Acta Astronautica 136 (2017) 101–114

Contents lists available at ScienceDirect

Acta Astronautica

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

# Robust, fast and accurate vision-based localization of a cooperative target used for space robotic arm

Zhuoman Wen[a,b], Yanjie Wang[a], Jun Luo[a,b,*], Arjan Kuijper[c,d], Nan Di[a], Minghe Jin[e]

[a] Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, Changchun 130033, China
[b] University of the Chinese Academy of Sciences, Beijing 100049, China
[c] Fraunhofer Institute for Computer Graphics Research, Darmstadt 64283, Germany
[d] Technische Universität Darmstadt, Darmstadt 64283, Germany
[e] State Key Laboratory of Robotics and System, Harbin Institute of Technology, Harbin 150080, China

ARTICLE INFO

*Keywords:*
Space robotic arm
Visual measurement
Cooperative target
Edge detection
Marker localization
Pose measurement

ABSTRACT

When a space robotic arm deploys a payload, usually the pose between the cooperative target fixed on the payload and the hand-eye camera installed on the arm is calculated in real-time. A high-precision robust visual cooperative target localization method is proposed. Combing a circle, a line and dots as markers, a target that guarantees high detection rates is designed. Given an image, single-pixel-width smooth edges are drawn by a novel linking method. Circles are then quickly extracted using isophotes curvature. Around each circle, a square boundary in a pre-calculated proportion to the circle radius is set. In the boundary, the target is identified if certain numbers of lines exist. Based on the circle, the lines, and the target foreground and background intensities, markers are localized. Finally, the target pose is calculated by the Point-3-Perspective algorithm. The algorithm processes 8 frames per second with the target distance ranging from 0.3m to 1.5 m. It generated high-precision poses of above 97.5% on over 100,000 images regardless of camera background, target pose, illumination and motion blur. At 0.3 m, the rotation and translation errors were less than 0.015° and 0.2 mm. The proposed algorithm is very suitable for real-time visual measurement that requires high precision in aerospace.

## 1. Introduction

Space robotic arms [1,2] play an essential role in outer space. They help reassemble space station, move transfer vehicles, assist astronauts to walk in space, and dock with spacecrafts [3]. In order to capture target, usually, a camera is assembled on the robotic arm to measure the pose between the target and the arm [4–6]. As shown in Fig. 1, a hand-eye camera and an arresting device are installed on the robotic arm [7]; a cooperative target and a to-be-arrested device are fixed on the object. The hand-eye camera has to identify the cooperative target quickly from complex scenes, localize the fiducial markers on the target, calculate the relative pose between the target and the camera based on the marker coordinates, and then transfer the pose to the one between the arresting and to-be-arrested device so that the moving path of the arm can be planned.

It is challenging to guarantee a high identification rate of the cooperative target. The distance between the target and the camera ranges from 0.3m to 1.5 m, and is subject to pitch, yaw and roll. As the pose changes, the target appears very differently in the image. The complex mechanical structure of the to-be-arrested device also interferes with the target identification. To make the robotic arm move as quick as possible, the algorithm must run in (near) real-time. It should also be robust to a small degree of motion blur. Performing in space and open air, some other constraints must be taken into consideration. One is irregular lighting. Lighting intensity may be too weak, too strong, or uneven. The metal on the arm and the to-be-arrested device may generate glittering points in the image. Therefore the difficulty of identification and marker localization are largely increased, especially the latter. Another obstacle is the limited storage and speed of the chips that runs the algorithm. For outer space execution, chips must resist radiation and high-speed particles, have low power supply and endure large temperature range; therefore they usually have lower speed and smaller memory than civil products [8]. As a consequence, algorithms should be relatively simple and the target needs to be provided with visual clues making identification and orientation easier.

We designed a cooperative target using a circle, a line and dots as markers, as shown in Fig. 2. The design simplifies the detection and localization algorithm and guarantees high identification rates. A novel

* Corresponding author at: Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, Changchun 130033, China.
*E-mail address:* luojun604@qq.com (J. Luo).

http://dx.doi.org/10.1016/j.actaastro.2017.03.008
Received 6 October 2016; Accepted 9 March 2017
Available online 12 March 2017

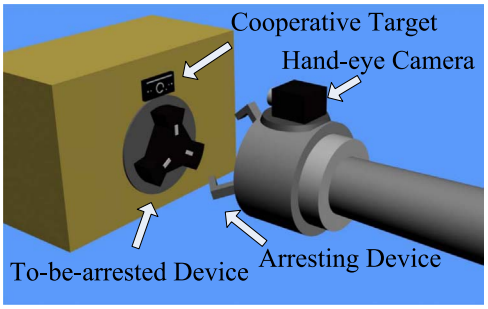


**Fig. 1.** Robotic arm with camera and grabber capturing an object with a target and connection structure.

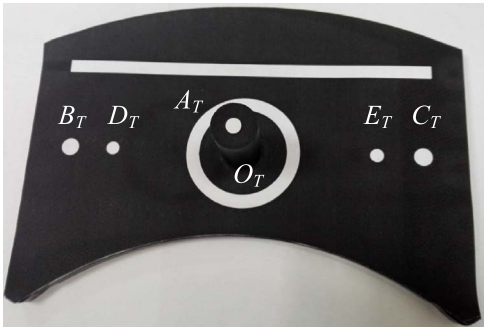


**Fig. 2.** Our cooperative target with distinct markers for optimal identification of location and orientation.

linking mechanism is proposed to obtain the interested single-pixel-width edges in a target image. Circles are detected from edges using isophote curvature. The target is identified if there exist one circle with certain number of straight lines in a boundary around the circle. The size of the boundary is related to a pre-computed ratio of the circle radius. In the target area, markers are localized using the target characteristics and a region growing algorithm. Based on the image coordinates of the markers, the target pose is measured with high precision at 8 frames per second. Our proposed algorithm is robust to lighting variance, complex scenes and small degree of motion blur, and very suitable for fast and accurate visual measurement with robotic arms.

## 2. Related work

Cooperative targets have been widely used in aerospace applications. In 1997, using a 2-point and a 3-point marker, the ETS-VII (Engineering Test Satellite VII) [9,10] developed by the JAXA (Japanese Space Agency) successfully demonstrated the RVD (Rendezvous and Docking) and RBT (space robotics) technologies. However, it is depending on a threshold commanded from the ground that the marker image is converted into a black and white image and then used to measure the pose. NASA (National Aeronautics and Space Administration) built the VGS (Video Guidance Sensor) in the mid 90 s and its improved version i.e. the AVGS (Advanced Video Guidance Sensor) in the early 21st centaury for automated spacecraft guidance. The VGS and AVGS use two wavelengths of lasers to illuminate a target that has a pattern of filtered retro-reflectors. One passes through the filters and generates a foreground image; the other is absorbed by the filters and produces the background image. Subtracting the former from the latter leaves an image with only the target's retro-reflectors visible. Then the target's pose is obtained by solving the perspective-N-point problem. However, the huge size and the requirement of two lasers as the illumination sources burden the hardware design and make the system costly. From 2012 to 2017, the Chinese Tiangong spaceships have performed a serious of autonomous dockings with the Shenzhou spacecrafts. In auto mode, a laser and rader system was used to measure the pose; in manual mode, a cross target was identified by the human eye. Our former work [11] proposes a visual method to identify a cooperative target with high accuracy rates. The target has one circle in the center and three lines adjacent with it, two long ones on the left and right, one short on the top. However, the circle's adjacent with lines increases the difficulty for both the circle and the line detection. This paper improves the former design by separating the circle and line, and increasing two abundant dot markers to guarantee pose measurement. Accordingly, we propose a new method that identifies the target with higher accuracy rates, and further explores the localization of the markers' centroids and high-precision target pose measurement.

A cooperative target could be identified using feature points or template matching. Well known point-feature descriptors like SIFT by [12] (scale invariant features transform) and SURF by [13] (speeded-up robust features) have robust identification and tracking capabilities. A method like Mutual Information – see e.g. [14] – is quite insensitive to changes in the lighting condition and to partial occlusions. However the huge computational load of these methods limits their hardware implementation in space.

For the identification of our target (see Fig. 2), edge extraction is an essential first step. The edges generated by the well-known Canny operator (see [15]) may be discontinuous or not single-pixel-width. To know the pixel coordinates of edges, another step i.e. edge tracking should be executed. Ref. [16] proposed a method that draws continuous and single-pixel-width edges in an image. It computes anchors first, and then link them to obtain edges. However, every time when the gradient direction of an edge point changes, say from vertical to horizontal, the linking algorithm has to try both the left and right walks to find edges, hence the complexity increases. To address this problem, we propose a novel linking method that makes the local optimum move at every step. It leads to smoother edges hence lays better foundation for further circle detection.

Circle detection is another key of target identification. Traditional methods like the circle Hough transform (CHT) by [17] are very slow and memory-demanding and produce many false detections. Improved method like randomized HT by [18] and local voting by [19] overcome some shortcomings but are either still slow or memory-demanding. Properties of isophotes (see [20,21]) makes them particularly suitable for object detection. Their shapes are independent of rotation and varying lighting conditions, hence isophotes curvature is used in this paper to detect circles.

Fiducial markers have been used for visual measurement by many researchers. Reference [22] arranged multiple dots in certain patterns as markers to measure the position and orientation of a spacecraft. Ref. [23] utilized a marker that consists of concentric contrasting circles to estimate the 12 Degrees of Freedom relative state for small inspection spacecrafts. Ref. [24] presented a coarse-to-fine dot array marker tracking method and implemented it in a vivo animal experiment. Ref. [25] implemented fiducial markers around a lung tumor for dynamic tumor tracking. However, these methods have not been tested in complex circumstances like uneven lighting or with motion blur.

The structure of this paper is as follows. Section 3 describes the cooperative target. The algorithm determining the relative pose is based on two steps:

1. Target Identification, presented in Section 4, consisting of the three steps 1) Single-pixel-width Edge Extraction, 2) Fast Circle Detection, and 3) Line Detection within a Boundary.
2. Marker Localization, explained in Section 5, using 1) Reference Values Extraction and 2) Accurate Marker Localization.Section 6 demonstrates exhaustive experiments. Conclusions are drawn in Section 7.

## 3. Cooperative target

Fig. 2 displays our cooperative target. It is painted by two flat paints, black as the background and white as the foreground. With the target center $O_T$ as the center, we designed a ring shape, for circles have no vertexes, hence are simpler and faster detected than other shapes. Above the ring, separated by a certain distance, we placed a long straight line. The ring and the line are used for target identification. The ring includes two circular edges, i.e. the inner circle and the outer circle, hence raises the possibility of identifying the target. The line approximately indicates the roll of the target so that the dot markers can be correctly labelled accordingly.

