DDETR-SLAM: A TRANSFORMER-BASED APPROACH TO POSE OPTIMISATION IN DYNAMIC ENVIRONMENTS

Feng Li*,** Yuanyuan Liu,* Kelong Zhang,** Zhengpeng Hu,** and Guozheng Zhang**

Abstract

Simultaneous localisation and mapping (SLAM) is a critical technology for accurate robot localisation and path planning. It has been an important area of research to improve localisation accuracy. In this paper, we propose a transformer-based visual semantic SLAM algorithm (DDETR-SLAM) to address the shortcomings of traditional visual SLAM frameworks, such as large localisation errors in dynamic scenes. First, by incorporating the deformable Detection Transformer (DETR) network as an object detection thread, the pose estimation accuracy of the system has been improved compared to ORB-SLAM2. Furthermore, an algorithm that combines the semantic information is designed to eliminate outlier points generated by dynamic objects, thereby improving the accuracy and robustness of SLAM localisation and mapping. Experiments are conducted on the public TUM datasets to verify the localisation accuracy, computational efficiency, and readability of the point cloud map of DDETR-SLAM. The results show that in highly dynamic environments, the absolute trajectory error (ATE), translation error, and rotation error are reduced by 98.45%, 95.34%, and 92.67%, respectively, when compared to ORB-SLAM2. In most cases, our proposed system outperforms DS-SLAM, DynaSLAM, Detect-SLAM, RGB-D SLAM, and YOLOv5+ORB-SLAM2. The relative pose error (RPE) is only 0.0076 m, the ATE is only 0.0063 m, and the dense mapping also has better readability.

Key Words

Simultaneous localisation and mapping, deformable DETR, object detection, dynamic environments

1. Introduction

Simultaneous localisation and mapping (SLAM) is commonly used in various fields, including indoor mobile robots, autonomous driving technology, virtual reality, and mobile augmented reality [1]. As significant progress has been made in visual SLAM methods, Davison *et al.* [2] introduced the first entirely visual SLAM method using monocular cameras, but with limited applicability to smaller scenes; Klein and Murray [3] proposed a SLAM system called PTAM and the method effectively reduces the computational cost of monocular SLAM; Mur-Artal *et al.* [4], [5] first proposed an open source monocular ORB-SLAM system within the algorithmic framework of [4]; Mur-Artal *et al.* [5] then proposed an ORB-SLAM2 system based on ORB-SLAM [6], which supports binocular and RGBD cameras. Combining motion-compensated image difference and maximum a posteriori estimation, Sun *et al.* [7] proposed a method to roughly detect moving objects.

However, before ensuring the robustness and effectiveness of the system, most of the current visual SLAM methods are based on static scenes [8]. This strict assumption is not in line with the actual situation. When there are dynamic objects in the indoor environment, the traditional SLAM algorithm will fail, because the SLAM system will produce errors in feature matching, pose estimation, looping and mapping [8]. Therefore, it is a key challenge to achieve high-precision visual SLAM to extract high-quality feature points in a dynamic environment, distinguish static points from dynamic points, remove outliers, and use static points for fundamental pose estimation and mapping. With the rapid advancement of deep learning and the maturation of algorithms like object detection [9] and semantic segmentation [10], the demand for deep learning-based dynamic indoor scene (SLAM) is growing [11], such as SOF-SLAM [12], DynaSLAM [13], Detect-SLAM [14], Dynamic SLAM [15]. Chen *et al.* [16] combined monocular SLAM with multi-object tracking using an object detection algorithm to provide initial semantic information to the SLAM system and eliminate all feature points in the detection frame. Most of the proposed methods use a combination of deep learning and geometry. Most of the proposed methods use a combination of deep learning and geometry. While some progress has been made, the real-time performance and stability of the slam system remain challenges in practical applications.

* School of Computer Science and Technology, Donghua University, Shanghai, China; e-mail: {lifeng, 2212490}@mail.dhu.edu.cn
** National Innovation Center of Advanced Dyeing and Finishing Technology, Shandong, China; e-mail: zkl19911219@163.com; huzp2019@foxmail.com; 1943220103@qq.com
Corresponding author: Yuanyuan Liu

Recommended by Raffaele Di Gregorio
(DOI: 10.2316/J.2024.206-1063)

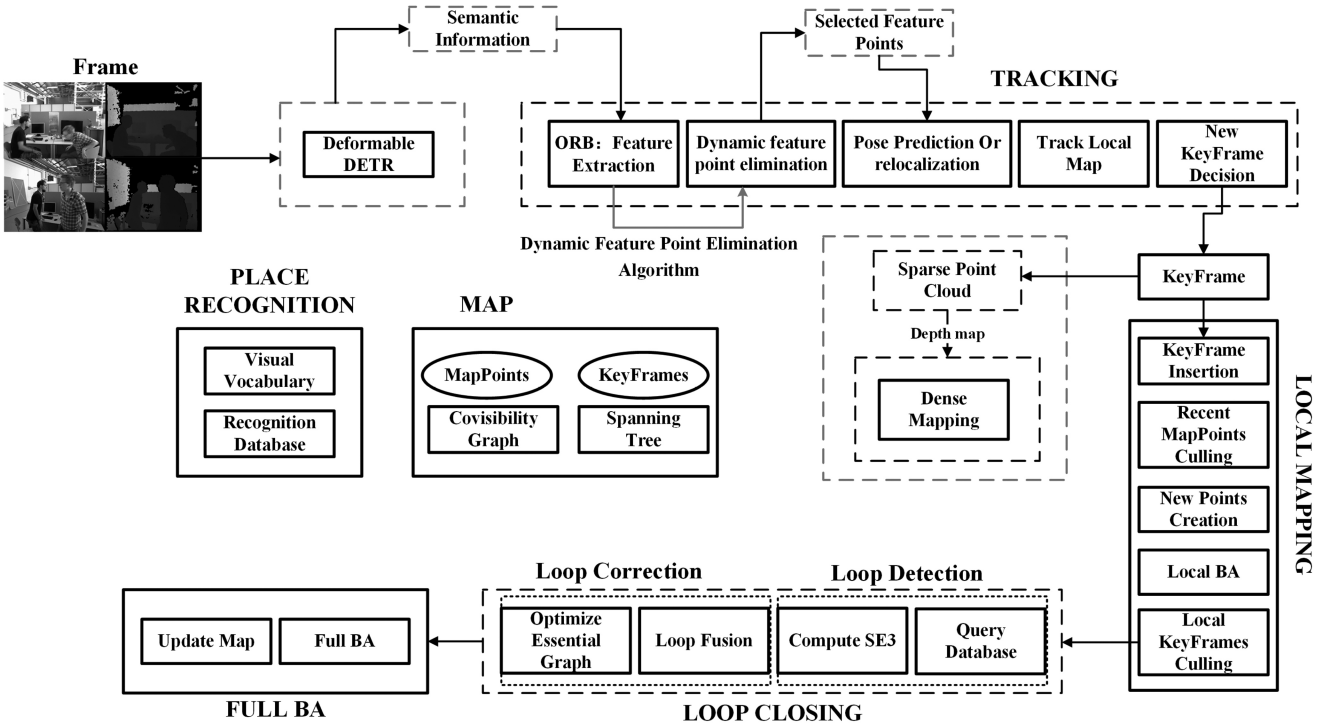


Figure 1. The SLAM framework proposed in this paper.

