We classify arcs into quadrants and a set of arcs consisting of an ellipse must rest on at least three different quadrants. Consequently, highly occluded ellipses and semi-ellipses cannot be detected. Moreover, we use parallel chords to estimate centres of ellipses. This is inaccurate for short arcs, leading to the failures of detecting the ellipses of minor axis with less than around 15 pixels.

V. EXPERIMENTS IN ROBOTIC MANIPULATIONS

Two experiments are performed to illustrate the capabilities of the proposed detector for robotic manipulation of cylindrical object. We deploy a test platform including a 6-DOF arm, a 2-fingered Robotiq gripper, and a Microsoft Kinect camera mounted on a Pan-Tilt tripod next to a table.

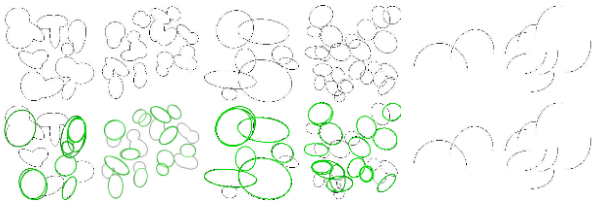

Fig. 10. The examples of detecting occluded, small, semi ellipses in synthetic images.

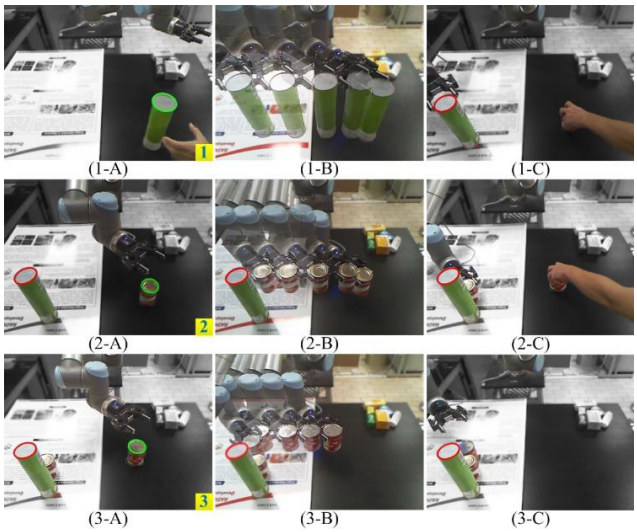

Fig. 11. Arranging cans in a dynamic scenario. Place the first can (1-A, B, C), the second can (2-A,B,C) and the third can (3-A,B,C).

A. Arranging Cylindrical Objects by Ellipse Detection

The first experiment is to use a robotic system to arrange cylindrical objects delivered by a human to a working station. In particular, the human puts the food cans randomly on the right side of a table while the robot must automatically detect and pick new objects and arrange them on the other end of the table. The proposed method is used for detecting a new cylindrical object and tracking the position of object till it is placed to the left side, as illustrated in Fig. 11. The new

