·Review·

A review of edge-based 3D tracking of rigid objects

Pengfei HAN[1*], Gang ZHAO[2*]

1. *School of Mechanical Engineering and Automation, Beihang University, Beijing* 100191, *China*

2. *MIIT Key Laboratory of Aeronautics Intelligent Manufacturing, Beihang University, Beijing* 100191, *China*

*** Corresponding author,** Hanpf@buaa.edu.cn; Zhaog@buaa.edu.cn

Received: 31 July 2019 **Revised:** 14 October 2019 **Accepted:** 19 October 2019

Supported by Special Program of the Ministry of Industry and Information Technology of China.

Citation: Pengfei HAN, Gang ZHAO. A review of edge-based 3D tracking of rigid objects. Virtual Reality & Intelligent Hardware, 2019, 1(6): 580—596
DOI: 10.1016/j.vrih.2019.10.001

Abstract Three-dimensional (3D) tracking of rigid objects plays a very important role in many areas such as augmented reality, computer vision, and robotics. Numerous works have been done to pursue more stable, faster, and more accurate 3D tracking. Among various tracking methods, edge-based 3D tracking has been widely used owing to its many advantages. Furthermore, edge-based methods can be mainly divided into two categories, methods without and those with explicit edges, depending on whether explicit edges need to be extracted. Based on this, representative methods in both categories are introduced, analyzed, and compared in this paper. Finally, some suggestions on the choice of methods in different application scenarios and research directions in the future are given.

Keywords Augmented reality; 3D tracking; Edge; CAD model

1 Introduction

Three-dimensional (3D) tracking, which requires real-time six-degrees-of-freedom (DOF) pose estimation of the camera/objects, is the core technology used in many research areas pertaining to computer vision and computer graphics. For example, augmented reality (AR) achieves the effect of virtual and real fusion by estimating in real time the camera pose and then superimposing computer-generated objects on the captured sequence of a real environment. Robotic arms rely on an accurate pose estimation of the target object to complete the grabbing movement. Robots self-localize by estimating the camera pose and then perform route planning in simultaneous localization and mapping (SLAM) applications. To achieve these effects through 3D tracking, researchers have proposed massive approaches using various techniques and hardware devices in the last two decades. Here, we mainly focus on edge-based tracking approaches for rigid objects with computer-aided design (CAD) models in monocular images (with a single camera).

In the research area pertaining to tracking using CAD models, natural features are mostly used to establish a connection between the 2D projected image and the 3D real world, and then solve the pose parameters using optimization methods. Local descriptors, which are scale and rotation invariant around a key point such as SIFT[1] and SURF[2], are some of the commonly used natural features. Descriptors are extracted and matched across images, and the relative camera pose is therefore estimated. To make the extraction of descriptors and the matching process more robust and faster, researchers have proposed many improved methods[3,4]. Meanwhile, derivative methods based on local descriptors have also been utilized to

solve tracking problems such as optical flow[5]. This type of natural feature is robust against partial occlusions, background clutter, and fast movements, and is also illumination invariant.

Although local descriptor-based approaches have many advantages, one obvious limitation reported is that it requires the target object to have a rich texture. However, products usually have a metallic luster or are painted in a fixed color (e. g., all parts are painted in light yellow in aircraft assembly shops) in industrial environments. In other words, there are not enough textures to generate local descriptors for tracking. To address this problem, researchers often adopt edge-based tracking methods. This type of methods utilizes edges, which are usually produced by the structures or contours of the objects, to estimate the relative pose between two sequential frames and, in this way, real-time 3D tracking is achieved. Owing to the calculation of image gradients, these methods are also illumination invariant even for materials and are very computationally efficient.

Edge-based tracking methods can be mainly divided into two categories depending on whether explicit edges are extracted:

(1) Methods without explicit edges. This type of methods attempts to search for strong gradients near the projections of control points, which are regularly sampled along the 3D edges located on the surface of the CAD model of the object, as candidate edges. These methods require less computational effort, which are their main advantage.

(2) Methods with explicit edges. This type of methods extracts explicit edges such as straight line segments, contours, circles, and corners. These methods attempt to establish correspondences between the extracted edges and the 3D model edges, and use them to recover the pose parameters. These methods are known to be more robust.

According to this category, the remainder of this paper will be organized as follows. To better understand the algorithms involved in this review, we will first briefly introduce the camera model as the mathematical foundation in Section 2. Then, edge-based tracking methods without and with explicit edges will be reviewed in Sections 3 and 4, respectively. The main features of various methods will be compared and discussed in Section 5. Suggested future work will be discussed in this section also. Finally, conclusions will be drawn in Section 6.

2 The camera model

In the area of computer vision and computer graphics, the pinhole model is commonly used to simulate the camera imaging process. As shown in Figure 1, there are several coordinate systems defined in the camera model:

(1) World Coordinate System $O_w X_w Y_w Z_w$. The absolute coordinate system of the objective world.

(2) Camera Coordinate System $O_c X_c Y_c Z_c$. The system established from the optical center of the camera.

(3) Image Plane Coordinate System $O_i xy$. The system defined on the image plane of the camera.

(4) Pixel Coordinate System $O_o uv$. The coordinate system used by digital images.

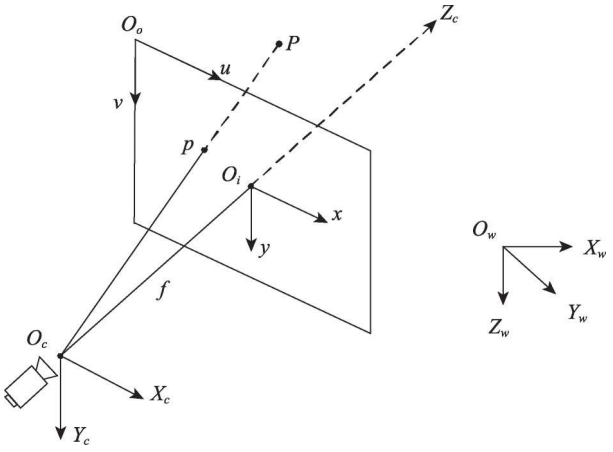

Figure 1 The camera model and the definition of the coordinate systems.

Assume that the homogeneous coordinates of a

point P in $O_wX_wY_wZ_w$ are $P_w(X_w,Y_w,Z_w,1)^T$ and that the coordinates of its projection point p in O_ouv are $p_o(u,v,1)^T$. The coordinate system conversion from P to p is

$$\begin{aligned}\begin{pmatrix}u\\v\\1\end{pmatrix}&=\frac{1}{Z_c}\begin{pmatrix}\frac{1}{d_x}&0&u_0\\0&\frac{1}{d_y}&v_0\\0&0&1\end{pmatrix}\begin{pmatrix}f&0&0&0\\0&f&0&0\\0&0&1&0\end{pmatrix}\begin{pmatrix}R&t\\0&1\end{pmatrix}\begin{pmatrix}X_w\\Y_w\\Z_w\\1\end{pmatrix}\\&=\frac{1}{Z_c}\begin{pmatrix}\alpha_x&0&u_0&0\\0&\alpha_y&v_0&0\\0&0&1&0\end{pmatrix}\begin{pmatrix}R&t\\0&1\end{pmatrix}\begin{pmatrix}X_w\\Y_w\\Z_w\\1\end{pmatrix}=\frac{1}{Z_c}K\begin{pmatrix}R&t\\0&1\end{pmatrix}\begin{pmatrix}X_w\\Y_w\\Z_w\\1\end{pmatrix}\end{aligned}\tag{1}$$

where $1/Z_c$ is the scale factor; d_x and d_y represent the physical size of each pixel in the x and y directions, respectively; (u_0,v_0) are the coordinates of O_i in O_ouv; f is the focal length; $\alpha_x = f/d_x$ and $\alpha_y = f/d_y$; R is a third-order rotation matrix; and t is a 3D translation vector, which represents the relative rotation and translation, respectively, between the world coordinate system and the camera coordinate system. K and $[R|t]$ are also termed the intrinsic and the external parameters, respectively, of the camera. Fast, accurate, and robust solving of $[R|t]$ is the key issue of pose estimation and 3D tracking. For more details on the mathematical tools, please refer to [6].