In this paper, an indoor dynamic environment DDETR-SLAM algorithm is proposed, which uses feature point method and fusion object detection algorithm to solve the challenge of inaccurate positioning and poor robustness of visual SLAM system in dynamic environment mapping. The transformer-based object detection thread has been added to ORB-SLAM2. At the same time, the semantic information is input into the tracking thread. The main contributions of this paper are as follows:

1. To solve the problem of poor localisation accuracy, we propose an improved deformable DETR + ORB-SLAM2 algorithm.
2. We propose a dynamic feature point elimination algorithm to reduce the false rejection rate of feature points.
3. We study and propose a dynamic feature point elimination algorithm to reduce the false rejection rate of feature points, and verify it in the actual scene, which proves the validity of the algorithm.

Static background reconstruction [17] has improved the legibility of the dense point cloud map in dynamic scenarios, making it easier for future robots to navigate and utilise the map. In addition, compared with other methods, the proposed method has relatively lower resource requirements.

2. System Framework

2.1 Structure of the DDETR-SLAM

The classical framework of visual SLAM consists of five components: sensor data, visual odometry, back end optimisation, loop closure detection, and mapping [17], [18]. ORB-SLAM2 is a remarkable open-source SLAM system that is compatible with monocular, binocular, and RGBD cameras. It performs exceptionally well in static settings by ensuring consistency between camera trajectory and map construction [18]. References [13] and [19] used Mask-RCNN [20] and SegNet [21] semantic segmentation networks, respectively, which are time-consuming and difficult to achieve real-time results in the whole SLAM system.

ORB-SLAM2, a typical SLAM framework based on the feature point approach, consists of three parallel threads: tracking, local mapping, and loop closing.

1. The tracking thread extracts and matches features of images using the ORB algorithm, optimises the camera pose based on the information from the matches, tracks the local map, and inserts the local map by selecting key frames from all the frames.
2. Keyframes connect the local mapping thread to the tracking thread, which removes redundant map points and keyframes, and optimises the local map using the bundle adjustment (BA) algorithm [22].
3. Loopback detection uses Bag of Word (BOW) [23] to determine if the same scene is being reached and to correct the camera pose, thus reducing the cumulative error.

The DDETR-SLAM system, shown in Fig. 1, improves the original threads of ORB-SLAM2 by incorporating object detection and dense mapping threads. The input image is processed by the deformable detection transformer (DETR) network, which provides initial semantic information for the tracking thread, including object position and class. This semantic information, combined with the dynamic feature point elimination algorithm, accurately identifies dynamic feature points from static ones and eliminates incorrect rejections from previous frames. The

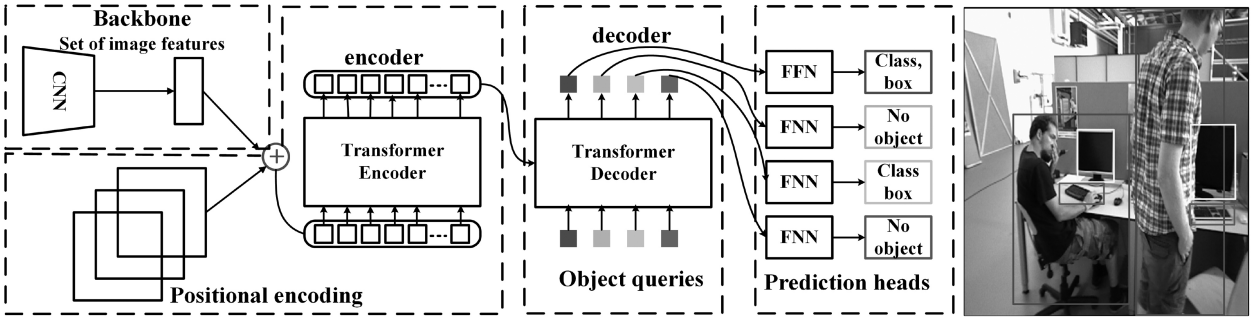


Figure 2. DETR network architecture [25].

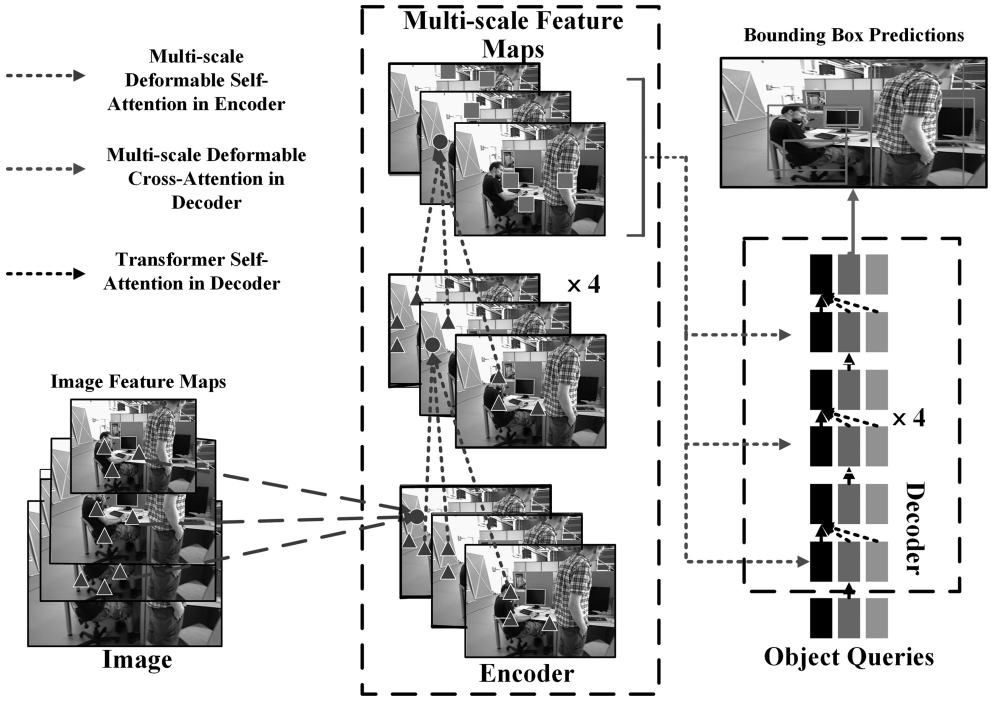


Figure 3. Deformable DETR network architecture [26].

remaining static points serve to enhance location precision and improve the comprehensibility of the mapping results in SLAM through camera position estimation.

2.2 Object Detection Algorithm

Transformer was first proposed for natural language processing (NLP) [24]. Later, Facebook AI researchers introduced DETR [25], which is the first application of transformer in the field of vision. DETR is an end-to-end network with a clear and efficient structure that replaces the anchor mechanism with object query and the original non-maximum suppression with bipartite graph matching. Three components make up its structure: a forward network FFN for prediction, a Transformer for encoding–decoding , and a CNN for feature extraction [25]. The DETR uses the global modeling capability of Transformer to treat object detection as an ensemble prediction problem, making model training, and deployment easier and with comparable performance to the SOTA approach. Figure 2 illustrates the structure of DETR.

