PAPER

DIG-SLAM: an accurate RGB-D SLAM based on instance segmentation and geometric clustering for dynamic indoor scenes

To cite this article: Rongguang Liang *et al* 2024 *Meas. Sci. Technol.* **35** 015401

IOP Publishing Measurement Science and Technology
Meas. Sci. Technol. **35** (2024) 015401 (13pp) https://doi.org/10.1088/1361-6501/acfb2d

DIG-SLAM: an accurate RGB-D SLAM based on instance segmentation and geometric clustering for dynamic indoor scenes

Rongguang Liang, Jie Yuan*, Benfa Kuang, Qiang Liu and Zhenyu Guo

School of Electrical Engineering, Xinjiang University, Urumqi, People's Republic of China

E-mail: yuanjie@xju.edu.cn

Received 12 June 2023, revised 12 September 2023
Accepted for publication 19 September 2023
Published 27 September 2023

Abstract

Simultaneous localization and mapping (SLAM) has emerged as a critical technology enabling robots to navigate in unknown environments, drawing extensive attention within the robotics research community. However, traditional visual SLAM ignores the presence of dynamic objects in indoor scenes, and dynamic point features of dynamic objects can lead to incorrect data correlation, making the traditional visual SLAM is difficult to accurately estimate the camera's pose when the objects in the scenes are moving. Using only point features cannot fully extract geometric information in dynamic indoor scenes, reducing the system's robustness. To solve this problem, we develop a RGB-D SLAM system called DIG-SLAM. Firstly, the objects' contour regions are extracted using the YOLOv7 instance segmentation method, serving as a prerequisite for determining dynamic objects and constructing a semantic information map. Meanwhile, the line features are extracted using the line segment detector algorithm, and the redundant line features are optimized via K-means clustering. Secondly, moving consistency checks combined with instance partitioning determine dynamic regions, and the point and line features of the dynamic regions are removed. Finally, the combination of static line features and point features optimizes the camera pose. Meanwhile, a static semantic octree map is created to provide richer and higher-level scene understanding and perception capabilities for robots or autonomous systems. The experimental results on the Technische Universität München dataset show that the average absolute trajectory error of the developed DIG-SLAM is reduced by 28.68% compared with the dynamic semantic SLAM. Compared with other dynamic SLAM methods, the proposed system shows better camera pose estimation accuracy and system's robustness in dynamic indoor environments and better map building in real indoor scenes.

Keywords: visual SLAM, instance segmentation, line feature extraction, K-means, octree map

* Author to whom any correspondence should be addressed.

1. Introduction

Simultaneous localization and mapping (SLAM) is a prerequisite technology for most mobile robotic applications [1, 2], helping them to position themselves in an unknown environment without any prior information while simultaneously creating maps of the surrounding environment [3]. Vision SLAM is relatively low-cost and can provide rich scene information to estimate the camera's poses and feature states. Therefore, it is widely used in autonomous driving scenarios, industrial automation, augmented reality, and virtual reality [4]. However, the data association problem becomes complex with motion and occlusion of objects in dynamic environments.

After decades of development, some mature visual SLAM systems have emerged, such as ORB-SLAM2 [5], ORB-SLAM3 [6], large-scale direct monocular SLAM (LSD-SLAM) [7], and real-time appearance-based mapping (RTAB-Map) [8]. However, these systems mainly assume a static environment, and the presence of dynamic objects in real indoor scenes leads to mismatching and incorrect data associations [9], reducing the camera's pose estimation accuracy. How to reduce the negative impact of dynamic objects on visual SLAM and improve the camera's pose estimation accuracy has become important issue in studying dynamic visual SLAM [10].

Common methods for removing dynamic objects uses the random sample consensus (RANSAC) algorithm [11], an iterative method for estimating mathematical model parameters, which can handle certain outliers in static or slightly dynamic environments. However, when multiple dynamic objects appear simultaneously, the RANSAC algorithm may not perform well [12]. In recent years, with the development of deep learning, some SLAM researchers have proposed combining deep learning to incorporate prior-semantic information and eliminating the influence of dynamic objects to improve the performance of SLAM. Therefore, many SLAM systems integrating deep learning have been investigated, such as Dyna-SLAM [13] and dynamic semantic SLAM (DS-SLAM) [14]. Although SLAM systems combining deep learning networks have performed well in eliminating the negative impact of dynamic objects, there are still some errors in detecting dynamic objects' edges, resulting in the loss of certain static features. Moreover, point features are susceptible to occlusion and scale changes factors, leading to instability in feature matching. Line features can also be used, and line features can provide rich geometric information to enhance the accuracy of camera pose estimation and system robustness.

To solve the problems of inaccurate camera pose estimation due to dynamic objects and reducing the system's robustness by solely relying on point features, we develop a dynamic instance segmentation and geometric clustering SLAM (DIG-SLAM) system. The developed DIG-SLAM is designed on the basis of the DS-SLAM [14] system, which uses the SegNet semantic segmentation. The developed system replaces SegNet semantic segmentation with pixel-level YOLOv7 instance segmentation for better accurately segmenting object outlines. Moreover, LSD algorithm [15] is used in the system to extract line features, enhancing the robustness of the system. The K-means clustering algorithm is used to reduce the redundancy of the line features retrieved by the LSD algorithm. Then, potential dynamic points are checked for motion consistency, and potential feature points are combined with the segmented mask to determine the dynamic region, eliminating dynamic features. Static point and line features are used to construct an error optimization model to solve for the camera's position at each moment. Finally, constructing a static semantic 3D octree map provides richer, higher-level scene understanding and perception capabilities for robots or autonomous systems. The experimental results show that the system has high camera's pose estimation accuracy and robustness in dynamic indoor environments.

The main contributions of this paper are as follows:

- A visual SLAM method is proposed based on the YOLOv7 instance segmentation model. The generated mask can finely cover the contour region of the object, providing a prerequisite for eliminating dynamic objects.
- A novel method for pose optimization is designed. The K-Means clustering is used to optimize the line features extracted by the LSD algorithm, and the point features are combined to construct point and line error models for pose optimization, improving the estimation accuracy of the camera pose and the robustness of the system.
- An accurate DIG-SLAM system is developed by verifying its accuracy on the Technische Universität München (TUM) dataset and building a static semantic 3D octree map.

The rest of this paper is structured as follows: A review of relevant work is provided in section 2. The developed DIG-SLAM is described in section 3. Section 4 presents the experimental findings from open TUM datasets and the real scene, along with the analysis. Section 5 presents conclusions.

2. Related works

In recent years, many researchers have focused on enhancing the camera pose estimation accuracy and system robustness of visual SLAM in dynamic indoor environments. The major challenge is to effectively extract dynamic features from the input frames and discard them as outliers, since the dynamic features should not be utilized for posture estimation and mapping. The approaches currently in use can be broadly split into two groups based on the extracted dynamic outliers: those based on conventional methods, and the ones based on deep learning.

2.1. Conventional methods for extracting dynamic features

The traditional methods recognizing dynamic regions are mainly based on optical flow (OF) and geometric methods. Literature [16] the LK (Lucas and Kanade) OF method tracks

the frames' feature points by computing and matching key frame descriptors to create data connections between key frames. Literature [17] proposes an algorithm for geometrically breaking up an input image frame into several image chunks. This algorithm then iterates to assess whether an image block is a dynamic zone by minimizing the photometric errors. Literature [18] uses the residual calculate OF between the predicted image and the binocular camera observation image, and observe moving objects by detecting points in the residual field. Literature [19] develops a method to detect moving objects by calculating the transformation matrix between two adjacent RGB-D images, compensating for the motion of the previous frame. Literature [20] uses the fundamental matrix to detect the inconsistency of feature points and then the depth images are clustered. When the outliers in a clustering area exceed the threshold, this area is identified as a part of the moving object. Literature [21] presents a sparse motion reduction mode that detects dynamic regions based on the similarity and differences between consecutive frames.

The above methods determine dynamic features based on OF methods or geometric errors without additional computational overhead, and perform well in most environments. However, OF methods are prone to false matches and have difficulty extracting high-quality point features in scenes with changing illumination. The geometry method typically requires a predefined threshold to determine whether the features are dynamic or static, which easily causes excessive recognition or inadequate recognition. Therefore, the traditional methods decrease in the accuracy of camera pose estimation and system robustness.

