Semantic-Independent Dynamic SLAM based on Geometric Re-clustering and Optical Flow Residuals

Hengbo Qi, Xuechao Chen, Zhangguo Yu, Chao Li, Yongliang Shi*,
Qingrui Zhao, and Qiang Huang, *Fellow, IEEE*

***Abstract*—Dynamic objects pose significant challenges to the accuracy of state estimation and map quality in Simultaneous Localization and Mapping (SLAM). While current dynamic SLAM methods often rely on semantic information to detect specific movable objects, this dependency on pre-trained models and semantic priors can lead to false dynamic detections. This paper presents a novel semantic-independent dynamic SLAM method that detects truly moving regions, without being constrained by the classes or motion patterns of dynamic objects. We introduce a geometric re-clustering approach to improve object clustering by addressing the under- and over-segmentation caused by the K-Means algorithm. Next, instead of simply classifying entire clusters as dynamic or static, we propose a method to detect dynamic regions within each cluster based on dense optical flow residuals. This enables the detection of partial object movements, such as a seated person moving only his hands. Dynamic detection results are propagated across consecutive frames as dynamic priors for calculating optical flow residuals. Additionally, to enhance map quality, we address the mis-detection of slowly or intermittently moving objects through depth consistency checks applied over a larger time interval. Extensive evaluations on public datasets (TUM and Bonn) and real-world scenes show that our method outperforms state-of-the-art semantic-based methods in terms of localization accuracy and generalizability across various scenarios, particularly when facing unknown dynamic objects. Our method also achieves clean and dense reconstructions, demonstrating its potential for applications like robot navigation in dynamic environments.**

***Index Terms*—Dynamic SLAM, dynamic region detection, moving object segmentation, dense construction, semantic-independent.**

(a) DynaSLAM on Bonn *placing_no_box*

(b) Ours on Bonn *placing_no_box*

Fig. 1. In (**a**), the estimated camera trajectory (green) drifts from the ground-truth trajectory (red), and the undetected box is reconstructed into the map. In (**b**), the estimated camera trajectory closely aligns with the ground truth, and the reconstructed map accurately represents the static environment, with dynamic elements being successfully removed.

I. Introduction

SIMULTANEOUS Localization and Mapping (SLAM) is a foundational technology for robotic perception systems, allowing robots to estimate their motion state and build a model of the surrounding environment concurrently [1]. Over the years, numerous visual SLAM algorithms [2], [3], [4] are developed based on the assumption of a static environment. They have been proven to deliver remarkable performance in a variety of scenarios. However, integrating SLAM into real-world applications remains challenging due to the prevalence of dynamic objects in common scenes.

As shown in Fig. 1(a), dynamic objects severely impair the performance of SLAM in two key ways. First, undetected dynamic objects introduce erroneous data associations, leading to drift or discontinuity in the estimated motion trajectory [5], [6]. Second, un-removed dynamic objects are incorporated into the map, compromising its accuracy for visual localization and navigation tasks [7]. Thus, detecting and removing regions occupied by dynamic objects are crucial for SLAM in dynamic environments, a field known as dynamic SLAM.

Dynamic SLAM can be classified into semantic-based and semantic-independent methods, depending on whether semantic information (i.e., the categories of objects) is used [8], [9]. Semantic-based methods [10], [11], [12], [13] leverage semantic priors (e.g., a person tends to be dynamic, while a chair tends to be static) and segmentation masks obtained from deep learning networks to detect dynamic objects. In essence, semantic-based methods detect movable objects, not truly moving ones [14]. They typically assign the same dynamic attributes to all pixels with the same semantic label, even if the dynamic object displays hybrid dynamic properties. For instance, a sitting person may be identified as

This work was supported in part by the National Natural Science Foundation of China, under Grant 62073041; in part by the "111" Project, under Grant B08043.

Hengbo Qi, Xuechao Chen, Zhangguo Yu, Chao Li, Qingrui Zhao, and Qiang Huang are with the School of Mechatronical Engineering, Beijing Institute of Technology (BIT), Beijing, China.

Yongliang Shi* is with the Qiyuan Lab, Beijing, China (* represents the corresponding author. E-mail: shiyongliang@qiyuanlab.com).

dynamic while only their hand is moving. This could result in fewer measurements being used for estimation, at the expense of accuracy [1]. Furthermore, semantic-based methods cannot identify objects absent from the pre-trained models, restricting their applicability in unknown environments.

On the other hand, semantic-independent methods [15], [16], [17] segment the image into regions through geometry-based clustering and then classify the dynamic properties of each region. However, they often grapple with under-segmentation (i.e., grouping different objects together) or over-segmentation (i.e., dividing one object into multiple clusters). This hinders the complete detection of dynamic regions, challenging accurate camera motion estimation and clean environment reconstruction. Additionally, semantic-independent methods may fail when dynamic objects cover a large image area, as no prior information is provided [16].

In this paper, we present a semantic-independent dynamic SLAM method to address four above-mentioned limitations: (**a**) incomplete object clustering, (**b**) improper handling of objects with hybrid dynamics, (**c**) limited detection in images with large dynamic regions, and (**d**) reliance on semantic information. To tackle limitation (**a**), we introduce geometric re-clustering, where under-segmented objects are split using geometric edges, and over-segmented objects are merged based on depth histogram analysis. As for limitation (**b**), rather than assigning uniform dynamic properties to the entire cluster, we identify regions with high optical flow residuals (the difference between camera-induced optical flow and that from dynamic objects) as dynamic within each cluster, while preserving static areas. Next, we leverage the temporal consistency of dynamic objects by propagating detected dynamic regions to subsequent frames as priors for optical flow residual calculation. This ensures robust dynamic detection, even in scenes characterized by limitation (**c**). Finally, during dense reconstruction, we remove misdetected dynamic objects with intermittent or slow motion to improve map quality. As our method requires no semantic priors or pre-trained models, it is inherently free from limitation (**d**). In summary, this work makes three contributions:

1) A novel semantic-independent dynamic SLAM method is proposed. Our method detects the truly moving regions, without relying on predefined movable objects or specific motion patterns, showing strong generalizability across diverse dynamic scenarios. It achieves superior localization accuracy and mapping quality compared to both state-of-the-art semantic-based and semantic-independent methods across most sequences on the TUM and Bonn datasets and real-world scenes. Our code will be released upon the publication of this paper [1].

2) A geometric re-clustering approach is presented, which enhances object segmentation by resolving under- and over-segmentation using geometric edges and depth histogram analysis. This approach results in complete and accurate object clustering, thereby facilitating a more thorough detection of dynamic regions.

3) A dynamic detection approach is presented, which combines geometric re-clustering with optical flow residuals to accurately distinguish between dynamic and static regions. This approach effectively handles scenes involving hybrid dynamic objects, extensive dynamic regions, and objects with slow or intermittent motion.

The remaining sections are structured as follows. Section II provides an overview of the related works in dynamic SLAM. Section III presents a comprehensive description of our method. Section IV includes comparative experimental results and discussions about the results. Finally, conclusions and future work are drawn in Section V.

II. Related Work

General SLAM systems utilize both color and geometric information to estimate camera motion. Early methods, such as MonoSLAM [18], are based on nonlinear filtering frameworks. In recent years, however, the field has shifted towards graph optimization-based frameworks due to their superior accuracy and efficiency. Graph optimization methods are typically categorized as direct or indirect. DSO [4] is a notable work of the direct approach, which estimates camera motion by minimizing photometric error across image frames. DSO also supports dense reconstruction by utilizing all pixel information. In contrast, ORB-SLAM2 [2] represents the indirect approach, where camera motion is estimated by minimizing reprojection error through feature point extraction. Moreover, ORB-SLAM2 further enhances accuracy through the integration of global relocalization and loop closure modules.

Recently, deep learning has garnered considerable attention across various fields [19] [20], including SLAM, where significant progress has been made in developing end-to-end SLAM systems. For instance, TartanVo [21] presents the first learning-based visual odometry (VO) model, which incorporates a novel up-to-scale loss function and integrates camera intrinsic parameters into the model. DROID-SLAM [22] achieves high accuracy and robustness by iteratively updating camera poses and per-pixel depth estimation through a differentiable Dense Bundle Adjustment layer. Despite these advances, deep learning-based SLAM is usually less explainable and struggles to generalize to unseen datasets or cameras with different calibrations [23]. Moreover, they lack the core modules of general SLAM systems, such as loop closure and global bundle adjustment.

While the above methods perform well in static environments, they often struggle with dynamic objects, prompting the development of dynamic SLAM. Some works, such as DyTanvo [24], utilize deep learning for dynamic object detection, but such methods are still in their early stages. The current mainstream dynamic SLAM approaches are built upon general SLAM systems, which can be divided into semantic-based and semantic-independent methods.

A. Semantic-based Dynamic SLAM

Semantic-based methods efficiently detect dynamic regions, leveraging two key advantages: semantic priors to identify dynamic objects and pixel-wise masks to delineate object regions. The classical semantic-based method DynaSLAM [10] employs Mask-RCNN [25] to detect movable objects, followed

[1]https://github.com/qimao7213/SInDSLAM

by a lightweight tracking using feature points outside these objects. Feature points within the movable objects are then classified by multi-view geometry constraints. This workflow is widely adopted by subsequent works like [26], [27], [28] to eliminate the impact of dynamic objects on pose estimation. For dense static reconstruction, pixel-level semantic masks facilitate the quick removal of dynamic objects. To address the imperfections in semantic masks that may contaminate the constructed map, Blitz-SLAM [11] carefully removes the noisy blocks from the depth image. OVD-SLAM [29], instead of exploiting time-costing semantic segmentation networks, employs the YOLOv5 [30] object detection network to generate object bounding boxes and segments the foreground and background within the boxes.

Semantic-based methods often require pre-defined semantic priors, assigning dynamic weights to objects based on semantic information [10], [31], [32]. Unfortunately, these methods are limited by the inherent subjectivity of human assignment, as object dynamic attributes are not consistently invariant. For instance, a stationary chair can be moved [33], a moving vehicle might stop temporarily [34], or a person may have only part of their body in motion [35]. Moreover, objects not included in the pre-trained dataset cannot be effectively detected, which presents a significant challenge.

B. Semantic-independent Dynamic SLAM

Semantic-independent methods detect dynamic regions or feature points through geometric constraints or motion consistency checks. We categorize them into point-based and region-based approaches according to the type of features being processed. Point-based approaches detect dynamic points, while region-based ones operate at the region level.

1) Point-based Approach: Points are commonly used for data association and camera tracking. Tan [36] observes the appearance and depth change of feature points between adjacent frames, and classifies points with significant changes as dynamic. Similarly, Shile [37] extracts edge points from depth image and assigns high static weights to points with small depth variations, which are then used for camera pose estimation. In [38], Cheng uses sparse optical flow to establish data associations between feature points in two frames and removes dynamic points using epipolar constraints, derived from the fundamental matrix [39]. In contrast to detecting dynamic feature points by changes between adjacent frames, LCCRF SLAM [40] constructs a long-term consistent conditional random field, resulting in better temporal consistency for dynamic detection. DSLAM [41] constructs a sparse graph by treating feature points as vertices and their correlations as edges. The feature points are grouped based on the change of edges, and the group containing most feature points is considered static. Dynam-SLAM [42] proposes a loosely coupled framework to detect dynamic feature points using stereo scene flow and inertial measurement unit (IMU) data. Dynamic and static feature points, along with IMU information, are then fed into a tightly coupled framework for joint optimization.

Although these methods can effectively reduce the impact of dynamic feature points on camera trajectory estimation, they only construct sparse metric maps, limiting their utility for advanced tasks and comprehensive environment representation.

2) Region-based Approach: The main idea of region-based approaches involves segmenting the image into clusters and identifying dynamic regions, then eliminating all pixels within these dynamic areas. Yang [16] and Jaimez [17] utilize the K-Means algorithm for image clustering and remove the feature points in dynamic clusters. While effective, they tend to oversegment objects and struggle to classify entire moving objects as dynamic. Additionally, some approaches determine dynamic regions by analyzing residuals. Zhang [43] and Sun [7] utilize optical flow residuals, while ReFusion [44] and StaticFusion [15] use depth residuals between the current depth image and the reconstructed 3D model. However, since these methods do not explicitly handle dynamic objects, they often fail to fully detect dynamic regions, leading to challenges in achieving accurate camera trajectory estimation. The reconstructed dense maps also exhibit obvious noise or distortion.