After DETR, deformable DETR [26] emerged, a new network model called deformable DETR [26] was developed. This model reduces the required training rounds, achieves faster convergence, and improves efficiency. This network architecture addresses DETR's limitation of performing well on large objects but poorly on small objects. The deformable DETR replaces the transformer attention module that processes the feature map with a multi-scale deformable attention module, which solves the problem of excessive computing and memory requirements [26].

Deformable DETR is selected to detect dynamic objects. The pre-trained model is used to detect objects in the image and extract semantic information from it. The pre-trained model is trained on the COCO data set (including 80 categories). Although this cannot cover all object types, in indoor and simple outdoor scenarios, 80 categories are sufficient to meet the requirements. Figure 3 is the structure of Deformable DETR.

For the purposes of tracking, we consider "people" to be dynamic objects and incorporate semantic information

into the process. Our proposed algorithm eliminates dynamic feature points. In the experimental section, we compare the performance of deformable DETR with the latest YOLOv5 to show the advantages of our approach in visual odometry.

3. Dynamic Feature Point Elimination

3.1 Dynamic Feature Point Elimination Algorithm

We use deformable DETR to detect objects in the image to distinguish static and dynamic feature points. To improve the tracking process, we integrate semantic information and masks as inputs. However, if dynamic objects occupy a significant portion of the scene, removing all feature points within the detection bounding boxes could lead to tracking failures and hinder pose optimisation. To solve this problem, our algorithm differentiates feature point characteristics to reduce the occurrence of false rejection and improve overall performance.

Algorithmic Approach: The deformable DETR is utilised for object detection in camera images. The objects in the scene belong to three classes: high, low, and static. Through classification of the detection boxes into these categories, we can better process each category in detail, which improves the flexibility and adjustability of the system. Qualitative classifications of bounding boxes as either dynamic or static are determined by the semantic information obtained from object detection. The elimination of feature points only occurs in specific situations.

$$P = [p_1, p_2, \ldots, p_m]^T, m = 1, 2 \ldots \tag{1}$$

$$E'\left(\widehat{\delta}\right) = E_d\left(\widehat{\delta}\right) \&\& E_s\left(\widehat{\delta}\right) \tag{2}$$

where p is the feature point set, $\widehat{\delta} \in [p_1, p_2, \ldots, p_m]^T$, $E_d()$ is the function to determine whether $p_{\ i}$ is a dynamic feature point, and $E_s()$ is the function to determine whether $p_{\ i}$ is a static feature point. $E'()$ is a function that assigns values to the state of the feature point $p_{\ i}$. The algorithm's pseudocode is shown in Algorithm 1. The pseudocode only explains the decision to classify points as dynamic or static, because the algorithm focuses on dealing with dynamic and static objects, rather than directly dealing with detection boxes with "low" attributes. Instead, we use the RANSAC algorithm in the original system to filter feature points with "low" attribute in the detection boxes to eliminate potential mismatches.

To guarantee the accuracy of outlier removal, we have observed that reducing the count of feature points can influence pose estimation in the tracking thread. Therefore, except for human subjects, we utilise the following algorithm to handle dynamic feature points. The dynamic feature point elimination algorithm employs the same approach to feature point quantity as this method. The pseudocode is shown below.

1: **Modify the number of feature points:**
2: Original number of feature points: n
3: Modified number of feature points: $1.5n$
4: **Motion object discrimination method:**

Algorithm 1 Dynamic Feature Point Elimination Algorithm

Input: Semantic information from Deformable ETR detect result; The extracted feature points of ORB, P
Output: $\acute{E}(\widehat{\delta})$
$\widehat{p}$= PointWithinBoundingBox(P, box_id);
$E_d\left(\widehat{\delta}\right)$= FindPointIsInDynamicBox($p_{\ i}$);
$E_s\left(\widehat{\delta}\right)$= FindPointIsInStaticBox($p_{\ i}$);
for k = 0; k <$\widehat{p}$.size(); ++k **do**
  if $E_d(k) == true \&\& E_s(k) == false$ **then**
    vbPointInDynamickeys.push_back(true);
    p → (-1,-1,-1);
  else
    vbPointInDynamickeys.push_back (false);
  endif
end for

5: Overlapping area threshold: $T = 0.9$
6: Bounding box information: d
7: **The current frame area**: p
8: **For everyframe, compare the bounding box information based on the category and determine whether to remove the feature points:**
9: **If** the ratio of the bounding box information d to the current frame area p exceeds the threshold value T, the feature points located in the border will not be removed.
10: **Else** remove the feature points.

3.2 Systematic Evaluation Metrics

relative pose error (RPE) and ATE are commonly used metrics to assess visual SLAM and odometry [27]. RPE quantifies translational and rotational drift in the trajectory, whereas ATE specifically measures translational error. RPE takes into account both translational and rotational errors. RPE is described as follows:

$$e_i := \left(G_i^{-1} G_{i+\Delta}\right)^{-1} \left(E_i^{-1} E_{i+\Delta}\right) \tag{3}$$

where $E_1, \ldots, E_n \in SE(3)$ is the estimated camera trajectory sequence and $G_1, \ldots, G_n \in SE(3)$ is the true camera trajectory sequence. Δ is time interval. The root mean square error (RMSE) is often used to evaluate the performance of the SLAM system over the entire time period, RMSE is described as follows:

$$\mathrm{RMSE}(e_{1:n}, \Delta) := \left(\frac{1}{m}\sum_{i=1}^{m} \left\|\mathrm{trans}(e_i)\right\|^2\right)^{\frac{1}{2}} \tag{4}$$

$$\mathrm{RMSE}(e_{1:n}, \Delta) := \left(\frac{1}{m}\sum_{i=1}^{m} \left\|\mathrm{rota}(e_i)\right\|^2\right)^{\frac{1}{2}} \tag{5}$$

Where $\mathrm{trans}(e_i)$ is the translation error component of RTE, $\mathrm{rota}(e_i)$ is the rotation error component of RTE, $m = n - \Delta$ is the relative pose error of individuals in the sequence, and we set $\Delta = 1$ in the experiment. To average

Table 1
Dataset Details [27]

Sequence	Duration	Ground-truth trajectory length	Avg. translational velocity	Avg. angular velocity
freiburg3_walking_xyz	28.83 s	5.791 m	0.208 m/s	5.490 deg/s
freiburg3_walking_halfsphere	35.81 s	7.686 m	0.221 m/s	18.267 deg/s
freiburg3_walking_rpy	30.61 s	2.698 m	0.091 m/s	20.903 deg/s
freiburg3_walking_static	24.83 s	0.282 m	0.012 m/s	1.388 deg/s
freiburg3_sitting_static	23.63 s	0.259 m	0.011 m/s	1.699 deg/s

all time intervals Δ, RMSE is defined as follows:

$$\mathrm{RMSE}\left(e_{1:n}\right) := \frac{1}{n}\sum_{\Delta=1}^{n}\mathrm{RMSE}\left(e_{1:n},\Delta\right) \tag{6}$$

ATE is used to evaluate the consistency of camera trajectories and algorithmic stability. Unlike RPE, ATE only contains translation errors. Under Time Step i, the ATE is defined as follows:

$$\theta_i := {G_i}^{-1} S E_i \tag{7}$$

$$\mathrm{RMSE}\left(\theta_{1:n}\right) := \left(\frac{1}{n}\sum_{i=1}^{n}\left\|\mathrm{trans}\left(\theta_i\right)\right\|^2\right)^{\frac{1}{2}} \tag{8}$$

Where S is a rigid transformation that maps $G_{1:n}$ to $E_{1:n}$ using a least squares solution.