Perpendicular to the target plane, is a black column $O_TA_T$ with the white dot $A_T$ on its top. At the left side of the ring are the dots $B_T$ and $D_T$, and the dots $C_T$ and $E_T$ are at the right. The center of the left and right dots lie in a line which passes through the target center $O_T$ and is parallel with the long straight line. $A_T$, $B_T$ and $C_T$ form an isosceles triangle, and so do $A_T$, $D_T$ and $E_T$. The five dots $A_T$ to $E_T$ are used as markers to calculate the relative position and orientation between the target and the hand-eye camera. In most cases, the centroids of marker $A_T$, $B_T$ and $C_T$ are used; however, markers $A_T$, $D_T$ and $E_T$ are used when $B_T$ or $C_T$ is outside the field view of the camera. The design of two extra markers guarantees the success of the pose measurement in case not all markers are detected.

## 4. Target identification

The identification process of the target includes three steps: edge extraction, circle detection and line detection. The following subsections will explain them in detail.

### 4.1. Single-pixel-width edge extraction

Given a target image, three steps are followed to obtain the interested single-pixel-width edges: calculating the gradients, computing the anchors and linking the anchors.

#### 4.1.1. Calculating gradients

The initial image obtained from the camera is filtered with a 5×5 Gaussian filter with kernel $\delta = 1$. Standard Sobel operators are then used to calculate the gradients in $x$ direction: $G_x$, and in $y$ direction: $G_y$. Using the equation $G = \sqrt{G_x^2 + G_y^2}$, the gradient magnitude $G$ is calculated. The boolean gradient direction of a pixel is noted as $G_d$, whose value is TRUE if $|G_x| \geq |G_y|$ which means a vertical edge passes through the pixel; otherwise a horizontal edge passes through and the value is FALSE.

The camera background is complex due to harsh lighting conditions, various objects in the background and different target poses. However, the target has only black and white colors, and its edges are quite strong. Taking this advantage, an adaptive threshold as proposed by [11] is used to obtain only the interested edge areas, i.e. pixels whose gradient magnitude are higher than the threshold. The final single-pixel-width edges must be within these areas.

#### 4.1.2. Computing anchors

Among the edge areas, we scan the image in every $l_{scan}$ row to find anchor points. Anchors are the pixels with a very high probability of being edge elements. Intuitively, they are pixels which edges are put over. The smaller the scan interval $l_{scan}$ is, the less anchors are computed, resulting in less edges. In other words, if one likes to obtain only the long edges in an image, $l_{scan}$ shall be large; otherwise, if one prefers extracting most edges including the short ones, $l_{scan}$ should be small. In the line that is scanned, if a pixel belongs to the edge areas, we determine whether it is an anchor by the principle visualized in Fig. 3. If $G_d(x, y)$ is TRUE, the pixel belongs to a vertical edge, its gradient magnitude $G(x, y)$ is compared to those of its left and right neighbors $G(x - 1, y)$ and $G(x + 1, y)$. If the differences between the pixel $(x, y)$ and its neighbors are both greater than a threshold $t_a$, the pixel is an anchor point; otherwise it is not. Similarly, if $G_d(x, y)$ is FALSE, a horizontal edge passes through the point, and we compare its gradient magnitude with its upper and lower neighbors.

#### 4.1.3. Linking anchors

Anchors are linked using a novel linking mechanism. From the left top of an image, in every $l_{scan}$ row we search for the anchor points. If an anchor is found, it is used as a starting point of a linking procedure.

The linking mechanism of the starting point is shown in Figs. 4 and 8, and 8 directions are defined in Fig. 5. If the gradient direction $G_d(x, y)$ of the anchor $(x, y)$ is TRUE, the edge is vertically aligned; otherwise the edge is horizontally oriented. Taking the way up of a vertical edge for instance, we search for the next edge point in direction 5, 6 and 7 and the pixel with the largest gradient magnitude value is the next edge point. Similarly, the way down takes direction 1, 2 and 3 into consideration, and the pixel with highest magnitude is the next edge pixel.

From the second edge point, we continue to search for edge pixels according to a local optimum principle. It follows 3 different rules in 3 different cases, i.e. Figs. 6–8.

In the first case (Fig. 6), the gradient direction $G_d$ of the current edge point (green) is the same as the last one (black), hence three new directions are considered i.e. the last direction (7), the nearest direction clockwise to it (0) and the one counterclockwise to it (6). Here, the last direction is 7, therefore pixels in direction 7, 6 and 0 are considered. Likewise, the pixel with the greatest magnitude is the edge point.

Case 2 demonstrates the situation that the gradient direction $G_d$ of the last edge pixel is horizontal, while that of the current edge point becomes vertical. As shown in Fig. 7(a), when last direction is 5 or 7, the edge shows a tendency to go up, hence we change the value of last direction to 6 (dashed red arrow), and in the upper directions, i.e. direction 5 6 7, we search for the next point. Likewise, in Fig. 7(b), the edge demonstrates a tendency to go down, therefore last direction is changed to 2, and in direction 1, 2 and 3 we look for the next point. Fig. 7(c) and (d) shows the cases in which the edge has no tendency to go either up or down. In such cases, the sum of the gradient magnitudes of the three upper neighbors and that of the three lower neighbors are compared, and the edge walks to the way with the smaller sum. In Fig. 7(c), the upper side has the smaller sum, hence last direction is changed to 6, and directions 5, 6 and 7 are considered. On the contrary, in Fig. 7(d), the sum of the lower side is smaller, therefore the edge goes down.

Likewise, in case 3, the gradient direction of last edge point is vertical whereas that of current point changes to horizontal, a similar principle is followed as shown in Fig. 8.

Fig. 9 demonstrates an example of how to link the anchors. The orange pixels are the ones whose gradient directions $G_d$ are TRUE, and the blue pixels are those whose gradient directions are FALSE. The values in the pixels are gradient magnitudes. Assuming the pixel with the red ring is the starting anchor, based on the linking mechanism of the starting point in Fig. 4, the edge walks left and right because the pixel's gradient direction is denoted as FALSE. Let us focus on the right walk.

1. Referring to Fig. 4(b), the next point is searched in direction 0, 1 and 7, hence we go in direction 1–557.
2. According to Fig. 7(b), direction 1, 2 and 3 are considered, and we go to 631.
3. Applying the rules in Fig. 6, the edge then goes to 635, 610 and 610.
4. Based on the principle in Fig. 8(b), the edge walks right to 635 and then continues to walk right.

A walk stops if an edge enters non-edge areas or there is no un-

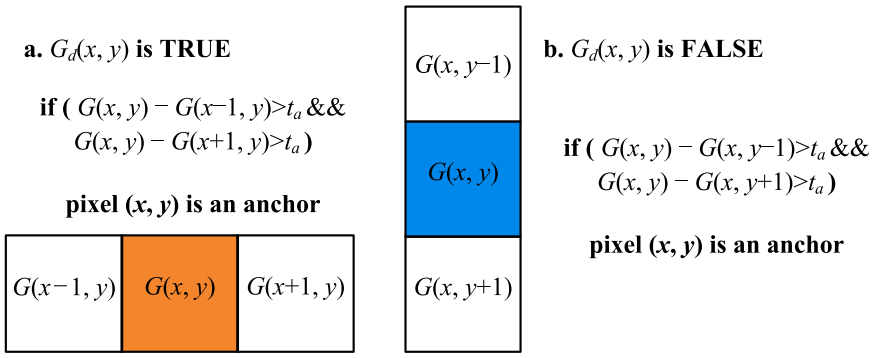


**Fig. 3.** Anchor computation of a vertical edge (left) and a horizontal edge (right).

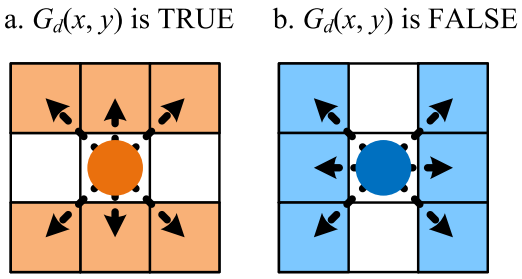


**Fig. 4.** Linking mechanism of the starting point. (a) Vertical edge. (b) Horizontal edge.

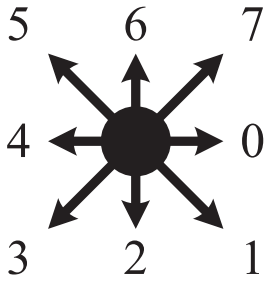


**Fig. 5.** Eight directions.

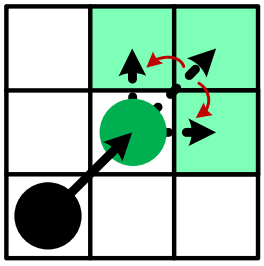

**Fig. 6.** Linking mechanism when gradient direction does not change. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article).

searched edge points in either of the three to-be-considered directions. When the two walks of the starting anchor stops, an edge is output. It starts from the end of one walk and ends at that of the other. Therefore, all the edge chains in an image are obtained when the linking step is completed.

Fig. 10 compares the proposed method with cvCanny. It is clear that the edges extracted by cvCanny show two pixel-width-edges (see the blue rectangle), notches (see the red rectangle), and discontinuities (see the yellow rectangle); however, our edges are single-pixel-width, smooth and continuous. That means our edges shall lead to higher circle detection rate and localization precision, hence increasing the target identification rate and localization accuracy. Another advantage of our method comparing to cvCanny is the use of an adaptive threshold calculated according to image gradient histogram. When the lighting intensity is high, our method ignores a large number of uninterested background edges while cvCanny obtains more unwanted edges (see Fig. 10(a)–(c)); therefore, using cvCanny shall occupy more computational load in the following target identification and pose measurement process. When the lighting intensity is low, cvCanny generates none target edges while the proposed method reserves the target edges well (see Fig. 10(d)–(f)); hence using cvCanny shall result in target identification failure.