2.2. Combining deep learning methods for extracting dynamic features

In addition to conventional extracting dynamic features approaches, the deep learning-based ones have emerged as promising alternatives for meeting the challenges of dynamic environments in visual SLAM systems [22], enhancing the systems' sensing the environment ability. Literature [14] develops a semantic visual SLAM system for dynamic environments (DS-SLAM). This system combines the semantic segmentation network SegNet providing semantic priori information with motion feature point detection to filter out dynamic objects in each frame, thus improving the accuracy of pose estimation while building a semantic octree map to meet a wider range of needs. Literature [23] designs a novel multimodal semantic SLAM system, which uses instance segmentation networks to provide semantic knowledge of the surrounding environment, directly removes ORB features from predefined dynamic objects, and combines multi-view geometric constraints with a K-means clustering algorithm to remove undefined but moving pixels. Literature [24] uses the lightweight YOLOv3 to change the backbone network of the detection model from darknet-53 to darknet-19, speeding up the detection efficiency by providing necessary semantic information in a dynamic environment. Additionally, the point cloud map format used in YOLO-SLAM may not be suitable for more intricate and advanced tasks. Literature [13] proposed the DynaSLAM using multi-view geometry and deep learning Mask R-CNN to detect dynamic objects, and repair the frame's background occluded by the dynamic objects, and build static maps. However, DynaSLAM sacrificing speed while gaining highly accurate. Literature [25] proposed DM-SLAM combining OF with semantic masks to achieve higher tracking accuracy in dynamic scenes. However, there is poor accuracy when objects suddenly transition from a stationary state to motion. Literature [26] proposes a missed detection compensation algorithm based on the motion invariance between adjacent frames. This algorithm improves the recall rate of object detection. By combining semantic prior knowledge of the SSD convolutional neural network, it determines dynamic objects. However, the application of deep learning is still limited and only applied to the frontend object detection module.

Although SLAM systems that combine deep learning networks have performed well in eliminating the impact of dynamic objects, there are still some errors in the detection of dynamic object edges, resulting in the loss of certain static features, leading to dynamic object surface feature points cannot be completely eliminate, the SLAM systems decrease the accuracy of camera pose estimation, and also lead to poor map construction. In this paper, we develop the DIG-SLAM to overcome the challenges in dynamic indoor scenes. For reducing the segmented dynamic object edge errors, we use YOLOv7 + Detectron2 for pixel-level instance segmentation. Furthermore, the proposed DIG-SLAM fully utilizes the line features in dynamic indoor scenes. As a result, our system improves the accuracy of camera pose estimation and system robustness in dynamic indoor environments, and better map construction.

3. System introduction

The DIG-SLAM system proposed in this paper is a novel approach to SLAM that utilizes a YOLOv7 pixel-level instance segmentation network to replace the original poor segmentation effect SegNet [14] semantic segmentation network. Along with adding new line features and performing K-means clustering to obtain high-quality geometric features. This section aims to provide a detailed explanation of the DIG-SLAM system, including its overall architecture and how it works.

3.1. Overview of DIG-SLAM

As one of the classic dynamic SLAM systems combined with deep learning, DS-SLAM uses ORB-SLAM2 as a framework to combine semantic segmentation with motion consistency checking to filter out dynamic objects in each frame, thus improving the accuracy of pose estimation, while building a semantic octree map. The DIG-SLAM system proposed in this paper is based on DS-SLAM using instance segmentation of the YOLOv7 network to replace the original SegNet semantic

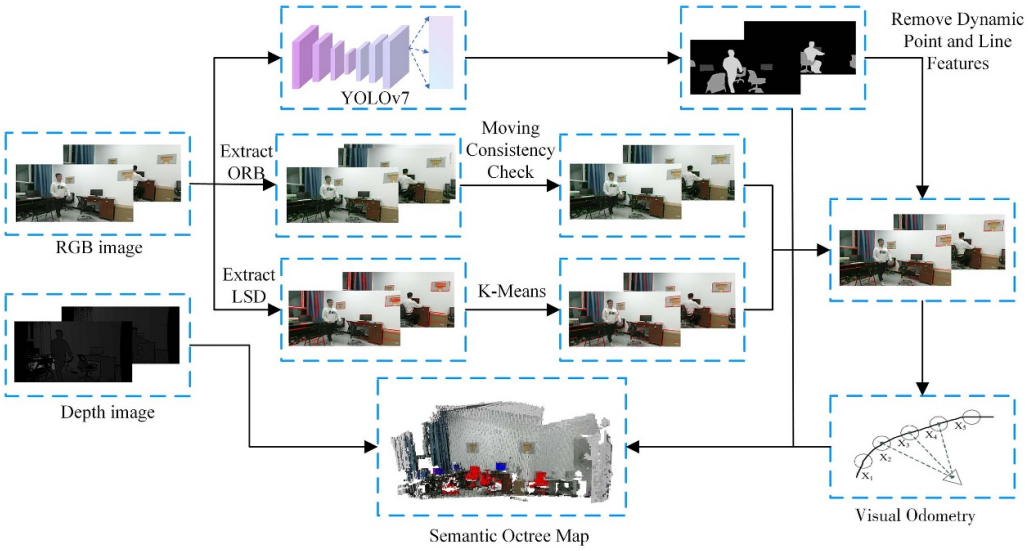


Figure 1. An overview of the developed DIG-SLAM system.

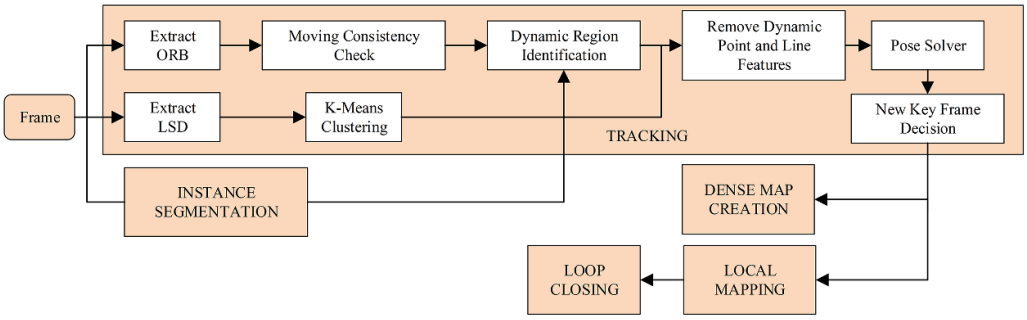


Figure 2. The framework of the developed system consisting of five concurrent threads: the tracking thread, the instance segmentation thread, the local mapping thread, the closed-loop detection thread, and the dense map creation thread.

segmentation network of the DS-SLAM, add new line features and optimize them with K-means clustering.

The overview of this system is shown in figure 1. Firstly, when the DIG-SLAM system processes each RGB image frame, the objects' contour regions are extracted by YOLOv7 instance segmentation. Then, point features for each frame are extracted using the ORB extractor. Next, potential dynamic point features are identified using the moving consistency check. Furthermore, each frame's line features are extracted using the LSD extractor. Then, the line features are optimized using K-means clustering to remove redundant line features. The optimized line features do not involved in dynamic object detection. Secondly, the contour regions and point features generated by instance segmentation are merged into the system with the optimized line features. Finally, a dynamic region is identified by whether the number of potentially dynamic point features in the region exceeds a set threshold, and removing dynamic point and line features from this dynamic region. Meanwhile, static point and line features are used to construct an error optimization model to solve the camera's pose at each moment. Finally, the semantic information provided by instance segmentation and pose estimation are utilized to construct a static semantic octree map to provide high-level scene understanding for robots or autonomous navigation systems. Figure 2 depicts the overall architecture of the developed system discussed in this study, which all uppercase letters represent thread names.

3.2. *YOLOv7 instance segmentation*

With the need for pixel-level dynamic object detection and semantic information mapping, this paper uses the YOLOv7 instance segmentation model to pre-train on the MS COCO dataset, which can segment 80 classes of objects commonly found in life [27]. YOLOv7 is compatible with Detectron2 [28] and adheres to its API and visualization tools, making it easier to run fast and accurate instance segmentation. The YOLOv7 network structure is shown in figure 3, where bounding boxes represent the location of potential dynamic objects with semantic labels in the image.

