Li JC, Zhong F, Xu SH *et al.* 3D object tracking with adaptively weighted local bundles. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 36(3): 555–571 May 2021. DOI 10.1007/s11390-021-1272-5

3D Object Tracking with Adaptively Weighted Local Bundles

Jia-Chen Li[1], Fan Zhong[2,*], *Member, CCF*, Song-Hua Xu[3], and
Xue-Ying Qin[1,*], *Senior Member, CCF, Member, IEEE*

[1] *School of Software, Shandong University, Jinan 250101, China*

[2] *School of Computer Science and Technology, Shandong University, Qingdao 266237, China*

[3] *College of Engineering and Computing, University of South Carolina, Columbia 29208, U.S.A.*

E-mail: ljc_sdu@mail.sdu.edu.cn; zhongfan@sdu.edu.cn; xus1@cec.sc.edu; qxy@sdu.edu.cn

Received January 6, 2021; accepted April 23, 2021.

Abstract The 3D object tracking from a monocular RGB image is a challenging task. Although popular color and edge-based methods have been well studied, they are only applicable to certain cases and new solutions to the challenges in real environment must be developed. In this paper, we propose a robust 3D object tracking method with adaptively weighted local bundles called AWLB tracker to handle more complicated cases. Each bundle represents a local region containing a set of local features. To alleviate the negative effect of the features in low-confidence regions, the bundles are adaptively weighted using a spatially-variant weighting function based on the confidence values of the involved energy terms. Therefore, in each frame, the weights of the energy items in each bundle are adapted to different situations and different regions of the same frame. Experiments show that the proposed method can improve the overall accuracy in challenging cases. We then verify the effectiveness of the proposed confidence-based adaptive weighting method using ablation studies and show that the proposed method overperforms the existing single-feature methods and multi-feature methods without adaptive weighting.

Keywords 3D tracking, local bundle, feature fusion, confidence map

1 Introduction

The 3D object tracking aims to estimate the six degrees of freedom (6DOF) relative pose between the camera and the target object with a known geometric model. It is a fundamental task in augmented reality because of its capability to simultaneously capture the camera pose and the registered 3D object model [1]. It is also widely used in various vision-related tasks such as human-computer interaction, robotics and medical navigation.

An important class of methods for 3D object tracking are focused on tracking the object pose based on local image features [2, 3]. Such methods have been extensively studied in the past decades. The tracking methods based on local features are often robust against lighting changes, partial occlusion and fast motion. Nevertheless, such methods are much more efficient for texture-rich objects and hence not applicable to texture-less objects [1]. To address this issue, a possible approach is to utilize depth cameras [4] and 3D tracking can be then performed using an ICP-like procedure [5]. There are however practical issues with using the depth for 3D tracking such as depth noise and misalignment, the limited distance between the camera and the object, and so on. In this paper, our focus is on the 3D tracking of texture-less objects based on monocular RGB video input.

Our objective is to perform 6DOF pose estimation for tracking and the video objects are assumed to undergo continuous transforms in their poses, where their initial pose in the first frame is also known. Note that

Regular Paper

Special Section of CVM 2021

This work was partially supported by Zhejiang Lab under Grant No. 2020NB0AB02, and the Industrial Internet Innovation and Development Project in 2019 of China.

*Corresponding Author (Xue-Ying Qin and Fan Zhong both supervised this work and provided funding support)

this is different from the 3D object detection and 6DOF pose estimation from a single image, which has been greatly advanced using learning-based approaches [6, 7].

In our approach for tracking, we only need to perform a local search in the pose space and our objective is to achieve a high precision while keeping the computational complexity as low as possible. This is crucial for achieving the high level of temporal coherence and real-time execution even in mobile devices (a common requirement of AR applications). For detection, an intensive global search is required, which is often more computational intensive than that of the tracking methods. In practice, both detection and tracking are required; however detection is only performed for initialization and/or re-localization in cases where the tracking is lost. Although several learning-based methods have been proposed for the 3D tracking [8, 9], considering efficiency and applicability, the methods based on hand-crafted features are still preferred in this research area.

Based on the features involved, texture-less 3D tracking methods can be categorized into edge-based methods [10–14] and region-based methods [15–19]. The edge-based methods are known to be sensitive to the cluttered background which presents disproportionate background edges that may easily force the optimization to fall into a local minimum [11]. Image edge detection is also sensitive to image blur which makes edge-based methods sensitive to fast-moving objects or camera. In the region-based methods, the optimal object pose is obtained through maximizing the color difference between the foreground and the background based on a statistical color model. Therefore, it can achieve a better performance in the images with a cluttered background. Nevertheless, in the scenes including foreground and background in the same color, the region-based methods become unstable. Since the statistical color model depends on the absolute color values, the region-based methods are often less robust against color and lighting changes.

As it is seen, in certain situations, various feature detection methods may become unreliable. Therefore, addressing the issue of unreliable features improves the accuracy of the tracking. Nonetheless, the reliability of features might be very different even for different parts of a single frame. Therefore, fusing features solely based on a uniform weighting function cannot achieve optimal combination of features. To address this issue, we propose the AWLB tracker, which uses adaptively weighted local bundles to define the energy function for a spatial-variant weighting of the features. In our proposed scheme, each bundle includes the aggregated evidence from a set of pixels in a local region (please refer to Subsection 3.2 for a detailed explanation). The adaptive weights of features are then obtained based on their confidence levels. The motion of each bundle is independently calculated and combined to obtain the pose transformation of the object. Color and edge features with different spatial support can be combined and adaptively weighted by packing them into bundles. Fig.1 shows some results of the proposed AWLB tracker in various challenging conditions.

Our main contributions in this paper are as the following.

• We propose the AWLB tracker, which uses adaptively weighted local bundles to suspend the negative effect of unreliable features. This results in a higher accuracy and reduces the sensitivity to the weighting parameters.

• We introduce techniques to compute the confidence of each feature, and establish the effectiveness in handling a variety of complex cases.

(a) (b) (c) (d) (e) (f)

Fig.1. Overall pose estimation results of the proposed AWLB tracker in various challenging conditions including (a) cluttered scene, (b) similar colored background, (c) occlusion, (d)(e) direct sunlight and (f) motion blur caused by fast movements.

• We demonstrate the complementarity of the color and edge features and propose an optimized method to fuse them to achieve a robust 3D object tracking in real-time.

The rest of the paper is organized as followings. Section 2 introduces related work. Section 3 presents the proposed the AWLB tracker in detail. The related experiments are illustrated in Section 4, and Section 5 concludes the paper.

2 Related Work

According to the main feature used, texture-less 3D object tracking with RGB images can be categorized as region-based and edge-based methods. Generally, color features are more informative than the edge features, while the edge features are less computationally intensive as fewer sample points are involved. Here we elaborate on these two tracking methods and then briefly review the state-of-the-art in this research area.

2.1 Edge-Based Methods

The first real-time 3D object tracking system is RAPID [10]. It starts with locating all the 3D-2D corresponding points, then utilizes a nonlinear optimization algorithm to minimize the square errors of each point, and then iteratively calculates the pose of the object. Based on [10], Marchand *et al.* [20] then replaced the gradient with the convolution kernel to select the best corresponding point with the largest response. Drummond and Cipolla [21] further proposed to weight each pair of 3D-2D points according to the number of candidate points to reduce the matching errors. Wuest *et al.* [22] picked up all candidate points to obtain the best result via an optimization with high computation time. Choi and Christensen [23] stored image templates in advance. During the tracking, [23] first estimates the initial pose based on the feature points of the current image and the template library, and then the pose optimization is completed according to the edge features thereafter.

Finding the best corresponding points of the contour points is the key function of the edge-based approach. The above-mentioned methods, however, are prone to failure in the cases where the background is complex. Some other ancillary information or strategies are therefore needed to address this issue. Seo *et al.* [11] proposed using the color model to select the best corresponding points. They first constructed the color model of the foreground and background, and then obtained the best corresponding points by maximizing the posterior probability. This method independently locates each corresponding point and hence tends to make wrong matching. Wang *et al.* [12] further utilized the geometric constraints of the image contour and the graph model to regularize the location of the edge points and improve the robustness against complex backgrounds.

Moreover, Wang *et al.* [24] proposed a tracking method based on the edge distance field. The method aims to minimize the value of the 3D contour projection points over the edge distance field to obtain the optimal pose. At the same time, to deal with fast movement and occlusion, particle filtering and robust estimation operators are introduced into the optimization method [24]. Wang *et al.* [13] further used the edge direction obtained by the image gradient to verify the confidence of edge matching to improve the robustness. They also proposed a strategy for re-localization, which records the key frame templates in real time, and performs pose recovery in cases where the object is lost.

2.2 Region-Based Methods

The region-based approach uses the level-set function [25] to represent the projection contour of the 3D object. The 6DOF pose is then optimized by maximizing the color difference between the foreground and the background. Such methods are often computationally intensive as they involve building the color model and computing the posterior probability. As the first real-time region-based approach, PWP3D [13], is accelerated by using a GPU and shown to reach the processing speed of around 20 FPS. It calculates the global foreground and background probabilities, and also uses the gradient descent technique to optimize the pose. Instead of using the gradient descent method, Tjaden *et al.* [16] embraced a Gauss-Newton-like method for optimization. They further adopted the Lie algebra to represent the pose, which enables the pose parameters to quickly converge during the optimization process.

Calculating the global foreground and background probability histograms is however difficult and it may lead to an inaccurate posterior probability. To deal with this, Hexner and Hagege [17] proposed to replace the global histogram with multiple local histograms, and then averaged them to improve the accuracy. Tjaden *et al.* [18] proposed to modify the representation of the local probability histogram, changing multiple local probability histograms of [17] to temporally consistent local color histograms, which significantly improve the accuracy of computed color probabilities. A pose recovery method is also proposed in [18] to handle the cases where the object is lost.

Based on the method of [18], Tjaden *et al.*[19] further re-weighted the energy function, and used the Gauss-Newton strategy to optimize the pose. This improves the convergence rate of the optimization. It is also shown that their method is capable of tracking multiple objects simultaneously. Zhong *et al.*[26] used overlapping fan-shaped regions to build the local color model without other sources to speed up the model building process. This requires fewer local regions and gets a similar or better segmentation result. They further proposed an explicit way to deal with the occlusion based on the distance and color information of the contour and edge points to determine the occlusion weight.