III. Method

A. System Overview

Figure 2 illustrates our method, which consists of three main modules: Image Clustering (Sec. III-B), Dynamic Region Detection (Sec. III-C), and Dense Map Reconstruction (Sec. III-D). Unlike previous semantic-independent methods, our method enhances dynamic region detection through geometric re-clustering and the introduction of dynamic priors. Specifically, in Image Clustering, our re-clustering improves initial results by producing more complete object clusters. In Dynamic Region Detection, dynamic information from the previous frame serves as dynamic priors to improve the robustness and accuracy of optical flow residual calculation. Based on the re-clustering results and optical flow residuals, we generate a dynamic mask, which classifies regions as dynamic (white), static (gray), or depth-invalid (black). Feature points in dynamic regions are discarded, while the remaining points are used for camera tracking in ORB-SLAM2. In Dense Map Reconstruction, the dynamic mask is further refined based on the depth reprojection error to address the misdetection of dynamic objects with slow or intermittent motion. Afterward, all local colorful point clouds from static regions are stitched together using the camera poses estimated by ORB-SLAM2, creating a dense static environment map. Our method takes a sequence of depth and RGB images as input, and outputs a dense point cloud map generated by Dense Map Reconstruction along with an estimated camera trajectory from ORB-SLAM2. This paper will use the notations shown in Table I.

B. Image Clustering

To mitigate the impact of dynamic objects on localization and mapping at region level, we first group the image into multiple clusters. Previous works [17] and [16] utilize the efficient K-Means algorithm for geometric clustering but often lead to cluttered results and incomplete dynamic detection. To address this limitation, we propose a novel geometric re-clustering approach, which enhances clustering results by employing

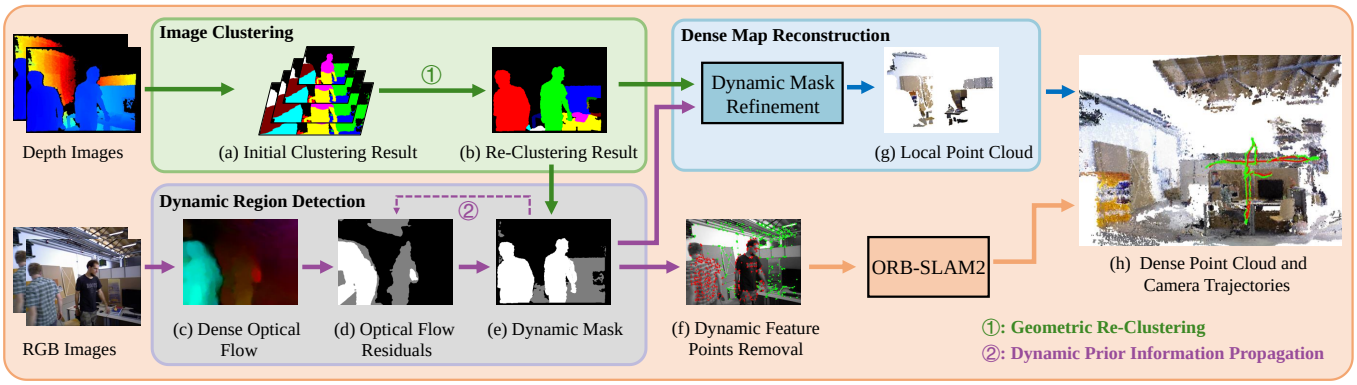


Fig. 2. System overview. Depth images are clustered to generate (**b**) and RGB images are used to compute optical flow residuals shown in (**d**). Based on the data from (**b**) and (**d**), a dynamic mask (**e**) is generated, indicating dynamic regions in the image. These dynamic regions and their feature points are removed to ensure accurate camera trajectory estimation and reconstruction of the static environment.

TABLE I
LIST OF NOTATIONS.

Notation	Explanation	Notation	Explanation
I	Image	E	Edge point set
C	Cluster	$\mathbf{M}$	Region adjacency graph Matrix
τ	Threshold	cmp	Histogram comparison function
λ	Scale factor	ω, φ	Weight

geometric edges to split under-segmented clusters and using depth histogram analysis to merge over-segmented clusters.

1) Initial Clustering: Following [17], we project each pixel from the depth image into 3D space to generate a point cloud and then use the K-Means algorithm for image clustering. Let a pixel of the depth image be $\mathbf{u} = (u, v)^T \in \mathbb{R}^2$, and its 3D spatial coordinate $\mathbf{p} = (x, y, z)^T \in \mathbb{R}^3$ can be calculated by the inverse camera projection model $\pi^{-1} : \mathbb{R}^2 \to \mathbb{R}^3$:

$$\mathbf{p} = \pi^{-1}(\mathbf{u}, I_D(\mathbf{u})), \tag{1}$$

where $I_D(\mathbf{u})$ is the depth value of the pixel $\mathbf{u}$, and π is determined by the camera intrinsic parameters. To accelerate the clustering convergence speed, we build a Gaussian pyramid (coarse-to-fine) for the depth image. At each level, the clustering result from the coarser level serves as the initial value. At the top level, we initialize it with the clustering result from the previous frame. Additionally, as the accuracy of depth images decreases with increasing depth values, pixels with depth values exceeding a certain threshold τ_{depth} (in meters) are considered depth-invalid, as suggested in [11]. The pixels with invalid or zero depth values are grouped into a separate cluster. We determine the number of clusters N based on the image size: $N = w \times h/25600$, where w and h are the image width and height, respectively.

2) Edge Extraction and Re-Clustering: Clustering based solely on spatial coordinates often results in over-segmentation or under-segmentation. We enhance object discrimination using geometric edges, as they typically indicate different objects. The gradient edges E_{grad} (white edges in Fig. 3(d)) in Bose's work [45] are non-closed, so we extract plane edges E_{plane} (green edges in Fig. 3(c)) as a complement to form complete edges. These edges are then used to divide under-segmented clusters, followed by a re-clustering process to merge over-segmented clusters (Fig. 4).

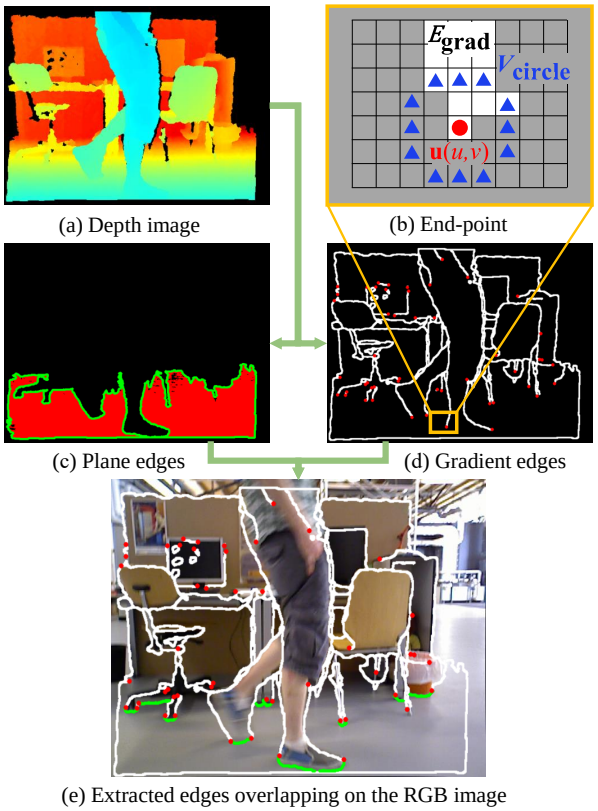


Fig. 3. Illustration of *Edge Extraction*. (**a**) is a normalized depth image where redder color indicates greater depth, bluer color indicates less. (**c**) shows a plane (red) extracted from the depth image along with its plane edges E_{plane} (green). (**d**) displays the gradient edges E_{grad} (white) and the end-points (red). (**b**) visualizes the definition of end-points, where the red point is an end-point, the blue triangles are its circular neighborhood V_{circle}, and the white area is E_{grad}. In (**e**), plane edges and gradient edges are connected by end-points to form closed edges, which are overlapped on the RGB image.

a) Gradient Edge Extraction: The maximum difference δ_{depth} between a pixel $\mathbf{u}$ and its neighboring pixel block $B(\mathbf{u})$ is computed as

$$\delta_{\text{depth}} = \max\left(|I_D(\mathbf{u}_{\text{ne}}) - I_D(\mathbf{u})|\right), \text{ for } \mathbf{u}_{\text{ne}} \in B(\mathbf{u}). \tag{2}$$

(a) Initial clustering result (b) Divided clustering result (c) Re-clustering result

Fig. 4. Illustration of *Re-Clustering*. (**a**) is divided into several clusters shown in (**b**) by using extracted edges from Fig.3(**e**), and then re-clustered to form (**c**). In (**b**), the white arrows indicate the directions of merging.

If δ_{depth} satisfies

$$\delta_{\text{depth}} > \max\left(\tau_1 I_D(\mathbf{u}), \tau_2\right), \tag{3}$$

it indicates a significant depth value change between $\mathbf{u}$ and $B(\mathbf{u})$, and $\mathbf{u}$ is classified as a point of E_{grad}. In (3), τ_1 is a proportional constant that adaptively adjusts threshold value based on $I_D(\mathbf{u})$, and the threshold τ_2 (in meters) is used to reject edge points caused by depth image noise, following the parameters in [46]. If the scene depth range is large or the depth image contains significant noise, τ_1 and τ_2 should be increased to enhance the stability and accuracy of edge extraction.

b) Plane Edge Extraction: Extracting gradient edges becomes challenging in regions where depth values change gradually. Fig. 3(a) provides an example, where the depth values (coded by colors) of the feet and the ground are similar. To overcome this issue, we extract plane edges as a complement. Planes are detected by PEAC [46], and initial plane edges E'_{plane} are extracted. E'_{plane} is divided into several segments by

$$\{E_{\text{seg}}\} = E'_{\text{plane}} - E_{\text{grad}}, \tag{4}$$

where the "$-$" symbol denotes the set difference operation and $\{E_{\text{seg}}\}$ is the set of the segmented plane edge E_{seg}. These segmented plane edges are connected to the gradient edges by end-points, which are defined in Fig. 3(b). Specifically, an end-point is a pixel whose circular neighborhood contains fewer than five pixels belonging to the gradient edges. If an E_{seg} region covers more than one end-point, it is retained to the plane edges set E_{plane}. Finally, we define the union set of E_{grad} and E_{plane} as E_{edge}, representing all edges collectively, as shown in Fig. 3(e).

c) Re-Clustering: Fig. 4(a) shows that the initial clustering result suffers from under-segmentation and over-segmentation. After using the extracted edges in Fig. 3(e) to split the initial clustering result, Fig. 4(b) illustrates that the feet are separated from the ground, resolving the under-segmentation issue. Moreover, in Fig. 3(e), the right leg and right foot are considered as a single entity, but in Fig. 4(a), they are divided into separate clusters, introducing a "fake edge". The presence of such a fake edge shared by two clusters suggests their potential to be merged. This strategy effectively handles the over-segmentation issue.

The divided initial clusters contain a set of clusters $\{C_1, C_2 \dots C_n\}$, where n is the total number of divided clusters. For each cluster C_i $(i \in (1, 2 \dots n))$, its attributes are denoted as $\left\{C^i_{\text{reg}}, C^i_{\mathbf{p}}, C^i_{\text{hist}}, C^i_{E_{\text{edge}}}, C^i_{E_{\text{fake}}}\right\}$, where C^i_{reg} represents the image region occupied by C_i, $C^i_{\mathbf{p}} = (x, y, z)$ is the 3D center point, C^i_{hist} is the histogram of depth values, $C^i_{E_{\text{edge}}}$ represents the edges of C^i_{reg} and $C^i_{E_{\text{fake}}}$ represents the set of fake edges, which are obtained by

$$C^i_{E_{\text{fake}}} = C^i_{E_{\text{edge}}} - E_{\text{edge}}. \tag{5}$$

Let $|C|$ represents the area (number of pixels) within a certain region C. The score s^i for each cluster C_i is calculated as

$$s^i = \lambda_1 |C^i_{\text{reg}}| - C^i_{\mathbf{p}}(z), \tag{6}$$

where λ_1 is a scale factor to balance the weights of $|C^i_{\text{reg}}|$ and $C^i_{\mathbf{p}}(z)$. A cluster with larger area and closer distance to the camera is assigned a higher score, indicating its greater importance in the image. After sorting all clusters in descending order according to their scores, we build a Region Adjacency Graph (RAG) matrix $\mathbf{M}_{\text{RAG}}$ to record the correlation between each pair of clusters. If the correlation between two clusters exceeds a certain threshold τ_{meg}, the two clusters are merged. $\mathbf{M}_{\text{RAG}}$ is calculated as

$$\mathbf{M}_{\text{RAG}} = (\mathbf{M}_1 \odot (\lambda_2 \mathbf{M}_2 + \mathbf{M}_3)) \odot \mathbf{M}_{\text{weight}} \odot \mathbf{M}_{\text{rej}}, \tag{7}$$

where $\odot$ denotes element-wise multiplication and $\mathbf{M}_1$, $\mathbf{M}_2$, $\mathbf{M}_3$ and $\mathbf{M}_{\text{weight}}$ are calculated by (8), (9), (10), (11), respectively. $\mathbf{M}_{\text{rej}}$ will be discussed later.