### 4.2. Fast circle detection

The detected edges are divided into two groups: closed ones and unclosed ones. The closed edges are fitted into circles using least square fitting method. If the fitting error is very low, say smaller than 1 pixel, and the edge covers more than half of the circle's circumference, this edge is regarded as a circle. Every unclosed edge is separated into several parts at the turning points which are computed using the isophote curvature $\kappa$ (the reciprocal of the subtended radius $r$). Eq. (1) gives the definition of $\kappa$.

$$\kappa = \frac{1}{r} = -\frac{G_y^2 G_{xx} - 2G_x G_{xy} G_y + G_x^2 G_{yy}}{(G_x^2 + G_y^2)^{3/2}}. \quad (1)$$

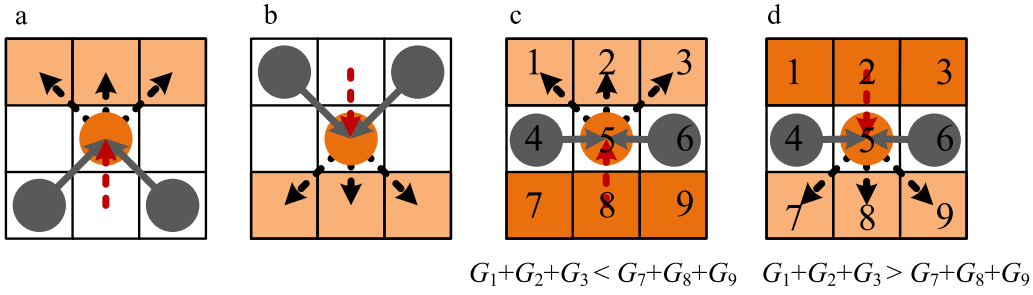


**Fig. 7.** Linking mechanism when gradient direction changes to TRUE. (a) Last direction shows a tendency of going up. (b) Last direction shows a tendency of going down. (c) Last direction shows no tendency whereas the sum of the upper 3 neighbors is smaller than that of the 3 lower neighbors. (d) Last direction shows no tendency whereas the sum of the upper 3 neighbors is greater than that of the 3 lower neighbors. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article).

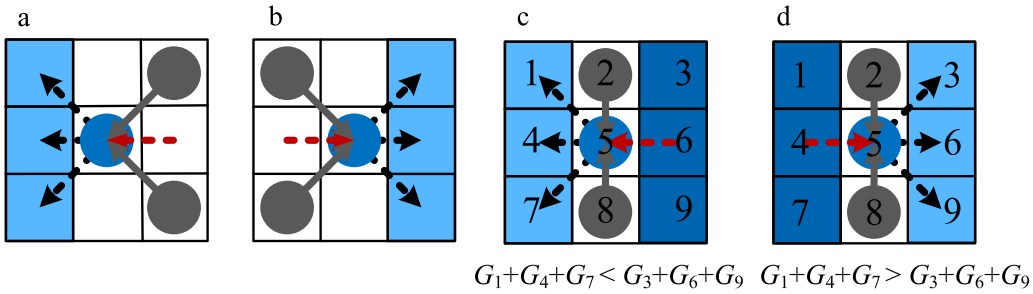


**Fig. 8.** Linking mechanism when gradient direction changes to FALSE. (a) Last direction shows a tendency of going left. (b) Last direction shows a tendency of going right. (c) Last direction shows no tendency whereas the sum of the left 3 neighbors is smaller than that of the 3 right neighbors. (d) Last direction shows no tendency whereas the sum of the left 3 neighbors is greater than that of the right 3 neighbors.

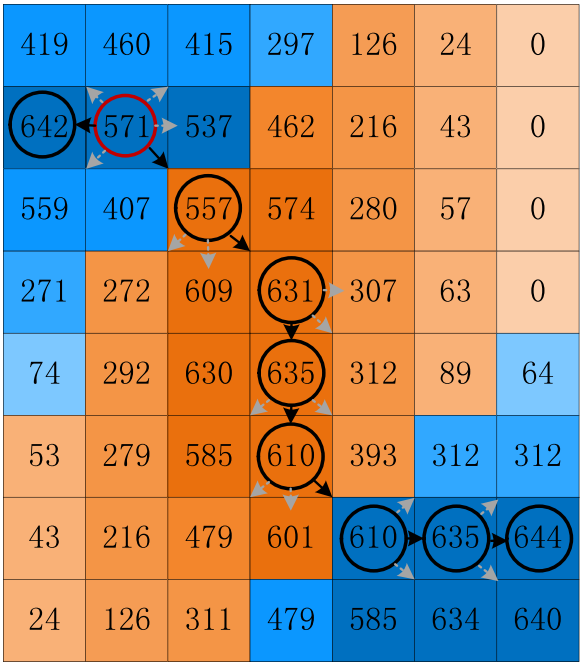


**Fig. 9.** An example of a linking procedure. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article).

where $G_x$, $G_y$ are the gradients (see Section 4.1.1), and $G_{xx}$, $G_{yy}$, and $G_{xy}$ are the second order derivatives.

If the isophote curvature of an edge point is too large, say greater than 1.5, it means that the edge that passes through this point changes its directions abruptly, hence it is a turning point.

From the example in Fig. 11, it is clear that the proposed method accurately detects the turning points (marked by crosses), including the rectangle vertices and the pixels on the rings where they are occluded by the column.

At these turning points, an edge is divided into several parts. Each part is then distinguished between arcs and non-arcs. As shown in Fig. 12, we calculate a parameter which indicates the ratio between the arc length and the chord length: $10\cdot(\frac{l_{arc}}{l_{chord}} - 1)$, where $l_{arc}$ is the length of the edge part and $l_{chord}$ is the distance between $(x_{start}, y_{start})$ and $(x_{end}, y_{end})$. If its value is larger than a threshold $k_{high}$ (say 0.15), the arc is long enough; if it is smaller than a threshold $k_{low}$(say 0.01), it means the edge resembles a line.

In both cases, four evenly distributed points $P_1$ to $P_4$ are chosen from the edge part (see Fig. 13). $P_{c1}$, the intersect point of the mid-perpendicular of chord $P_1P_2$ and chord $P_2P_3$, is then calculated; $r_{c1}$, the distanced between $P_{c1}$ and $P_2$, is obtained. Likewise, $P_{c2}$ and $r_{c2}$ are calculated. If the edge part is an arc, these values indicate the circle center and the radius. By Eq. (2), the two sets of results are compared.

$$\begin{cases} s_1 = \dfrac{|r_{c1} - r_{c2}|}{\max(r_{c1}, r_{c2})} \\ s_2 = \dfrac{\sqrt{(x_{P_{c1}} - x_{P_{c2}})^2 + (y_{P_{c1}} - y_{P_{c2}})^2}}{\max(r_{c1}, r_{c2})} \end{cases} . \tag{2}$$

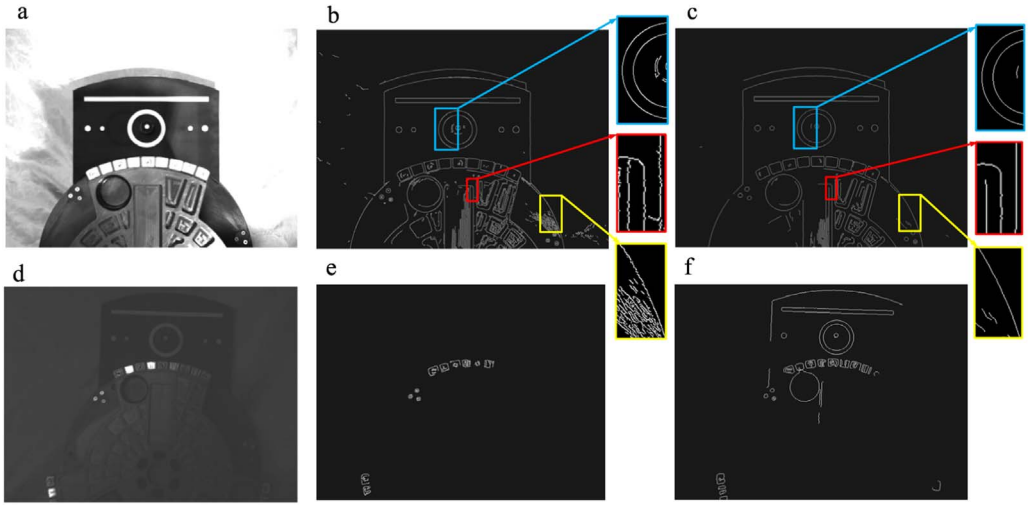


**Fig. 10.** Comparison of two edge extraction results. (cvCanny: low threshold=60, high treshold=120; the proposed: anchor scan interval=10, anchor treshold=4). (a) A target image with high lighting intensity. (b) Edges generated by cvCanny for (a). (c) Edges obtained by the proposed method for (a). (gradient threshold=61) (d). A target image with low lighting intensity. (e) Edges generated by cvCanny for (b). (f) Edges obtained by the proposed method for (b). (gradient threshold=15). (For interpretation of the references to color in this figure, the reader is referred to the web version of this article).

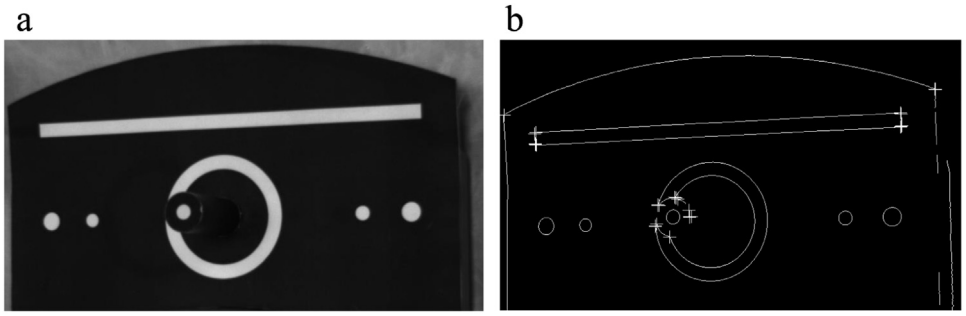


**Fig. 11.** Turning points. (a) Original Image. (b) Turning points on the edges computed by isophote curvature.

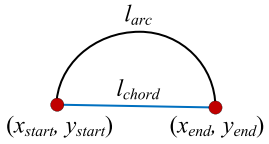


**Fig. 12.** Estimation of arc length and chord length.

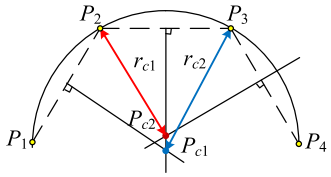


**Fig. 13.** Arc validation.

The value $s_1$ tells the similarity between the two radii ($r_{c1}$ and $r_{c2}$), and the value $s_2$ shows how close the two circle centers ($P_{c1}$ and $P_{c2}$) are to each other. If values of $s_1$ and $s_2$ are both in the interval of [0.6, 1], the edge part is fitted into a circle. If the fitting error is small enough, the edge part is deemed as a valid arc; otherwise it is invalid.

For the edge parts that do not belong to the two cases mentioned above, they are directly fitted into circles. Similarly, if the error is small, the edge part is also regarded as a valid arc.

After obtaining all the arcs in an image, they are sorted with their length in a descending order. The arcs are then joined together to detect circles, starting from the longest one. The longest arc is compared with the others one by one. If $s_1$ and $s_2$ in Eq. (2) belong to [0.75, 1], the two arcs belong to the same circle. They are joined together as a longer arc, and a circle center and a new radius are computed. After comparing the longest arc with all the others, a final fitting result of a joined arc is obtained. If the final arc spans more than half of the circumference of the great circle, and the fitting error is small, a circle is detected. Similarly, we then deal with the rest arcs, starting from the second longest one. In this way, all the circles in an image are quickly detected.