3 Tracking without explicit edges

Three-dimensional tracking methods that do not need to extract explicit edges mostly rely on searching for strong gradients near the projected edges in the orthogonal directions for each frame. RAPiD[7] is the first of this type of methods, and the subsequent methods proposed in recent years were mostly improvements or variants based on RAPiD. We will first introduce the basic RAPiD in Section 3.1. Then, several RAPiD-like approaches proposed to make it more robust or efficient will be described in Section 3.2.

3.1 RAPiD

RAPiD is a video rate object tracker proposed by Harris and Stennett[7], and it is the first tracker to run in real time. The core theory of RAPiD is the estimation of the relative pose between two consecutive frames by analyzing the displacement of image edges, which are the projections of 3D edges located on the CAD model of the object.

Obviously, not all free-form edges have precise mathematical expressions, and it is unwise to extract edge points one by one in pixels. To address this problem, Harris sampled a series of points, which are termed control points, on each 3D edge of the CAD model. After the initialization process is complete, the tracker starts working. For each frame, the pose calculated in the last frame is utilized to estimate the current pose. Assume that the pose for the last frame was already solved as the rotation matrix R and translation vector t. Therefore, for a control point P in the object/world coordinate system, its coordinates in the camera coordinate system for the last frame could be represented as $M = RP + t$ and its projection point m for the last frame could also be obtained through Equation 1. For the current frame, after a relative rotation ΔR and translation Δt, the coordinates of P in the camera coordinate system could be represented as $M' = \Delta R \cdot RP + t + \Delta t$. Furthermore, ΔR could be approximated as $\Delta R \approx I + \Omega$, where Ω is a skew-symmetric matrix including three parameters. Thus, M' and m' could be represented as a function with six parameters: $\delta p = (\Omega_x,\Omega_y,\Omega_z,\Delta t_x,\Delta t_y,\Delta t_z)$.

However, m', which is the projection point of P for the current frame, is difficult to locate directly. Instead of looking for m' precisely, RAPiD searches for strong gradients in orthogonal direction $\vec{n}$ (or only one of three directions, which are horizontal, vertical, and diagonal, is chosen for simplification), as shown in Figure 2.

The distance l is written as

$$l = \vec{n}^T(m' - m). \tag{2}$$

Therefore, for each control point P_i, we have

$$l_i = \vec{n}_i W_i \delta p \tag{3}$$

where W_i is a 2×6 matrix involving R, P, and t.

When enough control points are provided, δp can be calculated as the least-squares solution of equations:

$$\delta p = \underset{\delta p}{\operatorname{argmin}} \sum_i (\vec{n}_i W_i \delta p - l_i)^2. \tag{4}$$

Therefore, ΔR and Δt can be obtained respectively as

$$\begin{cases} \Delta R \approx I + \Omega \\ \Delta t = (\Delta t_x, \Delta t_y, \Delta t_z) \end{cases} \tag{5}$$

Finally, the pose for the current frame is estimated as $[\Delta R \cdot R | t + \Delta t]$. Similarly, this pose will be used to predict the pose for the next frame and, in this way, the camera/object pose will be updated for each frame continuously.

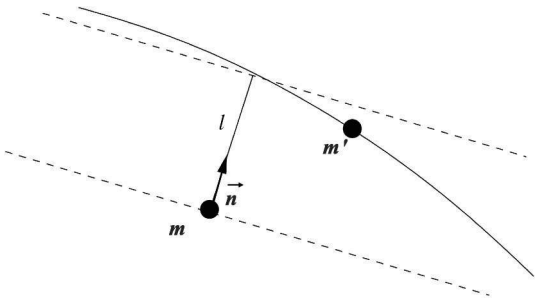

Figure 2 RAPiD searches for strong gradients in the orthogonal directions near the control points[6].

3.2 RAPiD-like approaches

Owing to the 2D search for strong gradients in the orthogonal directions and the linear approximation of small rotation ΔR, RAPiD is very computationally efficient and is the first tracker to run in real time. However, the main drawback of RAPiD is the lack of robustness, which is mainly caused by two reasons:

(1) The strong gradients searched near the control points may be generated by wrong edges such as cluttered environments, partial occlusions, and adjacent ambiguous edges. Unfortunately, RAPiD is not capable of recognizing the right edge intelligently and just simply picks the nearest one, which may lead to wrong 3D-2D correspondences and then to tracking failure.

(2) The linear approximation of rotation ΔR assumes that the relative pose between two consecutive frames is very small. In other words, the accuracy of pose estimation will decrease significantly if the camera/object moves fast.

To address these problems, researchers have proposed several improvements in recent years. In fact, the linearization of rotation is unnecessary if more efficient minimization approaches are adopted[6]. The distances between the extracted features and the projections of 3D elements can be minimized directly for each frame as

$$[R|t] = \underset{[R|t]}{\operatorname{argmin}} \sum_i dist(Proj(M_i, [R|t]), m_i') \tag{6}$$

where M_i represents 3D elements, $Proj(M_i, [R|t])$ denotes the projection of M_i under the pose $[R|t]$, and m_i' is the set of extracted features. This equation can be solved by nonlinear optimization algorithms such as Gauss-Newton[8], Levenberg-Marquardt[9], and Powell's dog-leg method[10]. In general, the Levenberg-Marquardt method has been the most often adopted method owing to its flexible steps in the optimization

process.

To make tracking more stable, Simon and Berger have introduced robust estimators to reduce the impact of outliers during the optimization[11]. Particularly, M-estimators, which are maximum likelihood-type estimators and special cases of extremum estimators, have been widely used and could be simply written as

$$\rho(r)=\begin{cases}|r| & |r|<r_{\max}\\ r_{\max} & |r|\geqslant r_{\max}\end{cases} \tag{7}$$

Furthermore, Tukey and Huber estimators are two typical M-estimators with the following respective forms:

$$\rho_{Tukey}(r)=\begin{cases}\dfrac{c^2}{6}\left[1-\left(1-\left(\dfrac{r}{c}\right)^2\right)^3\right] & |r|\leqslant c\\ \dfrac{c^2}{6} & |r|>c\end{cases} \tag{8}$$

and

$$\rho_{Huber}(r)=\begin{cases}\dfrac{r^2}{2} & |r|\leqslant c\\ c\left(|r|-\dfrac{c}{2}\right) & |r|>c\end{cases} \tag{9}$$

In [11], Equation 6 was modified to

$$[R|t]=\underset{[R|t]}{\operatorname{argmin}}\sum_i \rho_{Huber}\left(dist(Proj(M_i,[R|t]),m_i')\right). \tag{10}$$

A wire-frame tracking system that uses a combination of graphics rendering technology and constrained active edge tracking was proposed by Drummond and Cipolla[12]. In this system, the CAD model of the object is rendered with a binary space partition tree to remove the edges that are occluded and invisible. Therefore, only visible edges for the current frame are adopted to find their projected edges in the image and to establish the correspondences, which are used to estimate the pose. Moreover, in this way, the probability of edge mismatching is reduced. Moreover, robust estimators and an iterative reweighted least-squares method are utilized to solve optimization problems. The robustness of the tracking is improved significantly. As shown in Figure 3, an object with a complex structure was successfully tracked with partial occlusions.