For two clusters C_i and C_j, we calculate $\mathbf{M}_1$ as

$$\mathbf{M}_1(i,j) = \begin{cases} 1, & |C^i_{\text{reg}} \cap C^j_{\text{reg}}| > 0.4 * \min\left(\tau_3, |C^i_{\text{reg}}|, |C^j_{\text{reg}}|\right) \\ 0, & \text{otherwise} \end{cases}. \tag{8}$$

Since each cluster is mutually pixel-wise exclusive, we dilate them before computing intersections. Two clusters are considered adjacent if the number of intersecting pixels exceeds a certain threshold, making them eligible for potential merging. For $\mathbf{M}_2$, we have

$$\mathbf{M}_2(i,j) = |C^i_{E_{\text{fake}}} \cap C^j_{E_{\text{fake}}}|. \tag{9}$$

$\mathbf{M}_2$ indicates the presence of common fake edges between two clusters. The larger area of common fake edges, the higher probability of merging. For $\mathbf{M}_3$, we have

$$\begin{aligned} \mathbf{M}_3(i,j) = \text{cmp}_1(C^i_{\text{hist}}, C^j_{\text{hist}}) + (1 - \text{cmp}_2(C^i_{\text{hist}}, C^j_{\text{hist}})) \\ + \text{cmp}_3(C^i_{\text{hist}}, C^j_{\text{hist}}) \end{aligned}. \tag{10}$$

$\mathbf{M}_3$ represents the depth similarity between two clusters. For two depth histograms, cmp_1 compares the correlation, cmp_2 compares the Bhattacharyya distance, and cmp_3 compares the intersection (see Appendix). The higher depth similarity, the higher likelihood of merging. For $\mathbf{M}_{\text{weight}}$, we have

$$\mathbf{M}_{\text{weight}}(i,j) = \begin{cases} \omega_{\text{low}}, & \min(i,j) < 0.5 * n \\ \omega_{\text{high}}, & \min(i,j) > 0.7 * n \\ \omega_{\text{mid}}, & \text{otherwise} \end{cases} \tag{11}$$

where n is the total number of clusters. If $\min(i,j)$ exceeds $0.7 * n$, it indicates that C_i or C_j has a small score and is less significant in the image. Therefore, we assign higher

weights to them to promote their merging, while we discourage the merging of clusters with high scores by assigning lower weights. Regarding $\mathbf{M}_{\text{rej}}(i,j)$, if the connecting part between C_i and C_j contains E_{plane} or $\mathbf{M}_3(i,j)$ is below τ_{rej}, we directly reject their merging. Note that this strategy does not apply to clusters with small scores, as they are consistently encouraged to merge into other clusters.

After obtaining $\mathbf{M}_{\text{RAG}}$, we merge clusters in two steps. First, for clusters with middle or high scores (i.e., those with indices $i \leq 0.7 * n$), if $\mathbf{M}_{\text{RAG}}(i,j) > \tau_{\text{meg}}$ and $i < j$, we merge C_j into C_i and transfer all correlations related to C_j onto C_i. For example, if a cluster C_k is initially to be merged into C_j, it is now to be merged into C_i. In the second step, for clusters with low scores, we use a lower threshold of $0.2 * \tau_{\text{meg}}$ to decide whether to merge them into clusters with middle or high scores. If a cluster with a low score cannot be merged, it is preserved as a separate cluster.

τ_{meg} and τ_{rej} can be reduced for large scene depth range, as depth similarity decreases. When the scene's depth values are similar, reduce ω_{low} to avoid over-merging of significant clusters. If many small clusters appear in the re-clustering results, increase ω_{high} to promote their merging.

Figure 4(b) shows the merging process, and finally, we obtain Fig. 4(c), where the legs are successfully separated from the ground, with the right leg and foot being merged. The ground area is also more completely represented. Although our method may not achieve perfect performance in all cases and might lose some information, it provides enhanced clustering results essential for subsequent dynamic region detection. Moreover, our dynamic region detection algorithm exhibits sufficient robustness to tolerate the imperfection of this stage.

C. Dynamic Region Detection

In this section, we first compute optical flow residuals. While [44] and [47] use a residual threshold to detect dynamic regions, this may lead to inaccurate segmentation, particularly in cases of optical flow estimation errors. To address this issue, we dynamically adjust the thresholds using a proposed dual-threshold strategy and confine dynamic region detection within each cluster. Finally, we generate the I_{dMask}, which indicates regions of dynamic, static, and depth invalid with values of 255 (white), 125 (gray), and 0 (black), respectively.

Additionally, conventional outlier detection approaches, such as the Random Sample Consensus (RANSAC) algorithm, typically set an outlier limit of 50% or less, which becomes challenging in scenes with substantial dynamic regions [16]. To tackle this, we employ the Progressive Sample Consensus (PROSAC) algorithm [48], which enhances robustness by integrating dynamic prior information from the previous frame.

1) Optical Flow Residual Calculation: Optical flow represents the motion of pixels and indicates the 2D correspondence between consecutive images. We calculate dense optical flow to find the motion vectors for every image pixel, enabling pixel-wise matching. Given a pair of consecutive grayscale images I_G^t and I_G^{t-1}, a pixel $\mathbf{u}^t = (u,v) \in \mathbb{N}^2$ in I_G^t can find its corresponding pixel $\mathbf{u}'$ in I_G^{t-1} by

$$\mathbf{u}' = \mathbf{u}^t - \mathbf{v}(\mathbf{u}^t) \tag{12}$$

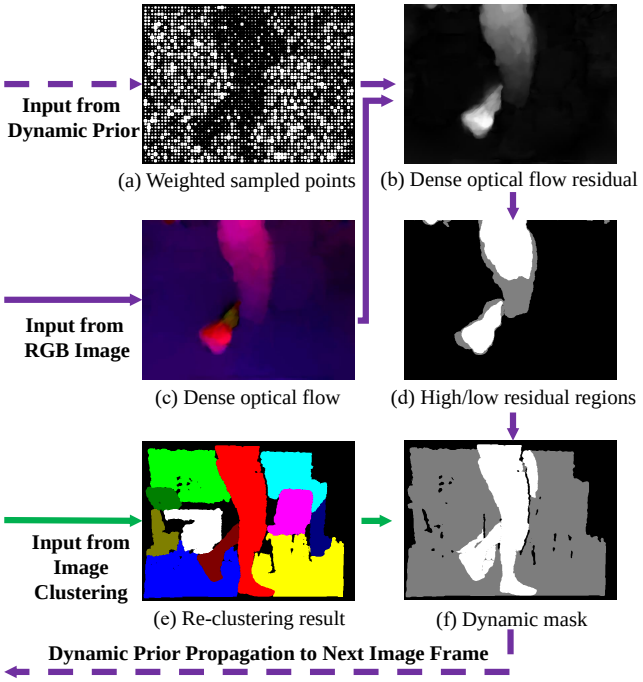


Fig. 5. Illustration of *Dynamic Region Detection*. In (**a**), brighter points have higher weights, increasing their likelihood of being identified as inliers in the $\mathbf{H}$ matrix calculation. These weights are determined based on dynamic priors from the previous frame. The residuals in (**b**) are categorized into high residuals (white), low residuals (gray), and static (black) regions shown in (**d**). The final dynamic mask (**f**) is created using (**d**) and (**e**), and is then propagated to the next image frame as dynamic priors.

where $\mathbf{v}(\mathbf{u}^t) \in \mathbb{R}^2$ is the 2D optical flow vector of $\mathbf{u}^t$. At the same time, the homography matrix from I_G^t to I_G^{t-1}, denoted as $\mathbf{H}_{t-1}^t \in \mathbb{R}^{3\times3}$, also represents the 2D correspondence between the two images. $\mathbf{u}^t$ can be mapped into I_G^{t-1} by $\mathbf{H}_{t-1}^t$ as

$$\mathbf{u}'' = \mathbf{H}_{t-1}^t \mathbf{u}^t / \xi(\mathbf{H}_{t-1}^t \mathbf{u}^t) \tag{13}$$

where the homogeneous coordinate of $\mathbf{u}^t$ is used, and $\xi(\cdot)$ represents taking the last dimension of a 3D vector. Similar to [7], we define the optical flow residual of $\mathbf{u}^t$ as $\delta(\mathbf{u}^t) = \|\mathbf{u}'' - \mathbf{u}'\|$. For static regions, the pixel changes between different images are mainly caused by camera motion, resulting in $\delta(\mathbf{u}^t)$ tending to be close to zero. In contrast, for dynamic regions, the pixel changes are caused by both camera motion and object motion, leading to larger optical flow residuals [49].

The homography matrix $\mathbf{H}_{t-1}^t$ can be obtained by minimizing the following least squares problem, considering all pixel correspondences estimated by dense optical flow:

$$\mathbf{H}_{t-1}^t = \arg\min_{\mathbf{H}} \sum_{\mathbf{u}^t \in I_G^t} \delta(\mathbf{u}^t). \tag{14}$$

We employ the PROSAC algorithm to estimate $\mathbf{H}_{t-1}^t$. Unlike RANSAC, which assigns equal weights to all pixel correspondences, PROSAC differentiates weights among them, where the pixel correspondences with higher weights are more likely to be selected as inliers. Specifically, considering the coherence of dynamic properties, i.e., a dynamic region in the

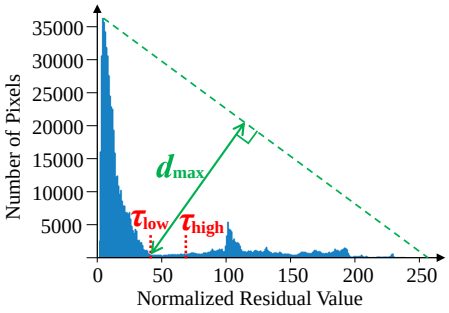


Fig. 6. Optical flow residual histogram of Fig. 5(**b**). The green dashed line connects the highest and lowest points of the histogram, and the green solid line presents the distance d between the green dashed line and the histogram. By maximizing d, we find the residual value τ_{low} and define $\tau_{\text{high}} = 1.3 * \tau_{\text{low}}$.

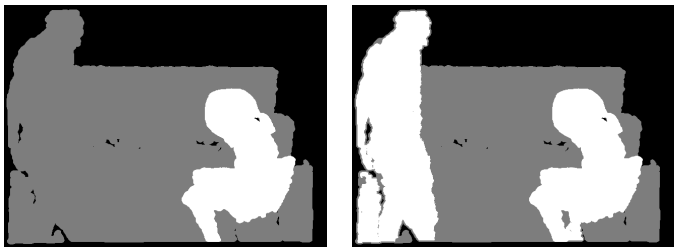
previous frame tends to remain dynamic in the current frame, our weighting strategy is as follows:

$$\omega(\mathbf{u}^t) = \mathbf{n} + \begin{cases} \varphi_{\text{high}} * \varphi, & I^{t-1}_{\text{dMask}}(\mathbf{u}^t) = 125 \\ \varphi_{\text{low}}, & I^{t-1}_{\text{dMask}}(\mathbf{u}^t) = 255 \\ \varphi_{\text{mid}}, & \text{otherwise} \end{cases}, \tag{15}$$

$$\varphi = 1 - |C^i_{\text{reg}} \cap \left(I^{t-1}_{\text{dMask}} = 255\right)| / |C^i_{\text{reg}}|, \tag{16}$$

where $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \mathbf{0.25})$, I^{t-1}_{dMask} is the dynamic mask of the previous frame, and C^i_{reg} is the cluster region to which the pixel $\mathbf{u}^t$ belongs. We assign a high weight φ_{high} to static regions, slightly higher than φ_{mid} for depth-invalid regions. This is because the optical flow in static regions is more reliable than in depth-invalid regions, which are primarily composed of distant pixels. The lower weight φ_{low} indicates our intention to reduce, rather than completely eliminate, the weight of dynamic regions from the previous frame. By adding a Gaussian noise $\mathbf{n}$, dynamic regions still have a chance to receive higher weights, improving robustness when static regions are misclassified as dynamic. Fig. 5(a) visualizes the assigned weights of the sampled pixels. After sorting all sampled pixel correspondences in descending order based on their weights, we input them into the PROSAC algorithm to compute the $\mathbf{H}^t_{t-1}$. Once matrix $\mathbf{H}^t_{t-1}$ is obtained, optical flow residuals can be calculated by (12) and (13), as shown in Fig. 5(b).