3.3 TUM Dataset

The TUM RGB-D dataset, provided by the University of Munich, is a widely used evaluation tool for SLAM system. The dataset was collected by Microsoft Xbox Kinect sensor with a resolution of 640 × 480 and a sampling rate of 30 Hz. Its limitation is that it focuses only on indoor scenes. Therefore, it is inadequate for representing outdoor environments or other diverse settings. This narrow emphasis might result in a bias in the dataset that impairs the model's capacity to generalise to real-world settings. Nevertheless, despite these limitations, the dataset provides a strong foundation for research in visual and depth perception in indoor environments. It helps to understanding the perception challenges in the indoor environments and provides valuable data resources for the domains of robotics and computer vision. Table 1 shows the dataset used in the experiment, categorised by camera motion: xyz, half-sphere, rpy, and static. The walking scenario aims to evaluate the effectiveness of visual SLAM and odometry algorithms in handling rapidly moving dynamic objects, while the sitting scenario evaluates the performance of these algorithms with slowly moving dynamic objects. Assessment metrics include RPE and ATE, with RMSE, SD, mean, and median used for evaluation.

4. Experiments and Analysis

In this section, we evaluate the feasibility of the proposed method and perform several experiments on the TUM RGB-D dataset to evaluate the effectiveness of the proposed method. In addition, we compared the performance of our method with OBR-SLAM2, DS-SLAM, Dyna-SLAM, Detect-SLAM, RGB-D SLAM, and YOLOV5+ORB-SLAM2 on the TUM RGB-D dataset. It is worth noting that the system architecture of YOLOv5+ORB-SLAM2 is identical to that of DDETR-SLAM, including the dynamic feature point elimination algorithm, except that the detection network is different. All experiments were performed on a desktop with Intel(R) Xeon(R) CPU and 4GB of memory running Ubuntu Linux 18.04 LTS.

4.1 Object Detection Algorithm Experiment

The object detection thread provides semantic information for the ORB-SLAM2 tracking thread, including bounding box masks and categories. The detection thread uses consistent resolutions (640 × 480) for all images to guarantee quantitative analysis. The inference threshold for individual images is set to 0.35, and the non-maximum suppression threshold for YOLOv5 is set to 0.5. As shown in Fig. 4(b) and (c), deformable DETR is sensitive to smaller objects. Moreover, under the same threshold, deformable DETR performs better than YOLOv5 in long-distance object detection, and the position accuracy of the object selected by the bounding box is higher. It is worth noting that deformable DETR can effectively detect smaller objects on the desktop. Moreover, the experimental results of dynamic point removal outlined in Section 4.4 show that the proposed algorithm combined with the deformable DETR thread is superior to other methods in optimising the camera position, which proves the effectiveness of our method.

4.2 Dynamic Point Removal Experiment

In ORB-SLAM2, all feature points (both dynamic and static) are used for matching and pose estimation. However, in highly dynamic environments, unprocessed dynamic points can compromise the accuracy of pose estimation. In order to solve this problem, our proposed algorithm eliminates external points in the dynamic scene when

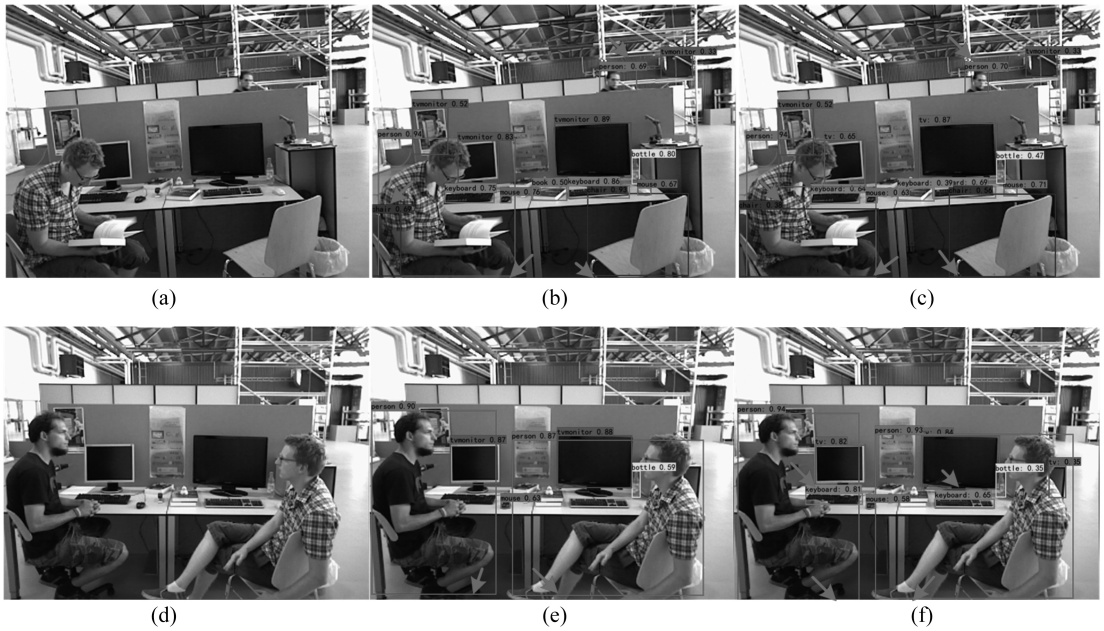


Figure 4. Detection results of YOLOv5 and deformable DETR on the TUM dataset: (a) and (d) is the dataset of fr3/walking_half and fr3/sitting_static; (b) and (e) is the detection result of the dataset using YOLOv5; and (c) and (f) is the detection result of the dataset using deformable DETR.