### 4.3. Line detection within a boundary

To guarantee high identification rates, other than the ring, the straight line on the target should also be utilized. All of the detected circles are sorted with their radius in a descending order. Starting from the largest one, a square boundary is set using the circle center as its center, and lines are detected within this boundary. If there exist certain numbers of lines that suit predetermined conditions, the target is identified. Once the target is found, the remaining circles are ignored.

One purpose of setting a boundary is to cut the two ends of the line, so that lines shall be easily detected among the edges. Another aim is to choose a relatively smaller area to reduce the amount of calculation. One crucial step of setting the boundary is to find the suitable size of the square so that the two ends of the line shall always be outside however the target pose changes.

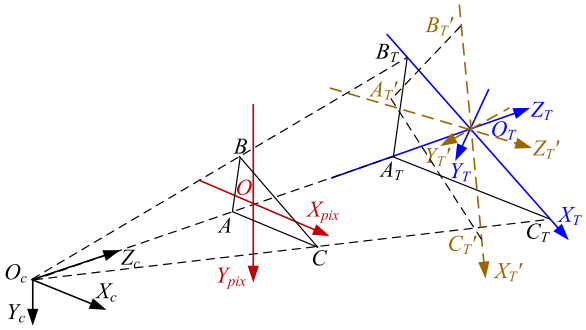


**Fig. 14.** Target coordinate and image pixel coordinate.

As shown in Fig. 14, a camera coordinate system $O_cX_cY_cZ_c$ is set up with its origin at $O_c$, i.e. the main point of the hand-eye camera. Its corresponding image pixel coordinate system is $OX_{pix}Y_{pix}$ whose origin is at the image main point $O$. Assuming the target is facing right towards the camera, an original target coordinate system $O_TX_TY_TZ_T$ is set with its origin at the target center $O_T$. Axis $X_T$ is parallel with the line on the target, axis $Y_T$ faces down, and axis $Z_T$ coincides with column $A_TO_T$ and $Z_c$ (the optical axis of the hand-eye camera). As the target rotates, $O_TX_TY_TZ_T$ changes to a new target coordinate system $O_TX'_TY'_TZ'_T$.

Fig. 15 (a) and (b) display the cooperative target in the original target coordinate system $O_TX_TY_TZ_T$ and in the image pixel coordinate system $OX_{pix}Y_{pix}$ respectively. The length of the line is noted as $l_{line}$. The distance between the lower edge of the line and the target center is noted as $d_{line}$. The inner and outer radius of the ring are $r_{inner}$ and $r_{outer}$, respectively. In most cases, the detected circle is the outer edge of the ring, and the circle's radius $r_{phy}$ is $r_{outer}$. However, in a few cases, the outer edge is not detected as a circle but the inner one is, hence $r_{phy}$ is $r_{inner}$. We set $P_T$ at the right bottom vertex of the line (the line is actually a rectangle) on the target. Its projection on the target image is $P$ (see Fig. 15(b)). Because the optical axis coincides with axis $Z_T$, the image pixel coordinate of $O$ shall remain at (0, 0) in $OX_{pix}Y_{pix}$ however the target rotates, and the minimum value of the horizontal coordinate of point $P$ shall be larger than half of the square's side.

In $O_TX_TY_TZ_T$, the coordinate of point $P_T$ is given by

$$P_T = \left[\frac{l_{line}}{2} \quad -d_{line} \quad 0\right]^T, \tag{3}$$

Assuming the three rotations of the target along the axes $X_T$, $Y_T$ and $Z_T$ are

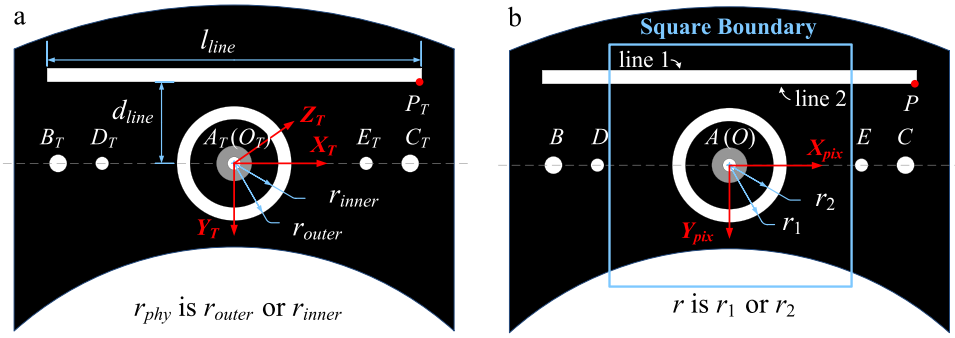


**Fig. 15.** Boundary setting. (a) Target coordinate system. (b) Image pixel coordinate system.

$$R_\alpha = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix}, R_\beta = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix}, R_\gamma = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{4}$$

the point $P_T$ after the rotation becomes $P'T$, and its coordinate in the original target coordinate system will be:

$$P_T{}' = R_\gamma R_\beta R_\alpha P_T = \begin{bmatrix} d_{line}(\cos\alpha\sin\gamma - \sin\gamma\sin\beta\cos\gamma) + \frac{l_{line}}{2}\cos\beta\cos\gamma \\ \frac{l_{line}}{2}\cos\beta\sin\gamma - d_{line}(\cos\alpha\cos\gamma + \sin\alpha\sin\beta\sin\gamma) \\ -\frac{l_{line}}{2}\sin\beta - d_{line}\sin\alpha\cos\beta \end{bmatrix}. \tag{5}$$

According to the Pinhole imaging principle, we have

$$\frac{x_P}{r} = \frac{P_T(1)}{r_{phy}} = \frac{l_{line}}{2r_{phy}} \tag{6}$$

and

$$\frac{x_P}{f} = \frac{P_T(1)}{d}. \tag{7}$$

In the above equations, $x_P$ is the horizontal image coordinate before the rotation, $r$ is the circle radius in the target image, $f$ is focal length, and $d$ is the distance between the target and the camera along axis $Z_T$. As shown in Fig. 15(b), the value of $r$ is $r_1$ in most cases (the outer edge is detected as a circle), and equals $r_2$ in some cases (the inner edge is detected as a circle). Likewise, after rotation, we have

$$\frac{x_{P'}}{f} = \frac{P_T{}'(1)}{d + P_T{}'(3)}, \tag{8}$$

where $x_{P'}$ is the horizontal coordinate of $P_T{}'$ in the image coordinate system. From Eqs. (6) and (8), $x_{P'}$ can be described as

$$x_{P'} = \frac{P_T{}'(1)}{d + P_T{}'(3)}\frac{d}{P_T(1)}x_p = \frac{P_T{}'(1)}{d + P_T{}'(3)}\frac{d}{r_{phy}}r. \tag{9}$$

We define the following target function:

$$g(\alpha, \beta, \gamma, d) = \frac{P_T{}'(1)}{d + P_T{}'(3)}\frac{d}{r_{phy}}, \tag{10}$$

where $\alpha \in [\alpha_{\min}, \alpha_{\max}]$, $\beta \in [\beta_{\min}, \beta_{\max}]$, $\gamma \in [\gamma_{\min}, \gamma_{\max}]$, $d \in [d_{\min}, d_{\max}]$. Because the range of the target pose is predetermined, the minimum value of $g(\alpha, \beta, \gamma, d)$ can be precalculated and corresponds to that of $x_{P'}$. The length of the square boundary size shall be less than $2{\cdot}x_{P'\min}$, and the center of the square shall be the circle center $O$.

With this boundary, we extract anchors with the scan interval $l_{scan}$ set at 1, and link all the edges. Then the edges are fitted into straight lines using least square fitting method. If the fitting error is less than 1 pixel, the line is long enough (say larger than $2.5r$), and the distance between the line and the circle center is within a range (say $\left[0.9{\cdot}\frac{d_{line}}{r_{outer}}r,\ 1.1{\cdot}\frac{d_{line}}{r_{inner}}r\right]$), the edge is regarded as one edge of the line on the target. When there exists certain numbers 1–3 for instance) of such lines, the target is identified.

## 5. Marker localization

Using the circle center as the center, a square target area is segmented from the target image. The size of the area is in proportion with the circle radius, and large enough to contain all the markers regardless of the target pose. In this area, some reference values are first extracted, and then the markers are localized based on them.

### 5.1. Reference values extraction

As shown in Fig. 16, the line function $y = ax + b$ calculated in Section 4.2 is stored. A point $Q$ is chosen on the edge that belongs to the circle. The grey scale intensities of Q's 5×5 neighborhood are then extracted. $I_{fore}$ is assigned the largest intensity among the 25 values and indicates the target foreground intensity. Similarly, the smallest intensity among the 25 values is assigned to $I_{back}$ which indicates the target background intensity. Because of such assigning mechanism, $I_{fore}$ and $I_{back}$ are adaptive to lighting conditions, hence are of better assistance for detecting markers. Another parameter $r_{dot}$ referring the radius of the markers is defined. Its value is estimated according to the circle radius $r$ in the image and the ratio between the physical value of the ring radius and that of the marker radius.

### 5.2. Accurate marker detection

Once again, we use the edge extraction method mentioned in Sections 4.1.2 and 4.1.3 to obtain all the edges in the target area with the scan interval $l_{scan}$ of 1 pixel.

#### 5.2.1. Marker candidates selected from edges

Referring to marker radius $r_{dot}$, the edges are picked out if they have suitable length and their start point is not too far away from its

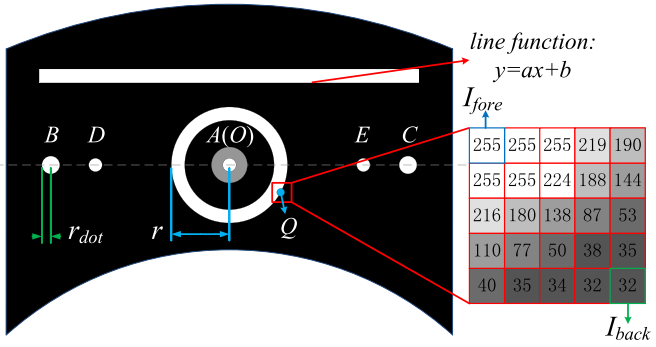


**Fig. 16.** Target information extraction.

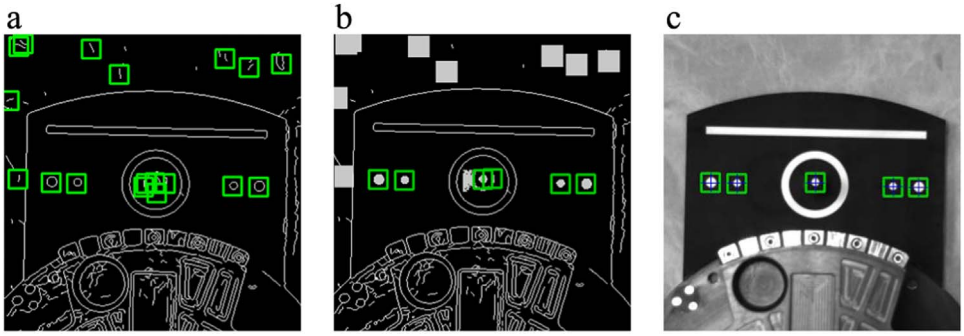


**Fig. 17.** Marker detection procedures. (a) Initial marker candidates. (b) Marker candidates after region grow. (c) Final marker detection result.

end point. For such edges, their centers of gravity are calculated. Among them, if their distance to the line $y = ax + b$ are within the range of $[k_1 r, k_2 r]$, where $\{k_1, k_2 \in R\}$, they are regarded as initial marker candidates. Fig. 17 (a) demonstrates an example. The square boxes are the extracted marker candidates.