2) Dynamic Region Determination: Statistical analysis of Fig. 5(b) yields the optical flow residual histogram in Fig. 6, allowing for discrimination between dynamic and static regions by selecting an appropriate threshold. To avoid inaccurate discrimination caused by a single fixed threshold, we adopt a double-threshold strategy. Specifically, as shown in Fig. 6, we use the Triangle algorithm [50] to find two residual values τ_{low} and τ_{high}. A pixel $\mathbf{u}^t$ is classified as high-residual if $\delta(\mathbf{u}^t) > \tau_{\text{high}}$ and as static if $\delta(\mathbf{u}^t) < \tau_{\text{low}}$. For the remaining pixels, temporarily classified as low-residual, their dynamic properties are determined by verifying their connectivity with high-residual regions. This verification process reduces errors in mistakenly classifying static regions as dynamic, as some low-residual regions may result from noise or lighting changes. A cluster C_i from Fig. 5(e) is classified as static if it does not contain high-residual regions. In contrast, if C_i contains high-residual regions, we select seed points from each high-residual region and use the residual-aware flood-fill algorithm to fill the high- and low- residual regions. If the area of the filled region exceeds $0.5 * |C^i_{\text{reg}}|$, the entire C_i is classified as dynamic; otherwise, only the filled region is classified as dynamic. The filling process is confined to a single cluster, thereby minimizing false positives. In Fig. 5, despite only a portion of the right leg being identified as high-residual, the entire leg is detected as dynamic after filling.



(a) Before refinement (b) After refinement

Fig. 7. An example of dynamic mask refinement. After refining, the person in the left side of the image is detected as dynamic.

D. Dense Map Reconstruction

Since our method relies on adjacent frames to detect dynamic regions, it may miss-detect dynamic objects moving slowly or intermittently, such as a person momentarily standing still. Although this missed detection does not significantly impact camera trajectory estimation, removing all dynamic regions during map reconstruction remains a goal. To address this issue, we refine the dynamic mask by frames captured at larger time intervals. Specifically, we select one frame out of every five, referred to as a sparse frame. The relative pose $\mathbf{T}^{k-1}_k$ between the k-th sparse frame I^k_D and the $(k-1)$-th one I^{k-1}_D can be obtained via ORB-SLAM2 camera tracking. We then re-project the pixel $\mathbf{u}$ from I^k_D to I^{k-1}_D by

$$\mathbf{u}' = \pi\left({T^k_{k-1}}^{-1} \pi^{-1}\left(\mathbf{u}, I^k_D(\mathbf{u})\right)\right). \tag{17}$$

If the depth reprojection error $\delta_{\text{diff}} = |I^k_D(\mathbf{u}) - I^{k-1}_D(\mathbf{u}')|$ exceeds $\tau_{\text{rep}} * I^k_D(\mathbf{u})$ and $\mathbf{u}$ is not a dynamic pixel in the original dynamic mask, $\mathbf{u}$ will be considered as a new dynamic pixel. For a cluster C_i, if the number of its new dynamic pixels exceeds $\tau_4 * |C^i_{\text{reg}}|$, the entire cluster will be considered as dynamic; otherwise, only the new dynamic pixels are considered as dynamic. A low τ_{rep} and τ_4 may cause the loss of some static regions. Fortunately, overlapping viewpoints often make static information redundant. Therefore, we set low τ_{rep} and τ_4 to aggressively remove dynamic objects and improve map quality. As shown in Fig. 7, a miss-detected person in the original dynamic mask is captured in the refined mask.

After removing the dynamic regions, the remaining pixels are projected into 3D space to generate local point clouds. These local point clouds are then aggregated to construct a global point cloud. However, this point cloud map has limitations. For instance, if a person remains stationary in a certain area, they will be reconstructed into the map, and the corresponding point cloud will not be removed even after they leave. Moreover, the point cloud map cannot represent

TABLE II
LIST OF IMPORTANT PARAMETER GROUPS.

Section	Par.	Value	Par.	Value	Par.	Value
Edge Ext.	τ_{depth}	6.0	τ_1	0.025	τ_2	0.08
Re-Clust.	τ_3	200	λ_1	0.05	λ_2	0.01
	ω_{low}	0.7	ω_{mid}	1.0	ω_{high}	2.0
	τ_{meg}	0.9	τ_{rej}	0.2		
Dyn. Reg. Det.	φ_{low}	0.4	φ_{mid}	1.0	φ_{high}	1.2
Den. Map Rec.	τ_4	0.4	τ_{rep}	0.13		

occupancy and free space, making it unsuitable for advanced missions. In contrast, Octomap [51] offers a more efficient mapping method by using an Octree structure to store point clouds. Octomap not only represents spatial occupancy but also allows online map update based on occupancy probabilities. Thus, we adopt Octomap to fuse the projected point clouds and complete the dense reconstruction.

E. Influence Among Parameter Groups

The important parameter groups used in this paper are listed in Table II. Below, we discuss how certain parameter groups influence others.

1) Edge Extraction on Re-Clustering: Lower τ_1 and τ_2 result in more refined edges, causing over-segmentation and many small clusters. In this case, increasing ω_{high} helps merge small clusters. Conversely, higher τ_1 and τ_2 can lead to under-segmentation, and ω_{low} should be lowered to reduce the likelihood of merging large clusters.

2) Re-Clustering on Dynamic Region Detection: Over-segmentation in the Re-Clustering stage may lead to dynamic regions being misclassified as static. Thus, a lower φ_{high} is required to reduce the weight for static regions. Conversely, under-segmentation may misclassify static regions as dynamic, requiring an increase in φ_{low}.

3) Dense Map Reconstruction: This parameter group is independent of others.

IV. EXPERIMENTS AND DISCUSSION

We extensively evaluate our method on publicly available TUM [52] and Boon [44] datasets and real-world scenes. The overall results of dynamic region detection are presented in Fig. 8. Next, a quantitative analysis of the trajectory estimation accuracy is performed to compare our method against the original ORB-SLAM2 and the state-of-the-art dynamic SLAM methods, including both semantic-independent and semantic-based methods. We also provide the results of dense mapping and a comprehensive discussion about experimental results.

A. Testing Datasets

1) TUM dataset: The TUM dataset consists of RGB-D images (640×480) at 30 Hz collected by a Kinect RGB-D camera in indoor environments, along with camera ground truth trajectories. The main dynamic objects are two people exhibiting two types of motion: sitting (*s*) and walking (*w*). Each scene includes four types of camera motion, denoted as *half*, *rpy*, *static*, and *xyz*. The sitting scene represents lowly dynamic scenes, where two people are sitting beside a desk, moving only parts of their bodies like hands, heads, or feet. This scene helps evaluate the ability to handle objects with hybrid dynamic properties. The walking scene represents highly dynamic scenes, where two people move back and forth in front of the camera, with dynamic regions occupying a significant portion of the field of view. This scene is used to assess the capability of completely removing dynamic objects.

2) Bonn dataset: The Bonn dataset has the same data format as the TUM dataset, but focuses entirely on highly dynamic scenes. In these scenes, people perform various tasks, and new dynamic objects such as moving boxes are introduced, which may pose challenges to semantic-based methods.

3) Real-world scene dataset: Besides using public datasets, we also collected data in real-world laboratory scenes. We captured RGB-D images (640×480) at 30Hz using a RealSense D455 camera and recorded the ground-truth camera trajectory with a VICON motion capture system at 100Hz. Our dataset features two scenes. In *Scene_1*, a person walks back and forth in front of the camera as a moving object. In *Scene_2*, the main moving object is a humanoid robot stepping in place, which is not typically included in existing datasets. Meanwhile, the operator remains static, contradicting common semantic priors. This dataset is proposed to analyze dynamic SLAM performance when lacking or containing incorrect prior information.

B. Experimental Setup

All experiments are performed on a desktop computer with a 2.1 GHz Intel Core i7-12700 CPU and an Nvidia RTX 4060Ti GPU. Image traversal is accelerated using OpenMP for parallel computing, and a separate thread is dedicated for *Optical Flow Residual Calculation*. BroxFlow [53] and DIP flow [54] are used for dense optical flow estimation. We use the implementation of BroxFlow in OpenCV with the parameters α=0.197, β=50.0, *scale_factor*=0.8, *inner_iterations*=10, *outer_iterations*=77, *solver_iterations*=10. The parameters for DIP flow is DIP_sintel.pth[2].

To evaluate the error between the estimated and ground-truth camera trajectories, three popular metrics are used: absolute trajectory error (ATE, m), translational relative pose error (tRPE, m/s), and rotational relative pose error (rRPE, deg/s). For each metric, the Root Mean Square Error (RMSE) and standard deviation (std) are computed. In addition to the original ORB-SLAM2 [2], the most advanced semantic-independent methods LCCRF SLAM [40], DSLAM [41], Re-Fusion [44], DKE-SLAM [16], and semantic-based methods DGS-SLAM [28], DynaSLAM [10], Blitz-SLAM [11] are compared with our method. All these methods, except Re-Fusion, are built based on ORB-SLAM2 or ORB-SLAM3 [3].

C. Trajectory Accuracy on TUM dataset

Table III and IV illustrate the results in lowly and highly dynamic scenes of the TUM dataset, respectively.