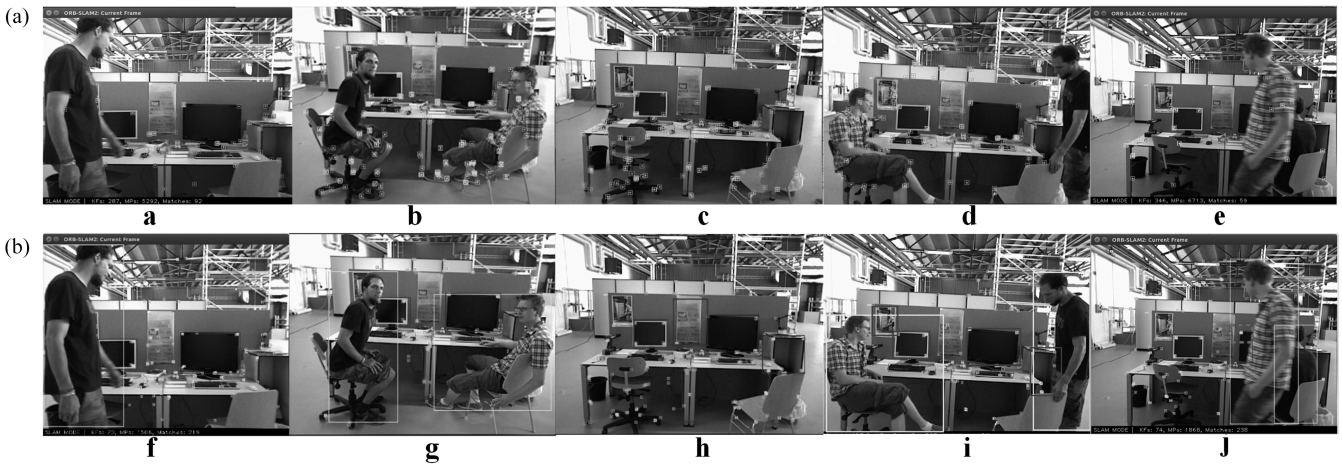


Figure 5. Results of feature point extraction by ORB-SLAM2 and DDETR-SLAM for the TUM dataset: (a), (b), (c), (d), and (e) are ORB-SLAM2 feature point extraction results with unfiltered dynamic feature points; (f), (g), (h), (i), and (j) are dynamic feature point filtering results of the proposed method in different scenes, where the yellow border is the position of the dynamic object identified by deformable DETR. Feature points on dynamic objects are removed in DDETR-SLAM.

extracting and matching feature points. By doing this, only static feature points are preserved, enhancing the reliability and stability of the system. Figure 5 shows the result of feature point extraction from ORB-SLAM2 and DDETR-SLAM in different scenarios, including unmanned scenes, one-person scenes, and two-person scenes.

The improved SLAM system does not eliminate all the feature points in the detection box classified as human, but selectively eliminates the dynamic feature points. This method is committed to selectively eliminating feature points, which effectively avoids the elimination of all feature points in the detection box when dynamic objects occupy significant space. By doing this, the system maintains a higher number of matched feature points, improving camera tracking and maintaining the stability of the SLAM system. Our proposed dynamic point elimination algorithm defines object attributes as high, low, and static based on semantic information provided by the detection thread. Objects with low attributes are not processed, while the RAndom SAmple Consensus (RANSAC) algorithm is used to filter dynamic feature points from objects with moderate attributes. In Fig. 5, we aim to retain static feature points for camera pose estimation and mapping. The experimental results show that the algorithm can effectively and accurately remove the dynamic feature points in the scene.

(a) (b)

Figure 6. In the real dynamic scene, the RGB-D camera is used to perform dynamic feature point elimination experiments: (a) real scene and (b) experiment scene.

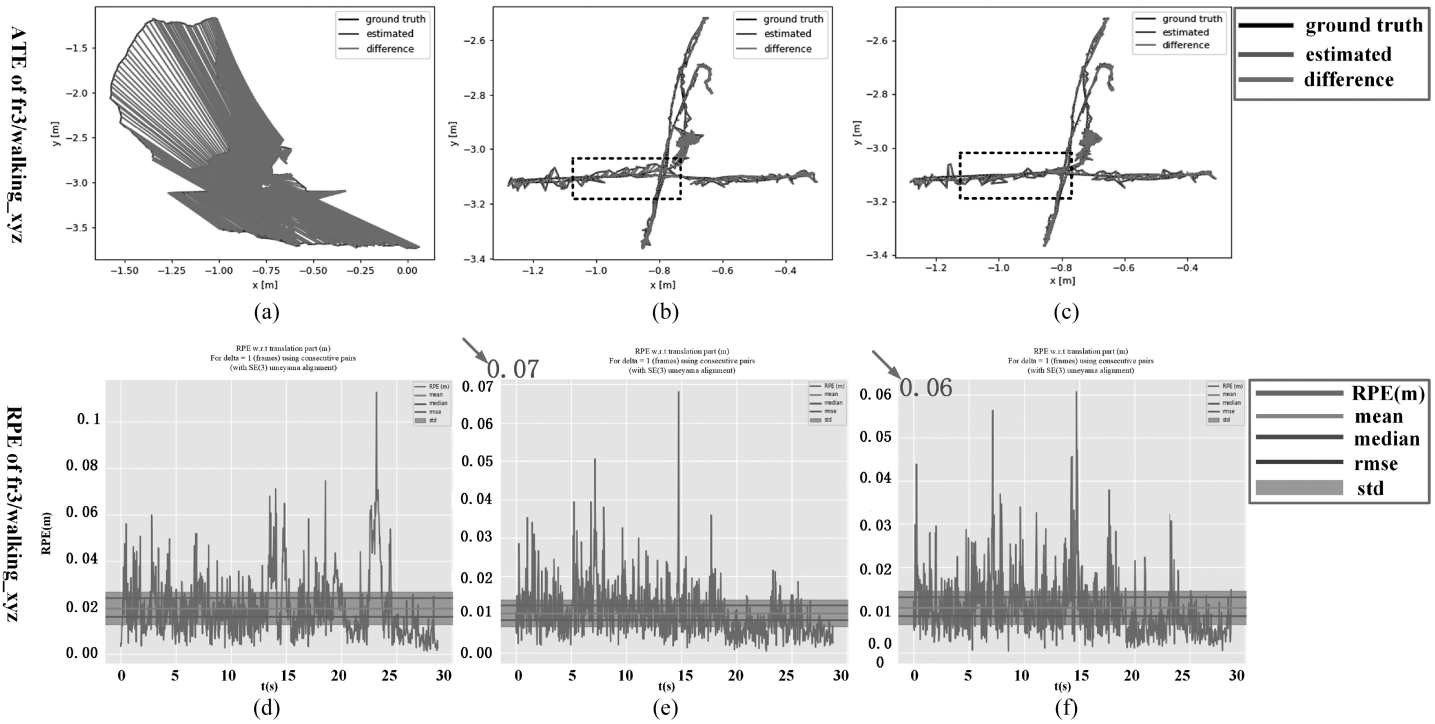


Figure 7. The ATE and RPE results of ORB-SLAM2, DDETR-SLAM and YOLOv5-SLAM. The diagrams are computed through the fr3/walking_xyz sequence. The graphs (a), (b), and (c) are ATE results and (d), (e), and (f) are RPE results.

4.2.1 Evaluation in Real Environment

To demonstrate the efficacy of our algorithm, we conducted experiments in a laboratory setting, capturing real-world dynamic scenarios. Figure 6(a) shows the actual scene captured using an RGB-D camera. In Fig. 6(b), the first column displays the results of object detection. The third and fourth columns of Fig. 6(b) show the feature point extraction results of ORB-SLAM2 and our algorithm, respectively.