In the proposed DIG-SLAM system, YOLOv7 is used to extract instance segmentation regions from the input image, serving as a prerequisite for determining dynamic objects and semantic information mapping. In the beginning, the input image is pre-processed to be scaled to 640 $\times$ 640, and the pixel values are normalized. The backbone network is made

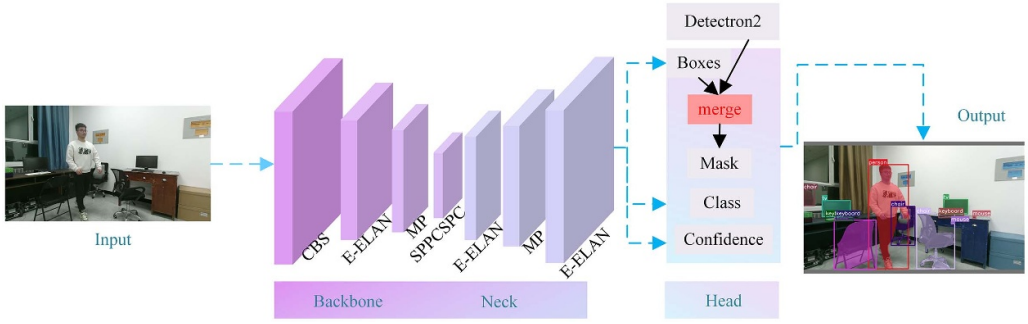


Figure 3. The network structure of YOLOv7. The YOLOv7 consists of three main parts: the backbone network layer, the feature fusion layer 'Neck', and the detection layer 'Head'.

up of Convolution BatchNorm SiLU layers, E-ELAN efficient aggregation network, and Max Pooling (MP) layers, which extract features by halving the layers and doubling the channels. Next, E-ELAN manages the connected paths owing to the E-ELAN's efficient learning and converge characteristics. The feature fusion layer, which comprises an E-ELAN-based efficient aggregation network, MP network, and space pyramidal pooling SPPCSPC network, efficiently processes the features retrieved from the backbone network. In the instance segmentation task, YOLOv7 uses the output of the target detection layer (Boxes) in the Detectron2 stage as the merged feature map. Based on the target object's position and size information provided by the output of the detection layer (Boxes), the precise contour information of the target object can be obtained to accomplish the instance segmentation task.

Acquiring semantic information about the surrounding environment serves two purposes. On the one hand, it provides priori information for the removal of potential dynamic features. In this study, a person label is defined as a dynamic object in an indoor environment. The segmented images are binarized according to the person label. Dynamic point and line features located on the predefined objects are then removed, and retained improve the accuracy of pose evaluation in the tracking thread and system robustness in dynamic environments. On the other hand, DIG-SLAM merges the semantic information of segmented static objects into the corresponding 3D points in the map construction thread, creating a static semantic octree map.

YOLOv7 outperforms SegNet and Mask_R-CNN in terms of segmentation effects and detail processing, and the mask edge obtained by YOLOv7 is clearer and more complete [29, 30]. As shown in figure 4.

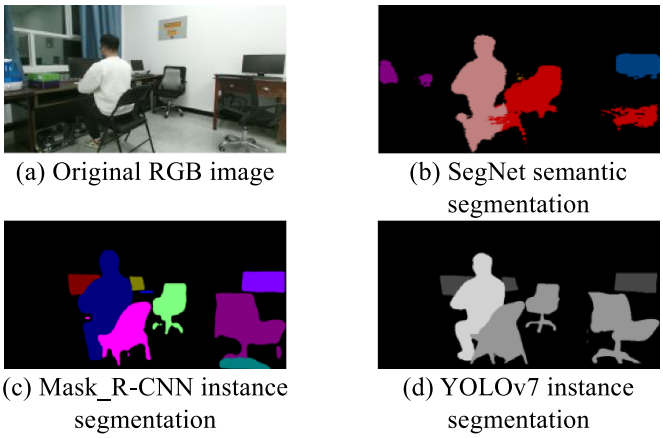


Figure 4. Comparison among SegNet semantic segmentation used by DS-SLAM, Mask_R-CNN-based instance segmentation used by DynaSLAM, and YOLOv7-based instance segmentation DIG-SLAM.

3.3. Line feature optimization based on K-means clustering

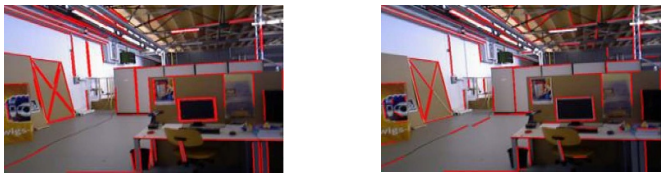
The LSD algorithm [15] uses gradient information and ranks lines to detect the pixel points with large gradient changes in the image, subsequently extracting line features and providing their starting and ending coordinates. However, at least one straight line is split into two or multiple ones by the LSD algorithm owing to its lack of merging and elimination mechanisms. The line features mismatching will grow with the increase of redundant line features, raising camera pose evaluation errors. This paper utilizes the K-means clustering algorithm to cluster the line features extracted by the LSD algorithm and screen and retain line features with higher response values in each cluster. To reduce redundant line features, lower the complexity of feature matching.

The K-means clustering algorithm is an unsupervised learning algorithm. By an iterative partitioning scheme of K clusters, the corresponding loss function is minimized. The loss function can be defined as the sum of the squares of the errors of the individual samples from the centroids of the clusters to which they belong, which is presented in formula 1

$$L(c,\mu) = \sum_{i=1}^{M} \|x_i - \mu_{c_i}\|^2, \tag{1}$$

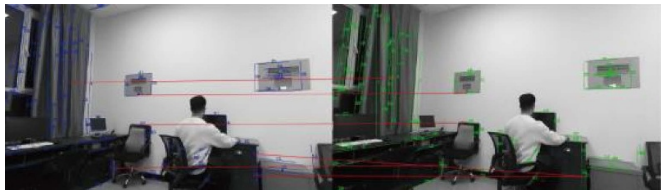
where X_i represents the ith sample, C_i is the cluster to which X_i belongs, μ_{c_i} presents the center of the cluster, and M is the total number of the samples.

The LSD algorithm extracts the line features from the image frames, divides the extracted line features into K clusters (the number of clusters is set to 100), calculates the distance of each line feature from the center of mass using the Euclidean distance, and assigns each sample to the cluster

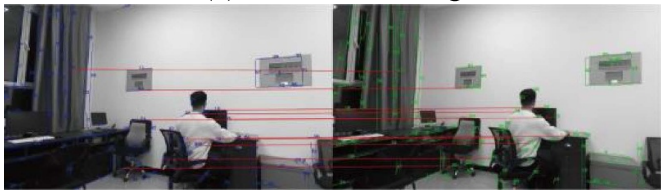


(a) (b)

Figure 5. (a) Line features before being clustered (b) Line features after being clustered.



(a)Before clustering



(b)After clustering

Figure 6. Clustering matching effect.

where the nearest center of mass is located. The center of mass is updated based on the average of all the line features in their respective clusters until it converges to a steady state. After the K-means algorithm converges to a steady state, the line segments of each cluster are examined, and the response value of each line feature is calculated based on the ratio of the line length to the maximum value between the width and height of the image. By selecting the line segments with the highest response values to replace the line features in the original cluster, the complexity and memory requirements for feature matching in subsequent processing steps are reduced.

The K-means algorithm does not easily to converge to the optimal solution in the case of random initialization due to the inherent limitation of K-means. Hence, we perform several experiments and take the average. As a result, the overall performance of the DIG-SLAM system is improved. It can be seen from figure 5 that there are significantly fewer line features in figure 5(b) compared with figure 5(a).

In figure 6, we can see that there are six successful matches with two incorrect matches before clustering. After clustering, there are 11 successful matches with 0 incorrect matches. Therefore, the feature matching performance is greatly improved after clustering.