[2]Available at https://github.com/zihuazheng/DIP

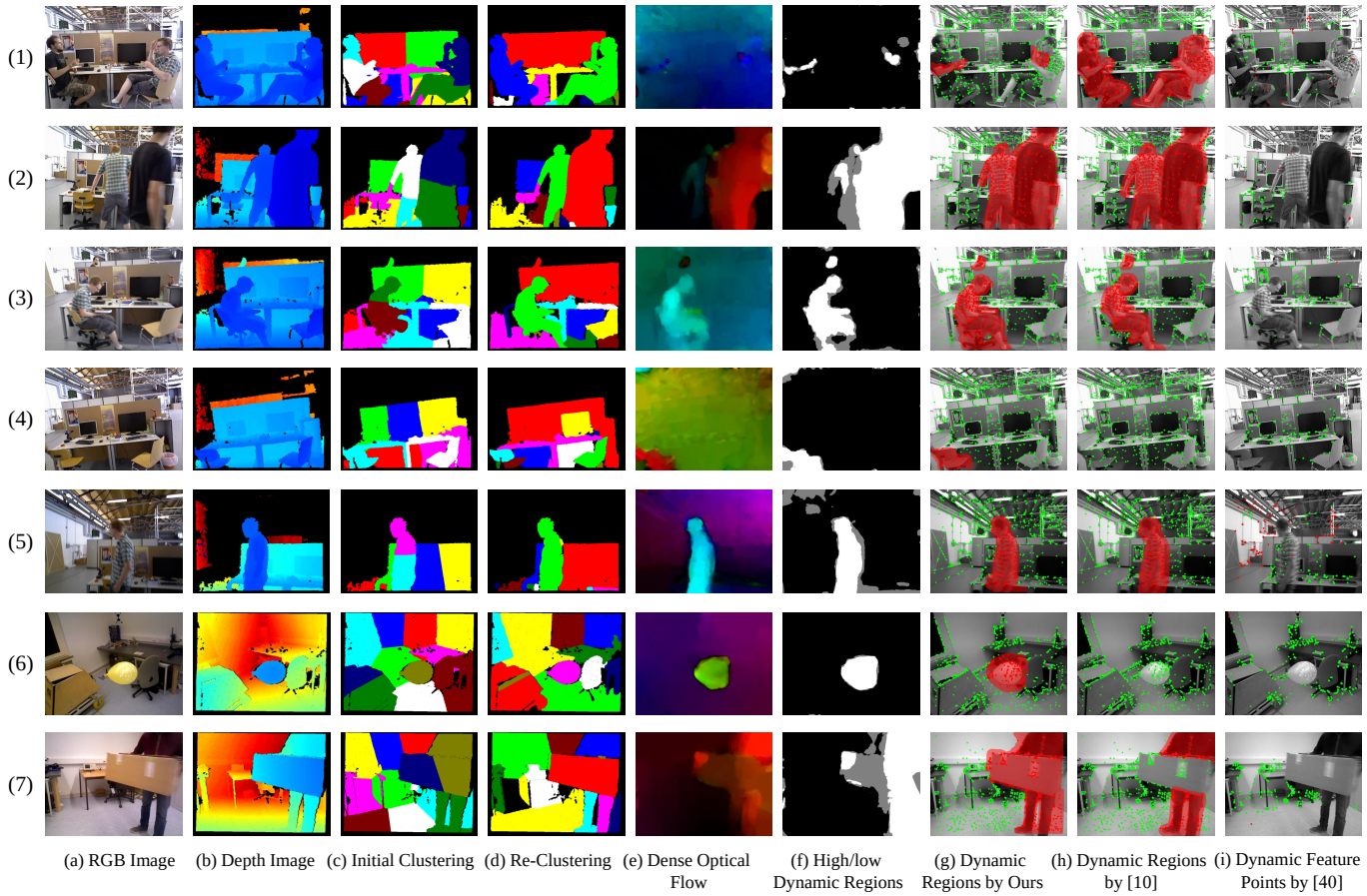


Fig. 8. Overall presentation of our dynamic region detection algorithm and comparison with DynaSLAM [10] and LCCRF SLAM [40]. (**a**) and (**b**) are the algorithm inputs. (**c**)∼ (**f**) are the intermediate results. The final output is shown in (**g**). In (**g**)∼(**h**), red color indicates dynamic and green color indicates static. In (**i**), fewer feature points are shown as LCCRF SLAM displays only the matched feature points.

1) Lowly Dynamic Scenes: ORB-SLAM2 effectively handles the few dynamic elements in lowly dynamic scenes by leveraging a robust outlier rejection strategy and thus achieves high accuracy. Although there is limited space for further improvement in such scenes, our method still provides better results than others, especially in the *fr3/s/half* sequence, where we achieve a 30% improvement in ATE RMSE and a 50% improvement in ATE std compared to ORB-SLAM2. However, we observe a significant decrease in ATE for semantic-based methods like DynaSLAM and Blitz-SLAM in the *fr3/s/xyz* sequence compared to ORB-SLAM2. This can be attributed to the approach employed by the semantic-based methods, which treats the entire human body as a dynamic region and remove it from the scene, even if it exhibits hybrid dynamic properties. Consequently, valuable static information is lost, resulting in degrading accuracy. Fig. 8(1) gives an example. In contrast, semantic-independent methods like ours and LCCRF SLAM accurately detect dynamic regions or points, preserving static information while also eliminating the influence of dynamic regions. As a result, our method does not degrade the performance of the original ORB-SLAM2 across nearly all sequences, which is important when seeking improvements based on an existing method.

2) Highly Dynamic Scenes: Table IV shows that, in general, semantic-based methods outperform semantic-independent ones, except our method, which achieves top-level performance (i.e., either optimal or sub-optimal) in 16 out of 24 evaluations, outperforming other methods. The remarkable performance of semantic-based methods can be explained by two reasons. First, in highly dynamic scenes, treat the entire human body as dynamic for removal is reasonable since people are mostly in motion. Second, dynamic priors from semantic information help semantic-based methods correctly identify human as dynamic. Our method benefits from these reasons and detects dynamic regions similar to DynaSLAM (Fig. 8(2)∼(5)), but without relying on semantic information. Through geometric re-clustering, we achieve more comprehensive object clustering results, facilitating the detection and removal of entire dynamic objects. In addition, the propagation of the dynamic mask further improves dynamic detection by providing dynamic priors, especially in scenes with abundant dynamic regions. As shown in Fig. 8(2), where two persons with different motion patterns occupy approximately half of the image area, our method successfully separates them into distinct clusters and identifies both as dynamic. Furthermore, our dynamic region detection also proves effective in other challenging scenes, such as when a person occupies only a small portion of the image (Fig. 8(3)) or when a chair is being moved by someone outside the image (Fig. 8(4)). These results

TABLE III
COMPARISON OF THE ATE (M), TRPE (M/S) AND RRPE (DEG/S) ON THE TUM LOWLY DYNAMIC SCENES. THE OPTIMAL RESULTS ARE SHOWN IN RED AND THE SUB-OPTIMAL IN BLUE. NOT ALL METHODS PROVIDE RESULTS FOR ALL SEQUENCES. THE (*) INDICATES THE SEMANTIC-BASED METHODS.

Sequence	ORB-SLAM2		LCCRF SLAM		DSLAM		ReFusion		DKE-SLAM		DGS-SLAM*		DynaSLAM*		Blitz-SLAM*		Ours	
	RMSE	std	RMSE	std	RMSE	std	RMSE	std	RMSE	std	RMSE	std	RMSE	std	RMSE	std	RMSE	std
ATE(m)																		
fr3/s/half	0.0197	0.0125	0.0207	0.0099	0.0235	-	-	-	-	-	0.0182	0.0086	0.0194	0.0086	0.0160	0.0076	0.0138	0.0062
fr3/s/rpy	0.0237	0.0161	0.0206	0.0119	0.0225	-	-	-	-	-	-	-	0.0436	0.0326	-	-	0.0225	0.0152
fr3/s/static	0.0099	0.0048	0.0078	0.0039	0.0096	-	-	-	0.0063	-	-	-	0.0068	0.0033	-	-	0.0085	0.0039
fr3/s/xyz	0.0087	0.0044	0.0107	0.0051	0.0091	-	-	-	-	-	0.0092	0.0060	0.0145	0.0060	0.0148	0.0069	0.0087	0.0043
tRPE(m/s)																		
fr3/s/half	0.0292	0.0183	0.0301	0.0133	0.0354	-	-	-	-	-	0.0276	0.0120	0.0285	0.0120	0.0165	0.0073	0.0200	0.0087
fr3/s/rpy	0.0335	0.0215	0.0298	0.0170	0.0320	-	-	-	-	-	-	-	0.0620	0.0421	-	-	0.0320	0.0200
fr3/s/static	0.0162	0.0084	0.0131	0.0064	0.0138	-	-	-	0.0093	-	-	-	0.0096	0.0045	-	-	0.0136	0.0065
fr3/s/xyz	0.0129	0.0061	0.0156	0.0070	0.0134	-	-	-	-	-	0.0134	0.0073	0.0208	0.0073	0.0144	0.0071	0.0131	0.0060
rRPE(deg/s)																		
fr3/s/half	0.7474	0.3155	0.8604	0.3744	0.8699	-	-	-	-	-	0.7876	0.3488	0.8342	0.3488	0.5981	0.2739	0.7326	0.3204
fr3/s/rpy	0.7620	0.3333	0.9235	0.4609	0.9047	-	-	-	-	-	-	-	0.9784	0.4872	-	-	0.8230	0.3706
fr3/s/static	0.4009	0.1761	0.3556	0.1552	0.3786	-	-	-	0.3400	-	-	-	0.3620	0.1703	-	-	0.3736	0.1649
fr3/s/xyz	0.5651	0.2972	0.5932	0.2832	0.5792	-	-	-	-	-	0.5938	0.2651	0.6249	0.2651	0.5024	0.2634	0.5671	0.2959

[1] ReFusion is not compared in lowly dynamic sequences because it is not developed based on ORB-SLAM2 or ORB-SLAM3.

TABLE IV
COMPARISON OF THE ATE (M), TRPE (M/S) AND RRPE (DEG/S) ON THE TUM HIGHLY DYNAMIC SCENES. THE OPTIMAL RESULTS ARE SHOWN IN RED AND THE SUB-OPTIMAL IN BLUE. NOT ALL METHODS PROVIDE RESULTS FOR ALL SEQUENCES. THE (*) INDICATES THE SEMANTIC-BASED METHODS.

Sequence	ORB-SLAM2		LCCRF SLAM		DSLAM		ReFusion		DKE-SLAM		DGS-SLAM*		DynaSLAM*		Blitz-SLAM*		Ours	
	RMSE	std	RMSE	std	RMSE	std	RMSE	std	RMSE	std	RMSE	std	RMSE	std	RMSE	std	RMSE	std
ATE(m)																		
fr3/w/half	0.4543	0.2524	0.0280	0.0129	0.0354	-	0.0474	0.0261	0.0268	-	0.0259	0.0157	0.0279	0.0157	0.0256	0.0126	0.0253	0.0111
fr3/w/rpy	0.5391	0.2283	0.0441	0.0264	0.1608	-	0.2573	0.1876	0.0351	-	0.0301	0.0190	0.0291	0.0190	0.0356	0.0220	0.0284	0.0158
fr3/w/static	0.3194	0.1819	0.0155	0.0085	0.0108	-	0.0174	0.0089	0.0074	-	0.0059	0.0032	0.0064	0.0032	0.0102	0.0052	0.0069	0.0033
fr3/w/xyz	0.7521	0.4712	0.0215	0.0106	0.0874	-	0.0739	0.0573	0.0177	-	0.0156	0.0086	0.0163	0.0086	0.0153	0.0078	0.0147	0.0079
tRPE(m/s)																		
fr3/w/half	0.3216	0.2629	0.0405	0.0184	0.0517	-	0.0703	0.0357	0.0382	-	0.0366	0.0149	0.0394	0.0149	0.0253	0.0123	0.0360	0.0176
fr3/w/rpy	0.3880	0.2823	0.0634	0.0357	0.2299	-	0.3609	0.2571	0.0505	-	0.0432	0.0262	0.0415	0.0262	0.0473	0.0283	0.0408	0.0205
fr3/w/static	0.1928	0.1773	0.0222	0.0114	0.0141	-	0.0254	0.0140	0.0107	-	0.0101	0.0044	0.0102	0.0044	0.0129	0.0069	0.0099	0.0045
fr3/w/xyz	0.4834	0.3663	0.0311	0.0147	0.1266	-	0.1027	0.0708	0.0254	-	0.0299	0.0119	0.0235	0.0119	0.0197	0.0096	0.0211	0.0105
rRPE(deg/s)																		
fr3/w/half	6.6515	5.3990	0.9320	0.4038	0.9854	-	2.3245	1.0633	0.9700	-	0.8848	0.4012	0.8839	0.4012	0.7879	0.3751	0.9009	0.3924
fr3/w/rpy	7.5906	5.4768	1.2468	0.6697	4.6327	-	9.2654	8.6493	1.1800	-	0.9213	0.5701	0.8788	0.5701	1.0841	0.6668	0.9010	0.4499
fr3/w/static	3.5991	3.2457	0.5001	0.2292	0.3293	-	0.5481	0.2598	0.3000	-	0.2639	0.1259	0.2659	0.1259	0.3038	0.1437	0.2795	0.1196
fr3/w/xyz	8.8419	6.6762	0.7721	0.4294	2.7413	-	2.3152	1.4864	0.6400	-	0.6425	0.3848	0.6212	0.3848	0.6132	0.3348	0.6330	0.3910

TABLE V
COMPARISON OF THE ATE (M) RMSE ON THE BONN DATASET. THE OPTIMAL RESULTS ARE SHOWN IN RED AND THE SUB-OPTIMAL IN BLUE. THE (*) INDICATES THE SEMANTIC-BASED METHODS.