2.3 Other State-of-the-Art Methods

In addition to the above two kinds of methods, several other tracking strategies are proposed and show excellent results. For instance, Tan *et al.*[4] used the random forest algorithm to regress the pose of the object. This method, however, needs the depth data. Convolutional neural networks (CNNs)[27] are also used for tracking[8, 9, 28]. However, these methods usually need a large amount of training data and often demonstrate a poor generalization performance. They also often require a pre-training process, which is computationally intensive and requires the GPU support. Real-time operation of such methods however is still not possible on ordinary devices.

In another development, Zhong and Zhang[29] fused statistical and photometric constraints for 3D tracking, incorporated the color features and geometric constraints into an energy function and used a weight coefficient to appropriately balance the metrics. This method combines the advantages of the two features. However, direct fusion is sub-optimal and it is thus unable to fully exploit the advantages of each feature. Besides, adjusting the balance parameter requires experiments, and the parameter may need to be re-adjusted for different environments.

3 Proposed Method

In this section, we devise an optimized multi-feature fusion method with adaptively weighted local bundles, named AWLB tracker. The AWLB tracker uses local bundles to fuse multiple features and adaptively adjusts their weights. The weights are adaptively adjusted using a spatially-variant function based on the confidence of each feature. In the following, we first introduce preliminaries and then elaborate on the proposed method.

3.1 Preliminaries

Given the object model with vertices $\boldsymbol{X}_i \in \mathbb{R}^3$, the camera internal parameters $\boldsymbol{K} \in \mathbb{R}^{3\times 3}$ and the pose of object $\boldsymbol{T} \in \mathbb{R}^{4\times 4}$, we can obtain the mapping from the object coordinate $\boldsymbol{X}_i \in \mathbb{R}^3$ to the image coordinate $\boldsymbol{x}_i \in \mathbb{R}^2$ based on the pinhole camera model as the following:

$$\boldsymbol{x} = \pi(\boldsymbol{K}(\boldsymbol{T}\tilde{\boldsymbol{X}})_{3\times 1}),$$

where $\tilde{\boldsymbol{X}} = (X, Y, Z, 1)^{\mathrm{T}}$ represents the homogeneous coordinates of $\boldsymbol{X}$, $\pi(\boldsymbol{X}) = (X/Z, Y/Z)^{\mathrm{T}}$.

The pose $\boldsymbol{T}$ of the object maps the model coordinate to the camera coordinate, which can be then represented as a 4×4 homogeneous matrix by Lie-group $\mathbb{SE}(3)$, i.e.:

$$\boldsymbol{T} = \begin{pmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{pmatrix} \in \mathbb{SE}(3),$$
$$\text{with } \boldsymbol{R} \in \mathbb{SO}(3) \text{ and } \boldsymbol{t} \in \mathbb{R}^3.$$

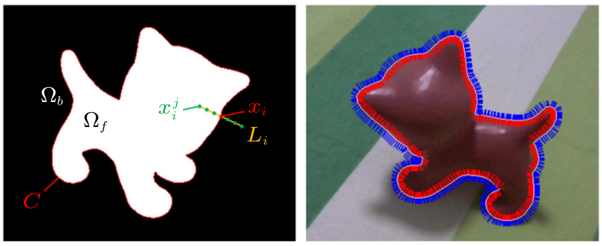
$\mathbb{SO}(3)$ constructs the orthometric group. Here we adopt the parametric form of Lie-algebra to optimize the pose. The Lie-algebra $\mathfrak{se}(3)$ corresponding to the Lie-group $\mathbb{SE}(3)$ is formulated as a vector $\boldsymbol{\xi} \in \mathbb{R}^6$ or its twist $\hat{\boldsymbol{\xi}} \in \mathbb{R}^{4\times 4}$. A detailed introduction to Lie-group and Lie-algebra can be found in [30]. The exponential mapping of the matrix establishes the relationship between Lie-group $\mathbb{SE}(3)$ and Lie-algebra $\mathfrak{se}(3)$:

$$\boldsymbol{T} = \exp(\hat{\boldsymbol{\xi}}) \in \mathbb{SE}(3).$$

3.2 3D Tracking with Local Bundles

In order to suspend the effect of unreliable observations in color and edge features, we can assign each pixel or edge point an individual weight adjusted with the confidence of features. This approach, however, ignores the local competition between different features, and the weight of each single feature is also not easy to be estimated stably. Both problems can be alleviated with the proposed local bundle model, which gathers features in different regions with a set of local structures (bundles) for a spatial-variant weighting, and the features inside each bundle are also competitively weighted to optimize their complementarity.

Specifically, given the pose $\boldsymbol{\xi}$ of the object, we can render the object contour C, as shown in Fig.2. For the contour point $\boldsymbol{x}_i$ on contour C, we calculate its normal vector according to the contour direction and then draw the local bundle L_i. The bundle L creates a sub-region that associates a contour point with its foreground and background. The length of L is set to 17 (including one contour point, eight foreground points, and eight background points). The choice of this value is the same as in [19], which is an empirical value. Combined with the multi-scale strategy, this value can get the optimal balance point in the calculation speed and accuracy. Furthermore, $\boldsymbol{x}_i^j$ is the region point on L_i. Notice that $\boldsymbol{x}_i \in \boldsymbol{x}_i^j$, i.e., the contour point $\boldsymbol{x}_i$ is one of the region points $\boldsymbol{x}_i^j$ on L_i.



(a) (b)

Fig.2. Illustration of the local bundles. Each bundle consists of a set of points on a line segment perpendicular to the contour: (a) bundle L_i at the contour point $\boldsymbol{x}_i$; (b) bundles around the object contour, with the red and the blue parts falling in the interior and the exterior regions of the object, respectively.

The energy function with local bundles is defined as follows:

$$E(\boldsymbol{\xi}) = \sum\nolimits_{\boldsymbol{x}_i \in C} \omega_i E_{\text{bundle}}(\boldsymbol{x}_i, \boldsymbol{\xi}),$$

where $E_{\text{bundle}}(\boldsymbol{x}_i, \boldsymbol{\xi})$ is the bundle energy cost corresponding to the i-th bundle, and ω_i is a spatially-variant adaptive weighting function. The bundle energy is defined as:

$$\begin{aligned} E_{\text{bundle}}(\boldsymbol{x}_i, \boldsymbol{\xi}) = {} & \alpha_i e_{\text{edge}}(\boldsymbol{x}_i, \boldsymbol{\xi}) + \\ & \beta_i \sum\nolimits_{\boldsymbol{x}_i^j \in L_i} \lambda e_{\text{color}}(\boldsymbol{x}_i^j, \boldsymbol{\xi}), \end{aligned} \quad (1)$$

where e_{edge} and e_{color} are the edge and the color energy terms, respectively. Further in (1), we borrow the energy function in [13] and [19], as

$$e_{\text{edge}}(\boldsymbol{x}_i, \boldsymbol{\xi}) = \left(\boldsymbol{D}(\pi(\boldsymbol{K}(\exp(\hat{\boldsymbol{\xi}})\tilde{\boldsymbol{X}}_i)_{3\times 1}))\right)^2, \quad (2)$$

$$\begin{aligned} e_{\text{color}}(\boldsymbol{x}_i^j, \boldsymbol{\xi}) = {} & -\log\left(H_e(\Phi(\boldsymbol{x}_i^j(\boldsymbol{\xi})))P_f(\boldsymbol{x}_i^j) + \right. \\ & \left.(1 - H_e(\Phi(\boldsymbol{x}_i^j(\boldsymbol{\xi}))))P_b(\boldsymbol{x}_i^j)\right). \end{aligned} \quad (3)$$

Specifically, (2) represents the value of the projection point $\boldsymbol{x}_i$ of the object's contour point $\boldsymbol{X}_i$ in the edge distance field $\boldsymbol{D}$, and (3) represents the color posterior probability of the region point $\boldsymbol{x}_i^j$. Then the optimal pose can be solved by minimizing the energy function. Note that in (1), α_i and β_i are the adaptive weights of the edge and the color energies, respectively. We further enforce $\alpha_i + \beta_i = 1$, thereby the edge and the color features are competitive with respect to their confidence (see Subsection 3.3.3). The constant parameter λ is also preserved to balance the overall effect of the color and edge features. We show that using the confidence-based adaptive weighting, λ can be easily set. Each bundle consists of one edge point and multiple color points. The multiple color points are also bundled together and share the same weight β_i. Therefore, we can easily define the competitive weights, and at the same time, improve the stability of the estimated β_i by summing up throughout each bundle.

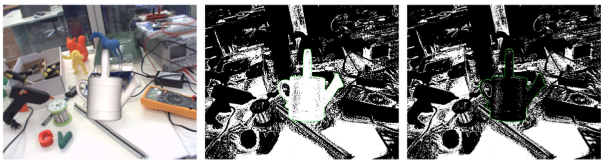
As it is seen, the bundles form a set of local regions that divide the sampled contour and region points into smaller subsets. This is mainly to deal with the spatial inconsistency of color and edge features. The energy terms in each bundle are independent, and this enables them to fit the particular case in each sub-region and take the full advantage of each existing feature. Although the bundles can be created in other ways, our method as illustrated in Fig.2 is a natural choice since it encodes the most informative features to estimate the object motion along the sample line.

3.3 Adaptive Weighting of Local Bundles

By fusing the color and edge features in each bundle, we can then weight the features based on their quality. To measure the quality of the features we introduce confidence. The confidence is obtained for the color and edge features to ensure their independence and it is also normalized to a value in $[0, 1]$. Using confidence, we can then measure the quality of each feature. The weights α_i, β_i, and ω_i are then adaptively obtained via a spatially-variant weighting function based on the confidence.

3.3.1 *Confidence of the Region Points*

We use Ω_f and Ω_b to represent the foreground and the background respectively. To distinguish the foreground and the background [19] uses local histograms and mean probabilities. However this method is unable to efficiently distinguish the foreground and the background in some complex environments. To address this issue, here we borrow the idea of [31] to construct an uncleared region Ω_u for indistinguishable colors, and then use it to calculate the confidence of the region points. Fig.3(a) is the input image and Fig.3(b) shows per-pixel segmentation visualized as $P_f(\boldsymbol{x}) - P_b(\boldsymbol{x}) > 0$. It is seen that the color of the object is similar to the background color, especially the lower part of the object, which may distract the optimization. Specifically, for the cases where $\boldsymbol{x}$ is in the foreground, but $P_f < P_b$, or $\boldsymbol{x}$ is in the background, but $P_b < P_f$, we obtain the color at $\boldsymbol{x}$ to Ω_u. Fig.3(c) illustrates the uncleared region Ω_u constructed according to the above, where the green line represents the contour of the object.