3.4. Remove dynamic point and line features

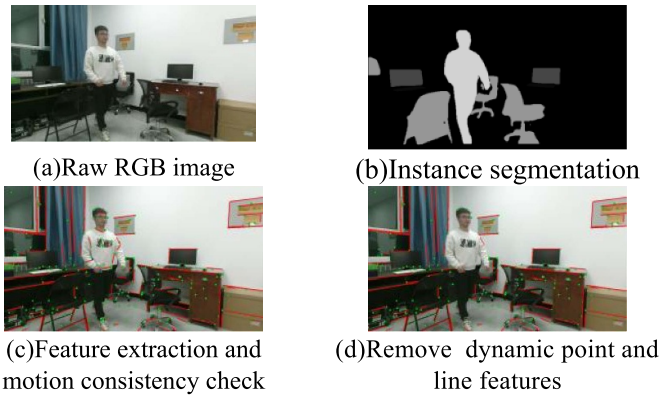
The extraction quality of dynamic point and line features determines the overall stability and performance of a SLAM system. In this study, YOLOv7 instance segmentation is used in the instance segmentation thread to extract the objects' contour regions. Meanwhile, the OF method is applied to associate with the point features generated from the previous and current one, and to remove point pairs that are prone to be mismatched. Then, the transformation matrix that satisfies most of the remaining matching point pairs is calculated. Finally, the matched point pairs are traversed, and the distance from the corresponding point feature to the epipolar line is gauged. If the distance exceeds a threshold, the point is considered a potential dynamic feature point. If there are multiple potential dynamic point features in a region generated by the instance segmentation, this region is labeled the dynamic object and the region is considered a dynamic region. The dynamic points of the region are then eliminated.

Algorithm 1. Dynamic region recognition.

Input: Set of potential dynamic object regions M; Set of feature points P_1 from the previous frame F_1; Current frame F_2;
Output: Dynamic labels for each region
Procedure1: Calculate the dynamic points using the optical flow method and the polar line distance
$P_2 = CalcOpticalFlowPyrLK(F_1,F_2,P_1)$
Remove outliers in P_2
$S = \{\}$
for each (p_1, p_2) in P_1, P_2 **do**
  $F_M = ComputeFundamentalMatrix(P_1, P_2)$
  $I_1 = FindEpipolarLine(p1, F_M)$ //Calculate the polar line corresponding to p1
  $D = CalcDistanceFromEpipolarLine(p_2, I_1)$
  if $D > \varepsilon$ **then** //Determine whether it is a dynamic point
    S.add(p_2)
Procedure2: Determine a dynamic object label region in the dynamic point set S
for each region m in M **do**
  $n = 0$
  for each dynamic point s in S **do**
    if s in region m **then**
      $n = n + 1$
    end if
    if $n > \xi$ **then**
      region $m.label$ = 'dynamic region'
      break
    end if
  end for
  if region $m.label$!= "dynamic region" **then**
    region $m.label$ = "static region"
  end if
end for

The system for identifying dynamic regions are identified as shown in algorithm 1. P_1 and P_2 in algorithm 1 are the sets of feature points from the previous and current frames, respectively; F_1 and F_2 are the previous and current frame images, respectively; F_M is the fundamental matrix, M is the set of potential dynamic object regions generated by the instance segmentation, ε is the threshold for the polar line distance, and ξ is the threshold for the number of dynamic points.

After the LSD algorithm extracts line features from the image, LBD descriptors [31] are computed for each line



(a)Raw RGB image (b)Instance segmentation

(c)Feature extraction and motion consistency check (d)Remove dynamic point and line features

Figure 7. The process of removing dynamic point and line features. A region is deemed to be dynamic when it contains several potential dynamic feature points (the blue points) in the labeled dynamic region (the person). The features in this dynamic area are then eliminated.

segment. Then, line segment matching between the previous frame and the current one is performed by pairwise geometric consistency constraints, and using the RANSAC Algorithm [11] to remove erroneous data associations. As far as line features in dynamic regions are concerned, the three-point method presented in [32] is used to transform line features into point features, removing the line features in dynamic regions. The pixel values of the start, end, and middle points of a line feature are obtained from the grayscale map. If any of the pixels at the above three points of the line feature has a labeled dynamic pixel value, the line feature is considered to be dynamic and subsequently removed. The process of removing dynamic point and line features is shown in figure 7.

3.5. Optimization model based on static features

After removing the dynamic features, the remaining point and line features are considered static features. These static features will be used to construct a geometrical optimization model for solving the camera's pose. In this paper, we use a reprojected 3D–2D projection method to calculate the reprojection errors. In this method, the 3D point and line are projected onto the camera's image plane and compared with the corresponding 2D feature point and line to calculate the reprojection error. By calculating the reprojection error of point and line features between frames, the pose of each frame is obtained.

Set the jth point in space at the kth moment in the world coordinate system with coordinates $P^k_{w,j}$ and camera pose T^k_c. The point is projected into the pixel coordinate system according to from the camera coordinate transformation relationship, which is presented in formula 2

$$P^{k\prime}_{uv,j} = f\left(P^k_{w,j}, T^k_c, \boldsymbol{K}\right), \tag{2}$$

where $T^k_c = \{R_k, t_k\}$ denotes the camera pose at the kth moment, and $\boldsymbol{K}$ is the internal calibration matrix of the camera.$R_k \in \mathbb{R}^{3\times3}$ denotes the rotation matrix, and $\mathrm{t}_k \in \mathbb{R}^3$ denotes the translation vector. Finally, the reprojection error between the matching point $P^k_{uv,j}$ and the projection point $P^{k'}_{uv,j}$ is calculated, which is presented in formula 3

$$e_{\mathrm{k},j} = P^k_{uv,j} - P^{k\prime}_{uv,j}. \tag{3}$$

The calculation of the reprojection error of the line feature is more complex than that of the point feature. After the line feature matching is completed, the corresponding 3D coordinates in the line feature of the previous frame are derived based on the corresponding depth map of the previous frame image, and the two endpoints of the 3D line feature are projected to the current frame to construct the reprojection error model. The reprojection error of the lth line in space at the kth moment can be expressed, which is presented in formula 4

$$e_{k,l} = \begin{bmatrix} d(X_s, l') \\ d(X_e, l') \end{bmatrix} = \begin{bmatrix} \frac{X_s^T l'}{\|l'\|} \\ \frac{X_e^T l'}{\|l'\|} \end{bmatrix}, \tag{4}$$

where l' denotes the space line l projected on the 2D plane.X_e and X_s represent the two endpoints on the matching line, and X'_e and X'_s denotes their projection points on the line l'. $\| l' \|$ represents the L2 norm of the vector l'.

Finally, the observation errors are set to follow the Gaussian distribution. A cost optimization function integrated with each error term is thus constructed, which is presented in formula 5

$$c = \sum_{k,j} \left(\rho_{\mathrm{p}} e^T_{k,j} \Omega^{-1}_{k,j} e_{k,j}\right) + \sum_{k,l} \left(\rho_{\mathrm{l}} e^T_{k,l} \Omega^{-1}_{k,l} e_{k,l}\right), \tag{5}$$

where c is the cost optimization function, ρ_{p} and ρ_{l} are the regularization constants associated with point and line features, respectively. $\Omega^{-1}_{k,j}$ and $\Omega^{-1}_{k,l}$ are the inverses of the observed covariance matrices for point and line features, respectively.

4. Experiments

In this paper, the performance of the developed DIG-SLAM is validated using the public dataset TUM RGB-D. Firstly, the DIG-SLAM compares with against the original DS-SLAM to verify the developed approach's performance improvement. Secondly, the DIG-SLAM is analyzed with other dynamic SLAM. Next, the DIG-SLAM is performed to the ablation experiments and time evaluation to validate the effectiveness of the developed system. Finally, experiments are conducted in a real scenario to evaluate the DIG-SLAM's practical application performance, and the static semantic octree map of a real scenario is constructed. All experiments are run on a laptop computer equipped with an Intel Core i7-8750H (2.20 GHz) CPU, 8 GB RAM, NVidia GTX 1050Ti GPU, Ubuntu 18.04 system.

4.1. Performance evaluation on TUM RGB-D dataset

This study uses the Freiburg3 series from the TUM RGB-D dataset. Freiburg3 consists of a high-dynamic scene sequence

marked 'walking', in which two people walk around a table, and a low-dynamic scene sequence marked 'sitting', in which two people sit in chairs with slight head or part of the limb movements. For simplicity, Freiburg3, walking, and sitting sequences are described as frb3, w, and s, respectively. Camera motion directions include *xyz* (camera rotates along *x*–*y*–*z* axes), static (camera remains still, no motion), *rpy* (camera roll–pitch–yaw along a main axis) and half (camera rotates along the *x*–*y*–*z* axes on a hemispherical surface).

Absolute trajectory error (ATE) is a metric for computing the estimated and actual poses, and it indicates the system's accuracy and the trajectory's global consistency. Relative pose error (RPE) is used to evaluate the trajectory's local accuracy with a fixed interval. Absolute pose error (APE) is used to measure the absolute error between the estimated and true poses.