Sequence	LCCRF SLAM	Dyna SLAM*	DGS-SLAM*	Ours
balloon	0.0310	0.0306	0.0228	0.0279
balloon2	0.0257	0.0255	0.0248	0.0239
balloon_tracking	0.0350	0.0423	0.0321	0.0290
balloon_tracking2	0.0272	0.0309	0.0277	0.0273
moving_no_box	0.0197	0.0295	0.0180	0.0190
moving_no_box2	0.0294	0.0301	0.0281	0.0307
person_tracking	0.0418	0.0386	0.0609	0.0374
person_tracking2	0.0356	0.0302	0.0484	0.0298
placing_no_box	0.0140	0.1254	0.0158	0.0163
placing_no_box2	0.0160	0.0217	0.0281	0.0172
placing_no_box3	0.0440	0.0650	0.0337	0.0260
removing_no_box	0.0120	0.0172	0.0149	0.0147
removing_no_box2	0.0192	0.0233	0.0199	0.0185
synchronous2	0.0072	0.0084	0.0063	0.0071
Mean	0.0239	0.0346	0.0254	0.0217
Max	0.0440	0.1254	0.0609	0.0374

convincingly validate our advantage in accurately detecting truly moving regions within the scene.

However, other semantic-independent methods often suffer from under-detection of dynamic points/regions and fail to match the accuracy of semantic-based methods. LCCRF SLAM focuses solely on local dynamic information without considering the dynamic properties of entire objects. It miss-detects or wrongly identifies dynamic feature points, as shown in Fig. 8(5). DSLAM assumes that the connected component with the most feature points is static. However, this assumption does not hold in all scenes, particularly when the SLAM system is just starting. ReFusion attempts to identify dynamic regions by comparing the current frame with the reconstructed 3D model but lacks explicit handling of dynamic objects, leading to discontinuity and inaccuracy in dynamic detection. DKE-SLAM, despite using a region-based approach similar to ours, faces challenges in identifying whole dynamic objects, especially when they occupy a large image area. This is because DKE-SLAM segments a single

dynamic object into multiple clusters without any strategies to merge these over-segmented clusters.

D. Trajectory Accuracy on the Bonn dataset

We evaluate our method on the Bonn dataset and compare it with LCCRF SLAM, DynaSLAM, and DGS-SLAM. The quantitative results in Table V illustrate that our method achieves top-level performances in 12 out of 14 evaluations, with the lowest MEAN of 0.0216m and MAX of 0.0374m, surpassing other methods. These results demonstrate the generalizability and stability of our method across various scenarios.

While DynaSLAM performs well on the TUM dataset, it exhibits significant inaccuracies in the *placing_no_box* and *placing_no_box3* sequences. This performance degradation is mainly due to the unexpected moving objects, such as balloons and boxes, which are not detected by the semantic segmentation module of DynaSLAM, as shown in Fig. 8(6) and (7). This highlights a critical limitation of semantic-based methods: their reliance on semantic priors and pre-trained models restricts their ability to handle unknown dynamic objects. In contrast, LCCRF SLAM and our method correctly detect the feature points on the box and balloon as dynamic. These results show that semantic-independent methods maintain consistent performance across different scenes. To reduce the reliance on semantic information, DGS-SLAM incorporates an additional dynamic object detection module. It only performs semantic segmentation on specific frames, while in other frames, dynamic detection is done using geometric and appearance information, similar to semantic-independent methods. This adjustment leads to a noticeable improvement when compared with DynaSLAM.

E. Real-world Scene

We compare our method with DynaSLAM, and the results are presented in Table VI. In *Scene_1* , DynaSLAM effectively detects the moving person and removes the corresponding feature points (Fig. 9(a)), resulting in an ATE of 0.0128m, which is very close to our ATE of 0.0105m. However, in *Scene_2*, DynaSLAM exhibits an error 80% higher than ours, which can be attributed to two factors: first, it falsely detects movable but static objects (e.g., the operators) as dynamic; second, it fails to detect objects not trained by the semantic segmentation networks (e.g., the humanoid robot), as shown in (Fig. 9(b)). As a result, incorrect dynamic features are used, and fewer static features contribute to pose estimation. Although retraining the network with specific object samples could improve performance, the additional training time limits the rapid deployment of semantic-based methods in real-world applications. Conversely, our method makes no assumptions about object types or motion patterns, and thus accurately classifies the humanoid robot as dynamic and the operators as static (Fig. 9(d)), highlighting superior generalizability when facing unknown objects.

F. Dense Reconstruction Results

To highlight the capability of our method in removing dynamic regions and accurately estimating camera trajectory, we present the dense reconstruction results in Fig. 10 and Fig. 11. In the original point cloud map of Fig. 11(a), a significant amount of ghosting point cloud from people can be observed. This occurs for two reasons. First, our dynamic detection fails to detect the moving person when he briefly remains stationary. Second, at the beginning and end of the image sequence, two people sitting at the same location are reconstructed into the map. By converting point cloud map into Octomap, most of the ghosting is removed. Furthermore, with the combination of Octomap and dynamic mask refinement, our method achieves the cleanest reconstruction, as shown in Fig. 11(c).

In Fig. 10, we compare the reconstruction results between our method and ReFusion. Evident noise, incompleteness and distortion can be seen in the results from ReFusion. In contrast, our method, benefiting from accurate camera trajectory estimation, dynamic mask refinement and Octomap integration, achieves correct and clean reconstructions of the static environment.

TABLE VI
COMPARISON OF THE ATE (M) RMSE ON THE REAL-WORLD SCENE. THE (*) INDICATES THE SEMANTIC-BASED METHODS.

Sequence	DynaSLAM*	Ours
Scene_1	0.0128	0.0105
Scene_2	0.0184	0.0104

TABLE VII
AVERAGE TIME CONSUMPTION PER FRAME OF DYNAMIC SLAM METHODS. THE (*) INDICATES THE SEMANTIC-BASED METHODS.

Method	DynaSLAM*	Blitz-SLAM*	DGS-SLAM*	LCCRF SLAM
Time	485.3ms	83.3ms	38.5ms	66.7ms
Method	DSLAM	DKE-SLAM	ReFusion	Ours
Time	31.25ms	33.3ms	125.1ms	117.8ms

TABLE VIII
TIME CONSUMPTION OF EACH PART IN OUR DYNAMIC SLAM.

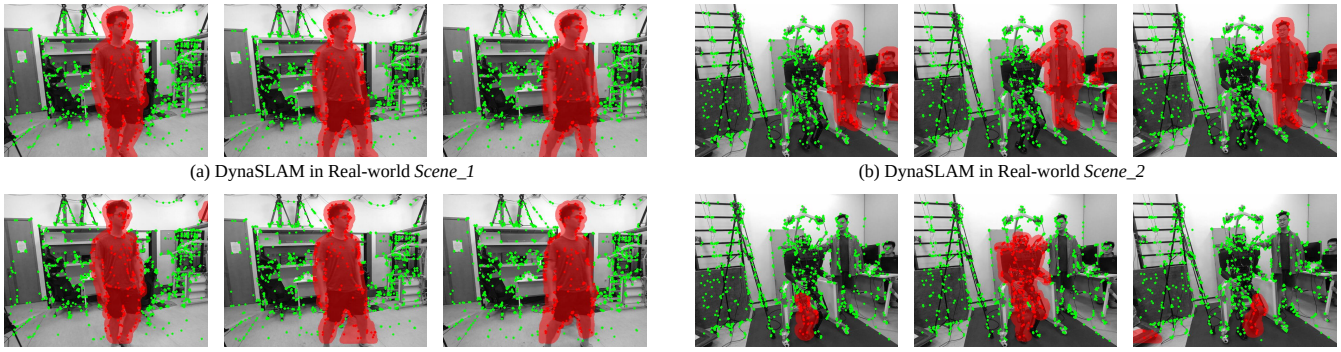
Thread 1		Thread 2	
Initial Clustering	17.7ms	Optical Flow Calculation	26.6ms
Edge Extraction	18.6ms	Residual Calculation	6.5ms
Re-Clustering	40.6ms		
Dynamic Region Determination		2.3ms	
ORB-SLAM2 Camera Tracking		30.5ms	
Total		117.8ms	

TABLE IX
HARDWARE UTILIZATION BEFORE AND DURING THE EXECUTION.

Measurement	CPU	Memory	GPU
Before execution	3.6%	9.3%	14.6%
During execution	43.5%	15.4%	23.7%

G. Analyses

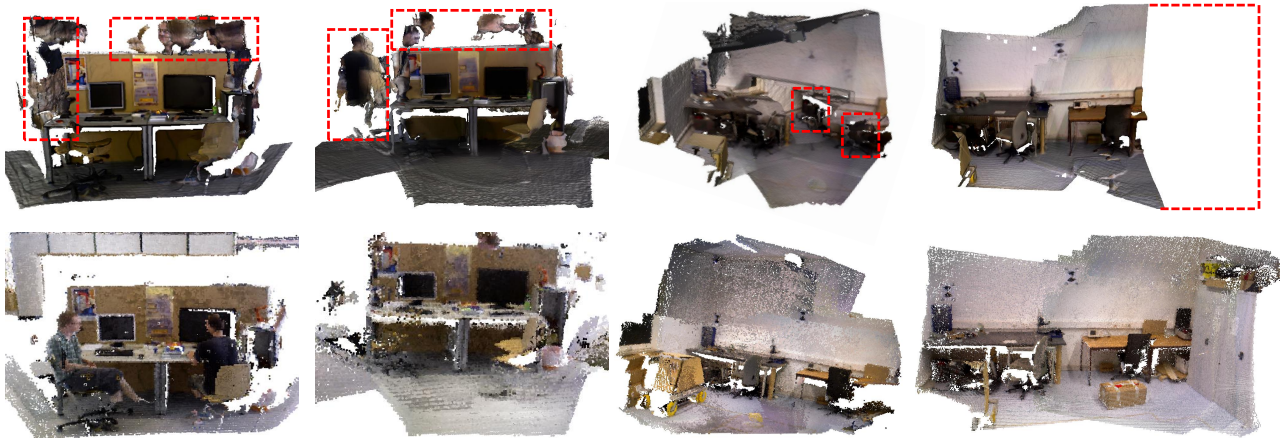
We analyze the current methods in terms of runtime efficiency, scalability, and scene robustness. For clarity, we categorize Blitz-SLAM, DynaSLAM, and DGS-SLAM as semantic-based methods. Semantic-independent methods are



(a) DynaSLAM in Real-world *Scene_1* (b) DynaSLAM in Real-world *Scene_2*

(c) Our method in Real-world *Scene_1* (d) Our method in Real-world *Scene_2*

Fig. 9. Comparison of dynamic region detection between DynaSLAM and our method. DynaSLAM correctly detects the dynamic person in *Scene_1*, but misclassifies the stepping humanoid robot and static operators in *Scene_2*. In contrast, our method accurately detects the dynamic objects in both scenes.



(a) TUM *fr3_w_static* (b) TUM *fr3_w_rpy* (c) Bonn *balloon_tracking2* (d) Bonn *placing_no_box3*

Fig. 10. Dense reconstruction results of ReFusion (top) and ours (bottom). In (**a**) and (**b**), ReFusion fails to completely remove all dynamic regions. In (**c**), significant pose estimation errors by ReFusion result in reconstructing the same stationary chair in two different locations and causing a distorted scene map. In (**d**), ReFusion can only provide an incomplete reconstruction.

(a) Point Cloud Map (b) Octomap (c) Ours

Fig. 11. Dense reconstruction on TUM *fr3_w_static*. Our method achieves the cleanest result through combinating dynamic mask refinement and Octomap.

divided into region-based methods (Re-fusion, DKE-SLAM and ours) and point-based methods (LCCRF SLAM and DSLAM).

1) Analysis for Runtime Efficiency: Table VII summarizes the runtime consumption of current methods. The runtime efficiency of semantic-based methods primarily depends on the used networks. For example, Blitz-SLAM uses the lightweight BlitzNet [55] for semantic segmentation, largely improving its speed compared with DynaSLAM, which uses Mask-RCNN [25]. DGS-SLAM futher enhances efficiency by applying semantic segmentation only on selected frames. Feature point-based methods are generally the most efficient, as they focus on dynamic feature points rather than entire regions. In contrast, region-based methods involve more pixel-level computations, requiring higher computational resources. Although as a region-based method, DKE-SLAM only performs initial image clustering, with subsequent process focusing on feature points.

2) Analysis for Scalability: Besides localization, SLAM is also required to construct environmental maps. The quality of constructed maps affects a SLAM method's scalability for extended applications. Point-based methods, while efficient for localization, only provide sparse maps, which lack detailed environmental information and are inadequate for advanced applications. Conversely, semantic-based and region-based methods can comprehensively detect dynamic regions and construct dense static maps, showing better scalability. These methods can also easily integrate dynamic object tracking or re-localization modules.