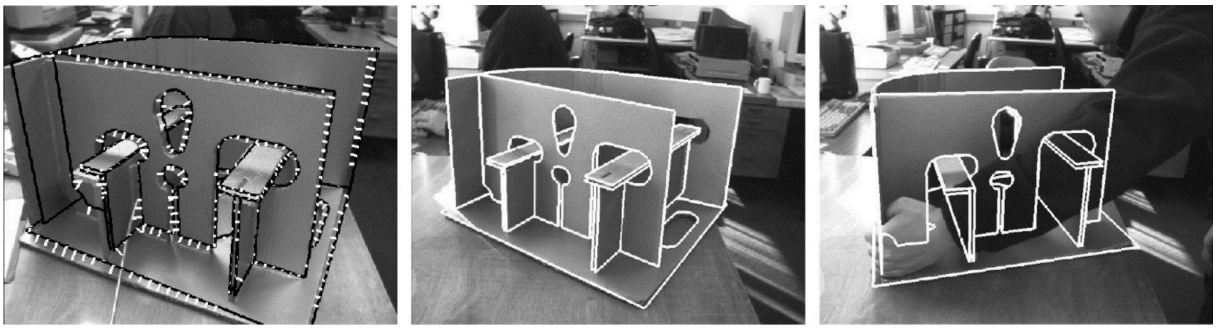

Figure 3 Use of robust estimators to improve the robustness of RAPiD against partial occlusions[12].

Marchand et al. proposed a tracking method for polyhedral objects[13]. There are two steps in this method. The first step uses 2D affine transformation to handle the displacements of image projections. Then, the 3D pose is refined by solving nonlinear optimization problems with robust estimators in the second step. The method works in real time and is robust against partial occlusions. However, this method is limited to polyhedral objects and currently cannot be used with objects with free-form shapes.

A modified Tukey estimator that considers multiple hypotheses when searching for strong gradients near the projection of control points was proposed by Vacchetti et al.[14]. For the projection of each control point, the original RAPiD only searches for the nearest point with strong gradients to minimize the displacement between them. This is not robust as the nearest one may be generated by wrong edges such as cluttered environments, partial occlusions, and adjacent ambiguous edges, which may lead to wrong 3D–2D correspondences and then to tracking failure. Therefore, all strong gradients along the orthogonal direction within a predefined distance threshold are considered and the choice of the correct candidate happens implicitly during the minimization with a modified Tukey estimator (Figure 4), which is expressed as

$$\rho^{*}_{Tukey}\left(r_1,\ldots,r_n\right) = \min_i \rho_{Tukey}\left(r_i\right) \tag{11}$$

and Equation 10 could be rewritten as

$$[R|t] = \underset{[R|t]}{\operatorname{argmin}} \sum_{i,j} \rho^{*}_{Tukey}\left(dist\left(Proj\left(M_i,[R|t]\right),m'_{ij}\right)\right) \tag{12}$$

where j is the index of multiple hypotheses.

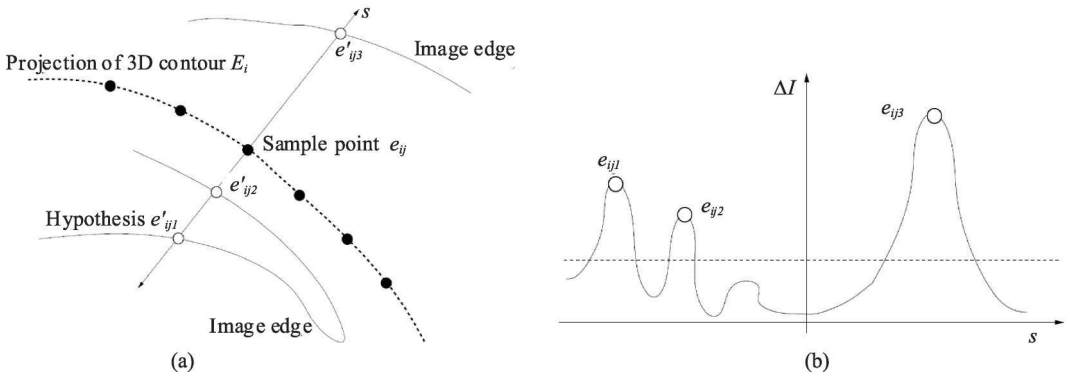

Figure 4 (a) Search for strong gradients near the projection of the control points by considering multiple hypotheses and (b) illustration of the proposed modified Tukey estimator [14].

On the basis of multiple hypotheses, Wuest proposed to describe the visual properties of an edge to help find the desired edges from captured images[15]. Moreover, the process of sampling control points is performed on the 2D projected edges instead of on the 3D edges of the CAD model to make the distribution of control points on the 2D edge more even, thus improving the robustness of pose estimation.

The control points are treated individually in the methods introduced above without considering that several control points are often placed on the same edge; hence, their measurements are correlated. Armstrong and Zisserman regarded the control points located on the same edge as a primitive, which could be adopted or rejected as a whole[16]. Moreover, they used RANSAC[17] to eliminate outliers among the control points that belong to the same primitive. If the number of inliers is less than a predefined threshold, the primitive would be excluded. In this way, the primitives have an analytic expression.

Wuest and Stricker proposed that a close prediction of the camera pose for each frame would provide great help because the local optimum could be avoided to some extent if the initial pose was close enough to the real one during the optimization process[18]. The correspondences between 3D control points and their projections searched along the orthogonal direction in the last frame are used as a predictor for the current frame. Instead of searching for points with strong gradients, the point with the highest similarity to that used in the last frame is adopted as the candidate.

Wang et al. proposed an edge-based method that has no explicit 3D-2D correspondences for 3D texture-less object tracking and that exploits the consistency of the edge direction to validate the estimated 3D pose[19]. The robust estimation adopts a direction-based estimator, which uses point-wise validation to

reduce the impact of outliers. A particle filter[20] with pose validation for a faithful weighting of particles is utilized to perform the nonlocal search. Wang et al.'s method showed better experimental results than those of distance-based methods.

To improve the tracking accuracy and robustness against fast movements, researchers often use feature points to achieve hybrid tracking[14,21–24]. Feature points are detected with a local descriptor, such as SIFT[1], SURF[2], ORB[25], BRIEF[26], BRISK[27], or FREAK[28], which is scale and rotation invariant. Feature points and edges are tracked separately, and the tracking information is merged to help with recovery in case one of them loses tracking or to provide validation for each other to improve the accuracy of the pose estimation.