(a) (b) (c)

Fig.3. (a) The first frame of the regular variant of the Can model in the RBOT dataset [19], where the color of the object is similar to the surrounding background. (b) Per-pixel segmentation visualized as $P_f(\boldsymbol{x}) - P_b(\boldsymbol{x}) > 0$. (c) Unclear region Ω_u of the image.

We collect Ω_u on the full image and update it every S frames. The reasons for adopting this strategy include: 1) collecting Ω_u on the full image uses the global color information and 2) the moving distance between the frames is small during the tracking; thus the change of Ω_u is negligible. It takes much less time to perform full image statistics every S frames than to perform local averaging every frame.

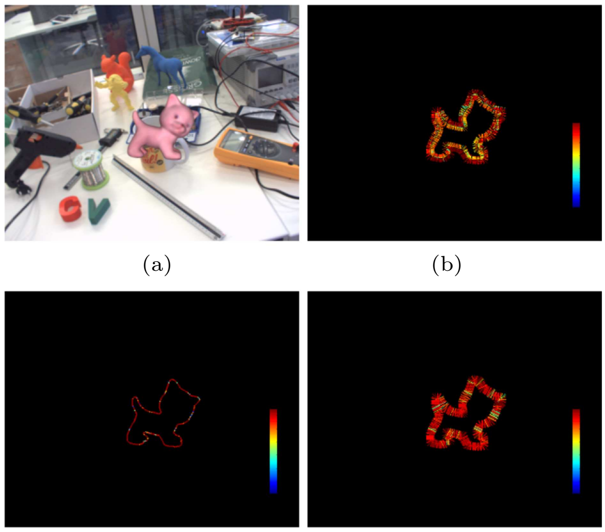
For each region point $\boldsymbol{x}_i^j$, we now can obtain its confidence $c_{\text{color}}(\boldsymbol{x}_i^j)$ by:

$$c_{\text{color}}(\boldsymbol{x}_i^j) = 1 - \frac{P(\boldsymbol{y}_i^j|\Omega_u)}{P(\boldsymbol{y}_i^j|\Omega_u) + P(\boldsymbol{y}_i^j|\Omega_f) + P(\boldsymbol{y}_i^j|\Omega_b)},$$

where $\boldsymbol{y}_i^j$ is the color value at $\boldsymbol{x}_i^j$ on the image, and $P(\boldsymbol{y}_i^j|\Omega_u)$, $P(\boldsymbol{y}_i^j|\Omega_f)$ and $P(\boldsymbol{y}_i^j|\Omega_b)$ indicate the color models of the uncleared region, foreground region and background region, respectively. We can see that the point with a higher probability in the unclear region will have lower confidence. Fig.4(b) shows an example of the color confidence. The color model of Ω_u is recursively adjusted by:

$$P(\boldsymbol{y}|\Omega_u) = (1-\tau)P^{t-S}(\boldsymbol{y}|\Omega_u) + \tau P^t(\boldsymbol{y}|\Omega_u),$$

where t is current frame index and τ is the decay factor.



(a) (b)

(c) (d)

Fig.4. (a) The first image of regular variant of the Cat model in the RBOT dataset, (b) the color confidence, (c) the contour confidence, and (d) weights of bundles corresponding to the input image.

In our proposed approach, the confidence of the region points is used to calculate the weight of the color energy term as mentioned in (1). However, if we do not use the feature fusion strategy, we can still use the confidence to weight the color energy term, i.e., by adding the confidence of each point to the corresponding cost term. This improves the performance of the region-based method, which is also illustrated in Subsection 4.2.

3.3.2 Confidence of the Contour Points

We use the gradient direction to calculate the confidence of the contour points because the gradient is the most important property of the edge. For the contour point $\boldsymbol{x}_i$ on image I, we formulate its confidence $c_{\text{edge}}(\boldsymbol{x}_i)$ as:

$$c_{\text{edge}}(\boldsymbol{x}_i) = |\cos(ori^I(\boldsymbol{x}_i) - ori^{I'}(\boldsymbol{x}_i))|,$$

where $ori^I(\boldsymbol{x}_i)$ represents the gradient direction at $\boldsymbol{x}_i$ on the image, and $ori^{I'}(\boldsymbol{x}_i)$ is the gradient direction of the object contour, which represents the normal direction of the contour point $\boldsymbol{x}_i$. Fig.4(c) shows an example of the contour confidence. This idea is inspired from [13], and we further use the normal information of the projected contour for geometric consistency, which combines the geometric properties of the object model. Furthermore, the confidence of the contour point is used to calculate the weight of the energy term, which is the same as the confidence of region points. The definition of confidence is also robust to the outliers (occlusion or disappearance of image edges). Because the edge direction of the outlier and the direction of the projection contour point do not match, the confidence at the outlier is small. Besides, this calculation method requires minimal computational resources.

Both c_{edge} and c_{color} are naturally distributed between 0 and 1 and do not involve threshold parameters. This enables our method to flexibly choose the weights of features and also become highly tolerant against the environmental variables.

3.3.3 Weights

We use a spatially-variant weighting function based on the confidence calculated above to adaptively weight the energy term. For the i-th local bundle L_i, we first calculate the average confidence of the region points it contains as:

$$\bar{c}^i_{\text{color}} = \frac{1}{|L_i|}\sum\nolimits_{\boldsymbol{x}_i^j \in L_i}^{|L_i|} c_{\text{color}}(\boldsymbol{x}_i^j).$$

The weights of the edge term α_i, the color term β_i, and the bundle term ω_i are also obtained as:

$$\begin{aligned}
\alpha_i &= \frac{c_{\text{edge}}(\boldsymbol{x}_i)}{\bar{c}^i_{\text{color}} + c_{\text{edge}}(\boldsymbol{x}_i)},\\
\beta_i &= \frac{\bar{c}^i_{\text{color}}}{\bar{c}^i_{\text{color}} + c_{\text{edge}}(\boldsymbol{x}_i)},\\
\omega_i &= \begin{cases} 0, & \text{if } \bar{c}^i_{\text{color}} < \gamma \,\&\, c_{\text{edge}}(\boldsymbol{x}_i) < \gamma,\\ \dfrac{\bar{c}^i_{\text{color}} + c_{\text{edge}}(\boldsymbol{x}_i)}{2}, & \text{otherwise,}\end{cases}
\end{aligned}$$

where α_i and β_i are normalized and directly obtained from the confidence. Note that ω_i weights each bundle and see Fig.4(d) for example. Considering that the number of bundles in each iteration may be different, we do not use a normalization strategy to ω_i. The bundle with lower weight can weaken the negative impact of the untrusted sub-region. For $\bar{c}^i_{\text{color}} < \gamma$ and $c_{\text{edge}}(\boldsymbol{x}_i) < \gamma$, the confidence of both the contour point and the region points on L_i is very small. In such cases, we simply eliminate it to avoid its negative effect. We further emphasize that our method does not need to calculate the costs when calculating weights, and also does not need to unify metrics, and the energy term and bundles are adaptively weighted.

3.4 Pose Optimization

For pose optimization we use the Gauss-Newton scheme presented in [19] and extend it to our multi-feature cost function. We also note that (3) does not have square terms and thus cannot directly use a second-order optimization strategy. We use the modified version of (3) as in [19] so that it can be optimized using the Gauss-Newton method. The color energy function is therefore rewritten as:

$$\widetilde{e}_{\text{color}}(\boldsymbol{x}_i^j, \boldsymbol{\xi}) = \frac{1}{2}\psi(\boldsymbol{x}_i^j)e^2_{\text{color}}(\boldsymbol{x}_i^j, \boldsymbol{\xi}),$$

with $\psi(\boldsymbol{x}_i^j) = 1/(e_{\text{color}}(\boldsymbol{x}_i^j, \boldsymbol{\xi}))$. For the edge energy term, (2) includes the square term and therefore does not require modification. The Jacobian of $e_{\text{color}}(\boldsymbol{x}_i^j, \boldsymbol{\xi})$ and $e_{\text{edge}}(\boldsymbol{x}_i, \boldsymbol{\xi})$ at the pose $\boldsymbol{\xi}$ are:

$$\begin{aligned}
\boldsymbol{J}_{\text{color}}(\boldsymbol{x}_i^j) &= \tfrac{\partial e_{\text{color}}(\boldsymbol{x}_i^j, \boldsymbol{\xi})}{\partial \boldsymbol{\xi}},\\
\boldsymbol{J}_{\text{edge}}(\boldsymbol{x}_i) &= \tfrac{\partial e_{\text{edge}}(\boldsymbol{x}_i, \boldsymbol{\xi})}{\partial \boldsymbol{\xi}}.
\end{aligned}$$

Specifically, for the i-th bundle, we express its Jacobian matrix and Hessian matrix as $\boldsymbol{J}_i$ and $\boldsymbol{H}_i$, respectively. Each of them can be divided into the edge part and the color part, i.e.:

$$\begin{aligned}
\boldsymbol{J}_i &= \alpha_i \boldsymbol{J}_i^{\text{edge}} + \beta_i \boldsymbol{J}_i^{\text{color}},\\
\boldsymbol{H}_i &= \alpha_i \boldsymbol{H}_i^{\text{edge}} + \beta_i \boldsymbol{H}_i^{\text{color}}.
\end{aligned}$$

For the edge part,

$$\begin{aligned}
\boldsymbol{J}_i^{\text{edge}} &= \boldsymbol{J}_{\text{edge}}(\boldsymbol{x}_i) = \frac{\partial e_{\text{edge}}(\boldsymbol{x}_i, \boldsymbol{\xi})}{\partial \boldsymbol{\xi}}\\
&= \frac{\partial e_{\text{edge}}(\boldsymbol{x}_i, \boldsymbol{\xi})}{\partial \boldsymbol{x}_i} \cdot \frac{\partial \boldsymbol{x}_i}{\partial \boldsymbol{\xi}},\\
\boldsymbol{H}_i^{\text{edge}} &= \boldsymbol{J}_i^{\text{edgeT}} \cdot \boldsymbol{J}_i^{\text{edge}} = \boldsymbol{J}_{\text{edge}}(\boldsymbol{x}_i)^{\mathrm{T}} \cdot \boldsymbol{J}_{\text{edge}}(\boldsymbol{x}_i).
\end{aligned}$$