As observed in the third column of Fig. 6(b), ORB-SLAM2 extracts a large number of feature points, including many dynamic feature points. However, in the fourth column of Fig. 6(b), our proposed algorithm effectively eliminates feature points on moving individuals, retaining only those associated with the static background. This observation underscores the viability of our proposed algorithm.

4.3 Performance Evaluation on TUM RGB-D Dataset

In this section, we select TUM RGBD datasets to evaluate the performance of the proposed algorithm in different scenarios. Figure 7(a) and (d) shows the visualisation of ORBSLAM2 on the fr3/walking_xyz , where the black

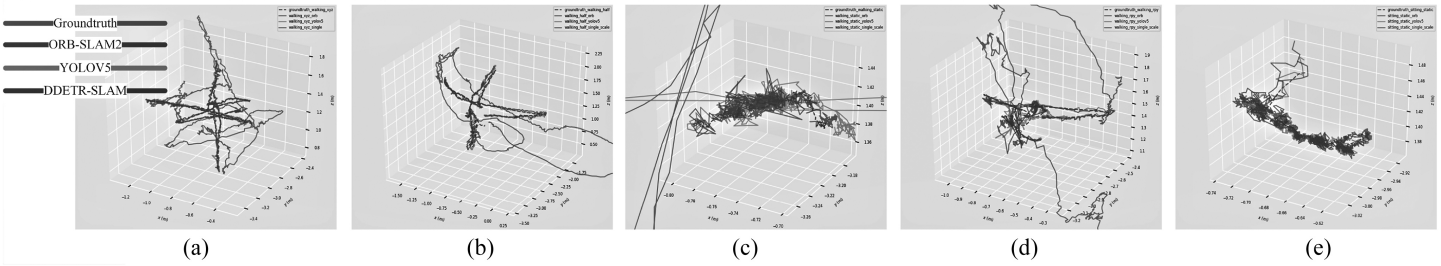


Figure 8. Ground truth and trajectories estimated by DDETR-SLAM, YOLOv5+ORBSLAM2 and ORB-SLAM2 in the TUM sequence fr3/walking_xyz: (a) fr3/walking_xyz; (b) fr3/walking_halfsphere; (c) fr3/walking_static; (d) fr3/walking_rpy; and (e) fr3/sitting_static.

line represents the actual camera trajectory, the blue line represents the estimated value, and the red line represents the difference between the two. Figure 7(b) and (e) demonstrates the visualisation results with YOLOv5 as the object detection framework, while Fig. 7(c) and (f) shows the results obtained using deformable DETR. However, ORB-SLAM2 struggles to filter dynamic feature points, leading to significant positioning and orientation inaccuracies in dynamic environments. To address this issue, we extended ORB-SLAM2 and compared it with the original system, achieving reduced ATE and RPE in camera pose estimation. This improvement enhances the pose estimation accuracy of the DDETR-SLAM system.

Fig. 7(b) illustrates that YOLOv5 + ORBSLAM2 achieves better camera pose matching compared to the ORB-SLAM2. However, the deformable DETR+ ORBSLAM2 result in Figs. 7(b) and (c) exhibits a more accurate camera pose estimation. The ORB-SLAM2 system initially produces a maximum relative positional inaccuracy of 1.506343 m within 1 s. In contrast, the maximum relative position deviation of the improved SLAM system is less than 0.075971 m in the same period. This improvement is attributed to our algorithm's ability to effectively reduce false elimination of dynamic points and remove dynamic feature points from the scene, so as to obtain superior localisation outcomes. Figure 8 shows a visualisation of our proposed algorithm on the TUM dataset. The black line represents the real trajectory, the red line represents ORB-SLAM2, the green line represents our algorithm, and the blue line represents YOLOv5+ORB-SLAM2.

Table 2 shows the comparison results of APE, where the best result is indicated in bold font. "-" denotes that the original paper does not provide the result. Compared with ORB-SLAM2, DDETR-SLAM has a significant improvement in both low-dynamic and high-dynamic scenarios. Our algorithm achieves the best results on fr3/walking_xyz, fr3/walking_rpy, and fr3/walking_static sequences. However, compared with ORB-SLAM2, this improvement is not obvious in low dynamic sequences. This can be attributed to the static characteristics of the scene, where ORB-SLAM2 removes a small number of dynamic feature points using the RANSAC algorithm. DynaSLAM obtains optimal results in fr3/walking_half and fr3/sitting_static sequences attributable to its precise semantic segmentation. Nevertheless, our algorithm also obtains almost the same results as DynaSLAM.

Tables 3 and 4 list the translation drift and rotation drift results of each algorithm in the TUM dataset. Due to the ability to identify objects and eliminate dynamic feature points, DDETR-SLAM has obvious advantages over other algorithms in reducing translation drift and rotation drift errors in high dynamic scenes. The lower RPE value indicates that the results generated by the SLAM algorithm have better stability and consistency. This is crucial for achieving precise positioning in real-time applications like navigation and robot path planning.

Tables 5–7 show the percentage improvement in ATE and RPE of DDETR-SLAM compared to ORB-SLAM2. The results in these tables are calculated as follows:

$$\Delta = \frac{\alpha - \beta}{\alpha} \times 100\% \tag{9}$$

where Δ is the improved value, α is the value of ORB-SLAM2 and β is the value of DDETR-SLAM.

Table 5 shows that the proposed method significantly reduces the ATE in the fr3/walking_xyz dataset, with RMSE of 98.45% and standard deviation of 98.44%. The results of Tables 6 and 7 show that in the case of complex camera motion, such as in a scene similar to fr3/walk_rpy, the improvement of rotation error is lower than that of translation error. The main reason is that the fast motion of the camera causes the image to be blurred, and the camera cannot effectively track the key frame.

The results show that the DDETR-SLAM system can improve the robustness and accuracy of visual SLAM in high dynamic environments. In the field of robot navigation, accurate camera pose is crucial for robot navigation and obstacle avoidance tasks. Therefore, the proposed method also improves the performance and efficiency of the entire navigation system. As ORB-SLAM2 uses RANSAC algorithm to deal with low dynamic object outliers, our algorithm has only a slight improvement in dealing with low dynamic environment. However, in dynamic scenes, the deviation result of camera pose estimation is better than ORB-SLAM2.