3) Analysis for scene robustness: Semantic-based methods rely heavily on semantic priors to detect dynamic objects, resulting in a double-edged sword effect. This reliance can cause significant performance drops when encountering unknown objects or situations where reality contradicts the prior knowledge. However, the benefit of semantic priors lies in reducing dependence on other data sources, like color and depth

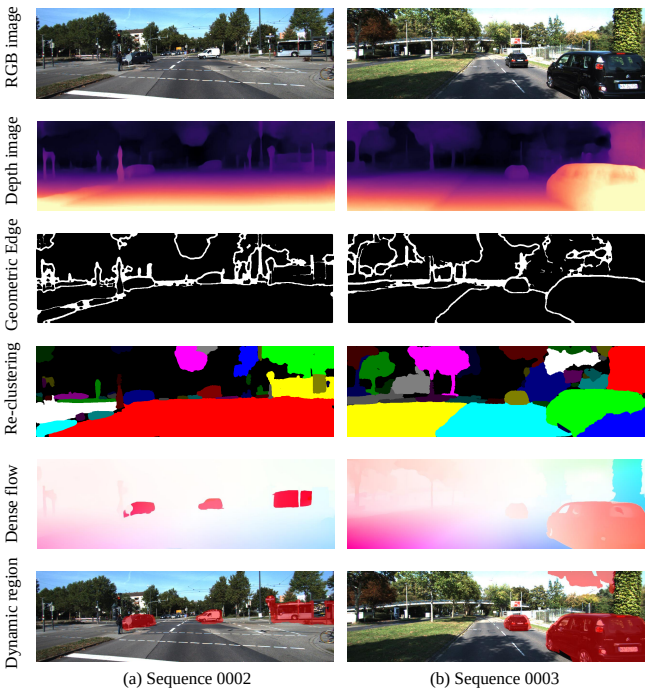
information, potentially leading to more stable performance under varying lighting conditions and fast camera motion. In summary, semantic-based methods are more sensitive to object semantic information but more robust to scene variations. Conversely, the robustness of semantic-independent methods exhibits opposite characteristics.

H. Discussion

We extensively compared our method with state-of-the-art dynamic SLAM methods on two public datasets and real-world scenes. We observe that semantic-independent and semantic-based methods excel in different dynamic settings. Semantic-independent methods, which focus on detecting truly moving regions using geometric and appearance information, perform better in lowly dynamic scenes. They preserve the static information on objects with hybrid dynamic properties, thus hardly degrading the performance of ORB-SLAM2. They also maintain accuracy when the dynamic object classes vary, as no assumptions on object classes are made. Our method, as a semantic-independent method, naturally inherits these advantages. On the other hand, semantic-based methods benefit from two aspects in highly dynamic scenes: semantic priors and pixel-wise masks. By disregarding movable objects and using static information for initial pose estimation, semantic-based methods effectively handle large dynamic regions. Moreover, pixel-wise masks help reduce the mis-detection of dynamic regions or features. Our method introduces dynamic prior propagation and geometric re-clustering to address these issues, which other semantic-independent methods struggle to handle. In summary, our method combines the advantages of both semantic-independent and semantic-based methods. Experimental results showcase that our method achieves superior camera trajectory estimation accuracy in both lowly and highly dynamic scenes, even when facing unknown dynamic objects. It shows significant potential for advanced tasks like robot navigation [56], scene description [57] and panoptic mapping [58] by providing accurate localization and clean environment reconstruction.

Our method still faces three limitations. First, our dynamic region detection algorithm is inevitably affected by illumination changes due to the use of dense optical flow. These changes may cause some static regions to be erroneously identified as dynamic, impacting camera trajectory estimation. This issue can be alleviated by adaptive histogram equalization [59] and photometric calibration [4]. Additionally, specific strategies can be designed to counteract errors in dynamic region detection caused by illumination changes. For instance, when illumination changes incorrectly classify most feature points as dynamic, they can still be used to maintain tracking and prevent system failure. This introduces only minor errors since illumination changes are brief.

Second, our method involves extensive image traversals and morphological operations, resulting in substantial processing time. At 117.8ms per frame (9Hz), our method is not faster than other methods (see Table VII, VIII and IX). Offloading the image traversal and morphological operations to the GPU is a potential solution to improve speed. Another solution is to modify the system framework. Specifically, the main thread focuses solely on detecting dynamic feature points by comparing them with previous frames where dynamic region detection has been completed. Dynamic region detection can run in a separate thread, parallel to camera tracking. Once a frame completes dynamic region detection, subsequent frames can reference it for dynamic feature point detection. This prevents the main thread from being blocked by computationally intensive dynamic region detection.



(a) Sequence 0002 (b) Sequence 0003

Fig. 12. The elevations on KITTI Tracking datasets.

TABLE X
LIST OF PARAMETERS FOR OUTDOOR SCENES. THE ONES THAT DIFFER FROM INDOOR SCENES ARE HIGHLIGHTED WITH UNDERLINE.

Section	Par.	Value	Par.	Value	Par.	Value
Edge Ext.	τ_{depth}	40.0	τ_1	0.1	τ_2	0.3
Re-Clust.	τ_3	200	λ_1	0.05	λ_2	0.01
	ω_{low}	0.7	ω_{mid}	1.0	ω_{high}	3.0
	τ_{meg}	0.7	τ_{rej}	0.15		
Dyn. Reg. Det.	φ_{low}	0.4	φ_{mid}	1.0	φ_{high}	1.2
Den. Map Rec.	τ_4	0.4	τ_{rep}	0.13		

Third, we note that the re-clustering approach in outdoor scenes needs further improvement. We conduct evaluations on the KITTI Tracking dataset [60], as shown in Fig. 12. The parameters used are list in X. We use Monodepth2 [61] for monocular depth estimation and DIP flow [54] for dense optical flow estimation. While our method performs well at close distances, the clustering becomes cluttered for distant and small regions. There are two main reasons: (1) Monocular depth estimation has difficulty in producing accurate results for distant regions and object edges; (2) The geometric clustering approach struggles to merge distant and small regions because

their depth values differ significantly. In future research, we plan to enhance the geometric clustering approach by integrating color information to merge clusters and utilizing optical flow contours to better distinguish between different objects.

V. CONCLUSION

In conclusion, this paper presents a novel semantic-independent dynamic SLAM method capable of identifying truly moving regions, regardless of object categories or motion patterns. We enhance object clustering through geometric re-clustering and detect dynamics within each cluster. The dynamic detection results are then propagated as priors to improve robustness and are refined for clean and dense reconstruction. Experimental results demonstrate the effectiveness of our method in terms of localization accuracy, environment reconstruction, and generalizability to diverse dynamic objects. This facilitates the rapid deployment of our method in unfamiliar settings, supporting applications that require precise visual understanding.

Future research will address challenges related to illumination variations and high computational demands. Additionally, we plan to extend our work to outdoor environments, which involve a broader range of objects, greater depth changes, and more dispersed scenes.

APPENDIX

During the re-clustering process, we decide whether to merge two clusters by comparing their depth information. Specifically, we normalize the depth value histograms of both clusters and perform a histogram comparison. Clusters that exhibit high similarity are merged. Let the depth value histograms of two clusters be H_1 and H_2, and their mean values be $\bar{H}_1$ and $\bar{H}_2$, calculate as

$$\bar{H}_k = \frac{1}{N}\sum\nolimits_i^N H_k(i), \quad \text{for } k = 1, 2 \tag{A1}$$

where N is the total number of bins in the histogram. The correlation comparison is denoted as cmp_1 and calculated as

$$\text{cmp}_1\left(H_1, H_2\right) = \frac{\sum_i^N \left(H_1(i) - \bar{H}_1\right)\left(H_2(i) - \bar{H}_2\right)}{\sqrt{\sum_i^N \left(H_1(i) - \bar{H}_1\right)^2 \sum_i^N \left(H_2(i) - \bar{H}_2\right)^2}}. \tag{A2}$$

The result of cmp_1 falls within $(-1, 1)$, with values closer to 1 indicating greater similarity, and vice versa. The Bhattacharyya distance comparison is denoted as cmp_2 and calculated as

$$\text{cmp}_2\left(H_1, H_2\right) = \sqrt{1 - \frac{1}{\sqrt{\bar{H}_1 \bar{H}_2 N^2}} \sum\nolimits_i^N \sqrt{H_1(i) \cdot H_2(i)}}. \tag{A3}$$

The result of cmp_2 falls within $(0, 1)$, with values closer to 0 indicating greater similarity. The normalized intersection comparison is denoted as cmp_3 and calculated as

$$\text{cmp}_3\left(H_1, H_2\right) = \frac{\sum_i^N \min\left(H_1(i), H_2(i)\right)}{\sum_i^N \max\left(H_1(i), H_2(i)\right)}. \tag{A4}$$

The result of cmp_3 falls within $(0, 1)$, with values closer to 1 indicating greater similarity. In summary, the similarity between two depth value histograms is represented as

$$S(H_1, H_2) = \text{cmp}_1\left(H_1, H_2\right) + \left(1 - \text{cmp}_2\left(H_1, H_2\right)\right) + \text{cmp}_3\left(H_1, H_2\right). \tag{A5}$$

REFERENCES

[1] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Transactions on robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[2] R. Mur-Artal and J. D. Tardós, "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE transactions on robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[3] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós, "Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.

[4] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 3, pp. 611–625, 2017.

[5] L. Zhao, K.-K. Ma, Z. Liu, Q. Yin, and J. Chen, "Real-time scene-aware lidar point cloud compression using semantic prior representation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 8, pp. 5623–5637, 2022.

[6] G. Wang, J. Zhong, S. Zhao, W. Wu, Z. Liu, and H. Wang, "3d hierarchical refinement and augmentation for unsupervised learning of depth and pose from monocular video," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 33, no. 4, pp. 1776–1786, 2022.

[7] Y. Sun, M. Liu, and M. Q.-H. Meng, "Motion removal for reliable RGB-D SLAM in dynamic environments," *Robotics and Autonomous Systems*, vol. 108, pp. 115–128, Oct. 2018.

[8] M. R. U. Saputra, A. Markham, and N. Trigoni, "Visual slam and structure from motion in dynamic environments: A survey," *ACM Computing Surveys (CSUR)*, vol. 51, no. 2, pp. 1–36, 2018.

[9] Z. Xu, Z. Rong, and Y. Wu, "A survey: which features are required for dynamic visual simultaneous localization and mapping?" *Visual Computing for Industry, Biomedicine, and Art*, vol. 4, no. 1, pp. 1–16, 2021.

[10] B. Bescos, J. M. Facil, J. Civera, and J. Neira, "DynaSLAM: Tracking, Mapping, and Inpainting in Dynamic Scenes," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 4076–4083, Oct. 2018.

[11] Y. Fan, Q. Zhang, Y. Tang, S. Liu, and H. Han, "Blitz-SLAM: A semantic SLAM in dynamic environments," *Pattern Recognition*, vol. 121, p. 108225, Jan. 2022.

[12] A. Li, J. Wang, M. Xu, and Z. Chen, "DP-SLAM: A visual SLAM with moving probability towards dynamic environments," *Information Sciences*, vol. 556, pp. 128–142, May 2021.

[13] C. Yu, Z. Liu, X.-J. Liu, F. Xie, Y. Yang, Q. Wei, and Q. Fei, "DS-SLAM: A Semantic Visual SLAM towards Dynamic Environments," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Madrid: IEEE, Oct. 2018, pp. 1168–1174.

[14] H. Wu, Y. Li, W. Xu, F. Kong, and F. Zhang, "Moving event detection from lidar point streams," *Nature Communications*, vol. 15, no. 1, p. 345, 2024.

[15] R. Scona, M. Jaimez, Y. R. Petillot, M. Fallon, and D. Cremers, "StaticFusion: Background Reconstruction for Dense RGB-D SLAM in Dynamic Environments," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. Brisbane, QLD: IEEE, May 2018, pp. 3849–3856.

[16] X. Yang, Z. Yuan, D. Zhu, C. Chi, K. Li, and C. Liao, "Robust and Efficient RGB-D SLAM in Dynamic Environments," *IEEE Transactions on Multimedia*, vol. 23, pp. 4208–4219, 2021.

[17] M. Jaimez, C. Kerl, J. Gonzalez-Jimenez, and D. Cremers, "Fast odometry and scene flow from RGB-D cameras based on geometric clustering," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. Singapore, Singapore: IEEE, May 2017, pp. 3992–3999.