Seo et al. proposed a well-established framework for real-time visual tracking of texture-less 3D objects on mobile platforms[29]. Basic RAPiD is used and optimized to track 3D objects in real time on mobile devices. Moreover, the distinctive geometric or photometric information, such as corners, colors, and lines, of the background is adopted to handle camera pose ambiguity.

Choi and Christensen proposed a tracking approach that uses edge and key point features in a particle filtering framework[30]. To address the limitation of most particle filtering-based approaches that an initial pose is assumed to be given, they employed key point features for the initialization of the filter. In the tracking phase, edge points are employed to estimate the inter-frame motions. A refinement process is performed to improve the edge correspondences between the 3D control points and the image edge points.

Edge-based tracking is also used by commercial companies such as Metaio[31], Vuforia[32], and VisionLib[33]. In the offline version, users have to import the CAD model of the target object into a desktop software, which is provided by a company and is used to extract the visible edges from a fixed point of view. These edges are then copied into projections, and the SDK will read and display them upon feedback from the camera to guide users to complete the initialization. In the online version, users manually move and rotate the camera to bring these edges close to the target object with a similar pose and the initialization will be automatically performed when they are close enough. At the same time, the tracking process also starts, as shown in Figure 5.

Figure 5 Edge-based tracking demos provided by commercial companies. Top left: A building being tracked by the Metaio SDK; Bottom left: VisionLib's tracking of industrial parts; Right: Model target tracking in Vuforia 7.

To reduce the offline work and make the whole process more convenient, Han[34] developed an edge-based mobile AR system (Figure 6). Instead of importing the CAD model into an individual desktop software, users can extract the edges within the app directly. At the beginning, the model will be rendered upon feedback from the camera and can be rotated and scaled freely through finger interaction. All the users need to do is to modify the model's pose and make it close enough to the target object, which means that users no longer have to set the camera to a fixed position. During the tracking, edges will be re-

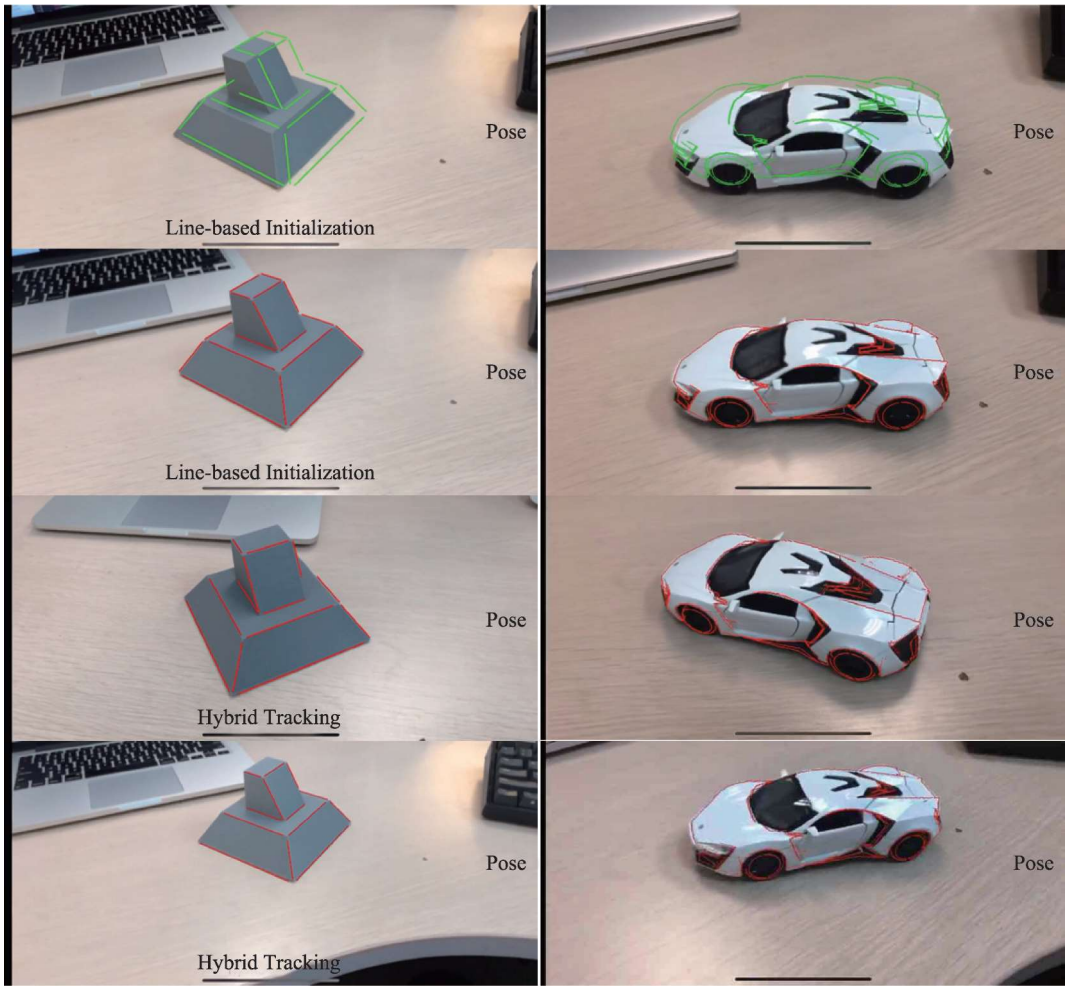

Figure 6 The edge-based mobile AR system developed by Han et al. Green edges are used to guide users through the initialization, whereas red edges indicate that the tracking is running successfully[34].

extracted and updated if the change in viewing angle is greater than a certain threshold to guarantee that all edges are visible.

4 Tracking with explicit edges

Three-dimensional tracking methods with explicit edges need to detect higher-level edge features such as straight line segments, contours, circles, and corners. These methods attempt to establish correspondences between the extracted features and the 3D model edges and use them to estimate the camera/object pose. They are known to be more robust as many outliers could be excluded through the extraction, although the computational effort will increase as a result.

Koller et al. used the Mahalanobis distance of straight line segments to establish correspondences between 3D model segments and 2D image segments[35], as shown in Figure 7. In their method, line segments are represented by

$$X = \left(c_x, c_y, \theta, l\right) \tag{13}$$

where $\left(c_x, c_y\right)$ are the coordinates of the middle point, and θ and l represent the orientation and the length, respectively[36]. Given X_m of a model line segment and X_d of an extracted line segment, the Mahalanobis distance between them could be defined as

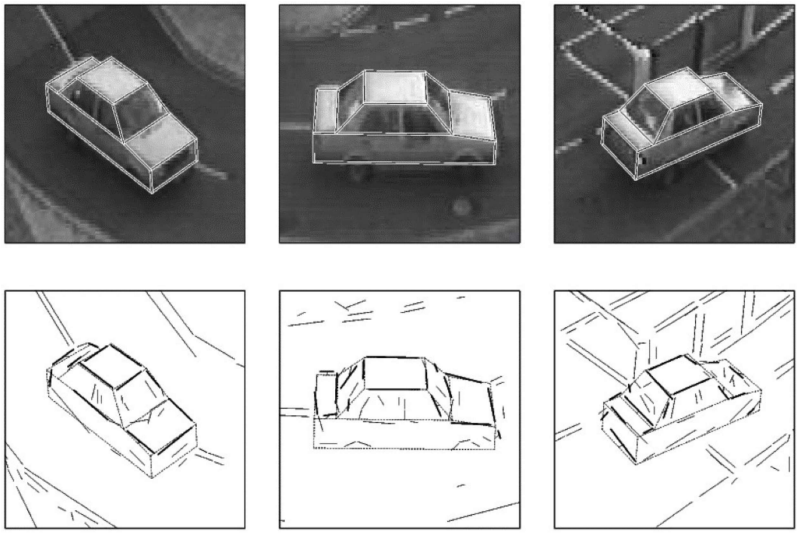

Figure 7 Pose estimation from the correspondences between 3D line segments and 2D line segments[35].