For the color part,

$$\begin{aligned}
\boldsymbol{J}_i^{\text{color}} &= \sum\nolimits_{\boldsymbol{x}_i^j \in L_i}^{|L_i|} \lambda \boldsymbol{J}_{\text{color}}(\boldsymbol{x}_i^j)\\
&= \sum\nolimits_{\boldsymbol{x}_i^j \in L_i}^{|L_i|} \lambda \frac{\partial e_{\text{color}}(\boldsymbol{x}_i^j, \boldsymbol{\xi})}{\partial \boldsymbol{\xi}}\\
&= \sum\nolimits_{\boldsymbol{x}_i^j \in L_i}^{|L_i|} \lambda G \delta_e \frac{\partial \Phi(\boldsymbol{x}_i^j(\boldsymbol{\xi}))}{\partial \boldsymbol{\xi}}, \qquad (4)\\
\boldsymbol{H}_i^{\text{color}} &= \sum\nolimits_{\boldsymbol{x}_i^j \in L_i}^{|L_i|} \lambda \psi(\boldsymbol{x}_i^j) J_{\text{color}}(\boldsymbol{x}_i^j)^{\mathrm{T}} J_{\text{color}}(\boldsymbol{x}_i^j). \qquad (5)
\end{aligned}$$

In (4), $G = \frac{P_b(\boldsymbol{x}_i^j) - P_f(\boldsymbol{x}_i^j)}{H_e(\Phi(\boldsymbol{x}_i^j))(P_f(\boldsymbol{x}_i^j) - P_b(\boldsymbol{x}_i^j)) + P_b(\boldsymbol{x}_i^j)}$ and $\delta_e = \delta_e(\Phi(\boldsymbol{x}_i^j))$ is the smoothed Dirac delta function. In (5), $\psi(\boldsymbol{x}_i^j) J_{\text{color}}(\boldsymbol{x}_i^j)^{\mathrm{T}} J_{\text{color}}(\boldsymbol{x}_i^j)$ is the Hessian matrix of one region point. Some optimization details can refer to [19].

The update step of each iteration for all bundles is also formulated as:

$$\Delta\boldsymbol{\xi} = -\boldsymbol{H}^{-1}\boldsymbol{J}^{\mathrm{T}} = -\left(\sum\nolimits_i^{|C|} \omega_i \boldsymbol{H}_i\right)^{-1} \sum\nolimits_i^{|C|} \omega_i \boldsymbol{J}_i^{\mathrm{T}},$$

where

$$\begin{aligned}\boldsymbol{J}_i &= \alpha_i \boldsymbol{J}_{\text{edge}}(\boldsymbol{x}_i) + \beta_i \sum\nolimits_{\boldsymbol{x}_i^j \in L_i}^{|L_i|} \lambda \boldsymbol{J}_{\text{color}}(\boldsymbol{x}_i^j),\\ \boldsymbol{H}_i &= \alpha_i \boldsymbol{J}_{\text{edge}}(\boldsymbol{x}_i)^{\mathrm{T}} \boldsymbol{J}_{\text{edge}}(\boldsymbol{x}_i) + \\ &\quad \beta_i \sum\nolimits_{\boldsymbol{x}_i^j \in L_i}^{|L_i|} \lambda \psi(\boldsymbol{x}_i^j) \boldsymbol{J}_{\text{color}}(\boldsymbol{x}_i^j)^{\mathrm{T}} \boldsymbol{J}_{\text{color}}(\boldsymbol{x}_i^j).\end{aligned}$$

Because we divide the optimization point by the local bundle L, the $\boldsymbol{J}^{\mathrm{T}}\boldsymbol{J}$ term must be calculated according to this division, and cannot be summed directly. Otherwise, it cannot play the role of weight item. We perform the optimization on three scales with four iterations on the 1/4 image, two iterations on the 1/2 image, and one iteration on the original image.

4 Experiments

We evaluate the performance of the proposed approach on a laptop equipped with an Intel® Core™ i7-8565U @1.8 GHz processor, 8 GB RAM, and an NVIDIA GeForce MX250 GPU. We use a set of default parameters for all experiments, including $S = 100$, $\tau = 0.8$, and $\gamma = 0.5$. We further set λ to 1, unless otherwise specified, and we also clip each model to a maximum of 5 000 vertices.

4.1 Comparisons in 3D Tracking Datasets

We compare the proposed AWLB tracker with the state-of-the-art methods, using the RBOT dataset [19] and the OPT dataset [32], respectively. And we further show two challenging examples based on real scenarios.

4.1.1 RBOT Dataset

The RBOT dataset [19] is a synthetic dataset of images with 640×512 px resolution, where the background is a real scene image, and the object model is used to render the foreground. The dataset consists of 18 objects and each contains four sets of variants, including regular (Reg.), dynamic light (Dyn.), noisy+dynamic light (Noi.), and occlusion (Occ.).

Here, we use the same evaluation method as in [19]. For the k-th frame at the j-th sequence, we then obtain the tracking error for translation and rotation as follows:

$$\begin{aligned}e_k^j(\boldsymbol{t}) &= \|\boldsymbol{t}^j(k) - \boldsymbol{t}_{gt}^j(k)\|_2,\\ e_k^j(\boldsymbol{R}) &= \cos^{-1}\left(\frac{\text{trace}\left(\boldsymbol{R}^j(k)^{\mathrm{T}} \boldsymbol{R}_{gt}^j(k)\right) - 1}{2}\right).\end{aligned}$$

If $e_k^j(\boldsymbol{t}) < 5$ cm and $e_k^j(\boldsymbol{R}) < 5°$, the pose is successfully tracked. Otherwise, the pose is reset to the ground truth pose. The accuracy of all poses in the sequence is then obtained by counting the instances.

Table 1 presents a detailed accuracy of the proposed method as well as the other four tracking methods [18, 19, 26, 29], and all the results are taken from their corresponding references. The bold value in the table corresponds to the method with the highest accuracy. The results confirm that our method illustrates the effectiveness of the multi-feature fusion strategy with confidence and it is performing better than the others, and especially in the noisy+dynamic light variant, the average accuracy rate is improved by about 7.8% compared with [26]. This is because the image of the noisy+dynamic light variant contains a lot of random noise. This makes the color histograms of the foreground and the background similar to each other, leading to an unreliable corresponding probability. The proposed method fuses the advantage of the edge feature which is less affected by the noise and hence greatly improves the accuracy.

For the regular and dynamic light variables, our method also improves by 3.1% and 5.4%. For occlusion variant, our mean accuracy is still 1.9% higher than that in [26] even if it adopts an explicit way to handle occlusion. [29] is also a feature fusion method that fuses color features and photometric constraints. But this method adopts a direct fusion way, and its result is only slightly better than the region-based [19] and far less than that of the proposed method. This confirms the effectiveness of our proposed approach.

4.1.2 OPT Dataset

The OPT dataset [32] is a real dataset with six objects, i.e., bike, chest, house, ironman, jet and soda, where each object contains seven motion patterns. We evaluate our method by using all RGB image sequences at 1 920×1 080 px resolution. The pose error of the k-th frame at the j-th sequence is:

$$e_k^j = \frac{1}{n}\sum\nolimits_{i=1}^{n} \|(\boldsymbol{T}^j(k)\tilde{\boldsymbol{X}}_i - \boldsymbol{T}_{gt}^j(k)\tilde{\boldsymbol{X}}_i)_{3\times 1}\|_2.$$

In this setting, the pose is successfully tracked if $e_k^j < \lambda_e d_m$, where λ_e is a predefined threshold, and d_m is the largest distance between the vertices of the model. Within the tracking process, only the first frame of the ground truth is used for initialization. If the tracking fails, no recovery is taken. For a given λ_e, we then obtain the accuracy which is between 0 and 100. The final tracking accuracy is measured by AUC (area under curve) score for all $\lambda_e \in [0, 0.2]$, meaning the AUC score is between 0 and 20.

Table 1. Tracking Accuracy (%) for the Proposed Method on the RBOT Dataset Compared with Other State-of-the-Art Methods

Variant	Method	Ape	Baking Soda	Bench Vise	Broccoli Soup	Camera	Can	Cat	Clown	Cube	Driller	Duck	Egg Box	Glue	Iron	Koala Candy	Lamp	Phone	Squirrel	Avg.
Reg.	ICCV17 [18]	62.1	30.5	95.8	66.2	61.6	81.7	96.7	89.1	44.1	87.7	74.9	50.9	20.2	68.4	20.0	92.3	64.9	98.5	67.0
	TPAMI19 [19]	85.0	39.0	**98.9**	82.4	79.7	87.6	95.9	93.3	78.1	93.0	86.8	74.6	38.9	81.0	46.8	**97.5**	80.7	99.4	79.9
	IJCV19 [29]	82.6	40.1	92.6	85.0	82.8	87.2	98.0	92.9	81.3	84.5	83.3	76.2	56.1	84.6	57.6	90.5	82.6	95.6	80.8
	TIP20 [26]	88.8	41.3	94.0	85.9	86.9	**89.0**	98.5	93.7	83.1	87.3	86.2	78.5	**58.6**	86.3	57.9	91.7	85.0	96.2	82.7
	Proposed	**92.8**	**42.6**	96.8	**87.5**	**90.7**	86.2	**99.0**	**96.9**	**86.8**	**94.6**	**90.4**	**87.0**	57.6	**88.7**	**59.9**	96.5	**90.6**	**99.5**	**85.8**
Dyn.	ICCV17 [18]	61.7	32.0	94.2	66.3	68.0	84.1	96.6	85.8	45.7	88.7	74.1	56.9	29.9	49.1	20.7	91.5	63.0	98.5	67.0
	TPAMI19 [19]	84.9	42.0	**99.0**	81.3	84.3	88.9	95.6	92.5	77.5	94.6	86.4	77.3	52.9	77.9	47.9	**96.9**	81.7	**99.3**	81.2
	IJCV19 [29]	81.8	39.7	91.5	85.1	82.6	87.1	98.1	90.7	79.7	87.4	81.6	73.1	51.7	75.9	53.4	88.8	78.6	95.6	79.0
	TIP20 [26]	89.7	40.2	92.7	86.5	86.6	**89.2**	98.3	93.9	81.8	88.4	83.9	76.8	55.3	79.3	54.7	88.7	81.0	95.8	81.3
	Proposed	**93.5**	**43.1**	96.6	**88.5**	**92.8**	86.0	**99.6**	**95.5**	**85.7**	**96.8**	**91.1**	**90.2**	**68.4**	**86.8**	**59.7**	96.1	**91.5**	99.2	**86.7**
Noi.	ICCV17 [18]	55.9	35.3	75.4	67.4	27.8	10.2	94.3	33.4	8.6	50.9	76.3	2.3	2.2	18.2	11.4	36.6	31.3	93.5	40.6
	TPAMI19 [19]	77.5	**44.5**	91.5	82.9	51.7	38.4	95.1	69.2	24.4	64.3	88.5	11.2	2.9	46.7	32.7	57.3	44.1	96.6	56.6
	IJCV19 [29]	80.5	35.0	80.9	85.5	58.4	53.5	96.7	65.9	38.2	71.8	85.8	29.7	17.0	59.3	34.8	61.1	60.8	93.6	61.6
	TIP20 [26]	79.3	35.2	82.6	86.2	65.1	56.9	96.9	67.0	37.5	75.2	85.4	35.2	18.9	63.7	35.4	64.6	66.3	93.2	63.6
	Proposed	**89.1**	44.0	**91.6**	**89.4**	**75.2**	**62.3**	**98.6**	**77.3**	**41.2**	**81.5**	**91.6**	**54.5**	**31.8**	**65.0**	**46.0**	**78.5**	**69.6**	**97.6**	**71.4**
Occ.	ICCV17 [18]	55.2	29.9	82.4	56.9	55.7	72.2	87.9	75.7	39.6	78.7	68.1	47.1	26.2	35.6	16.6	78.6	50.3	77.6	57.5
	TPAMI19 [19]	80.0	42.7	91.8	73.5	76.1	81.7	89.8	82.6	68.7	86.7	80.5	67.0	46.6	64.0	43.6	88.8	68.6	86.2	73.3
	IJCV19 [29]	77.7	37.3	87.1	78.7	74.6	81.0	93.8	84.3	73.2	83.7	77.0	66.4	48.6	70.8	49.6	85.0	73.8	90.6	74.1
	TIP20 [26]	83.9	38.1	**92.4**	81.5	81.3	**85.5**	**97.5**	**88.9**	76.1	87.5	81.7	72.7	52.2	**77.2**	**53.9**	88.5	79.3	92.5	78.4
	Proposed	**89.3**	**43.3**	92.2	**83.1**	**84.1**	79.0	94.5	88.6	**76.2**	**90.4**	**87.0**	**80.7**	**61.6**	75.3	53.1	**91.1**	**81.9**	**93.4**	**80.3**