The following experiments qualitatively compare the DIG-SLAM system with DS-SLAM and ORB-SLAM2 according to the metrics of ATE, RPE and APE. The dynamic sequences frb3_w_xyz and frb3_w_rpy are selected from the open datasets TUM. The Evaluation of Odometry and SLAM (EVO) [33] builder tool is used to compare the trajectories generated by the aforementioned three systems with the real trajectories. As shown in figure 9, the ORB-SLAM2 system is displayed in red, the DS-SLAM system is shown in green, the DIG-SLAM system is presented in blue, and the actual trajectory is show by dashed lines in black.

As shown in figures 9(a) and (b), the estimated trajectory of ORB-SLAM2 has a significant trajectory drift compared with the actual trajectory because of the influence of dynamic objects. By using DS-SLAM combined with semantic segmentation, the effect of eliminating the point features in dynamic regions is improved, but the estimated trajectory still deviates from the real trajectory due to unsatisfactory segmentation. Figure 9(c) shows that the system used in this study has a lower RPE value than DS-SLAM for the same amount of time local drift. Figure 10 shows that the DIG-SLAM estimated camera trajectories are more consistent with the actual ones. Since the instance segmentation based on YOLOv7 with tighter bounds is implemented, and static point and line features are fully used, this developed DIG-SLAM achieves higher robustness, and the trajectory's global consistency in a dynamic indoor environment.

Figures 11(a) and (b) shows the comparison of the trajectory errors among ORB-SLAM2, DS-SLAM, and the developed DIG-SLAM on the dynamic sequence frb3_w_rpy. The SegNet semantic segmentation is employed in the DS-SLAM, and the segmentation is inaccurate when the camera is rotated, resulting in a severe trajectory drift. As shown in figure 8, the DIG-SLAM developed in this paper using YOLOv7 achieves a better segmentation effect when the camera is rotated. Furthermore, point and line features are used to optimize camera pose estimation, and thus obtains a relatively small trajectory error. As shown in figure 11(c), the estimated maximum trajectory drifts by the ORB-SLAM2, DS-SLAM, and the developed system in this paper are 0.75 m, 0.68 m, and 0.127 m, respectively. As shown in figure 12, the camera estimation accuracy and trajectory global consistency of our system are reflected, and its performance of absolute camera pose estimation is better than ORB-SLAM2 and DS-SLAM systems.

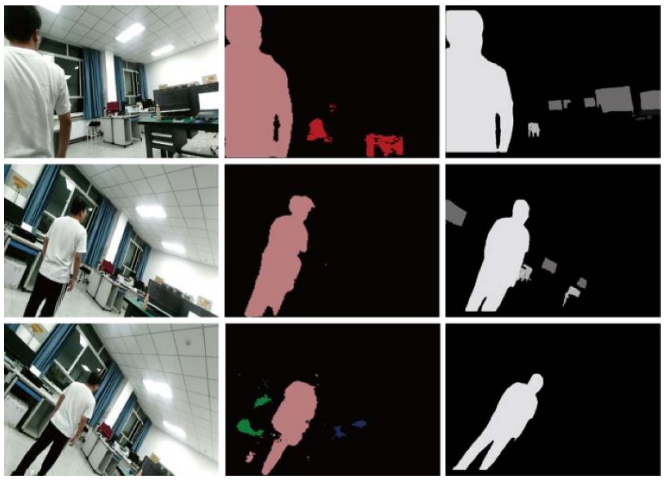
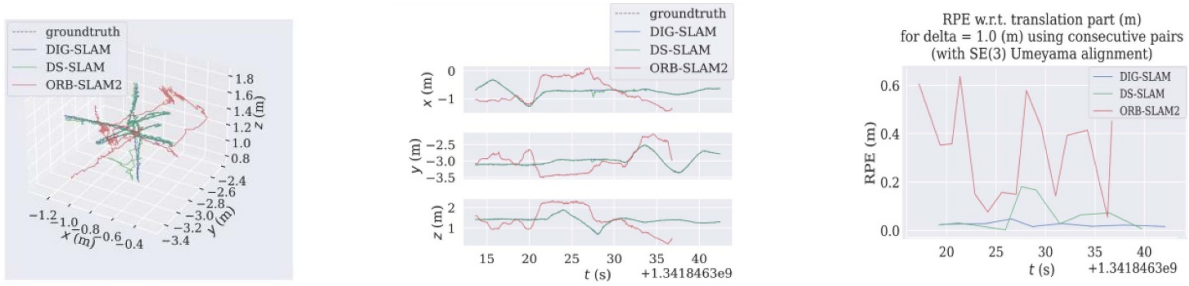


Figure 8. (a) Raw RGB image; (b) DS-SLAM segmentation; (c) DIG-SLAM segmentation.

To further validate the performance of our developed system in dynamic indoor scenarios, quantitative analysis experiments are conducted using the frb3 sequences dataset. The evaluation metrics [34] of the root mean squared error (RMSE), mean error (Mean), median error (Median), and standard deviation (STD) of the absolute trajectories are used. Among them, RMSE and STD better reflect the accuracy and system robustness. The quantitative comparison results are shown in tables 1–3, in which the values marked in bold denote the optimal results of camera pose estimation.

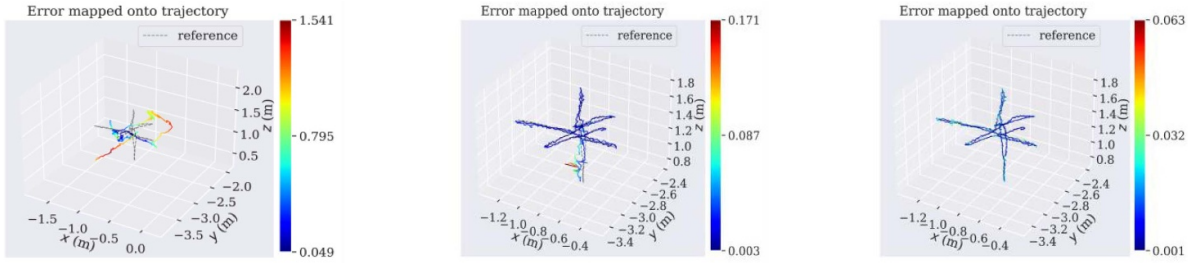
Table 1 shows that the performance of the developed DIG-SLAM is superior to DS-SLAM on dynamic dataset sequences. The improvement is obvious on the highly dynamic dataset sequences frb3_w_xyz and frb3_w_rpy, and the decrement of RMSE and S.D. reaches 90.17% and 86.51% on the frb3_w_rpy dataset, respectively. Due to the poor performance of the semantic segmentation based on DS-SLAM, some dynamic points are not effectively eliminated, affecting the accuracy of camera pose estimation. The YOLOv7 instance segmentation used in this study has stronger segmentation capabilities and solves the problem of insufficient segmentation of DS-SLAM. The results show that the developed DIG-SLAM in this paper has higher accuracy in the estimated trajectories and system robustness in high dynamic scenes. However, the advantage of the developed system in this paper is not obvious on low dynamic sequences, such as frb3_s_rpy sequences. The reason is that, the dynamic object motion is small in low dynamic scenes and the elimination of dynamic features has little impact on camera pose estimation accuracy of the system.

The developed DIG-SLAM is compared with the dynamic SLAM systems of DynaSLAM, CFP-SLAM, Blitz-SLAM [35], YOLO-SLAM, optical-flow dynamic (OPF-SLAM) [36], point and line features for dynamic scenes (PLD-SLAM) [37], and ORB-SLAM2. The quantitative experimental results



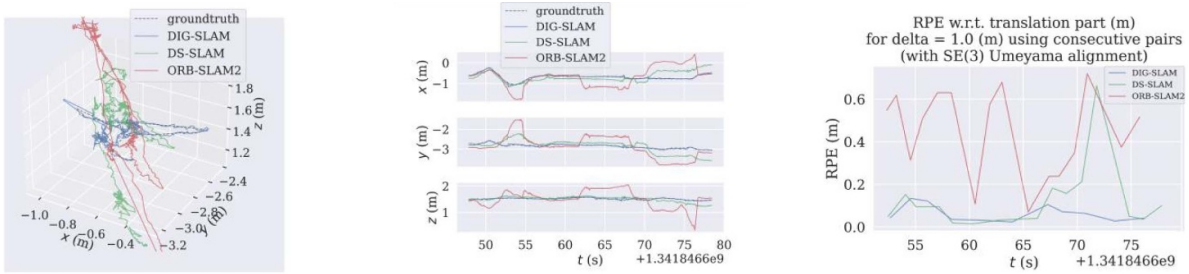
(a) trajectories (b) xyz_view (c) RPE_view

Figure 9. Comparison of absolute trajectory error (ATE) and relative pose error (RPE) among ORB-SLAM2, DS-SLAM and DIG-SLAM on the frb3_w_xyz sequence.