$$d = \left(X_m - X_d\right)^T \left(\Lambda_m + \Lambda_d\right)^{-1} \left(X_m - X_d\right) \tag{14}$$

where Λ_m and Λ_d are the covariance matrix of X_m and X_d, respectively. Then, the pose is estimated by

$$[R|t] = \underset{[R|t]}{\operatorname{argmin}} \sum_i \left(X_d^i - X_m^i([R|t])\right)^T \Lambda_d^i \left(X_d^i - X_m^i([R|t])\right). \tag{15}$$

This equation is solved using the Levenberg−Marquardt algorithm[11].

Shahrokni et al. proposed an initialization method for AR using line segments[37]. They assumed that a parallelogram, which is detected by line segments, corresponds to a face of a 3D object. The pose of the polyhedral object is estimated by generating hypothetical correspondences between the vertices of this parallelogram and the vertices of the face. This method can only be applied to objects with simple shapes such as a rectangular solid because it requires several models depending on how the face appears.

David and DeMenthon presented an approach to recognize partially occluded objects in cluttered environments through the corresponding line segments[38] (Figure 8). Line correspondences are found in three steps. First, 3D−2D correspondences of one or two lines are used to generate many approximate pose hypotheses. Second, the pose hypotheses are ranked by comparing the local image neighborhoods with the corresponding local model neighborhoods using fast nearest-neighbor and range search algorithms. Third, the best approximate poses in the last step are refined and verified. However, a few model lines need to be integrated into an image.

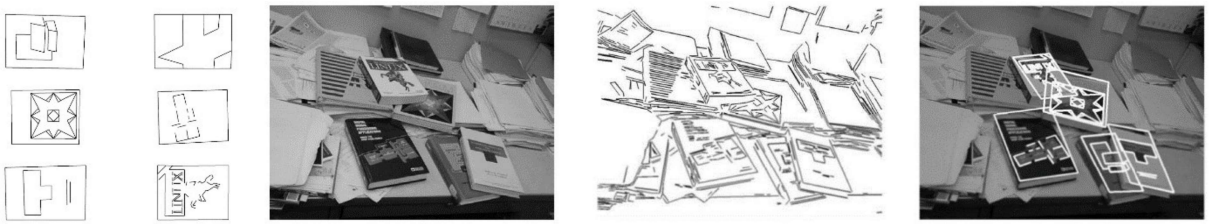

Figure 8 Recognition of books with partial occlusions in a cluttered background[38].

Kotake et al. proposed a hybrid camera pose estimation method that uses an inclination sensor and correspondence-free line segments[39]. Possible azimuths of the camera pose are hypothesized by a voting method under an inclination constraint. Then, some camera positions for each possible azimuth are calculated on the basis of the detected line segments that affirmatively voted for the azimuth. The most consistent one is selected as the camera pose out of the multiple sets of camera positions and azimuths.

However, randomly picking from possible corresponding line segments may lead to uncertain computational effort or less robustness, and once many line segments are clustered with a one-directional vector, the problem of combinatorial explosion will happen.

Kim et al. presented a texture-less object tracking approach for an indoor mobile robot; it relies on line segments and pairwise geometric relationships between segments[40]. This approach was motivated by the need for recognition strategies that can handle many indoor objects that have little textures on their surfaces but have strong geometrical consistency within the same object class. A training process is needed, and the computational complexity is high.

An approach to track 3D objects using line structure correspondences was presented by Lu et al.[41]. Line structures that humans usually use to describe an object, such as parallelism and intersection, are used to represent the target object. A set of such feature representations that share the same properties with the corresponding model line structures is generated and evaluated to contribute a pose hypothesis with a transformation matrix. The main problem with this method is that some line structures will change when 3D lines are projected onto the 2D image.

Álvarez and Borro proposed a junction-assisted 3D pose retrieval method for un-textured 3D objects (Figure 9)[42]. An automatic process extracts the junctions and contours of the model. The junctions provide an efficient mechanism to generate candidate matches, while the contours select the correct match based on a robust shape similarity evaluation. Moreover, a comprehensive and time-consuming virtual key-frame sampling process from different virtual camera viewpoints is required in this method.

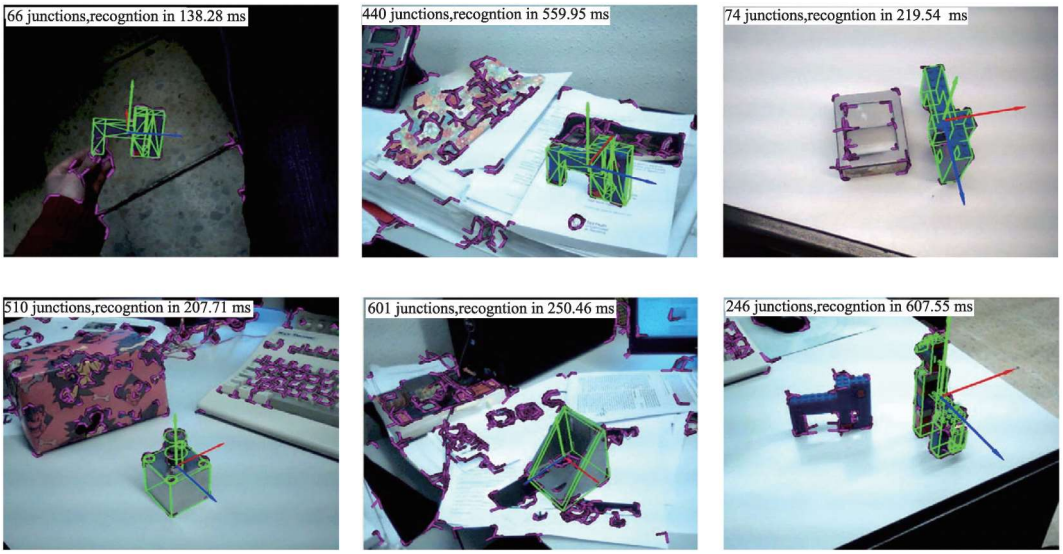

Figure 9 Use of junctions to estimate the 3D pose for un-textured objects[42].

Qiu and Wei presented a recognition method for man-made objects based on the geometric feature of line segment characteristics[43]. This method disperses the overall coordinate transformation calculation in every local plane homography calculation to reduce the complexity of the solution. Feature points are used to solve the plane homography matrix between scenes and models, and line segments on the homography plane are used to verify the assumption. Therefore, this method is not suitable for objects with only a few textures or even without any texture at all.