Table 2 presents the results obtained based on our approach compared with seven other state-of-the-art methods, where the bold value indicates the highest score in each column. In Table 2, PWP3D [15], MTAP19 [13], TPAMI19 [19] and TIP20 [26] are 3D Tracking methods, UDP [33] is a pose estimation method, and ElasticFusion [34] and ORB-SLAM2 [35] are visual SLAM methods. Their results except [13] are available in [19,26,32]. For [13], the results are obtained using the code provided by the authors. We only use one particle, which does not use color information as a constraint and thus it is easy to be lost. Besides, [13] only uses one re-projection process in the calculation and utilizes the L-M method for optimization. This may result in the object being easily trapped in a local minimum during the tracking and thus lost.

In Table 2, ORB-SLAM2 obtains the best results because the objects are well textured and thus stable feature points can be easily found. Our method performs significantly better than UDP, ElasticFusion, PWP3D and MTAP19, and slightly better than TPAMI19 and TIP20. In the OPT dataset, the background surrounding the objects is a white region. Thus the color feature [19, 26] is able to segment the foreground and the background; therefore adding edge features does not significantly improve the result.

4.1.3 Visual Analysis

Fig.5 demonstrates two typical cases and proves that our method can outperform the region-based method of TPAMI19 [19].

The first case is demonstrated under a condition where light changes drastically, as shown in Fig.5(a) and Fig.5(b). The image is obtained from the OPT dataset and zoomed for better visualization. We can see that the color of the image changes drastically under the spotlight. In [19], only the color feature is used, and the update of the color model cannot catch up with the color change, resulting in a computed color probability map with errors and leading to a further tracking failure, especially in rotation. Our method incorporates edge features in an optimized way, which is not effected by the illumination changes, and thus can be more robust in this case.

The second one handles a case where the color of the foreground is similar to the background's, and also the object is under the highlight, as shown in Fig.5(c) and Fig.5(d). The Bunny model has a similar color to the background, where it is unreliable to estimate the foreground probability with color distribution. Although local color distribution is adopted in [19] to improve the robustness of estimated color probability, we still often encounter cases that the color feature is less accurate than the edge feature. By fusing both color and edge features with local bundle, our method can achieve better robustness in handling different challenge situations.

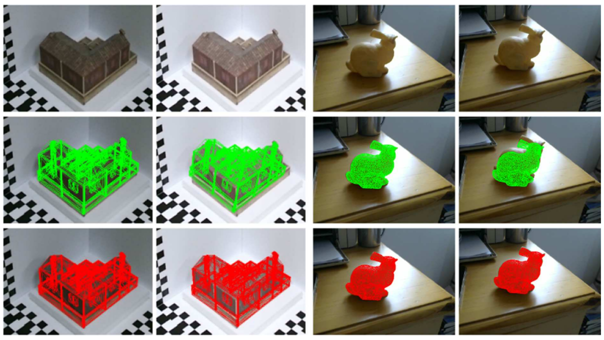
4.2 Ablation Studies

The proposed method is based on adaptively weighted local bundles for fusing the features. Here we evaluate the accuracy of the part to analyze their corresponding contribution in improving efficiency. Here we use the RBOT dataset for our performance evaluations.

In Table 3, the bold value means the highest accuracy in each variant, and the proposed method gets the best results in all variants. In the following, "conf." means "confidence", and "w." and "wo." mean "with" and "without", respectively. The second row is the result of [19] which is considered as the baseline because we borrowed its energy equation as the color feature energy term. The third row shows the results of using the color feature only with confidence. The role played by each region point is weighted by the confidence, and the edge energy terms e_{edge} and the bundle

Table 2. AUC Scores on OPT Dataset of the Proposed Approach Compared with Other Methods

Method	Bike	Chest	House	Ironman	Jet	Soda	Avg.
UDP [33]	6.097	6.791	5.974	5.250	2.342	8.494	5.825
ElasticFusion [34]	1.567	1.534	2.695	1.692	1.858	1.895	1.874
ORB-SLAM2 [35]	10.410	**15.531**	**17.283**	11.198	9.931	**13.444**	**12.966**
PWP3D [15]	5.358	5.551	3.575	3.915	5.813	5.870	5.014
MTAP19 [13]	1.053	8.669	5.599	3.895	1.596	9.055	4.978
TPAMI19 [19]	11.903	11.764	10.150	11.986	13.217	8.861	11.314
TIP20 [26]	12.831	12.240	13.613	11.214	**15.441**	9.012	12.392
Proposed	**12.848**	14.922	13.577	**13.443**	10.642	8.996	12.405



(a) (b) (c) (d)

Fig.5. Two typical cases that our method (the third row) outperforms the region-based method of [19] (the second row). (a)(b) The light is drastically changed between the two frames. (c)(d) The foreground object has similar colors as the background, and the object is under highlight. In both cases it is difficult to get accurate foreground probability with color distribution and so the region-based method will be error-prone.

Table 3. Average Tracking Accuracy (%) on the RBOT Dataset of Different Parts of the Proposed Approach

Varaint	Reg.	Dyn.	Noi.	Occ.
[19]	79.92	81.16	56.64	73.27
w. conf. & wo. edge	83.54	85.13	60.68	77.01
wo. conf. & w. edge	84.58	85.34	68.44	78.42
Proposed	**85.78**	**86.73**	**71.38**	**80.27**

L_i are not included. The results show that the accuracy rate is 3%–4% higher than that in [19]. This confirms the effectiveness of the confidence and confirms that it is applicable in different scenarios.

Further in Table 3, the fourth row shows the results of using fusion features without confidence, i.e., the local bundle structure is removed, and the confidence values are set to a fixed number ($\bar{c}^i_{\rm color} = 1.0$, $c_{\rm edge}(\boldsymbol{x}_i) = 1.0$ and $\omega_i = 1.0$). The improvement is particularly evident in the noisy+dynamic light variant, which shows that using the edge features compensates for the disadvantages associated with the color features. The accuracy of the other three variables is also improved by 4%–5%. This further confirms the effectiveness of multi-feature fusion. In addition, the results show that the effect of adding multiple features is greater than that of adding confidence alone.

The last row in Table 3 shows the results of the proposed method, which uses the confidence and the local bundle to fuse two features. Adjusting the weight of the color energy term and the edge energy term enables them to be complemented. The above results confirm that the weighting based on the confidence value is effective.

Table 4 presents another set of experiments that makes the comparison based on the edge-based method [13], where the bold value means the highest accuracy in each variant. The second row is the result of [13]. For a fair comparison, we modify its optimization strategy to align with the proposed method, i.e., we change the number of re-projection operations to 7 in the pyramid and pick the Gauss-Newton method for the optimization. The results are shown in the third row as [13]+. As it is seen here the results are significantly worse than those of the region-based method [19]. This indicates that on a single feature, the color constraint information is significantly stronger than the edge constraint information. In [13]+, we also eliminate the confidence term.

Table 4. Average Tracking Accuracy (%) on the RBOT Dataset Compared with the Edge-Based Method [13]

Variant	Reg.	Dyn.	Noi.	Occ.
[13]	21.84	22.02	20.74	21.57
[13]+	40.34	43.97	39.46	42.18
w. conf.	43.36	46.92	42.40	44.65
Proposed	**85.78**	**86.73**	**71.38**	**80.27**

The fourth row is the result with confidence, which improves the accuracy by about 3%. It is seen that the accuracy of each variable is slightly different, which also shows that the influence of dynamic light and noise on edge features is small. Adding the edge features also compensates for the disadvantages of the color features.

4.3 Analysis and Discussions

This subsection analyzes the proposed method and discusses its function by presenting some intermediate results. All example images are taken from the RBOT dataset.