4.4 Dense Point Cloud Mapping Experiment

For tasks such as robot navigation, the reconstruction of the static background is crucial. In Fig. 9, The deformable DETR is employed for image detection, specifically for dynamic object detection. Subsequently, the detection

Table 2
Comparison of the RMSE of ATE(m) of DDETR-SLAM against OBR-SLAM2, DS-SLAM, DynaSLAM, Detect-SLAM and YOLOV5+ORB-SLAM2 for TUM RGB-D Dataset

Sequences		fr3/walking_xyz	fr3/walking_half	fr3/walking_static	fr3/walking_rpy	fr3/sitting_static
ORB-SLAM2	RMSE	0.9749	0.6303	0.3953	0.9292	0.0085
	S.D.	0.5199	0.2871	0.156	0.483	0.0042
	mean	0.8247	0.5610	0.3632	0.7938	0.0074
	median	0.7441	0.4568	0.3093	0.8166	0.0064
DS-SLAM [19]	RMSE	0.0247	0.0303	0.0081	0.4442	0.0065
	S.D.	0.0161	0.0159	0.0036	0.235	0.0033
	mean	0.0186	**0.0258**	0.0073	0.3768	**0.0055**
	median	0.0151	0.0222	0.0067	0.2835	**0.0049**
DynaSLAM [13]	RMSE	0.0164	**0.0296**	0.0068	0.0354	0.0064
	S.D.	0.0086	**0.0157**	0.0032	0.0190	–
	mean	–	–	–	–	–
	median	–	–	–	–	–
Detect-SLAM	RMSE	0.0241	0.0514	–	0.2959	–
	S.D.	0.0159	0.0231	–	0.1481	–
	mean	–	–	–	–	–
	median	–	–	–	–	–
YOLOv5+ORB-SLAM2	RMSE	0.0171	0.0356	0.0202	0.0312	0.0069
	S.D.	0.0089	0.0188	0.0175	0.0194	0.0040
	mean	0.0147	0.0296	0.0103	0.0243	0.0062
	median	0.0123	0.0270	0.0075	0.0189	0.0057
The proposed approach	RMSE	**0.0151**	0.0307	**0.0067**	**0.0297**	**0.0063**
	S.D.	**0.0081**	0.0178	**0.0031**	**0.0184**	**0.0032**
	mean	**0.0139**	0.0287	**0.0069**	**0.0233**	0.0061
	median	**0.0122**	**0.0214**	**0.0059**	**0.0176**	0.0051

The bold fonts represent the best results

results of the current frame are used to remove dynamic objects from the depth map. ORB-SLAM2 then integrates the processed depth map and color image. The final output is a concatenated 3-D point cloud.

Figure 10 displays the visualisation of the static background reconstruction outcome in a highly dynamic environment. The mapping results show that the improved method effectively eliminates the appearance of dynamic objects in the map and enhances the readability of the map.

4.5 Real-time Performance Evaluation

The evaluation of algorithm performance involves not only assessing localisation accuracy but also evaluating real-time performance. Tracking time refers to the time required for the tracking thread to complete tasks, such as single-frame image extraction, feature matching, and pose estimation. Table 8 shows the time consumption for processing a single image by different networks. Tables 9 and 10 provide the average tracking times for various systems. For our algorithm, the time spent on semantic extraction is relatively minimal. In high-dynamic scenes, the tracking time is slightly higher than that of ORB-SLAM2. In low-dynamic scenes, such as fr3/walking_static and fr3/sitting_static, the per-frame tracking time is comparable to the original system. Overall, in terms of tracking thread efficiency, our proposed system significantly outperforms DynaSLAM [13], which utilises Mask-RCNN for semantic segmentation. In terms of tracking time, this system is on par with ORB-SLAM2,

Table 3
Comparison of the Metric Translational Drift (m/frame) of DDETR-SLAM against OBR-SLAM2, DS-SLAM, DynaSLAM, RGB-D SLAM, and YOLOV5+ORB-SLAM2 for TUM RGB-D Dataset

Sequences		fr3/walking_xyz	fr3/walking_half	fr3/walking_static	fr3/walking_rpy	fr3/sitting_static
ORB-SLAM2	RMSE	0.4395	0.4222	0.2099	0.3917	0.0089
	S.D.	0.3004	0.3207	0.1905	0.2787	0.0044
	mean	0.3208	0.2746	0.0883	0.2752	0.0078
	median	0.1961	0.1031	0.0142	0.1494	0.0069
DS-SLAM [19]	RMSE	0.0333	0.0297	0.0102	0.1503	0.0078
	S.D.	0.0229	0.0152	0.0048	0.1168	**0.0038**
	mean	0.0238	0.0256	0.0091	0.0942	**0.0068**
	median	0.0181	**0.0226**	0.0082	0.0457	**0.0061**
DynaSLAM [13]	RMSE	0.0218	**0.0272**	0.0102	0.0529	0.0085
	S.D.	0.0121	**0.0102**	0.0041	0.0342	0.0079
	mean	–	–	–	–	–
	median	–	–	–	–	–
RGB-D SLAM [28]	RMSE	0.1210	0.167	0.084	0.175	–
	S.D.	–	–	–	–	–
	mean	0.089	0.108	0.045	0.136	–
	median	–	–	–	–	–
YOLOv5+ORB-SLAM2	RMSE	0.0219	0.0333	0.0275	0.0447	0.0079
	S.D.	0.0111	0.0227	0.0239	0.0268	0.0038
	mean	0.0189	0.0288	0.0135	0.0358	0.0070
	median	0.0166	0.0252	0.0093	0.0287	0.0062
The proposed approach	RMSE	**0.0205**	0.0305	**0.0099**	**0.0412**	**0.0076**
	S.D.	**0.0101**	0.0169	**0.0039**	**0.0241**	0.0043
	mean	**0.0178**	**0.0243**	**0.0089**	**0.0333**	0.0080
	median	**0.0161**	0.0233	**0.0082**	**0.0272**	0.0068

The bold fonts represent the best results

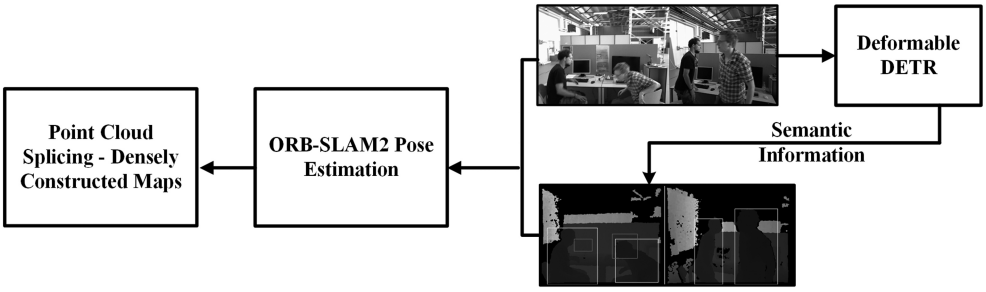


Figure 9. Mapping process.

but excels in terms of localisation accuracy. The system achieves a frame rate of 18 frames per second on the CPU hardware platform, primarily due to the introduction of object detection threads and dynamic feature point elimination algorithms. These additional computations consume some processing time. It is worth noting that the system's performance speed can be enhanced through the utilisation of GPU acceleration.