[18] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 6, pp. 1052–1067, 2007.

[19] R. Zhang, J. Tan, Z. Cao, L. Xu, Y. Liu, L. Si, and F. Sun, "Part-aware correlation networks for few-shot learning," *IEEE Transactions on Multimedia*, 2024.

[20] R. Zhang, Z. Cao, S. Yang, L. Si, H. Sun, L. Xu, and F. Sun, "Cognition-driven structural prior for instance-dependent label transition matrix estimation," *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

[21] W. Wang, Y. Hu, and S. Scherer, "Tartanvo: A generalizable learning-based vo," in *Conference on Robot Learning*. PMLR, 2021, pp. 1761–1772.

[22] Z. Teed and J. Deng, "Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras," *Advances in neural information processing systems*, vol. 34, pp. 16 558–16 569, 2021.

[23] Z. Min, Y. Yang, and E. Dunn, "Voldor: Visual odometry from log-logistic dense optical flow residuals," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 4898–4909.

[24] S. Shen, Y. Cai, W. Wang, and S. Scherer, "Dytanvo: Joint refinement of visual odometry and motion segmentation in dynamic environments," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 4048–4055.

[25] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[26] M.-F. Yu, L. Zhang, W.-F. Wang, and J.-H. Wang, "Scp-slam: Accelerating dynaslam with static confidence propagation," in *2023 IEEE Conference Virtual Reality and 3D User Interfaces (VR)*. IEEE, 2023, pp. 509–518.

[27] Y. Liu and J. Miura, "Rds-slam: Real-time dynamic slam using semantic segmentation methods," *Ieee Access*, vol. 9, pp. 23 772–23 785, 2021.

[28] L. Yan, X. Hu, L. Zhao, Y. Chen, P. Wei, and H. Xie, "DGS-SLAM: A Fast and Robust RGBD SLAM in Dynamic Environments Combined by Geometric and Semantic Information," *Remote Sensing*, vol. 14, no. 3, p. 795, Feb. 2022.

[29] J. He, M. Li, Y. Wang, and H. Wang, "Ovd-slam: An online visual slam for dynamic environments," *IEEE Sensors Journal*, 2023.

[30] G. Jocher, A. Stoken, J. Borovec, L. Changyu, A. Hogan, L. Diaconu, J. Poznanski, L. Yu, P. Rai, R. Ferriday *et al.*, "ultralytics/yolov5: v3. 0," *Zenodo*, 2020.

[31] J. Cheng, H. Zhang, and M. Q.-H. Meng, "Improving Visual Localization Accuracy in Dynamic Environments Based on Dynamic Region Removal," *IEEE Transactions on Automation Science and Engineering*, pp. 1–12, 2020.

[32] L. Xiao, J. Wang, X. Qiu, Z. Rong, and X. Zou, "Dynamic-slam: Semantic monocular visual localization and mapping based on deep learning in dynamic environment," *Robotics and Autonomous Systems*, vol. 117, pp. 1–16, 2019.

[33] S. Cheng, C. Sun, S. Zhang, and D. Zhang, "Sg-slam: a real-time rgb-d visual slam toward dynamic scenes with semantic and geometric information," *IEEE Transactions on Instrumentation and Measurement*, vol. 72, pp. 1–12, 2022.

[34] N. Brasch, A. Bozic, J. Lallemand, and F. Tombari, "Semantic monocular slam for highly dynamic environments," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 393–400.

[35] C. Xu, E. Bonetto, and A. Ahmad, "Dynapix slam: A pixel-based dynamic slam approach," *arXiv preprint arXiv:2309.09879*, 2023.

[36] W. Tan, H. Liu, Z. Dong, G. Zhang, and H. Bao, "Robust monocular slam in dynamic environments," in *2013 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 2013, pp. 209–218.

[37] S. Li and D. Lee, "RGB-D SLAM in Dynamic Environments Using Static Point Weighting," *IEEE Robotics and Automation Letters*, vol. 2, no. 4, pp. 2263–2270, Oct. 2017.

[38] J. Cheng, Y. Sun, and M. Q.-H. Meng, "Improving monocular visual slam in dynamic environments: An optical-flow-based approach," *Advanced Robotics*, vol. 33, no. 12, pp. 576–589, 2019.

[39] A. Kundu, K. M. Krishna, and J. Sivaswamy, "Moving object detection by multi-view geometric techniques from a single camera mounted robot," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009, pp. 4306–4312.

[40] Z.-J. Du, S.-S. Huang, T.-J. Mu, Q. Zhao, R. R. Martin, and K. Xu, "Accurate Dynamic SLAM Using CRF-Based Long-Term Consistency," *IEEE Transactions on Visualization and Computer Graphics*, vol. 28, no. 4, pp. 1745–1757, Apr. 2022.

[41] W. Dai, Y. Zhang, P. Li, Z. Fang, and S. Scherer, "Rgb-d slam in dynamic environments using point correlations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 1, pp. 373–389, 2020.

[42] H. Yin, S. Li, Y. Tao, J. Guo, and B. Huang, "Dynam-slam: An accurate, robust stereo visual-inertial slam method in dynamic environments," *IEEE Transactions on Robotics*, vol. 39, no. 1, pp. 289–308, 2022.

[43] T. Zhang, H. Zhang, Y. Li, Y. Nakamura, and L. Zhang, "FlowFusion: Dynamic Dense RGB-D SLAM Based on Optical Flow," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. Paris, France: IEEE, May 2020, pp. 7322–7328.

[44] E. Palazzolo, J. Behley, P. Lottes, P. Giguère, and C. Stachniss, "ReFusion: 3D Reconstruction in Dynamic Environments for RGB-D Cameras Exploiting Residuals," Aug. 2019.

[45] L. Bose and A. Richards, "Fast depth edge detection and edge based RGB-D SLAM," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. Stockholm, Sweden: IEEE, May 2016, pp. 1323–1330.

[46] C. Feng, Y. Taguchi, and V. R. Kamat, "Fast plane extraction in organized point clouds using agglomerative hierarchical clustering," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 6218–6225.

[47] Y. Sun, M. Liu, and M. Q.-H. Meng, "Improving RGB-D SLAM in dynamic environments: A motion removal approach," *Robotics and Autonomous Systems*, vol. 89, pp. 110–122, Mar. 2017.

[48] O. Chum and J. Matas, "Matching with PROSAC — Progressive Sample Consensus," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1. San Diego, CA, USA: IEEE, 2005, pp. 220–226.

[49] X. Miao, Y. Bai, H. Duan, Y. Huang, F. Wan, X. Xu, Y. Long, and Y. Zheng, "Ds-depth: Dynamic and static depth estimation via a fusion cost volume," *IEEE Transactions on Circuits and Systems for Video Technology*, 2023.

[50] G. W. Zack, W. E. Rogers, and S. A. Latt, "Automatic measurement of sister chromatid exchange frequency." *Journal of Histochemistry & Cytochemistry*, vol. 25, no. 7, pp. 741–753, 1977.

[51] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: An efficient probabilistic 3d mapping framework based on octrees," *Autonomous robots*, vol. 34, pp. 189–206, 2013.

[52] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *2012 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2012, pp. 573–580.

[53] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, "High accuracy optical flow estimation based on a theory for warping," in *Computer Vision-ECCV 2004: 8th European Conference on Computer Vision, Prague, Czech Republic, May 11-14, 2004. Proceedings, Part IV 8*. Springer, 2004, pp. 25–36.

[54] Z. Zheng, N. Nie, Z. Ling, P. Xiong, J. Liu, H. Wang, and J. Li, "Dip: Deep inverse patchmatch for high-resolution optical flow," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 8925–8934.

[55] N. Dvornik, K. Shmelkov, J. Mairal, and C. Schmid, "Blitznet: A real-time deep network for scene understanding," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 4154–4162.

[56] X. Hu, Y. Lin, S. Wang, Z. Wu, and K. Lv, "Agent-centric relation graph for object visual navigation," *IEEE Transactions on Circuits and Systems for Video Technology*, 2023.

[57] T. Yu, X. Lin, S. Wang, W. Sheng, Q. Huang, and J. Yu, "A comprehensive survey of 3d dense captioning: Localizing and describing objects in 3d scenes," *IEEE Transactions on Circuits and Systems for Video Technology*, 2023.

[58] Z. Ying, X. Yuan, B. Song, Y. Song, F. Zhou, and W. Sheng, "Accurate and efficient 3d panoptic mapping using diverse information modalities and multidimensional data association," *IEEE Transactions on Circuits and Systems for Video Technology*, 2023.

[59] Y. Chang, C. Jung, P. Ke, H. Song, and J. Hwang, "Automatic contrast-limited adaptive histogram equalization with dual gamma correction," *Ieee Access*, vol. 6, pp. 11 782–11 792, 2018.

[60] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE conference on computer vision and pattern recognition*. IEEE, 2012, pp. 3354–3361.

[61] C. Godard, O. Mac Aodha, M. Firman, and G. J. Brostow, "Digging into self-supervised monocular depth estimation," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 3828–3838.

Hengbo Qi received the B.S. and M.S. degrees in aerospace science and technology from the Beijing Institute of Technology (BIT), Beijing, China, in 2018 and 2021, respectively. He is currently working toward the Ph.D. degree with the Intelligent Robotics Institute, School of Mechatronic Engineering, BIT, Beijing, China.

His current research interests include Visual-SLAM and Lidar-SLAM in humanoid robots.

Xuechao Chen received the B.S. and Ph.D. degrees in mechatronics engineering from the Beijing Institute of Technology (BIT), Beijing, China, in 2007 and 2013, respectively. He was a Visiting Student at the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA, in 2012, and a Visiting Scientist at the Sibley School of Mechanical and Aerospace Engineering, Cornell University, Ithaca, NY, USA, in 2018. He served as a Lecturer from 2013 to 2018, an Associate Professor from 2018 to 2021, and is currently a Professor at the School of Mechatronics Engineering, BIT. His research interests include bipedal locomotion and humanoid robotics.

Dr. Chen received a First Class Prize of the Ministry of Education Award for Technology Invention.

Zhangguo Yu received the B.S. degree in electronics engineering and the M.S. degree in control engineering from the Southwest University of Science and Technology (SWUST), Mianyang, China, in 1997 and 2005, respectively, and the Ph.D. degree in mechatronics engineering from the Beijing Institute of Technology (BIT), Beijing, China, in 2009.

He is currently a Professor at BIT and serves as the Director of the Intelligent Robotics Institute. His research interests include motion planning and control of biped robots.

Chao Li received the B.S. degree from the Hubei University of Automotive Technology (HUAT) in 2014 and the M.S. degree from the Beijing Institute of Technology (BIT) in 2020. He is currently working toward the Ph.D. degree in mechanical engineering with the Intelligent Robotics Institute, School of Mechatronic Engineering, BIT.

His research interests include environment perception for humanoid robots.

Yonglaing Shi obtained his Ph.D. in mechatronics engineering from the Beijing Institute of Technology, Beijing, China, in 2021. He is currently serving as an assistant researcher at Qiyuan Lab. His research focuses on Neural Radiance Fields (NeRF), simultaneous localization and mapping (SLAM), and multi-sensor fusion-based localization for robots.

Qingrui Zhao received the B.S. degree in mechatronics engineering, in 2022, from the Beijing Institute of Technology (BIT), Beijing, China. He is currently working toward the M.S. degree with the Intelligent Robotics Institute, School of Mechatronic Engineering, BIT.

His research interests include state estimation of biped robots and sensor fusion.

Qiang Huang (Fellow, IEEE) received the B.S. and M.S. degrees from the Harbin Institute of Technology, Harbin, China, in 1986 and 1989, respectively, both in electrical engineering and the Ph.D. degree in mechanical engineering from Waseda University, Tokyo, Japan, in 1996.

He was a Research Fellow with the National Institute of Advanced Industrial Science and Technology, Chiyoda-ku, Japan, between 1996 and 1999. He was a Research Fellow with the University of Tokyo, Bunkyo City, Japan, between 1999 and 2000. Since 2000, he has been a Professor with the Beijing Institute of Technology (BIT), Beijing, China. He is currently the Executive Director of Beijing Advanced Innovation Center for Intelligent Robots and systems, BIT. His research interests include biped locomotion and biorobotic systems.

Dr. Huang was the recipient of the IFToMM award of merit.