4.3.1 Adaptivity to Different Cases

The parameter λ is important for balancing the effect of the color and edge features. The proposed adaptive weighting method is also helpful for setting λ so that the optimal fusion of the features can be achieved in different cases. To verify this we conduct the experiments explained in Table 5, which shows the changes of accuracy on the RBOT dataset for different values of λ. We also obtain the results with and without the confidence-based adaptive weights for different cases. The bold value indicates the highest accuracy among all λ values. As it is seen, the proposed method with adaptive weights achieves the highest accuracy for different

Table 5. Sensitivity to λ in Different Cases

λ	w. conf.						wo. conf.					
	0.5	0.8	1.0	1.2	1.5	2.0	0.5	0.8	1.0	1.2	1.5	2.0
Reg.	83.66	84.84	**85.78**	84.97	84.74	84.69	83.58	84.49	**84.58**	84.38	84.32	84.18
Dyn.	84.88	85.95	**86.73**	86.09	85.96	85.92	84.17	85.41	85.34	**85.78**	**85.78**	85.43
Noi.	70.96	71.13	**71.38**	70.02	70.11	68.12	**70.09**	69.85	68.44	68.59	67.50	66.63
Occ.	79.05	79.37	**80.27**	79.49	79.39	78.92	78.91	**79.52**	78.42	79.34	78.98	78.87

cases all achieved by $\lambda = 1$. In other words, by setting $\lambda = 1$ we get the optimal fusions of color and edge features for all of the different cases. On the contrary, as shown in Table 5, without using the adaptive weights, the highest accuracy for different cases is achieved by very different values of λ. This indicates that the optimal fusion is not achieved by a constant value of λ.

Our method is adaptive to different cases mainly because the weights are adaptively estimated to suspend the effect of unreliable features. The remaining features have high confidence values and the optimal fusion is achieved by uniform weights.

4.3.2 Probability Map

The color-based method mainly depends on the quality of the color probability model. If the background is clear and the foreground color and the background color are distinguishable, then the region-based method can generally get ideal results. Fig.6 shows the intermediate results of the probability map. Fig.6(b)–Fig.6(d) represent the foreground probability map, the background probability map, and the color confidence map calculated by the proposed method respectively. All the maps shown here are calculated on the last iteration during optimization. It should be noted that these calculations do not have to be performed on the full image in the actual optimization and here are just shown on the full map to analyze our results.

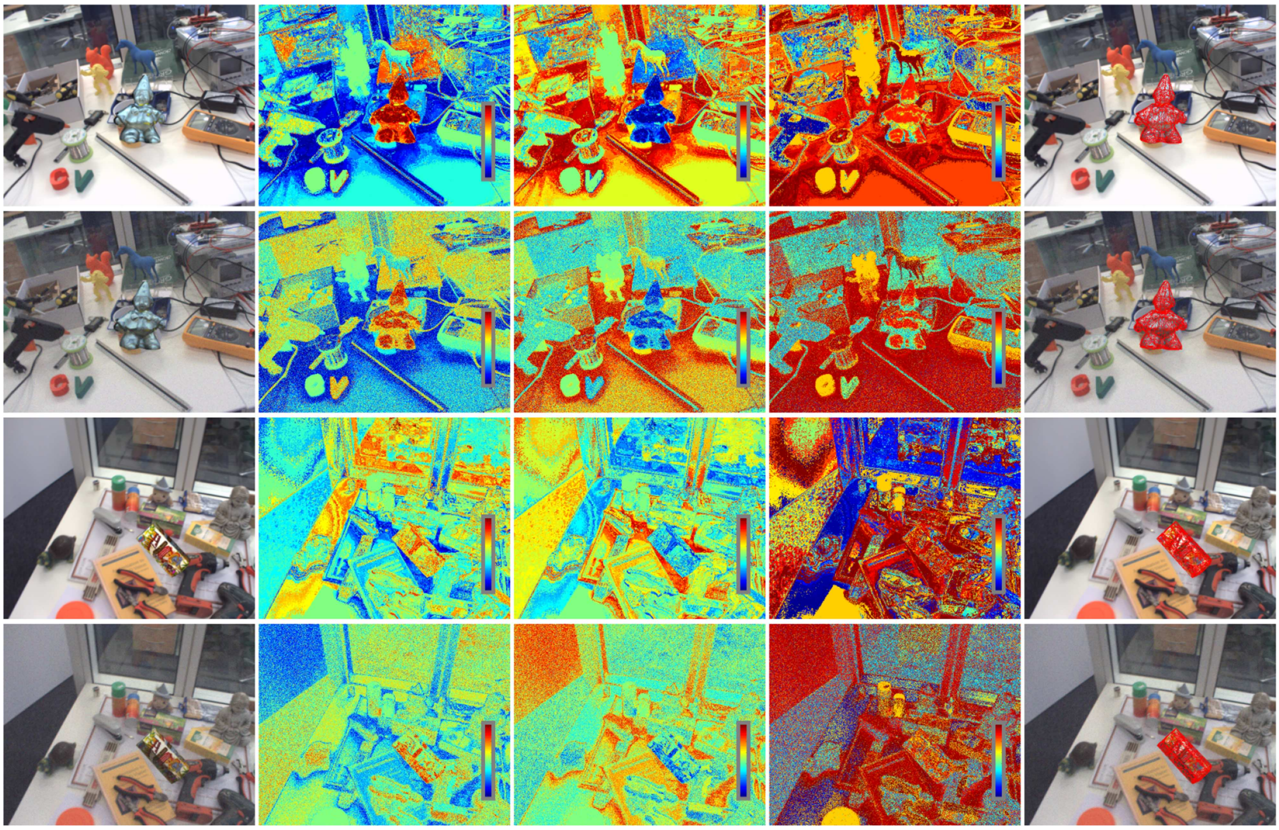
In Fig.6, the first row of the image (frame 0 of the Clown model of regular variant) shows a situation where the foreground color and the background color are easy to identify. In this case, the objects can be easily segmented according to the foreground and the background probability maps. And the calculated confidence values are high in the foreground and surrounding areas. Gaussian noisy and dynamic light are added to the input image in the second row (noisy+dynamic light variant) based on the first one. Although the probability map can also distinguish the foreground and the background regions, its quality has decreased, especially the impact of noise. In this case, our color confidence can play its role. By assigning lower values to areas where the difference is not obvious, the negative impact is reduced.

The third and the fourth rows show another set of examples (frame 528 of the Koalacandy model of regular and noisy+dynamic light variants). Although the foreground color of the image in the third row is complex and the probability map is indiscriminative, the contour of the object can still be segmented from the color around the object. And the confidence of the color around the object is also very credible. This tells us that even if the difference between the probability maps of the foreground and the background is not very obvious (compared with the first row), the color model still works, and the confidence can be used as a complement to the color model. In the fourth row, due to the addition of noise and dynamic light, it is difficult to distinguish the position of the object in the probability map, and the value of the confidence map is generally lower. At this time, we need to add edge features to overcome this shortcoming.

4.3.3 Confidence Values

To better analyze the effect of the confidence values, we select several typical images, as shown in Fig.7. Each row represents an image and its corresponding confidence map. All the confidence maps are obtained at the last iteration during optimization. Fig.7(a)–Fig.7(e) represent the input image, per pixel segmentation visualized as $P_f(\boldsymbol{x}) - P_b(\boldsymbol{x}) > 0$, the confidence value of the region points, the confidence value of the contour points, and the weights of the local bundles, respectively.

For the input image of the first row (frame 11 of the Phone model of regular variant), the color of the phone is similar to the background, especially the part located inside the red box, which is more likely that the color of the background belongs to the foreground. Therefore, if we use the color model directly for optimization, the points inside the red box may negatively affect the result. Furthermore, we can see that the confidence of the region points in the red box is mostly between 0.4–0.8,



(a) (b) (c) (d) (e)

Fig.6. Probability analysis of typical images. (a) Input image. (b) Foreground probability map. (c) Background probability map. (d) Color confidence of each image point. (e) Tracking result.

which reduces the negative impact of similar colors. Although the foreground and the background colors in the red box area are similar, a clear edge is still detectable between them. Therefore, the confidence values of the contour points are very high and most of them are above 0.8. Finally, we can calculate the weight of each local bundle using their confidence values. All the points in the red box are calculated, but the color term plays a smaller role than the edge term. Therefore, the negative effect of the similar colors is reduced and the advantages of contour features are highlighted, resulting in a higher accuracy than that of the other methods.

In the second row (noisy+dynamic light variant), the colors of the foreground and the background overlap with each other, partially due to the dynamic light and the Gaussian noise. This is a challenge for the region-based approach. Compared with the confidence of the region points in the first row, the overall confidence in the second row is lower. However, the addition of these dynamic lights and the Gaussian noise has little impact on edge detection. The confidence of the contour points still reaches a high value. Finally, because the contour points have high confidence, all bundles are still involved in the calculation, but the role of the color term is reduced.

The third and the fourth row are another set of examples (frame 171 of the Camera model of regular and noisy+dynamic light variants), similar to the previous set. It is worth noting that in cases where the confidence of the region points and the confidence of the contour points are both low, the points on the bundle do not participate in the optimization (for details, please refer to the points in the lower-left corner of the red box). Our optimized fusion method weights the features in each bundle based on their confidence; therefore it takes advantage of different features in each local region to achieve better results.

4.3.4 Weights of Features

In this subsection, we analyze the weight of energy terms and bundles, which are both adaptively weighted by confidence. We select the sequence (regular variant of the Ape model) for analysis.