Guerra and Pascucci introduced a 3D tracking scheme based on line segments that uses the Hausdorff distance[44] for matching[45]. The model and the image object are both represented in terms of line segments. Several extended variants of the Hausdorff distance have been defined to compare the models and to overcome the weakness of the Hausdorff distance, which is its sensitivity to partial occlusions. A matching

algorithm, which approximates a minimization process of the Hausdorff distance based on Tabu search[46], was presented by Glover and Laguna to establish the correspondences efficiently. However, this approach has a problem regarding accuracy.

Han and Zhaoalso used contours and line segments to perform the initialization and tracking process for mobile AR[45,46]. In their method[47], the geometric correspondence between the mass center of the object and its projection onto a 2D image is used to estimate the camera pose. To hypothesize the possible azimuths of the object, their method renders the 3D CAD model of the object off-screen at different azimuths under an assumed inclination constraint provided by the inertial sensors of mobile devices; it then compares the model with the input image by contours matching using similarity evaluation. This method only needs the CAD model of the object in the offline version and is computationally suitable for mobile devices; however, it is sensitive to partial occlusion.

In [48], Han and Zhao proposed a fast line-based tracking method for mobile AR. Based on Kotake et al.'s method[39], the geometric relationship between 3D world lines and their projections onto the camera image is built to estimate the pose including the rotation and translation, respectively. The inertial sensors of mobile devices are used to provide partial rotation information directly, and the rest is obtained through a voting process. A color-based region-of-interest mask, which is used to roughly approximate the real translation, and a method for extracting vertical lines, which is used to detect the projections of 3D vertical lines from the image directly (Figure 10), were proposed to shrink the search space and therefore improve the speed and robustness of the pose estimation.

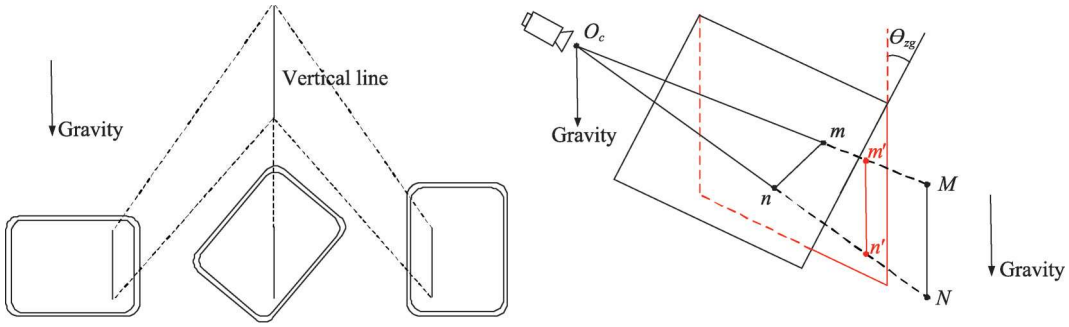

Figure 10 Illustration of the method for extracting vertical lines. (a)The projected line will always be parallel to the gravity no matter how we rotate the device if the image plane is also parallel to the gravity; (b) Assume an invisible virtual vertical plane behind the mobile device and use it to simulate the case of (a) by back-projecting the image lines onto this plane[48].

5 Discussion

Tracking methods without explicit edges are mainly based on RAPiD, which looks for strong gradients along the orthogonal direction of projected edges and minimizes the distance. Subsequent methods under this category proposed in recent years were mostly improvements or variants of RAPiD to make it more robust and efficient. This type of methods can handle free-form edges and is very fast owing to the 1D search for strong gradients.

Tracking with explicit edges needs to detect higher-level edge features such as straight line segments, contours, and corners. Therefore, detection algorithms are necessary to extract these features before the pose estimation and, as a result, the computational effort will increase, which is especially critical for mobile devices. As a trade-off, the robustness is improved as many outliers can be excluded during the

detection and the cost function of optimization can be defined more accurately owing to the explicit expression of the edges.

To make a clearer comparison, we summarize the limitations and characteristics of each method cited here in Table 1.

Table 1 Comparison between several representative methods

Methods	Explicit edges	Availability for untextured objects	Characteristic
Harris and Stennett[7]	No	Yes	Runs in real time but lacks robustness.
Simon and Berger[11]	No	Yes	Uses a Huber estimator to reduce the impact of outliers.
Drummond and Cipolla[12]	No	Yes	Removes invisible edges. Uses iterative re-weighted least-squares.
Marchand et al.[13]	No	Yes	Runs in real time and is robust against occlusions but only for polyhedral objects.
Vacchetti et al.[14]	No	Yes	Uses a modified Tukey estimator that considers multiple hypotheses.
Wuest[15]	No	Yes	Makes the distribution of control points on the edge more even.
Armstrong and Zisserman[16]	No	Yes	Regards the control points located on the same edge as a primitive.
Wuest and Stricker[18]	No	Yes	Uses a close prediction of the camera pose for each frame.
Wang et al.[19]	No	Yes	Exploits the consistency of edge direction to validate the estimated 3D pose.
Metaio[31]	No	No	Obtains edges manually and initializes them from a fixed view. Uses hybrid tracking.
VisionLib[33]	No	Yes	Obtains edges manually and initializes them from a fixed view.
Vuforia[32]	No	Yes	Initializes from an arbitrary view with artificial intelligence. Uses optional hybrid tracking.
Han[34]	No	Yes	Extracts edges from an arbitrary view within the app through user interaction.
Koller et al.[35]	Yes	Yes	Uses the Mahalanobis distance to establish line correspondences.
Shahrokni et al.[37]	Yes	Yes	Can only be applied to objects with simple shapes.
David and DeMenthon[38]	Yes	Yes	Recognizes occluded objects in a clutter through the corresponding line segments.
Kotake et al.[39]	Yes	Yes	Hypothesizes a pose by using a voting method under an inclination constraint.
Kim et al.[40]	Yes	Yes	Is only suitable for objects having strong geometrical consistency.
Lu et al.[41]	Yes	Yes	Uses line structures such as parallelism and intersection.
Álvarez and Borro[42]	Yes	Yes	Uses junctions and contours to retrieve a 3D pose.
Qiu and Wei[43]	Yes	No	Solves plane homography by using feature points and verifies assumptions by using lines.
Guerra and Pascucci[45]	Yes	Yes	Uses the extended Hausdorff distance to compare 2D and 3D models.
Han and Zhao[47]	Yes	Yes	Uses the correspondence between the mass center of the object and its projection onto a 2D image.
Han and Zhao[48]	Yes	Yes	Uses inertial sensors and color information to shrink the search space.

Obviously, methods with explicit edges have a limited scope of application. In most cases, RAPiD-like approaches are recommended for wider applicability, simpler implementation, and higher computational efficiency. Meanwhile, the robustness has also been improved to some extent with the use of robust estimators and hybrid tracking that combines point features. Methods with explicit edges can only be considered in some specific scenes such as tracking of polyhedral objects (vehicles, robot arms, etc.).

In the future, more robust and stable tracking is suggested for further study. We should consider a more accurate definition of cost functions, a more intelligent search strategy to exclude outliers, a smoother pose update to make tracking more stable, etc. Hybrid tracking has been adopted by many projection methods in recent years and has gradually become a new trend. The tracking information of edges is merged with those of feature points (SLAM[49], VO[50], etc.) to help with recovery in case one of them fails or to provide validation for each other to improve the accuracy of the pose estimation.