#### 5.2.2. Region growing

Starting from each candidate pixel, we perform a region growing. As long as the following equation holds, the region grows:

$$I_{cur} - I_{back} > k_3(I_{start} - I_{back}), \tag{11}$$

where $I_{cur}$ and $I_{start}$ are the grey scale intensity of the current point and the starting point (i.e. the center of gravity of an edge) respectively. A candidate is very likely to be valid if the total area of the final region after the region growing is reasonably large (for instance within $[0.5{\cdot}\pi r_{dot}^2, 1.5{\cdot}\pi r_{dot}^2]$), and both the horizontal and vertical distances between the start and end point are not too large. The centers of gravity of such regions are computed and stored as renewed marker candidates. Duplicated candidates are then deleted. Fig. 17(b) shows the marker candidates left after this step. It is clear that the majority of the invalid candidates are excluded by the two conditions above. However, there still remains one invalid marker in this example.

#### 5.2.3. Marker validation and coordinate assignment

To obtain the final set of markers, we compute the distance from each of the remaining marker candidates to the target center. Since the central marker $A$ is at the top of the column, and may fall out of the ring, we define that if the distance is within $[0, k_4 r]$, $\{k_4 \in R, 1 < k_4 < 2\}$ (say 1.5), the corresponding maker candidate should be the central marker $A$; otherwise it shall be one of the left or right markers ($B$ to $E$).

If the number of the central marker is larger than 1, the one whose grey scale intensity is closest to the target foreground $I_{fore}$ is regarded as the valid one. Because of the target pattern, the left and right markers should always fall in a line that goes through the target center. Therefore, if the total number of the left and right markers are larger than 4, they are combined into groups of four, and each group generates a line. The valid group is the one whose line fitting error is small and has the circle center lies the closest to its line. Similarly, if the total number is equal to or less than 4, these rules should also apply. According to the positions of the line and the circle in the target image, the pixel coordinates of marker $A$ to $E$ are allocated.

Based on the pixel coordinates of the markers' centroids, the camera intrinsics, and the target physical size, the relative pose between the target and the camera is calculated with high precision by applying P3P algorithm [26].

## 6. Results

The proposed algorithm was validated by a series of experiments. Section 6.1 displays several intermediate experiment results, i.e. extracted edges, and detected circles, lines, and markers. Section 6.2 shows the proposed algorithm's performance under different circumstances, such as complex backgrounds, harsh lighting conditions, and various target attitudes. Section 6.3 analyzes the pose measurement precision.

A Balser acA 1300-30 gm camera was used. Its resolution is 1280×960 pixel, with pixel size of 3.75 μm × 3.75 μm. The focal length was 6 mm and the field of view was 43.6028° × 33.3985°.

### 6.1. Intermediate experiment results

Regardless of the target pose, the edges of the ring on the cooperative target are relatively long and strong. Therefore, given a target image, the interested single-pixel-width edges were first extracted with an adaptive gradient magnitude threshold and a relatively large scan interval $l_{scan}$ (see Section 4.1). Fig. 18 (a) gives an example of the extracted edges. Circles were then detected from the edges (see Fig. 18(b)). For each circle, straight lines that are in a certain distance to the circle center were detected. If certain number of such lines existed, the target was identified (see Fig. 18(c)). Finally, in the target area, the five markers were detected as shown in Fig. 18(d). From the image pixel coordinates of the five markers, the relative pose between the cooperative target and the hand-eye camera was calculated by P3P algorithm. The four results took 36.4 ms, 14.3 ms, 13.7 ms, and 32.8 ms, so in total less than 100 ms, measured in a PC with a 3.4 GHz Intel (R) Core (TM) i3-2130 CPU and 1.96G SDRAM.

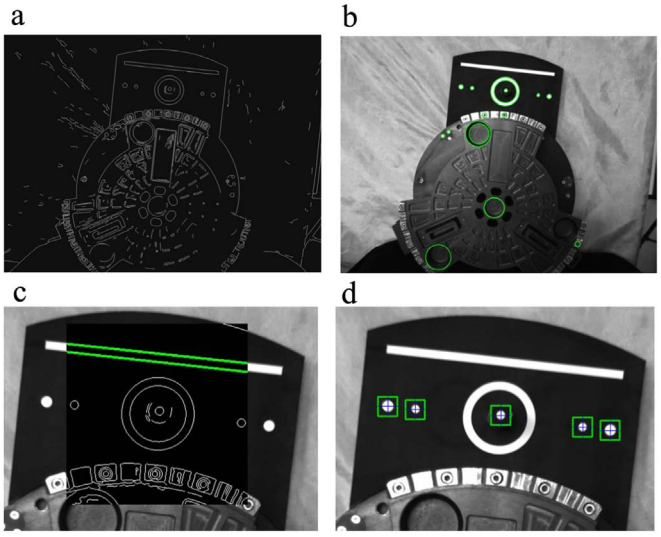


**Fig. 18.** Intermediate experiment results. (a) Single-pixel-width edges. (b) Detected circles. (c) Identified target with two lines detected. (d) Detected Markers.

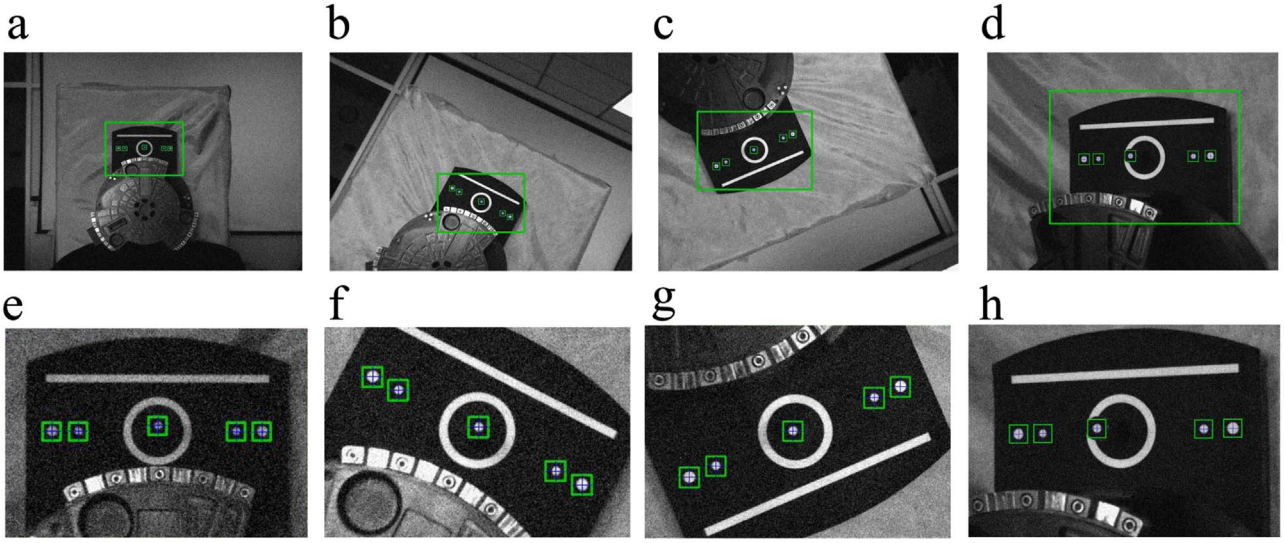


**Fig. 19.** Detection results when the target pose varies. (a) Far distance. (b) Roll. (c) Up side down. (d) Partial blocked ring. (e)–(h) Local images of (a), (b), (c) and (d), respectively. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article).

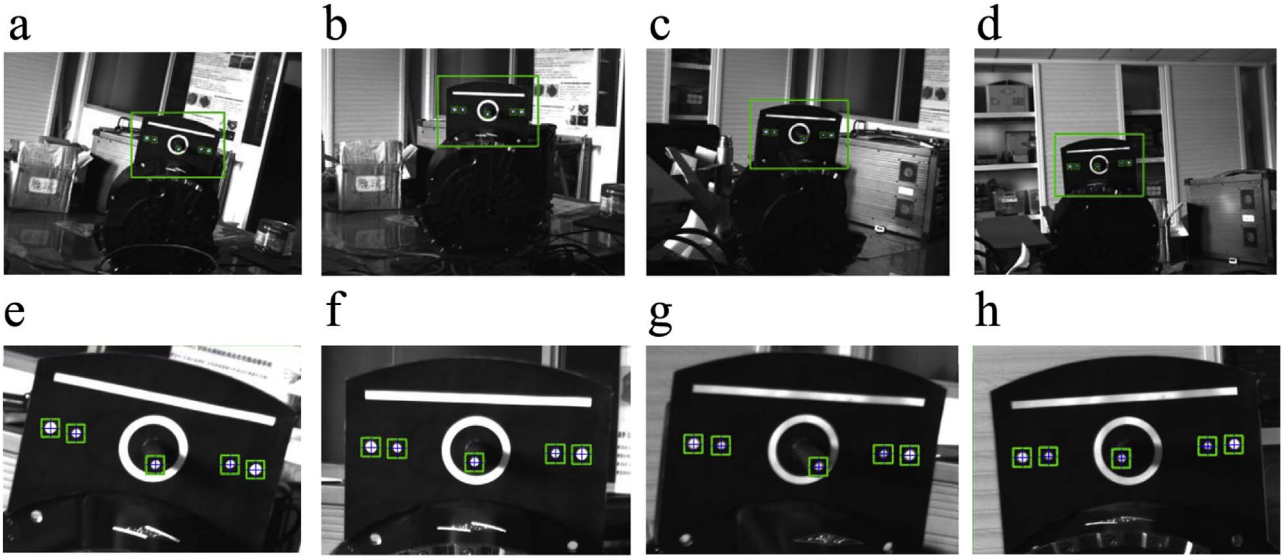


**Fig. 20.** Detection results in complex background. (a) Background with box, CD and poster. (b) Messy wires in foreground. (c) Irrelevant circles in background and several objects in foreground. (d) Background with tools. (e)–(h) Local image of (a), (b), (c) and (d) respectively.