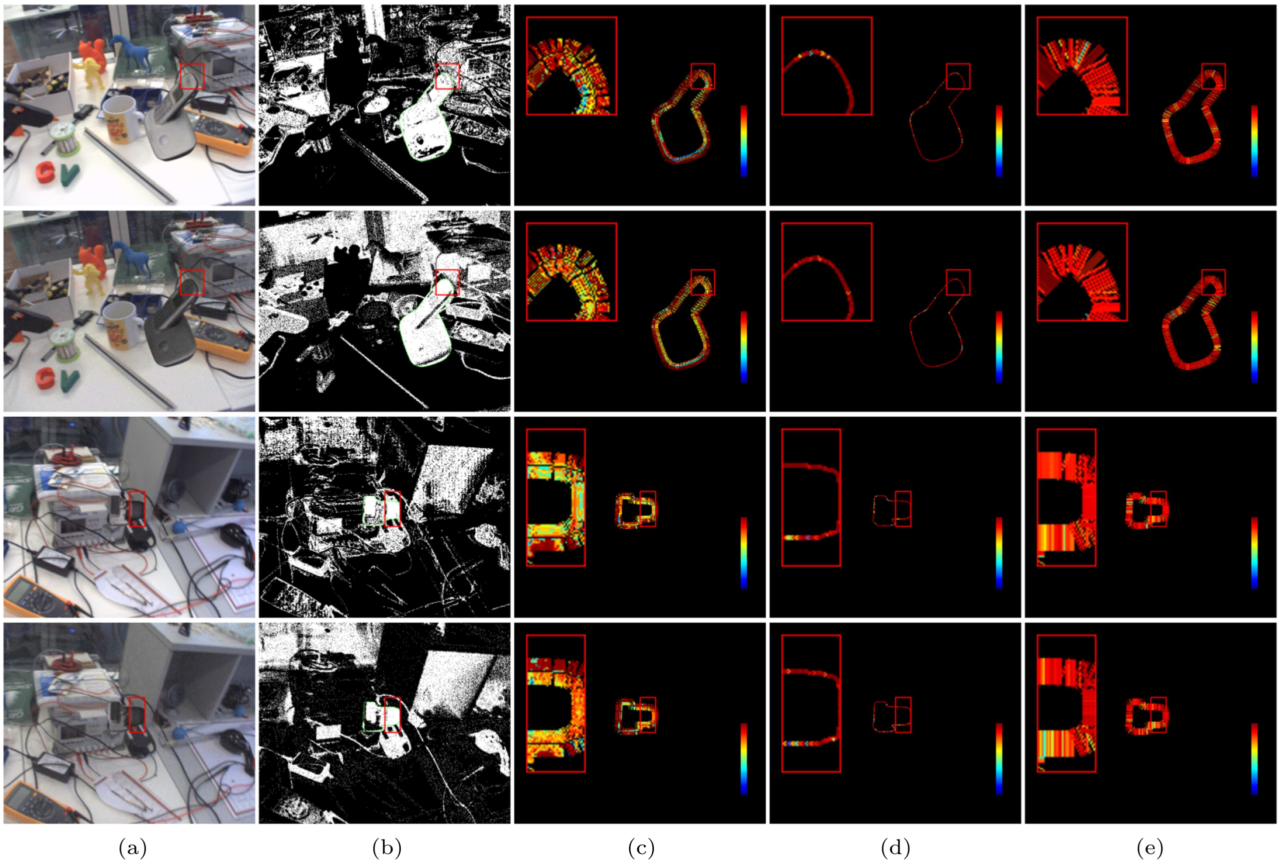


Fig.7. Confidence analysis of typical images. (a) Input image. (b) Per pixel segmentation visualized as $P_f(\boldsymbol{x}) - P_b(\boldsymbol{x}) > 0$. (c) Confidence of the region points. (d) Confidence of the contour points. (e) Weights of the local bundles.

Fig.8(a) shows the trend of color energy term weights. β represents the mean weight of the color term of all the bundles and $\alpha = 1 - \beta$ is the mean weight of the edge term. Besides, β shown in Fig.8(a) stands for the last iteration of the tracking. We can see that the weight of the color term is mainly distributed between 0.4 and 0.6, and its average value is 0.48. In general, the average impact of the color term and the edge term is the same. However, through adaptive adjustment, we can fully exploit their respective advantages. For the weight distribution at other iterations, there is no significant difference from the last iteration. Therefore they are not listed here.

Further, we pick out the images corresponding to the maximum and the minimum values of β. In cases where β reaches its minimum, as shown in the first image, the object is at a position similar to its color. This reduces the weight of the color term. In cases where β reaches its maximum, part of the object exceeds the image, leading to the ill-matched edges. The color item weight then becomes larger. It can be seen that the weights are adjusted accordingly to improve the performance of the algorithm.

Fig.8(b) further illustrates the trend of ω. Here ω represents the average of the weights of all bundles, which can reflect the participation of all optimization points in the calculation. When the value of ω is small, the optimization points have lower reliability and thus lower participation in the calculation. ω is mainly distributed between 0.8–0.9, and its average is 0.86. We select two images corresponding to the minimum value. The first image is the case in which the object exceeds the boundary, and the second image shows the object which is located in a position with a similar color. Corresponding to Fig.8(a), these two cases are the situations where one feature is invalid, and two figures together can reflect the validity of the weight setting.

4.3.5 Robust to Outlier

We further show that the confidence of the contour points can handle occlusion situations. Fig.9(a) shows the image when the object is occluded, and we zoom it to facilitate observation. We use a red box to mark the occluded part. Fig.9(b) shows the confidence of the

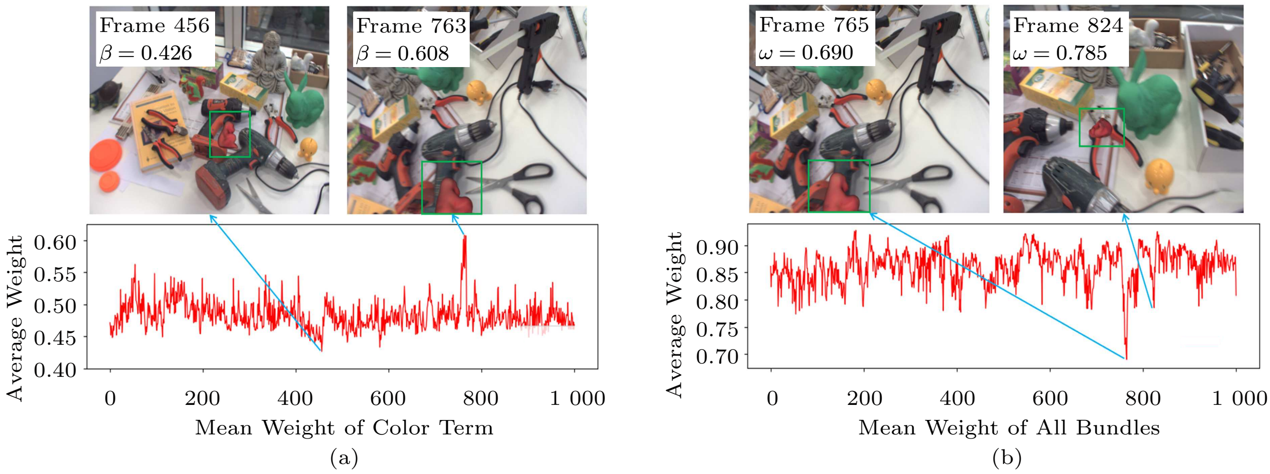


Fig.8. (a) β distribution of the Ape model of regular variant and (b) ω distribution of the Ape model of regular variant. A smaller ω indicates a lower participation level in the overall optimization point. The horizontal axis represents the video frame index.

corresponding contour points. The confidence of the occluded part is generally below 0.4, indicating its robustness to the outliers. Although the confidence of the region points is not designed to consider the influence of the occlusion, due to the contour point confidence, the corresponding bundle still has only a small weight, which can reduce the negative impact of outliers, as shown in Fig.9(c).

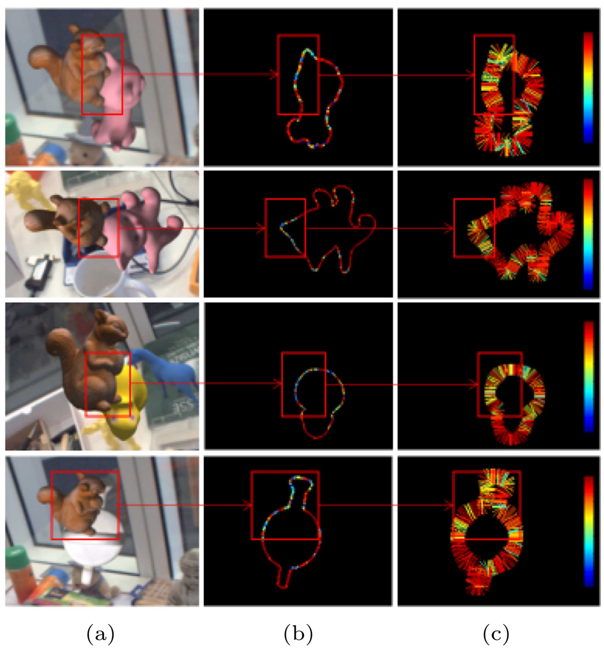


Fig.9. The confidence of the contour points can shield the outliers in the occlusion scene. (a) Input occlusion image. (b) Confidence of the contour points. (c) Weights of the bundles. Our method gives low confidence to the contour points in the occlusion area and makes the corresponding bundle weight lower, shielding the outliers.

4.4 Time Cost

In our experimental environment, the average speed of the proposed method on all sequences of the RBOT dataset is 32.1 ms. We also list the average time of other methods in Table 6. The average time of TPAMI19 [19], ICCV17 [18], and MTAP19 [13] is testing in our experimental environment. For IJCV19 [29] and TIP20 [26], their average time is taken from [26].

Table 6. Runtime Performance Compared with Other Methods

Method	Avg. Time (ms)	Std.
ICCV17 [18]	27.3	2.42
MTAP19 [13]	9.8	0.81
TPAMI19 [19]	26.2	2.31
IJCV19 [29]	47.0	–
TIP20 [26]	41.2	–
Proposed	32.1	3.05

4.5 Limitations

An optimized multi-feature fusion method with adaptively weighted local bundles is proposed in this paper. It performs better than the previous methods when the background is complex or the color of the foreground and the background is similar. However, it still has some limitations.

A complete 3D tracking system also includes initialization and relocation modules. In this paper, we only focus on the tracking module, and the 3D detection modules [36, 37] can be added to our method to complete the system. In addition, we can perform the tracking for each object to do the multi-object tracking. This is however limited by the computational efficiency. Efficient multi-object tracking methods will be explored in future work.

The color feature is distinguishable enough to get a proper segmentation when the background color is pure. In this case, the performance will not be significantly improved when we merge the edge feature to the energy function (see the experiment in Table 2). Besides, our method is disturbed when the background is particularly complex, or objects are severely occluded. Finally, objects with symmetry and translucency are still challenging.

5 Conclusions

This work proposed an optimized way to fuse multi-feature for 3D object tracking. To achieve optimal fusion and avoid the side effects of simple feature fusion with uniform weights, we proposed to group the region and edge features as a set of local bundles, which are adaptively weighted based on the confidence values of the involved features. The benefits of using local bundles are two-fold. First, the spatial-variant weights (ω_i) can be estimated more reliably by averaging over features of each bundle. Second, the color and the edge features can compete via spatial-variant weights (α_i,β_i) despite their spatial inconsistency. Quantitative experiments showed that our proposed approach significantly overperformed the previous single-feature methods and multi-feature methods without adaptive weighting. Extended experiments also showed that the proposed method enabled balancing the overall effect of each feature in different conditions. We further demonstrated that the overall weighting parameter λ was not essential. For the future work, additional texture features fusing might be taken into consideration to handle the textured and texture-less objects simultaneously.