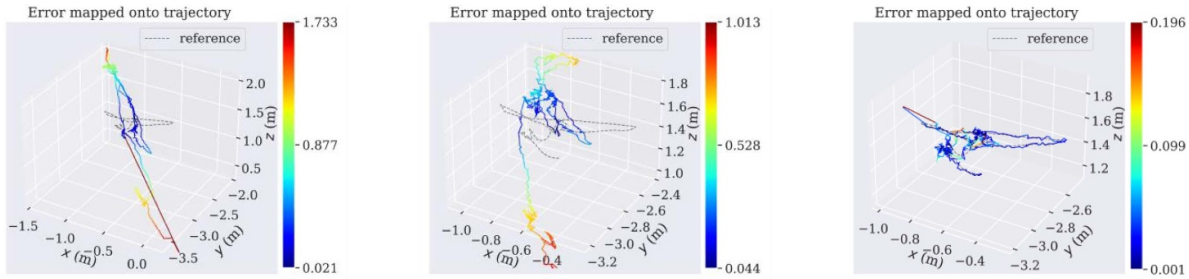
(a) ORB-SLAM (b) DS-SLAM (c) DIG-SLAM

Figure 10. Comparison of absolute pose error (APE) among ORB-SLAM2, DS-SLAM, and DIG-SLAM (corresponding from left to right) on the frb3_w_xyz sequence. The color bar on the right indicates that the larger the error is, the redder the color becomes; the smaller the error is, the bluer the color becomes.



(a) trajectories (b) xyz_view (c) RPE_view

Figure 11. Comparison of absolute trajectory error (ATE) and relative pose error (RPE) among ORB-SLAM2, DS-SLAM and DIG-SLAM on the frb3_w_rpy sequence.



(a) ORB-SLAM (b) DS-SLAM (c) DIG-SLAM

Figure 12. Comparison of absolute pose error (APE) among ORB-SLAM2, DS-SLAM and DIG-SLAM on the frb3_w_rpy sequence. The color bar on the right indicates that the larger the error is, the redder the color becomes; the smaller the error is, the bluer the color becomes.

Table 1. Comparison of camera pose estimation between DS-SLAM and DIG-SLAM according to the absolute trajectory error (ATE [m]).

	DS-SLAM				DIG-SLAM (ours)				Improvements of DIG-SLAM			
Sequences	RMSE	Mean	Median	STD	RMSE	Mean	Median	STD	RMSE	Mean	Median	STD
frb3_w_xyz	0.0288	0.0206	0.0166	0.0201	0.0140	0.0120	0.0109	0.0072	**51.16%**	**41.47%**	**34.27%**	**64.04%**
frb3_w_static	0.0076	0.0069	0.0063	0.0036	0.0067	0.0058	0.0052	0.0033	12.50%	15.63%	18.38%	8.53%
frb3_w_rpy	0.4752	0.4051	0.2856	0.2484	0.0466	0.0325	0.0225	0.0335	**90.17%**	**91.97%**	**92.10%**	**86.51%**
frb3_w_half	0.0283	0.0242	0.0209	0.0147	0.0237	0.0204	0.0173	0.0120	16.32%	15.54%	17.33%	18.47%
frb3_s_xyz	0.0102	0.0088	0.0081	0.0051	0.0090	0.0077	0.0069	0.0045	11.55%	13.14%	14.90%	12.52%
frb3_s_static	0.0065	0.0055	0.0048	0.0030	0.0056	0.0047	0.0041	0.0030	13.37%	14.10%	14.85%	11.40%
frb3_s_rpy	0.0213	0.0155	0.0118	0.0146	0.0193	0.0149	0.0109	0.0122	9.58%	3.82%	7.05%	16.58%
frb3_s_half	0.0148	0.0138	0.0120	0.0069	0.0134	0.0118	0.0108	0.0064	8.83%	14.42%	10.03%	7.50%

Table 2. Comparisons of absolute pose error of among different systems (ATE [m]).

Sequences	Dyna SLAM [13]	CFP-SLAM [9]	Blitz-SLAM [35]	YOLO-SLAM [24]	OPF-SLAM [36]	PLD-SLAM [37]	ORB-SLAM2 [5]	DIG-SLAM
frb3_w_xyz	0.0158	0.0141	0.0153	0.0194	0.3060	0.0144	0.5359	**0.0140**
frb3_w_static	0.0080	0.0066	0.0102	0.0094	0.0411	**0.0065**	0.3540	0.0067
frb3_w_rpy	0.0402	0.0368	**0.0356**	0.0933	0.1040	0.2212	0.7408	0.0466
frb3_w_half	0.0274	**0.0237**	0.0256	0.0268	0.3072	0.0261	0.3962	**0.0237**
frb3_s_xyz	0.0130	**0.0090**	0.0148	—	0.0130	0.0092	0.0095	**0.0090**
frb3_s_static	0.0064	**0.0053**	—	0.0089	0.0134	0.0063	0.0083	0.0056
frb3_s_rpy	0.0302	0.0253	—	—	**0.0160**	0.0222	0.0214	0.0193
frb3_s_half	0.0191	0.0147	0.0160	—	0.0257	0.0145	0.0201	**0.0134**

Table 3. Ablation experiments camera pose estimation according to the metric of ATE (m).

Sequences	YOLOv7	YOLOv7 + LSD	YOLOv7 + LSD + K-means
frb3_w_xyz	0.014 767	0.014 451	**0.014 078**
frb3_w_static	0.006 969	0.006 999	**0.006 727**
frb3_w_rpy	0.047 328	**0.042 701**	0.046 695
frb3_w_half	0.026 741	0.025 511	**0.023 711**
frb3_s_xyz	0.009 585	**0.008 882**	0.009 098
frb3_s_static	0.006 235	0.005 791	**0.005 664**
frb3_s_rpy	0.018 9297	**0.018 387**	0.019 319
frb3_s_half	**0.013 123**	0.014 171	0.013 496

are shown in table 2,—indicates that the corresponding data were not provided in the original literature. In the eight experimental data sequences, the camera's pose estimation accuracy using the developed system is improved compared with the ORB-SLAM2 systems. Our DIG-SLAM and CFP-SLAM in frb3_w_half and frb3_s_xyz reach the same results. Compared with DynaSLAM, Blitz-SLAM, YOLO-SLAM, OPF-SLAM and PLD-SLAM, this developed system performs better on frb3_w_xyz, frb3_w_half, frb3_s_xyz, and frb3_s_half datasets, and obtains suboptimal results on the remaining frb3_s_static and frb3_s_rpy datasets.

To further validate the performance of camera pose estimation using instance segmentation and K-means clustering line features, this study performs ablation experiments on the TUM RGB-D dataset using the RMSE metric of absolute trajectory errors. According to the experimental results presented in table 3, the proposed camera pose estimation system performs better on frb3_w_xyz, frb3_w_static, frb3_w_half and frb3_s_static sequences when the LSD algorithm and K-means are added, decreasing the ATE values. These results show that the accuracy of camera pose estimation is improved by including instance segmentation and K-means clustering line features. Finally, we provide the octree map results for all sequences of the DIG-SLAM system as shown in figure 13.

4.2. Real world scenario evaluation

To further evaluate the performance of our developed system in real scenarios, the images with the size of 960 × 540 are captured using a Kinect2 camera in a real indoor dynamic scene, and the collected image data are processed according to the TUM dataset format. The comparison effects of extracted static features between DS-SLAM and DIG-SLAM in real scenarios are shown in figure 14. Figure 14(a) presents a sequence in which a person gets up from a chair beside the left-hand table, moves towards the middle table, and sits down. The comparison demonstrates that DIG-SLAM outperforms DS-SLAM in eliminating dynamic point and line features and in effectively utilizing of line features in the environment.