### 6.2. Algorithm performance in various real scenes

Four experiments in real scenes were designed to test the performance of the proposed algorithm. In each of the first three experiment, the proposed algorithm ran 4 h with a frame rate of 8 FPS, in other words, a number of 115,200 images were tested.

#### 6.2.1. Varying target pose

Fig. 19 shows some results of the experiment which tests the proposed algorithm's performance when the target pose varies. During the experiment, the target remained still while the camera was moved by a turntable with six degrees of freedom. From Fig. 19(a)–(d), the distance between the target and the camera along the $Z$ axis gradually decreases.

In Fig. 19(a), the target is quite far from the camera. The rotations are quite large in both Fig. 19(b) and (c). In Fig. 19(d), the ring is partially blocked by the column in target center, and the outer and inner edges of the ring are actually visible as ellipses. In each of the images, the target was quickly identified (large green rectangle), and the five markers were accurately localized (small green squares). The intersect points of the two blue lines in each of the green squares indicates the gravity centers of the markers. In this experiment, the proposed algorithm demonstrated a 100% target identification rate and a 99.47% marker detection rate.

#### 6.2.2. Complex background

The difficulties in identifying the target are mostly contributed by the complex edges of the to-be-arrested device below the target (see Fig. 18(a)). As shown in Fig. 20, we increased the difficulty by placing the target and the to-be-arrested device in our laboratory with various tools in the background. The relative pose of the target also changed, but not in such large ranges as in the first experiment. In 99.73% of the tested image, the cooperatively target was accurately identified, and in 99.62% of them, the markers were correctly localized.

#### 6.2.3. Various lighting conditions

The third experiment tested the algorithm with different lighting conditions. During the 4 h in this experiment, the target remained still to the camera. With the time goes by, the height of the sun changed, the curtains were opened and closed occasionally. A strong LED light was

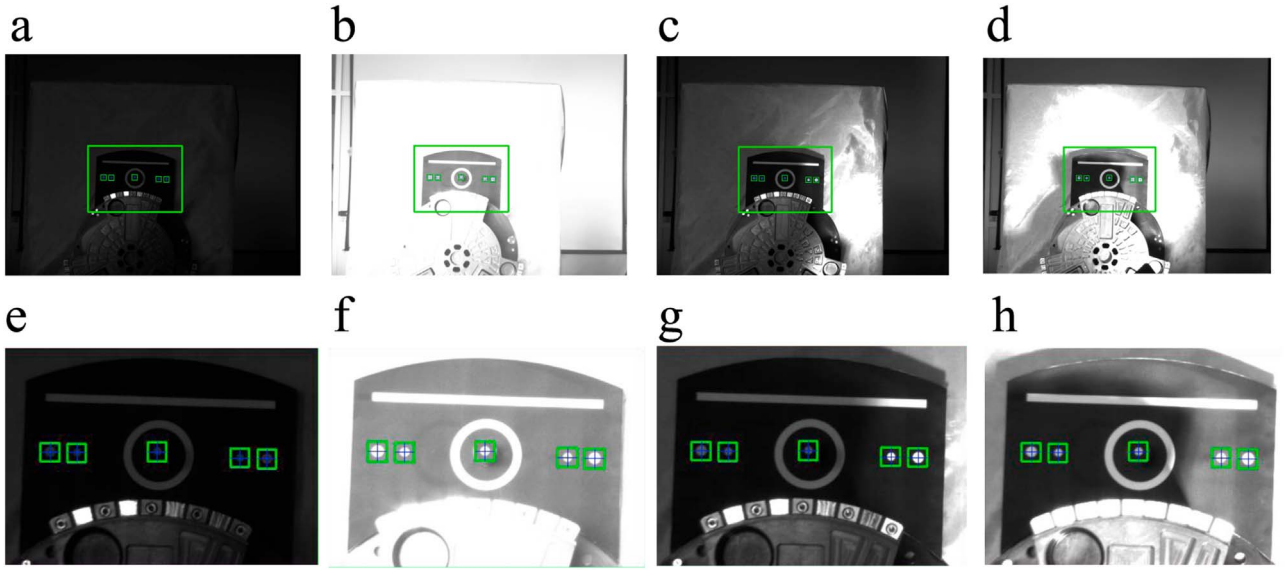


**Fig. 21.** Detection results under various lighting conditions. (a) Low lighting intensity. (b) High lighting intensity. (c) Partial light. (d) High dynamic range. (e)–(h) Local image of (a), (b), (c) and (d) respectively.

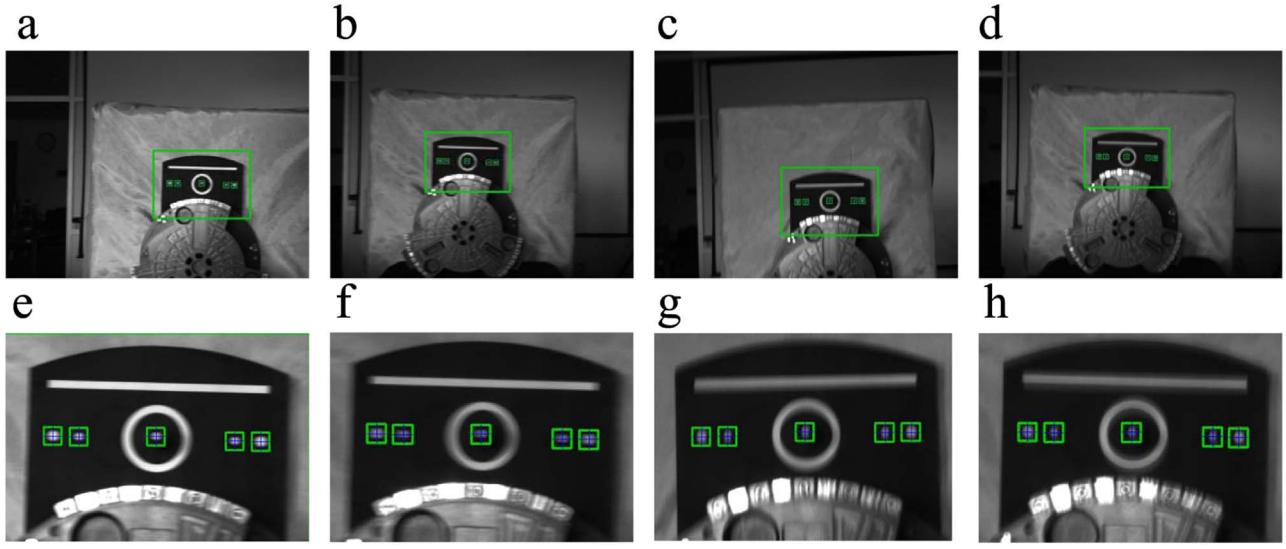


**Fig. 22.** Detection results when target images are blurred by movement. (a) Movement in horizontal direction. (b) Large movement in horizontal direction. (c) Movement in vertical direction. (d) Target moved up and down. (e)–(h) Local image of (a), (b), (c) and (d)respectively.

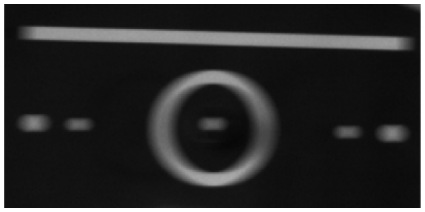

**Fig. 23.** Local image of the original target image corresponding with Fig. 22(b).

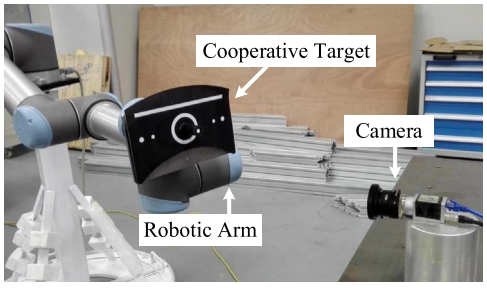


**Fig. 24.** Experimental set up.

used to light the target, and its intensity and relative position to the target constantly changed.

Fig. 21 (a) and (b) show the results when the lighting intensity is extremely low and high. The overall lighting intensity in Fig. 21(c) is quite low. However, the right part of the target has quite strong lighting intensity. Therefore, the central and left markers have quite low grey scale values, but the two right ones have fairly high grey scale values. It increases the difficulty for marker detection. In Fig. 21(d), even though the target is illuminated by strong lights, part of the target is in the shadow of the to-be-arrested device placed below the target. The contrast between the left and right markers are quite intense. Harsh lighting conditions create huge obstacles for both the target identification and marker detection, especially the latter. In spite of the difficulty, the target identification rate in this experiments was still quite high, at 98.95%, and the marker localization rate was at 97.83%.