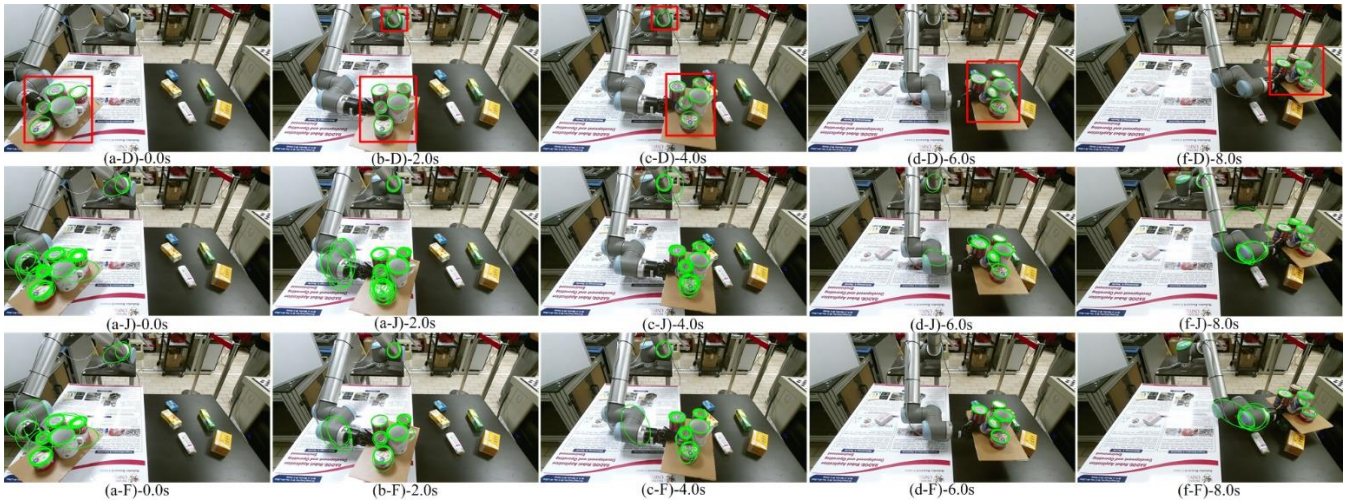

Fig. 12. Snapshots of dynamic detections at for the presented detector (the 1st row), Jia's detector (the 2nd row) and Fornaciari's detector (the 3rd row).

detected ellipse is depicted in green, and the color is changed to red when it is tracked to the other side of the table. The robot must also obtain 3D coordinate information to pick the objects. We convert the 2D pixel coordinates of the ellipse's centroid by the depth information from the RGB-D camera so that the robotic manipulator can move to the location and grasp the object. The motion planning is addressed by OMPL [28] to grasp and to move the object to the destination and finally release it at a position based on a given 3D location. The results indicate that the ellipse detector is fast and robust to automate the execution, the monitoring of the workflow and termination of the task.

B. *Tracking Cylindrical Objects by Ellipse Detection*

In the second experiment, we evaluate the performance of the presented detector in a dynamic scenario to track multiple cylindrical objects carried by the robotic arm at a constant speed. Our method is compared with two real-time detectors (Fornaciari 2014 and Jia 2017). The camera's video rate is 30 Hz. Screenshots of the captured video with ellipses detected in real-time by our method, Jia's method, and Fornaciari's method are shown in Fig. 12. Our ellipse detector shows the excellent performance in dynamic scenarios. But the other two detectors cannot discard false ellipses since close ellipses were merged together, which result in excessive detections. Fmeasure, Precision and Recall of our method is 0.7873, 0.9571 and 0.6814, respectively. The detectors proposed by Fornaciari and Jia have worse performances with low Fmeasure (0.2934, 0.3130), Precision (0.3412, 0.3845) and Recall (0.2573, 0.2639), respectively.

VI. Conclusion and Future Work

In this paper, we consider the problem of detecting ellipses in various scenarios, especially for the robotic manipulation of cylindrical objects in real time. Compared with six existing methods, our extensive experiments demonstrate the significant advantages of the proposed method for detecting ellipses in real images with complex backgrounds by concerning a trade-off between the detection effectiveness and detection time. In our robotic experiments, the detector successfully tracks multiple cylindrical objects in real time that enables the robot to recognize and sort cylindrical objects from ones with different shapes in a cluttered environment by detecting ellipses. Future works will focus on further enhancements to detect even smaller ellipses, heavily occluded ellipses cases, and well-shaped semi-ellipses.

References

[1] C. Korpela, M. Orsag, and P. Oh, "Towards valve turning using a dual-arm aerial manipulator," in Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on, 2014, pp. 3411-3416.

[2] H. Dong, E. Asadi, C. Qiu, J. Dai, and I.-M. Chen, "Geometric design optimization of an under-actuated tendon-driven robotic gripper," Robotics and Computer-Integrated Manufacturing, vol. 50, pp. 80-89, 2018.

[3] K. Liu, H. Li, and Z. Sun, "Ellipse Detection-Based Bin-Picking Visual Servoing System," in Engineering Creative Design in Robotics and Mechatronics, ed: IGI Global, 2013, pp. 114-121.

[4] J. A. Piepmeier and H. Lipkin, "Uncalibrated eye-in-hand visual servoing," The International Journal of Robotics Research, vol. 22, pp. 805-819, 2003.

[5] K. Ono, T. Ogawa, Y. Maeda, S. Nakatani, G. Nagayasu, R. Shimizu, et al., "Detection, localization and picking up of coil springs from a pile," in Robotics and Automation (ICRA), 2014 IEEE International Conference on, 2014, pp. 3477-3482.

[6] J. Su, Z.-Y. Liu, H. Qiao, and C. Liu, "Pose-estimation and reorientation of pistons for robotic bin-picking," Industrial Robot: An International Journal, vol. 43, pp. 22-32, 2016.