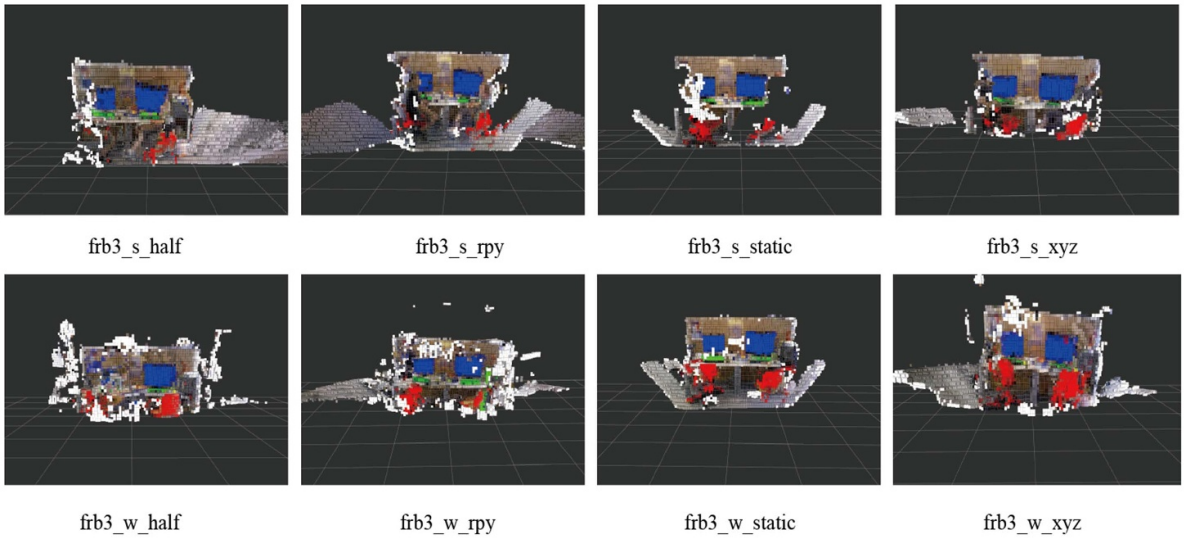


Figure 13. DIG-SLAM octree map results for all sequences.

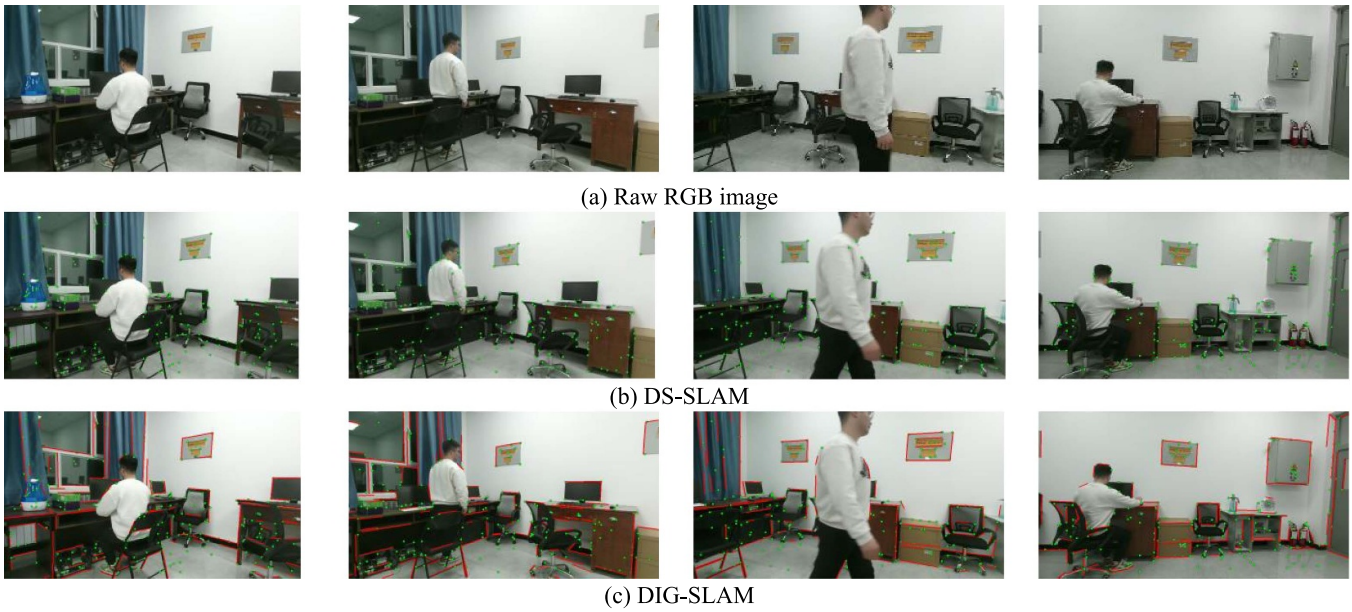


Figure 14. Comparison of extracted static feature sequences. (a) Shows the collected raw image data; (b)presents the static points features extracted by DS-SLAM; (c) shows the static point and line features extracted by DIG-SLAM.

In addition to the comparison of extracted static features in figure 14, we also built a static semantic 3D octree map of the real scene to further evaluate the mapping performance of our system. In figure 15, the monitor, wall, and seat are represented by blue, white, and red voxels, respectively. The display on the left cannot be successfully constructed due to the influence of the pedestrian in figure 15(a). In contrast, the system in this paper makes full use of static point and line features and uses YOLOv7 with more fine segmentation, thus achieving better map creation for the left and middle monitors and seats. To make it easier to see we have marked it with yellow boxes, which is shown in figure 15(b).

4.3. Runtime analysis

In this study, we tested the time consumption of the main modules to process each frame, and the tracking time is closely related to the computer performance and the number of extractions. In this paper, the average processing time per frame in the tracking thread is 126 ms. As shown in table 4, a time comparative analysis is conducted to assess the average processing time of DS-SLAM and the developed system in terms of target segmentation and feature extraction in the real-world and frb3_w_xyz scenes.

Table 4 presents the average time consumption for processing each frame evaluation result between DS-SLAM and DIG-SLAM. From table 4, we can see that in the feature

(a) DS-SLAM (b) DIG-SLAM

Figure 15. Comparison of octree mapping in a dynamic environment. DS-SLAM static semantic octree map is shown in (a), and our system's static semantic octree map is presented in (b).

Table 4. Average time consumption for processing each frame evaluation (ms).

	DS-SLAM		DIG-SLAM	
Sequences	Feature extraction (ORB)	SegNet-based semantic segmentation	Feature extraction (ORB and clustered lines)	YOLOv7-base instance segmentation
frb3_w_xyz	11.33	65.25	46.50	117.13
Real-world scenario	13.00	67.74	48.42	120.46

extraction part, DIG-SLAM consumes longer time in extracting features per frame due to the addition of line feature extraction compared to DS-SALM which only extracts point features. In each frame image segmentation to extract object region information, YOLOv7 has a finer segmentation and consumes longer time compared to SegNet, with an average time of 118.8 ms.

5. Conclusion

In this paper, we propose a novel semantic RGB-D SLAM system for modeling indoor dynamic environments using YOLOv7 instance segmentation and line features clustering. To improve camera pose estimation accuracy and system robustness, pixel-level instance segmentation is carried out using YOLOv7 to obtain precise contour and semantic information on the corresponding objects and identify persons as the dynamic object labels. Moreover, add line features, K-means clustering is used to remove redundant line features extracted by the LSD algorithm, reducing incorrect matching of line segments. Next, potential dynamic point features are examined through motion consistency. A segmented object is considered dynamic if this object contains potential dynamic point features. In this case, the object's region is labeled as dynamic, the point features in this dynamic region are removed, and the line features in this region are deleted using the three-point method. Then, optimize camera pose estimation using the remaining static point and line features combined with depth information from the depth images. Finally, combining positional optimization and depth information from depth images to create a static semantic 3D octree map to provide richer, higher-level scene understanding and perception capabilities for robots or autonomous systems.

To assess the performance of the developed DIG-SLAM, experiments were conducted on the sequences from the publicly available TUM RGB-D dataset as well as actual scenes. The experimental results demonstrate that our developed system increases the camera's pose estimation accuracy and system robustness. Especially in highly dynamic scenarios when compared with the original DS-SLAM and other dynamic SLAM. On the low dynamic dataset sequences, the camera's pose estimation accuracy is somewhat enhanced over the original DS-SLAM. These results demonstrate the developed DIG-SLAM is effective.