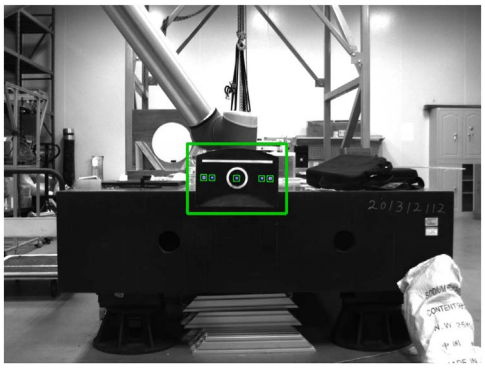

**Fig. 25.** An example of the target image.

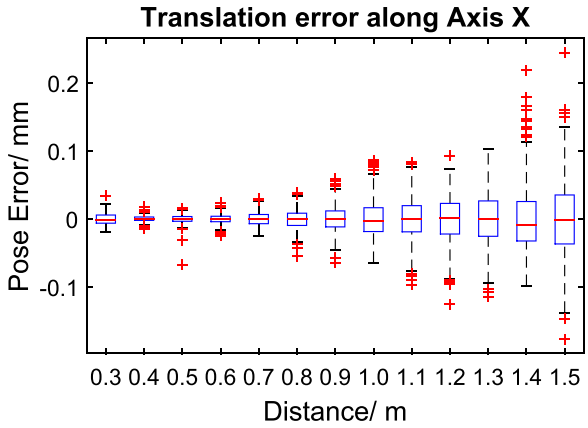


**Fig. 26.** Translation error along axis X.

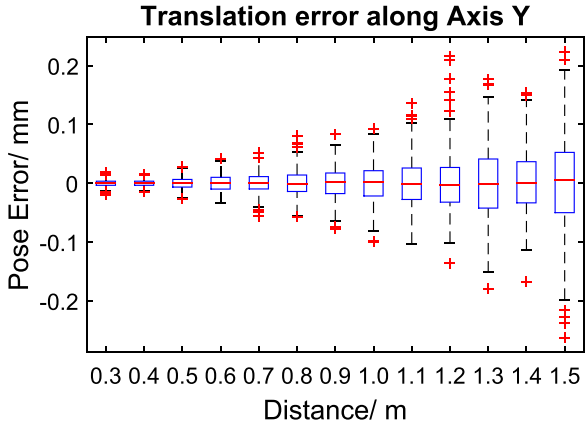


**Fig. 27.** Translation error along axis Y.

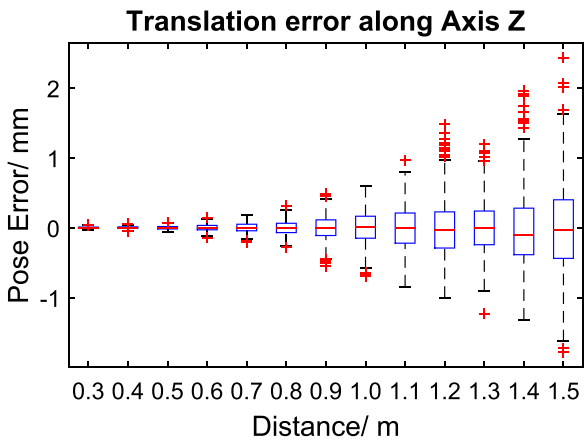


**Fig. 28.** Translation error along axis Z.

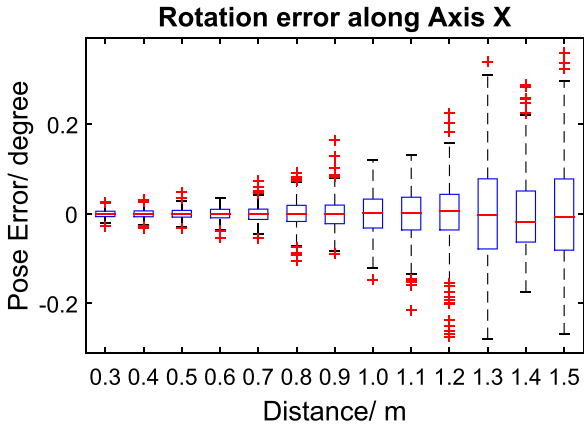


**Fig. 29.** Rotation error along axis X.

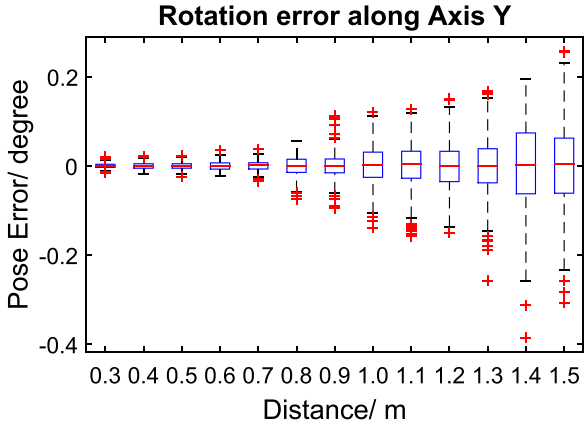


**Fig. 30.** Rotation error along axis Y.

### 6.2.4. *Motion blur*

The relative movement between the target and the camera may blur the image captured by the camera. Due to motion blur, the edges of the ring, line and dots on the target will not be sharp. Hence, target identification and especially marker detection may fail. The proposed algorithm was tested with motion blurred images, and Fig. 22 shows the test results. As shown in Figs. 22 and 23, it is clear that image is degraded strongly, and the markers are no longer dots. In Fig. 22(c), human eyes can barely tell the position of the line edges. Even with such strong motion blur, the target is identified and the markers are localized.

## 6.3. *Pose error analysis*

Aiming at analyzing the precision of target pose measurement, a robotic arm with the rotation precision of 0.001° and the translation precision of 0.005 mm was used. As shown in Fig. 24, the cooperative target is fixed at the end of the robotic arm. During the experiments, the arm moved, while the camera remained still. The pose between the target and the arm does not change, hence it was calibrated in advance. With this pose relationship, the proposed algorithm first calculated the pose between the target and the camera, and then was able to transfer it to the pose between the arm and the camera. In the experiments, the lighting intensity fluctuated because of the flashing of fluorescent lamps; the camera background included irrelevant circles and lines. Fig. 25 gives an example of the target image. The algorithm's repetitive and absolute precision are analyzed in Sections 6.3.1 and 6.3.2 respectively.

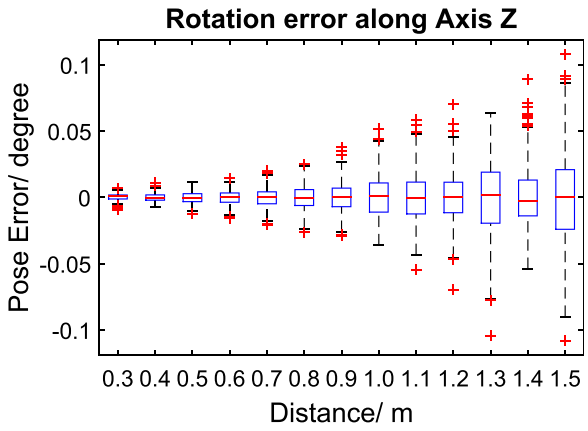


**Fig. 31.** Rotation error along axis Z.

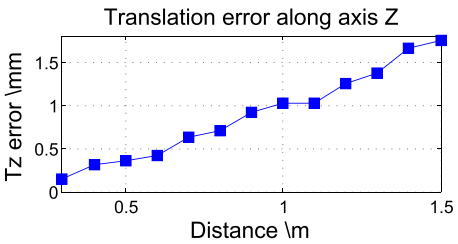


**Fig. 32.** Translation error along axis Z.

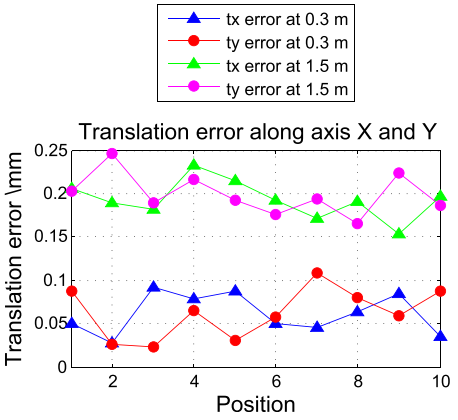


**Fig. 33.** Translation error along axis X or Y.

### 6.3.1. Repetitive precision

The arm was firstly moved to a position where the measured pose was (0,0,0,0,0,300) (the pose was defined as ($r_x$, $r_y$, $r_z$, $t_x$, $t_y$, $t_z$)), which meant that there existed only a 0.3 m translation along the Z axis between the arm and the camera. That is to say, the arm was 0.3 m away from the camera and facing right towards it. This distance was then increased by 100 mm at each time, to 400 mm, 500 mm, and ultimately to 1.5 m. At each of these 13 positions, 500 images were taken, and all their corresponding poses were calculated. As shown from Figs. 26–31, for each DOF (degree of freedom) of the pose, the 500 measurements' deviations from their mean values at each of the 13 positions are represented by a box. On each box, the central mark is the median, the edges of the box are the 25th and 75th percentiles, the whiskers extend to the most extreme data points the algorithm considers to be not outliers, and the outliers are plotted individually. Accordingly, the six degrees' standard deviations at the 13 positions are given in Table 1.

As the distance grows from 0.3m to 1.5 m, the errors all showed an increasing trend. This is reasonable for as the distance increases, the images become increasingly blur and the areas of the dot markers projected onto the camera gradually decrease. The marker localization accuracy drops accordingly, thus leads to larger errors.

Another interesting phenomenon was that the rotation errors along axis Z are always smaller than those along axis X and Y; whereas the translation errors about axis Z remained larger than those about the other two axes. The reason lies in the fact that when the 3D target is projected into 2D images, more pixels change their intensities when the target rotates around Axis $Z_T$ than along Axis $X_T$ or $Y_T$; similarly, pixel values are less sensible in distance than in horizontal or vertical translations.

At the farthest distance, i.e. 1.5 m, the maximum rotation error in the X, Y and Z axis are 0.3598°, 0.2572° and 0.1078° respectively; the maximum translation errors are 0.2442, 0.2238 and 2.4341 mm respectively. These values are still quite small considering the large distance. When the target is the nearest to the camera, at 0.3 m, the maximum rotation errors in axis X, Y and Z decrease alarmingly to 0.0257°, 0.0205° and 0.0067° respectively; and the maximum translation errors are 0.0337, 0.0183 and 0.0418 mm respectively. The main elements that lead to these errors are: lighting variance caused by flickering of fluorescent light, image blur caused by camera defocusing, slight movement of the camera because of the platform instability.

### 6.3.2. Absolute precision

Because the pose has six degrees of freedom and its required precision is quite high, it is difficult to use other instruments to calibrate the absolute pose value between the target and the hand-eye camera. In consideration that the robotic arm is able to measure the relative pose between two of its own movements with high precision, we took that as the ground truth data. Firstly, the mean value of 100 measured poses at Z=250 mm (the repetitive errors are smaller than Z=0.3 m) was regarded as the original value $t_{z0}$. Then the distance was increased by 50–0.3 m, hence the ground truth value of $t_z$ at this position is $t_{z0} + 50$ mm. Similarly, the distance was then increased by 100 mm each time to 400 mm, 500 mm, until 1.5 m. At each of these 13 positions, the difference between the ground truth and the average $t_z$ value of 100 images was regarded as the $t_z$ error.

As seen from Fig. 32, the $t_z$ error gradually increases from 0.1448mm to 1.7649 mm as the distance grows from 0.3m to 1.5 m. Such a small error (less than 0.2 mm) at the distance of 0.3 m is insignificant for the locking between the arresting and the to-be-arrested device (see Fig. 1 for reference).