[7] R. O. Duda and P. E. Hart, "Use of the Hough transformation to detect lines and curves in pictures," Communications of the ACM, vol. 15, pp. 11-15, 1972.

[8] L. Xu, E. Oja, and P. Kultanen, "A new curve detection method: randomized Hough transform (RHT)," Pattern recognition letters, vol. 11, pp. 331-338, 1990.

[9] E. Kim, M. Haseyama, and H. Kitajima, "Fast and robust ellipse extraction from complicated images," in Proceedings of IEEE information technology and applications, 2002.

[10] L. Libuda, I. Grothues, and K.-F. Kraiss, "Ellipse detection in digital image data using geometric features," in Advances in Computer Graphics and Computer Vision, ed: Springer, 2007, pp. 229-239.

[11] D. K. Prasad, M. K. Leung, and C. Quek, "ElliFit: An unconstrained, non-iterative, least squares based geometric Ellipse Fitting method," Pattern Recognition, vol. 46, pp. 1449-1465, 2013.

[12] D. K. Prasad, M. K. Leung, and S.-Y. Cho, "Edge curvature and convexity based ellipse detection method," Pattern Recognition, vol. 45, pp. 3204-3221, 2012.

[13] F. Mai, Y. Hung, H. Zhong, and W. Sze, "A hierarchical approach for fast and robust ellipse extraction," Pattern Recognition, vol. 41, pp. 2512-2524, 2008.

[14] H. Dong, D. K. Prasad, and I.-M. Chen, "Accurate detection of ellipses with false detection control at video rates using a gradient analysis," Pattern Recognition, vol. 81, pp. 112-130, 2018.

[15] A. Y.-S. Chia, S. Rahardja, D. Rajan, and M. K. Leung, "A split and merge based ellipse detector with self-correcting capability," IEEE Transactions on Image Processing, vol. 20, pp. 1991-2006, 2011.

[16] T. Lu, W. Hu, C. Liu, and D. Yang, "Effective ellipse detector with polygonal curve and likelihood ratio test," IET Computer Vision, vol. 9, pp. 914-925, 2015.

[17] H. Dong, I.-M. Chen, and D. K. Prasad, "Robust ellipse detection via arc segmentation and classification," in Image Processing (ICIP), 2017 IEEE International Conference on, 2017, pp. 66-70.

[18] M. Fornaciari, A. Prati, and R. Cucchiara, "A fast and effective ellipse detector for embedded vision applications," Pattern Recognition, vol. 47, pp. 3693-3708, 2014.

[19] X. Bai, C. Sun, and F. Zhou, "Splitting touching cells based on concave points and ellipse fitting," Pattern recognition, vol. 42, pp. 2434-2446, 2009.

[20] Q. Jia, X. Fan, Z. Luo, L. Song, and T. Qiu, "A Fast Ellipse Detector Using Projective Invariant Pruning," IEEE Transactions on Image Processing, 2017.

[21] J. E. Hershberger and J. Snoeyink, Speeding up the Douglas-Peucker line-simplification algorithm: University of British Columbia, Department of Computer Science, 1992.

[22] D. S. Barwick, "Very fast best-fit circular and elliptical boundaries by chord data," IEEE transactions on pattern analysis and machine intelligence, vol. 31, pp. 1147-1152, 2009.

[23] A. S. Aguado, M. E. Montiel, and M. S. Nixon, "On using directional information for parameter space decomposition in ellipse detection," Pattern Recognition, vol. 29, pp. 369-381, 1996.

[24] S.-C. Zhang and Z.-Q. Liu, "A robust, real-time ellipse detector," Pattern Recognition, vol. 38, pp. 273-287, 2005.

[25] A. Fernandes, "A correct set of equations for the real-time ellipse hough transform algorithm," Technical Report2009.

[26] C. Basca, M. Talos, and R. Brad, "Randomized Hough transform for ellipse detection with result clustering," in Computer as a Tool, 2005. EUROCON 2005. The International Conference on, 2005, pp. 1397-1400.

[27] Z.-Y. Liu and H. Qiao, "Multiple ellipses detection in noisy environments: A hierarchical approach," Pattern Recognition, vol. 42, pp. 2421-2433, 2009.

[28] I. A. Sucan, M. Moll, and L. E. Kavraki, "The open motion planning library," IEEE Robotics & Automation Magazine, vol. 19, pp. 72-82, 2012.