Despite the promising results of our proposed DIG-SLAM system, there still exists a limitation of its longer extra time consumption due to the instance segmentation and line feature clustering. In a future study, Firstly, we will focus on faster instance segmentation model to reduce time costs. Secondly, in the case of random initialization, K-means clustering is probably not able to converge to the optimal solution, and cannot reach the optimization of line features for scenes with complex textures. Next, we will use adaptive clustering algorithm for the study. Finally, our system does not involve optimization of loop closure detection, which will also be a work in progress.

Data availability statement

All data that support the findings of this study are included within the article (and any supplementary files).

Acknowledgments

This research was funded by the National Natural Science Foundation of China under Grant 62263031, the Natural Science Foundation of Xinjiang Uygur Autonomous Region under Grant 2022D01C53, Xiangyang Key R&D Project (High-tech Field) (2020ABH001799).

ORCID iDs

Rongguang Liang https://orcid.org/0009-0006-4935-5314
Jie Yuan https://orcid.org/0000-0002-5758-0267
Benfa Kuang https://orcid.org/0000-0003-2712-5973
Qiang Liu https://orcid.org/0000-0002-5173-187X
Zhenyu Guo https://orcid.org/0009-0003-6089-8780

References

[1] Giubilato R, Chiodini S, Pertile M and Debei S 2019 An evaluation of ROS-compatible stereo visual SLAM methods on a nVidia Jetson TX2 *Meas. Conf.* **140** 161–70
[2] Aladem M and Rawashdeh S A 2018 Lightweight visual odometry for autonomous mobile robots *Sensors* **18** 2837
[3] Cadena C, Carlone L, Carrillo H, Latif Y, Scaramuzza D, Neira J, Reid I and Leonard J J 2016 Past, present, and future of simultaneous localization and mapping: toward the robust-perception age *IEEE Trans. Robot.* **32** 1309–32
[4] Pumarola A *et al* 2017 PL-SLAM: real-time monocular visual SLAM with points and lines *IEEE Int. Conf. on Robotics and Automation (ICRA)* pp 4503–8
[5] Mur-Artal R and Tardós J D 2017 Orb-slam2: an open-source slam system for monocular, stereo, and RGB-D cameras *IEEE Trans. Robot.* **33** 1255–62
[6] Campos C, Elvira R, Rodriguez J J G, Montiel J M and Tardos D J 2021 Orb-slam3: an accurate open-source library for visual, visual–inertial, and multimap slam *IEEE Trans. Robot.* **37** 1874–90
[7] Engel J, Schöps T and Cremers D 2014 LSD-SLAM: large-scale direct monocular SLAM *Computer Vision–ECCV 2014* vol 8690 (https://doi.org/10.1007/978-3-319-10605-2_54)
[8] Labbé M and Michaud F 2019 RTAB-Map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation *J. Field Robot.* **36** 416–46
[9] Hu X *et al* 2022 CFP-SLAM: a real-time visual SLAM based on coarse-to-fine probability in dynamic environments *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)* pp 4399–406
[10] Dai W, Zhang Y, Li P, Fang Z and Scherer S 2020 Rgb-d slam in dynamic environments using point correlations *IEEE Trans. Pattern Anal. Mach. Intell.* **44** 373–89
[11] Fischler M A and Bolles R C 1981 Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography *Commun. ACM* **24** 381–95
[12] Leung T-S and Medioni G 2014 Visual navigation aid for the blind in dynamic environments *Proc. IEEE Conf. Computer Vision and Pattern Recognition Workshops* pp 565–72
[13] Bescos B, Facil J M, Civera J and Neira J 2018 DynaSLAM: tracking, mapping, and inpainting in dynamic scenes *IEEE Robot. Autom. Lett.* **3** 4076–83
[14] Yu C *et al* 2018 DS-SLAM: a semantic visual SLAM towards dynamic environments *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)* pp 1168–74
[15] Von Gioi R G, Jakubowicz J, Morel J-M and Randall G 2012 LSD: a line segment detector *Image Process. Online* **2** 35–55
[16] Zhang T and Zhang H 2020 Flowfusion: dynamic dense rgb-d slam based on optical flow *IEEE Int. Conf. on Robotics and Automation (ICRA)* pp 7322–8
[17] Jaimez M *et al* 2017 Fast odometry and scene flow from RGB-D cameras based on geometric clustering *IEEE Int. Conf. on Robotics and Automation (ICRA)* pp 3992–9
[18] Derome M, Plyer A, Sanfourche M and Besnerais G L 2015 Moving object detection in real-time using stereo from a mobile platform *Unmanned Syst.* **3** 253–66
[19] Sun Y, Liu M and Meng M Q H 2017 Improving RGB-D SLAM in dynamic environments: a motion removal approach *Robot. Auton. Syst.* **89** 110–22
[20] Ma S, Guo P, You H, He P, Li G and Li H 2021 An image matching optimization algorithm based on pixel shift clustering RANSAC *Inf. Sci.* **562** 452–74
[21] Wang H, Wang L and Fang B 2020 Robust visual odometry using semantic information in complex dynamic scenes *Cognitive Systems and Information Processing (ICCSIP)* pp 594–601
[22] Zhong F *et al* 2018 Detect-SLAM: making object detection and SLAM mutually beneficial *IEEE Winter Conf. on Applications of Computer Vision (WACV)* pp 1001–10
[23] You Y *et al* 2022 MISD-SLAM: multimodal semantic SLAM for dynamic environments *Wireless Communications and Mobile Computing* vol 2022 pp 1–13
[24] Wu W *et al* 2022 YOLO-SLAM: a semantic SLAM system towards dynamic environment with geometric constraint *Neural Comput. Appl.* **34** 6011–26
[25] Cheng J, Wang Z, Zhou H, Li L and Yao J 2020 DM-SLAM: a feature-based SLAM system for rigid dynamic scenes *ISPRS Int. J. Geo-Inf.* **9** 202
[26] Xiao L, Wang J, Qiu X, Rong Z and Zou X 2019 Dynamic-SLAM: semantic monocular visual localization and mapping based on deep learning in dynamic environment *Robot. Auton. Syst.* **117** 1–16
[27] Wang C Y, Bochkovskiy A and Liao H Y M 2022 YOLOv7: trainable bag-of-freebies sets new state-of-the-art for real-time object detectors (arXiv:2207.02696)
[28] Zhang Y *et al* 2022 Bytetrack: multi-object tracking by associating every detection box *Computer Vision–ECCV* pp 1–21
[29] Yasir M, Zhan L, Liu S, Wan J, Hossain M S, Isiacik Colak A T, Liu M, Islam Q U, Raza Mehdi S and Yang Q 2023 Instance segmentation ship detection based on improved Yolov7 using complex background SAR images *Front. Mar. Sci.* **10** 1113669
[30] Cao L, Zheng X and Fang L 2023 The semantic segmentation of standing tree images based on the Yolo V7 deep learning algorithm *Electronics* **12** 929
[31] Zhang L and Koch R 2013 An efficient and robust line segment matching approach based on LBD descriptor and pairwise geometric consistency *Vis. Commun. Image Represent.* **24** 794–805
[32] Kuang B, Yuan J and Liu Q 2022 A robust RGB-D SLAM based on multiple geometric features and semantic segmentation in dynamic environments *Meas. Sci. Technol.* **34** 015402
[33] Michael G 2022 Python package for the evaluation of odometry and SLAM (available at: https://github.com/MichaelGrupp/evo)
[34] Sturm J *et al* 2018 A benchmark for the evaluation of RGB-D SLAM systems *IEEE/RSJ Int. Conf. on Intelligent Robots abd systems(IROS)* pp 573–80
[35] Fan Y, Zhang Q, Tang Y, Liu S and Han H 2022 Blitz-SLAM: a semantic SLAM in dynamic environments *Pattern Recognit.* **121** 108225
[36] Cheng J, Sun Y and Meng M Q H 2019 Improving monocular visual SLAM in dynamic environments: an optical-flow-based approach *Adv. Robot.* **33** 576–89
[37] Zhang C, Huang T, Zhang R and Yi X 2021 PLD-SLAM: a new RGB-D SLAM method with point and line features for indoor dynamic scene *ISPRS Int. J. Geo-Inf.* **10** 163