As a revolutionary new technology, machine/deep learning (ML/DL) has been applied to visual tracking[51−56]. Mathieu and Jean presented a temporal six-DOF tracking method that leverages DL to achieve state-of-the-art performance on challenging datasets of real-world capture[52]. Hyeonseob and Bohyung proposed a novel visual tracking algorithm based on the representations from a discriminatively trained convolutional neural network (CNN) [55]. Samuel et al. presented an image-based multi-object tracking algorithm using deep learning detections. Scheidegger et al. used CNN to detect and track objects[56]. Therefore, combining ML/DL technology with edge-based tracking should be considered as a direction worth pursuing. For example, it could be speculated that using learning approaches to roughly estimate the pose of the target object would provide a close prediction for the initialization and recovery process of RAPiD-like methods. In this way, the tracking robustness and efficiency will be improved.

Various hardware devices such as RGB depth (RGB-D) sensors and graphics processing units (GPUs) have also been adopted to benefit 3D tracking in different ways[57−60]. For example, Petit et al. proposed a method to track in real time a 3D texture-less object that undergoes large deformations such as elastic ones, and rigid motions, using point cloud data[58]. Choi and Christensen proposed a particle filtering approach, which is massively parallelized in a modern GPU, for six-DOF object pose tracking[59]. Park et al. proposed a texture-less object detection and 3D tracking method that automatically extracts the information it needs on the fly from color images and the corresponding depth maps provided by an RGB-D sensor[60]. Nowadays, more and more powerful sensors such as dual cameras, inertial measurement units, RGB-D sensors, and time-of-flight cameras are being integrated into mobile devices. We can anticipate that the additional information provided by these sensors can be combined with edge features to improve 3D tracking significantly.

6 Conclusion

Three-dimensional tracking of rigid objects plays a very important role in many areas. Among various tracking methods, edge-based 3D tracking has been widely used owing to its many advantages. Depending on whether explicit edges are extracted, edge-based tracking methods can be mainly divided into two categories: with or without explicit edges. In this review, we introduced both types of approaches. Moreover, the limitations and characteristics of these methods were compared and analyzed. Open discussions about the challenges and new trends were also carried out. We hope that this review will provide a valuable reference for researchers in this field.

References

1 Lowe D G. Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision, 2004, 60(2): 91−110
DOI:10.1023/b:visi.0000029664.99615.94

2 Bay H, Ess A, Tuytelaars T, van Gool L. Speeded-up robust features (SURF). Computer Vision and Image Understanding, 2008, 110(3): 346−359
DOI:10.1016/j.cviu.2007.09.014

3 Deriche R, Giraudon G. A computational approach for corner and vertex detection. International Journal of Computer Vision, 1993, 10(2): 101−124
DOI:10.1007/bf01420733

4 Smith S M and Brady J M. SUSAN-A New approach to low level image processing. International Journal of Computer Vision, 1997, 23(1): 45−78
DOI:10.1023/a:1007963824710

5 Horn B K P, Schunck B G. Determining optical flow. Artificial Intelligence, 1981, 17(1/2/3): 185−203

DOI:10.1016/0004-3702(81)90024-2

6 Lepetit V, Fua P. Monocular model-based 3D tracking of rigid objects: A survey. Foundations and Trends® in Computer Graphics and Vision, 2005, 1(1): 1−89
DOI:10.1561/0600000001

7 Harris C and Stennett C. RAPID-a video rate object tracker. BMVC. 1990: 1−6
DOI:10.5244/C.4.15

8 Floudas C A and Pardalos P M. Encyclopedia of optimization. Reference Reviews, 2009, 584(4): 31−52
DOI:10.1016/0375-6505(85)90011-2

9 Levenberg K. A method for the solution of certain non-linear problems in least squares. Quarterly of Applied Mathematics, 1944, 2(2): 164−168
DOI:10.1090/qam/10666

10 Powell M J D. An efficient method for finding the minimum of a function of several variables without calculating derivatives. The Computer Journal, 1964, 7(2): 155−162
DOI:10.1093/comjnl/7.2.155

11 Simon G, Berger M O. A two-stage robust statistical method for temporal registration from features of various type. In: Sixth International Conference on Computer Vision. IEEE, 1998, 261−266
DOI:10.1109/iccv.1998.710728

12 Drummond T, Cipolla R. Real-time visual tracking of complex structures. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2002, 24(7): 932−946
DOI:10.1109/tpami.2002.1017620

13 MarchandÉ, Bouthemy P, Chaumette F. A 2D−3D model-based approach to real-time visual tracking. Image and Vision Computing, 2001, 19(13): 941−955
DOI:10.1016/s0262-8856(01)00054-3

14 Vacchetti L, Lepetit V, Fua P. Combining edge and texture information for real-time accurate 3D camera tracking. In: Third IEEE and ACM International Symposium on Mixed and Augmented Reality Arlington. VA, USA, 2004, 48−56
DOI:10.1109/ismar.2004.24

15 Wuest H, Vial F, Strieker D. Adaptive line tracking with multiple hypotheses for augmented reality. In: Fourth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'05). Vienna, Austria, IEEE, 2005
DOI:10.1109/ismar.2005.8

16 Armstrong M, Zisserman A. Robust object tracking. Proceedings of the Asian Conference on Computer Vision, 1995: 5−8

17 Fischler M A, Bolles R C. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM, 1981, 24(6): 381−395
DOI:10.1145/358669.358692

18 Wuest H, Stricker D. Tracking of industrial objects by using cad models. JVRB-Journal of Virtual Reality and Broadcasting, 2007, 4(1)
DOI:10.20385/1860-2037/4.2007.1

19 Wang B, Zhong F, Qin X Y. Robust edge-based 3D object tracking with direction-based pose validation. Multimedia Tools and Applications, 2019, 78(9): 12307−12331
DOI:10.1007/s11042-018-6727-5

20 Moral P D. Nonlinear filtering: Interacting particle resolution. Markov Processes and Related Fields, 1996, 2 (4): 555−580
DOI:10.1016/s0764-4442(97)84778-7

21 DeCarlo D, Metaxas D. The integration of optical flow and deformable models with applications to human face shape and motion estimation. In: Proceedings CVPR IEEE Computer Society Conference on Computer Vision and Pattern Recognition. San Francisco, CA, USA, 1996
DOI:10.1109/cvpr.1996.517079

22 Jurie F, Dhome M. Hyperplane approximation for template matching. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2002, 24(7): 996−1000

DOI:10.1109/tpami.2002.1017625

23 Masson L, Jurie F, Dhome M. Contour/texture approach for visual tracking//Image Analysis. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003: 661–668
DOI:10.1007/3-540-45103-x_88

24 Bugaev B, Kryshchenko A, Belov R. Combining 3D model contour energy and keypoints for object tracking// Computer Vision–ECCV 2018. Cham: Springer International Publishing, 2018: 55–70
DOI:10.1007/978-3-030-01258-8_4

25 Rublee E, Rabaud V, Konolige K, Bradski G. ORB: An efficient alternative to SIFT or SURF. In: 2011 International Conference on Computer Vision. Barcelona, Spain, IEEE, 2011
DOI:10.1109/iccv.2011.6126544

26 Calonder M, Lepetit V, Strecha C, Fua P. BRIEF: binary robust independent elementary features//Computer Vision–ECCV 2010. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010: 778–792
DOI:10.1007/978-3-642-15561-1_56