**Table 1**
Six degrees' standard deviations in different distances.

| | **Standard deviation of each degree in different distances/m** | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 | 1.0 | 1.1 | 1.2 | 1.3 | 1.4 | 1.5 |
| $t_x$/0.01 mm | 0.76 | 0.44 | 0.64 | 0.61 | 1.01 | 1.42 | 1.79 | 2.53 | 3.04 | 3.27 | 3.83 | 4.80 | 5.78 |
| $t_y$/0.01 mm | 0.61 | 0.54 | 0.99 | 1.35 | 1.58 | 2.11 | 2.52 | 3.24 | 3.92 | 4.75 | 4.76 | 5.55 | 7.91 |
| $t_z$/0.01 mm | 1.33 | 1.65 | 2.40 | 4.81 | 6.74 | 9.72 | 17.38 | 23.78 | 29.62 | 40.88 | 44.48 | 57.43 | 64.74 |
| $r_x$/0.01° | 0.88 | 1.00 | 1.16 | 1.38 | 1.76 | 2.87 | 3.33 | 4.58 | 5.68 | 7.09 | 6.89 | 9.96 | 11.24 |
| $r_y$/0.01° | 0.58 | 0.73 | 0.74 | 1.01 | 1.05 | 2.24 | 2.62 | 4.21 | 5.05 | 5.01 | 6.19 | 9.21 | 9.13 |
| $r_y$/0.01° | 0.26 | 0.28 | 0.40 | 0.51 | 0.66 | 0.87 | 1.08 | 1.53 | 1.83 | 1.93 | 2.39 | 2.81 | 3.28 |

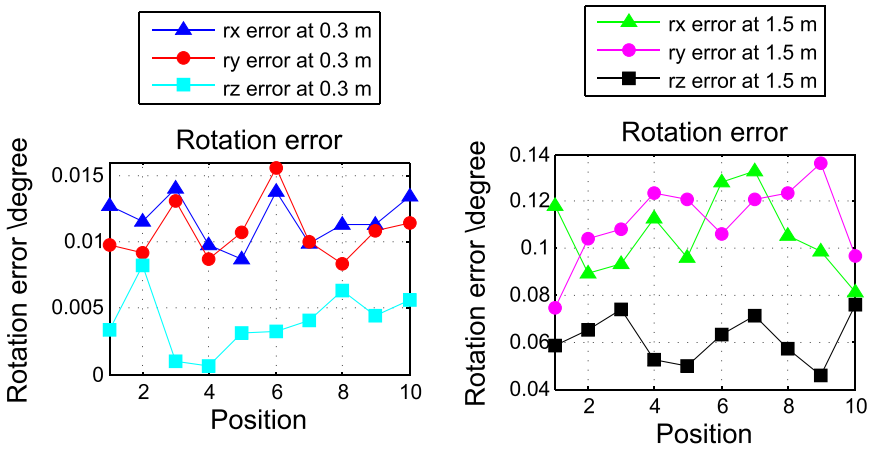


**Fig. 34.** Rotation error along three axes.

Distance significantly affects the translation precision along axis X and Y. For the sake of conciseness, the $t_x$ and $t_y$ precision at 0.3 m (nearest) and 1.5 m (farthest) were measured. Initially, the pose was set at (0,0,0,0,0,300), and the mean value of $t_x$ measured according to 100 images at this position was considered as the original value $t_{xo}$. The arm was then moved left 5 times with the interval of 10 mm at each step, so that ground truth value of $t_x$ becomes −10, −20, −30, −40 and −50 mm. Likewise, the arm moved right for 5 times. At these 10 positions, the mean $t_x$ value of 100 images were considered as the measured results; hence 10 $t_x$ errors were obtained. The $t_x$ precision at 1.5 m and the $t_y$ precision at 300 and 1.5 m were measured similarly, only with different intervals between each two adjacent positions. Fig. 33 displays the overall results.

When the distance is at 0.3 m, the $t_x$ and $t_y$ error are below 0.1 mm; while when the distance reaches 1.5 m, their errors turn a bit larger, at between 0.15 mm and 0.25 mm.

For the translation errors ($t_x$, $t_y$ and $t_z$) at 0.3 m are all less than 0.2 mm, and those at 1.5 m are all smaller than 2 mm, we can safely draw the conclusion that the translation precision of the proposed algorithm are suitable for space robotic arm pose measurement.

The rotation error along axis X, Y and Z were also measured with the distance at both 0.3 m and 1.5 m, using similar procedures as the those of $t_x$ and $t_y$. As shown in Fig. 34, at the distance of 0.3 m, the $r_x$ and $r_y$ errors are in the interval of (0.008,0.015) degree; the $r_z$ errors are relatively smaller, all less than 0.01°. When the distance is at 1.5 m, the $r_x$ and $r_y$ errors are always between 0.07° and 0.14°; the $r_z$ errors are in the interval of 0.04–0.08°.

The pose was measured by P3P algorithm according to the pixel coordinates of marker centroids. Because the P3P algorithm generates approximately zero error, the errors mainly come from inaccurate positioning of marker coordinates which are mainly contributed by illumination, image noise and motion blur.

## 7. Conclusion

We designed a cooperative target and proposed an algorithm that 1) quickly identifies it, 2) accurately localizes the markers on the target, and 3) measures its relative pose to the camera with high precision. Experimental results show that regardless of target pose, lighting condition, complex background, and motion blur, the detection rate of the proposed algorithm remained high: above 97.5%. The proposed algorithm runs in near real-time, at 8 frames per second, and applies to a large distance range, from 0.3m to 1.5 m. At 0.3 m, its rotation and translation errors are less than 0.015° and 0.2 mm respectively. The proposed algorithm is very suitable for real-time vision measurement which requires high precision and robustness (see e.g. [27,28]). We plan to further this work by implementing it in integrated circuit boards which includes in Field Programmable Gate Arrays and Digital Image Processors.

## Acknowledgement

This project was supported by the National Basic Research Program of China (No. 2013CB733103).

## References

[1] X. Liu, H. Li, J. Wang, G. Cai, Dynamics analysis of flexible space robot with joint friction, Aerosp. Sci. Technol. 47 (2015) 164–176.
[2] P. Huang, Z. Hu, Z. Meng, Coupling dynamics modelling and optimal coordinated control of tethered space robot, Aerosp. Sci. Technol. 41 (2015) 36–46.
[3] A. Flores-Abad, O. Ma, K. Pham, S. Ulrich, A review of space robotics technologies for on-orbit servicing, Prog. Aerosp. Sci. 68 (8) (2014) 1–26.
[4] G. Dong, Z.H. Zhu, Position-based visual servo control of autonomous robotic manipulators, Acta Astronaut. 115 (2015) 291–302.
[5] G. Dong, Z.H. Zhu, Autonomous robotic capture of non-cooperative target by adaptive extended Kalman filter based visual servo, Acta Astronaut. 122 (2016) 209–218.
[6] N.W. Oumer, G. Panin, Q. Mlbauer, A. Tseneklidou, Vision-based localization for on-orbit servicing of a partially cooperative satellite, Acta Astronaut. 117 (2015) 19–37.
[7] J.P. Alepuz, M.R. Emami, J. Pomares, Direct image-based visual servoing of free-floating space manipulators, Aerosp. Sci. Technol. 55 (2016) 1–9.
[8] G.W. Donohoe, Low-power reconfigurable processor, in: Proceedings of the Aerospace Conference Proceedings, IEEE, vol. 4, 2002, pp. 4-1969–4-1973.
[9] M. Mokuno, I. Kawano, In-orbit demonstration of an optical navigation system for autonomous rendezvous docking, J. Spacecr. Rockets 48 (6) (2015) 1046–1054.
[10] N. Inaba, M. Oda, Autonomous satellite capture by a space robot: world first on-orbit experiment on a japanese robot satellite ETS-VII, in: Proceedings of the Robotics and Automation, IEEE, vol. 2, 2000, pp. 1169–1174.
[11] Z. Wen, Y. Wang, A. Kuijper, N. Di, J. Luo, L. Zhang, M. Jin, On-orbit real-time robust cooperative target identification in complex background, Chin. J. Aeronaut. 9 (5) (2015) 1451–1463.
[12] D.G. Lowe, Distinctive image features from scale-invariant keypoints, Int. J. Comput. Vis. 60 (2) (2004) 91–110.
[13] H. Bay, A. Ess, T. Tuytelaars, L. Van Gool, Speeded-up robust features (surf), Comput. Vis. Image Underst. 110 (3) (2008) 346–359.
[14] A. Kuijper, Mutual information aspects of scale space images, Pattern Recognit. 37 (12) (2004) 2361–2373. http://dx.doi.org/10.1016/j.patcog.2004.04.014 (URL ⟨http://dx.doi.org/10.1016/j.patcog.2004.04.014⟩).
[15] J. Canny, A computational approach to edge detection, IEEE Trans. Pattern Anal. Mach. Intell. 6 (1986) 679–698.
[16] C. Topal, C. Akinlar, Edge drawing: a combined real-time edge and segment detector, J. Vis. Commun. Image Represent. 23 (6) (2012) 862–872.
[17] P.V. Hough, Method and means for recognizing complex patterns, Tech. Rep., 1962.
[18] K.-L. Chung, Y.-H. Huang, S.-M. Shen, A.S. Krylov, D.V. Yurin, E.V. Semeikina, Efficient sampling strategy and refinement strategy for randomized circle detection, Pattern Recognit. 45 (1) (2012) 252–263.
[19] H. Yang, J. Luo, Z. Shen, W. Wu, A local voting and refinement method for circle detection, Opt.-Int. J. Light Electron Opt. 125 (3) (2014) 1234–1239.
[20] T. De Marco, D. Cazzato, M. Leo, C. Distante, Randomized circle detection with isophotes curvature analysis, Pattern Recognit. 48 (2) (2015) 411–421.
[21] M. Van Ginkel, J. Van de Weijer, L.J. Van Vliet, P. Verbeek, Curvature estimation from orientation fields, in: Proceedings of the 11th Scandinavian Conference on Image Analysis, SCIA'99, Kangerlussuaq, Greenland, June 7–11 1999, pp. 545–

552.
[22] M. Pertile, M. Mazzucato, L. Bottaro, S. Chiodini, Uncertainty evaluation of a vision system for pose measurement of a spacecraft with fiducial markers, Metrol. Aerosp. (2015) 283–288.
[23] B.E. Tweddle, A. Saenzotero, Relative computer vision-based navigation for small inspection spacecraft, J. Guid. Control Dyn., 38(5).
[24] J. Wang, E. Kobayashi, I. Sakuma, Coarse-to-fine dot array marker detection with accurate edge localization for stereo visual tracking, Biomed. Signal Process. Control 15 (2015) 49–59.
[25] M. Nakamura, M. Takamiya, M. Akimoto, N. Ueki, M. Yamada, H. Tanabe, N. Mukumoto, K. Yokota, Y. Matsuo, T. Mizowaki, et al., Target localization errors from fiducial markers implanted around a lung tumor for dynamic tumor tracking, Phys. Med. 31 (8) (2015) 934–941.
[26] D. Dementhon, L.S. Davis, Exact and approximate solutions of the perspective-3-point problem, IEEE Trans. Pattern Anal. Mach. Intell. 14 (11) (1998) 1100–1105.
[27] S. Kahn, U. Bockholt, A. Kuijper, D.W. Fellner, Towards precise real-time 3D difference detection for industrial applications, Comput. Ind. 64 (9) (2013) 1115–1128. http://dx.doi.org/10.1016/j.compind.2013.04.004.
[28] A. De Stefano, R. Tausch, P. Santos, A. Kuijper, G. Di Gironimo, D.W. Fellner, B. Siciliano, Modeling a virtual robotic system for automated 3D digitization of cultural heritage artifacts, J. Cult. Herit. x(x), 2016, xx–xx. URL ⟨http://dx.doi.org/10.1016/j.culher.2015.11.008⟩.