References

[1] Lepetit V, Fua P. Monocular model-based 3D tracking of rigid objects: A survey. *Found. Trends® in Comput. Graph. Vis.*, 2005, 1(1): 1-89. DOI: 10.1561/0600000001

[2] Vacchetti L, Lepetit V, Fua P. Stable real-time 3D tracking using online and offline information. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2004, 26(10): 1385-1391. DOI: 10.1109/TPAMI.2004.92.

[3] Lourakis M I A, Zabulis X. Model-based pose estimation for rigid objects. In *Proc. the 9th International Conference on Computer Vision Systems*, July 2013, pp.83-92. DOI: 10.1007/978-3-642-39402-7_9.

[4] Tan D J, Tombari F, Ilic S, Navab N. A versatile learning-based 3D temporal tracker: Scalable, robust, online. In *Proc. the 2015 IEEE International Conference on Computer Vision*, December 2015, pp.693-701. DOI: 10.1109/ICCV.2015.86.

[5] Besl P J, McKay N D. A method for registration of 3-D shapes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 1992, 14(2): 239-256. DOI: 10.1109/34.121791.

[6] Peng S, Liu Y, Huang Q, Zhou X, Bao H. PVNet: Pixel-wise voting network for 6DoF pose estimation. In *Proc. the 2019 IEEE Conference on Computer Vision and Pattern Recognition*, June 2019, pp.4561-4570. DOI: 10.1109/CVPR.2019.00469.

[7] Ye Y, Zhang C, Hao X. ARPNET: Attention region proposal network for 3D object detection. *Sci. China Inf. Sci.*, 2019, 62(12): Article No. 220104. DOI: 10.1007/s11432-019-2636-x.

[8] Garon M, Lalonde J. Deep 6-DOF tracking. *IEEE Trans. Vis. Comput. Graph.*, 2017, 23(11): 2410-2418. DOI: 10.1109/TVCG.2017.2734599.

[9] Li Y, Wang G, Ji X, Xiang Y, Fox D. DeepIM: Deep iterative matching for 6D pose estimation. *Int. J. Comput. Vis.*, 2020, 128(3): 657-678. DOI: 10.1007/s11263-019-01250-9.

[10] Harris C, Stennett C. RAPID—A video rate object tracker. In *Proc. the 1990 British Machine Vision Conference*, September 1990, pp.73-77. DOI: 10.5244/C.4.15.

[11] Seo B, Park H, Park J, Hinterstoisser S, Ilic S. Optimal local searching for fast and robust textureless 3D object tracking in highly cluttered backgrounds. *IEEE Trans. Vis. Comput. Graph.*, 2014, 20(1): 99-110. DOI: 10.1109/TVCG.2013.94.

[12] Wang G, Wang B, Zhong F, Qin X, Chen B. Global optimal searching for textureless 3D object tracking. *The Visual Computer*, 2015, 31(6/7/8): 979-988. DOI: 10.1007/s00371-015-1098-7.

[13] Wang B, Zhong F, Qin X. Robust edge-based 3D object tracking with direction-based pose validation. *Multimedia Tools Appl.*, 2019, 78(9): 12307-12331. DOI: 10.1007/s11042-018-6727-5.

[14] Zhang Y, Li X, Liu H, Shang Y. Comparative study of visual tracking method: A probabilistic approach for pose estimation using lines. *IEEE Trans. Circuits Syst. Video Technol.*, 2017, 27(6): 1222-1234. DOI: 10.1109/TCSVT.2016.2527219.

[15] Prisacariu V A, Reid I D. PWP3D: Real-time segmentation and tracking of 3D objects. *Int. J. Comput. Vis.*, 2012, 98(3): 335-354. DOI: 10.1007/s11263-011-0514-3.

[16] Tjaden H, Schwanecke U, Schömer E. Real-time monocular segmentation and pose tracking of multiple objects. In *Proc. the 14th European Conference on Computer Vision*, October 2016, pp.423-438. DOI: 10.1007/978-3-319-46493-0_26.

[17] Hexner J, Hagege R R. 2D-3D pose estimation of heterogeneous objects using a region based approach. *Int. J. Comput. Vis.*, 2016, 118(1): 95-112. DOI: 10.1007/s11263-015-0873-2.

[18] Tjaden H, Schwanecke U, Schömer E. Real-time monocular pose estimation of 3D objects using temporally consistent local color histograms. In *Proc. the 2017 IEEE International Conference on Computer Vision*, October 2017, pp.124-132. DOI: 10.1109/ICCV.2017.23.

[19] Tjaden H, Schwanecke U, Schömer E, Cremers D. A region-based gauss-newton approach to real-time monocular multiple object tracking. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2019, 41(8): 1797-1812. DOI: 10.1109/TPAMI.2018.2884990.

[20] Marchand É, Bouthemy P, Chaumette F. A 2D-3D model-based approach to real-time visual tracking. *Image Vis. Comput.*, 2001, 19(13): 941-955. DOI: 10.1016/S0262-8856(01)00054-3.

[21] Drummond T, Cipolla R. Real-time visual tracking of complex structures. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2002, 24(7): 932-946. DOI: 10.1109/TPAMI.2002.1017620.

[22] Wuest H, Vial F, Stricker D. Adaptive line tracking with multiple hypotheses for augmented reality. In *Proc. the 4th IEEE/ACM International Symposium on Mixed and Augmented Reality*, October 2005, pp.62-69. DOI: 10.1109/ISMAR.2005.8.

[23] 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.

[24] Wang B, Zhong F, Qin X. Pose optimization in edge distance field for textureless 3D object tracking. In *Proc. the 2017 Computer Graphics International Conference*, June 2017, Article No. 32. DOI: 10.1145/3095140.3095172.

[25] Osher S, Sethian J A. Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations. *Journal of Computational Physics*, 1988, 79(1): 12-49. DOI: 10.1016/0021-9991(88)90002-2.

[26] Zhong L, Zhao X, Zhang Y, Zhang S, Zhang L. Occlusion-aware region-based 3D pose tracking of objects with temporally consistent polar-based local partitioning. *IEEE Trans. Image Process.*, 2020, 29: 5065-5078. DOI: 10.1109/TIP.2020.2973512.

[27] Lecun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998, 86(11): 2278-2324. DOI: 10.1109/5.726791.

[28] Crivellaro A, Rad M, Verdie Y, Yi K M, Fua P, Lepetit V. Robust 3D object tracking from monocular images using stable parts. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2018, 40(6): 1465-1479. DOI: 10.1109/TPAMI.2017.2708711.

[29] Zhong L, Zhang L. A robust monocular 3D object tracking method combining statistical and photometric constraints. *Int. J. Comput. Vis.*, 2019, 127(8): 973-992. DOI: 10.1007/s11263-018-1119-x.

[30] Ma Y, Soatto S, Košecká J, Sastry S S. An Invitation to 3-D Vision: From Images to Geometric Models (1st edition). Springer-Verlag New York Publishers, 2004.

[31] Zhong F, Qin X, Chen J, Hua W, Peng Q. Confidence-based color modeling for online video segmentation. In *Proc. the 9th Asian Conference on Computer Vision*, September 2009, pp.697-706. DOI: 10.1007/978-3-642-12304-7_66.

[32] Wu P, Lee Y, Tseng H, Ho H, Yang M, Chien S. A benchmark dataset for 6DoF object pose tracking. In *Proc. the 2017 IEEE International Symposium on Mixed and Augmented Reality Adjunct*, October 2017, pp.186-191. DOI: 10.1109/ISMAR-Adjunct.2017.62.

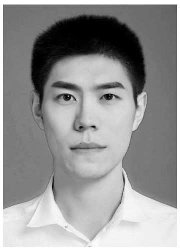
[33] Brachmann E, Michel F, Krull A, Yang M Y, Gumhold S, Rother C. Uncertainty-driven 6D pose estimation of objects and scenes from a single RGB image. In *Proc. the 2016 IEEE Conference on Computer Vision and Pattern Recognition*, June 2016, pp.3364-3372. DOI: 10.1109/CVPR.2016.366.

[34] Whelan T, Leutenegger S, Salas-Moreno R F, Glocker B, Davison A J. ElasticFusion: Dense SLAM without a pose graph. In *Proc. the 2015 Robotics: Science and Systems*, July 2015. DOI: 10.15607/RSS.2015.XI.001.

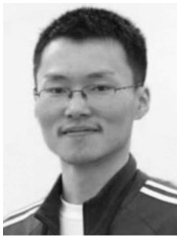
[35] Mur-Artal R, Tardós J D. ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras. *IEEE Trans. Robotics*, 2017, 33(5): 1255-1262. DOI: 10.1109/TRO.2017.2705103.

[36] Marchand É, Uchiyama H, Spindler F. Pose estimation for augmented reality: A hands-on survey. *IEEE Trans. Vis. Comput. Graph.*, 2016, 22(12): 2633-2651. DOI: 10.1109/TVCG.2015.2513408.

[37] Cheng M, Liu Y, Lin W, Zhang Z, Rosin P L, Torr P H S. BING: Binarized normed gradients for objectness estimation at 300fps. *Comput. Vis. Media*, 2019, 5(1): 3-20. DOI: 10.1007/s41095-018-0120-1.



Jia-Chen Li is a Ph.D. candidate of School of Software, Shandong University, Jinan. He received his B.E. degree in exploration technology and engineering from Ocean University of China, Qingdao, in 2016. His research interests include augmented reality, 3D object tracking, pose estimation, head posture analysis, etc.



Fan Zhong is an associate professor of School of Computer Science and Technology, Shandong University, Qingdao. He received his Ph.D. degree in mathematics from Zhejiang University, Hangzhou, in 2010. His research interests include image and video editing, 3D detection, pose estimation, augmented reality, etc.

Song-Hua Xu is a computer scientist. He received his Ph.D. degree in computer science from Yale University, New Haven, CT, USA, in 2010. His research interests include healthcare informatics, information retrieval, knowledge management and discovery, intelligent web and social media, visual analytics, user interface design, and multimedia.

Xue-Ying Qin is a professor of School of Software, Shandong University, Jinan. She received her Ph.D. degree in engineering from Hiroshima University, Hiroshima, Japan, in 2001. Her main research interests are augmented reality, pose estimation, video-based analyzing, photorealistic rendering, etc.