27 Leutenegger S, Chli M, Siegwart R Y. BRISK: Binary Robust invariant scalable keypoints. In: 2011 International Conference on Computer Vision. Barcelona, Spain, IEEE, 2011, 2548–2555
DOI:10.1109/iccv.2011.6126542

28 Alahi A, Ortiz R, Vandergheynst P. FREAK: fast retina keypoint. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition. Providence, RI, IEEE, 2012, 510–517
DOI:10.1109/cvpr.2012.6247715

29 Seo B K, Park J, Park H, Park J I. Real-time visual tracking of less textured three-dimensional objects on mobile platforms. Optical Engineering, 2013, 51(12): 127202
DOI:10.1117/1.oe.51.12.127202

30 Choi C, Christensen H I. Robust 3D visual tracking using particle filtering on the special Euclidean group: A combined approach of keypoint and edge features. The International Journal of Robotics Research, 2012, 31(4): 498–519
DOI:10.1177/0278364912437213

31 Metaio: Edge-based initialization (2015). http://www.metaio.com/

32 Vuforia Engine (2018). https://developer.vuforia.com/

33 VisionLib (2019). https://visionlib.com/

34 Han P. Edge-based real-time tracking for mobile augmented reality on iphone x. https://v. youku. com/v_show/id_XNDA5MTQzNTUwOA==.html?spm=a2h3j.8428770.3416059.1.

35 Roller D, Daniilidis K, Nagel H H. Model-based object tracking in monocular image sequences of road traffic scenes. International Journal of Computer Vision, 1993, 10(3): 257–281
DOI:10.1007/bf01539538

36 Deriche R, Faugeras O. Tracking line segments. Image and Vision Computing, 1990, 8(4): 261–270
DOI:10.1016/0262-8856(90)80002-b

37 Shahrokni A, Vacchetti L, Lepetit V, Fua P. Polyhedral object detection and pose estimation for augmented reality applications. In Proceedings of Computer Animation2002. Geneva, Switzerland, IEEE Comput. Soc, 2002, 65–69
DOI:10.1109/ca.2002.1017508

38 David P, DeMenthon D. Object recognition in high clutter images using line features. In: Tenth IEEE International Conference on Computer Vision. Beijing, China, IEEE, 2005, 1581–1588
DOI:10.1109/iccv.2005.173

39 Kotake D, Satoh K, Uchiyama S, Yamamoto H. A fast initialization method for edge-based registration using an inclination constraint. In: 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality. Nara, Japan, IEEE, 2007, 239–248
DOI:10.1109/ismar.2007.4538854

40 Kim G, Hebert M, Park S K. Preliminary development of a line feature-based object recognition system for textureless indoor objects//Recent Progress in Robotics: Viable Robotic Service to Human. Berlin, Heidelberg, 2007, 255–268
DOI:10.1007/978-3-540-76729-9_20

41 Lu Z, Li Y, Wang J, et al. 3D Object Recognition Using Line Structure. Proceedings of SPIE-The International Society

for Optical Engineering, 2011, 8009(4):41
DOI: 10.1117/12.896087

42 Álvarez H, Borro D. Junction assisted 3D pose retrieval of untextured 3D models in monocular images. Computer Vision and Image Understanding, 2013, 117(10): 1204–1214
DOI:10.1016/j.cviu.2012.08.012

43 Qiu Z Y, Wei H. Line segment based man-made object recognition using invariance. In: 2009 International Joint Conference on Artificial Intelligence. Hainan Island, China, IEEE, 2009, 460–464
DOI:10.1109/jcai.2009.149

44 Rockafellar R T, Wets R J B. Variational analysis. Springer Science & Business Media, 2009, 317
DOI: 10.1007/978-3-642-02431-3

45 Guerra C, Pascucci V. Line-based object recognition using Hausdorff distance: from range images to molecular secondary structures. Image and Vision Computing, 2005, 23(4): 405–415
DOI:10.1016/j.imavis.2004.11.002

46 Glover F, Laguna M. Tabu search//Handbook of Combinatorial Optimization. Boston, MA: Springer US, 1998, 2093–2229
DOI:10.1007/978-1-4613-0303-9_33

47 Han P F, Zhao G. CAD-based 3D objects recognition in monocular images for mobile augmented reality. Computers & Graphics, 2015, 50: 36–46
DOI:10.1016/j.cag.2015.05.021

48 Han P F, Zhao G. Line-based initialization method for mobile augmented reality in aircraft assembly. The Visual Computer, 2017, 33(9): 1185–1196
DOI:10.1007/s00371-016-1281-5

49 Durrant-Whyte H, Bailey T. Simultaneous localization and mapping: part I. IEEE Robotics & Automation Magazine, 2006, 13(2): 99–110
DOI:10.1109/mra.2006.1638022

50 Nister D, Naroditsky O, Bergen J. Visual odometry. In: Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. Washington, DC, USA, IEEE, 2004, 1: I–I
DOI:10.1109/cvpr.2004.1315094

51 Brunetti A, Buongiorno D, Trotta G F, Bevilacqua V. Computer vision and deep learning techniques for pedestrian detection and tracking: A survey. Neurocomputing, 2018, 300: 17–33
DOI:10.1016/j.neucom.2018.01.092

52 Garon M, Lalonde J F. Deep 6-DOF tracking. IEEE Transactions on Visualization and Computer Graphics, 2017, 23 (11): 2410–2418
DOI:10.1109/tvcg.2017.2734599

53 Schulter S, Vernaza P, Choi W, Chandraker M. Deep network flow for multi-object tracking. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition. Honolulu, HI, 2017: 6951–6960
DOI:10.1109/cvpr.2017.292

54 Wang N, Yeung D Y. Learning a deep compact image representation for visual tracking. Advances in neural information processing systems. 2013: 809–817
DOI:http://respository.ust.hk/ir/Record/1783.1-61168

55 Nam H, Han B. Learning multi-domain convolutional neural networks for visual tracking. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Las Vegas, NV, USA, IEEE, 2016, 4293–4302
DOI:10.1109/cvpr.2016.465

56 Scheidegger S, Benjaminsson J, Rosenberg E, Krishnan A, Granstrom K. Mono-camera 3D multi-object tracking using deep learning detections and PMBM filtering. In: 2018 IEEE Intelligent Vehicles Symposium (IV). Changshu, IEEE, 2018, 433–440
DOI:10.1109/ivs.2018.8500454

57 Pauwels K, Ivan V, Ros E, Vijayakumar S. Real-time object pose recognition and tracking with an imprecisely calibrated moving RGB-D camera. In: 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems.

Chicago, IL, USA, IEEE, 2014, 2733–2740
DOI:10.1109/iros.2014.6942936

58 Petit A, Lippiello V, Siciliano B. Real-time tracking of 3D elastic objects with an RGB-D sensor. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Hamburg, Germany, IEEE, 2015, 3914–3921
DOI:10.1109/iros.2015.7353928

59 Choi C, Christensen H I. RGB-D object tracking: A particle filter approach on GPU. In: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems. Tokyo, New York, IEEE, 2013, 1084–1091
DOI:10.1109/iros.2013.6696485

60 Park Y, Lepetit V, Woo W. Texture-less object tracking with online training using an RGB-D camera. In: 2011 10th IEEE International Symposium on Mixed and Augmented Reality. Basel, New York, 2011, 121–126
DOI:10.1109/ismar.2011.6162879