Table 4
Comparison of the Metric Rotational Drift (deg/frame) of DDETR-SLAM against OBR-SLAM2, DS-SLAM, DynaSLAM, RGB-D SLAM, and YOLOV5+ORB-SLAM2 for TUM RGB-D Dataset

Sequences		fr3/walking_xyz	fr3/walking_half	fr3/walking_ static	fr3/walking_rpy	fr3/sitting_static
ORB-SLAM2	RMSE	8.3994	8.7824	3.8036	7.5574	0.2867
	S.D.	5.8007	6.5599	3.4272	5.3568	0.1247
	mean	6.0747	5.8393	1.6497	5.3309	0.2582
	median	0.0563	0.0415	0.006	0.0511	0.0043
DS-SLAM [19]	RMSE	0.8266	0.8142	0.2690	3.0042	0.2735
	S.D.	0.5826	0.4101	**0.1182**	2.3065	0.1215
	mean	0.5836	**0.7033**	**0.2416**	1.9187	**0.2450**
	median	0.4192	0.6217	0.2259	0.9902	0.2351
DynaSLAM [13]	RMSE	0.6284	**0.7842**	**0.2612**	0.9894	–
	S.D.	0.3848	**0.4012**	0.1259	0.5701	–
	mean	–	–	–	–	–
	median	–	–	–	–	–
RGB-D SLAM [28]	RMSE	3.2350	5.0100	2.0490	4.3750	–
	S.D.	–	–	–	–	–
	mean	2.2560	3.2880	1.0550	3.3600	–
	median	–	–	–	–	–
YOLOv5+ORB-SLAM2	RMSE	0.6356	0.8809	0.5365	0.9699	**0.2729**
	S.D.	0.3890	0.4477	0.4273	0.5745	**0.1182**
	mean	0.5037	0.7586	0.3243	0.7814	0.2459
	median	0.0075	0.0116	0.0043	0.0111	0.0041
The proposed approach	RMSE	**0.6159**	0.8189	0.2743	**0.9372**	0.2745
	S.D.	**0.3715**	0.4158	0.1296	**0.5444**	0.1231
	mean	**0.5027**	0.7054	0.2417	**0.7751**	0.2454
	median	**0.0072**	**0.0113**	**0.0039**	**0.0110**	**0.0040**

The bold fonts represent the best results

Table 5
Improvement Results of ATE Performance (unit:m)

Sequences	ORB-SLAM2		The proposed approach		Improvements	
	RMSE	S.D.	RMSE	S.D.	RMSE	S.D.
fr3/walking_xyz	0.9749	0.5199	0.0151	0.0081	98.45%	98.44%
fr3/walking_half	0.6303	0.2871	0.0307	0.0178	95.13%	93.80%
fr3/walking_static	0.3953	0.156	0.0067	0.0031	98.31%	98.01%
fr3_walking_rpy	0.9292	0.483	0.0297	0.0184	96.80%	96.19%
fr3/sitting_static	0.0085	0.0042	0.0063	0.0032	25.88%	23.81%

Table 6
Improvement Results of Translational Error (unit:m)

Sequences	ORBSLAM2		The proposed approach		Improvements	
	RMSE	S.D.	RMSE	S.D.	RMSE	S.D.
fr3/walking_xyz	0.4395	0.3004	0.0205	0.0101	95.34%	96.64%
fr3/walking_half	0.4222	0.3207	0.0305	0.0169	92.78%	94.73%
fr3/walking_static	0.2099	0.1905	0.0099	0.0039	95.28%	97.95%
fr3_walking_rpy	0.3917	0.2787	0.0412	0.0241	89.48%	91.35%
fr3/sitting_static	0.0089	0.0044	0.0076	0.0043	14.61%	2.27%

Table 7
Improvement Results of Rotation Error (unit: deg)

Sequences	ORB-SLAM2		The proposed approach		Improvements	
	RMSE	S.D.	RMSE	S.D.	RMSE	S.D.
fr3/walking_xyz	8.3994	5.8007	0.6159	0.3715	92.67%	93.60%
fr3/walking_half	8.7824	6.5599	0.8189	0.4158	90.68%	93.66%
fr3/walking_static	3.8036	3.4272	0.2743	0.1296	92.79%	96.22%
fr3_walking_rpy	7.5574	5.3568	0.9372	0.5444	87.60%	89.84%
fr3/sitting_static	0.2867	0.1247	0.2745	0.1231	4.26%	1.28%

(a) (b)

Figure 10. The dense mapping results of ORB-SLAM2 and DDETR-SLAM. On the left is the ORB-SLAM2 mapping result; on the right is the DDETR-SLAM mapping result. The improved system successfully removes the dynamic feature points: (a) ORB_SLAM2; (b) DDETR-SLAM.

5. Conclusion

In this paper, we introduce DDETR-SLAM, a new system for visual SLAM designed specifically for dynamic indoor environments. To improve accuracy and stability in such environments, DDETR-SLAM uses a dynamic feature point elimination algorithm and combines Deformable DETR for object detection. The experimental results show that compared with the traditional visual SLAM framework, DDETR-SLAM shows significant improvements in positioning accuracy and map readability. In particular, the performance of the algorithm is better than other advanced SLAM systems, showing advantages in both low-dynamic and high-dynamic scenarios. Simulation results

Table 8
Comparison of Detection Times

Algorithm	Network Model	Time of single image processing (ms/image)
YOLOv5+ORB-SLAM2	YOLOv5	51.0
DynaSLAM	Mask R-CNN	198.0
DDETR-SLAM	Deformable DETR	**42.0**

Table 9
Comparison of Tracking Time on the TUM Datasets

Sequence	Algorithm	Track each frame time (ms)
fr3/walking_xyz	ORB-SLAM2	65.1
	The proposed approach	66.0
fr3/walking_half	ORB-SLAM2	60.2
	The proposed approach	63.0
fr3/walking_rpy	ORB-SLAM2	51.7
	The proposed approach	48.9
fr3/walking_static	ORB-SLAM2	56.9
	The proposed approach	51.5
fr3/sitting_static	ORB-SLAM2	52.4
	The proposed approach	51.6

Table 10
The Tracking Thread Time and Per Frame Time of the Improved Algorithm and Other Algorithms

Method	Tracking the Average Time of Each Frame of Image (ms)	Tracking Thread Time Consuming (ms)	Hardware Platform
ORB-SLAM2	57.26	37.36	**Intel (R) Xeon (R) CPU**
DynaSLAM	672.38	345.3	GeForceRTX1650
The proposed approach	56.20	45.60	**Intel (R) Xeon (R) CPU**

show that compared to ORB-SLAM2 and other leading SLAM methods, the proposed method has significantly improved positioning accuracy, computational efficiency, and readability of point cloud images. Although progress has been made in positioning accuracy and real-time performance, the following problems still exist. On the one hand, the real-time performance of the system needs to be improved and further optimised. On the other hand, we intend to test different environments on the Turtlebot robot to further strengthen its position.

Conflict of Interest

The authors declare no conflict of interest with any commercial entity or other organization in conducting this study.

Acknowledgement

We would like to thank all those who contributed to this paper. We are also grateful for the support of the project of "Introducing Urgently Needed Talents into Key Supporting Regions of Shandong Province, Construction of Intelligent System for Textile Printing and Dyeing Products" and "National Innovation Center of Advanced Dyeing and Finishing Technology".

References

[1] J. Fuentes-Pacheco, J. Ruiz-Ascencio, and J.M. Rendón-Mancha, Visual simultaneous localization and mapping: A survey, *Artificial Intelligence Review*, *43*(1), 2015, 55–81.

[2] 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, 29(6), 2007, 1052–1067.

[3] G. Klein and D. Murray, Parallel tracking and mapping for small AR workspaces, *Proc. 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, Nara, 2007, 225–234.

[4] R. Mur-Artal, J.M.M. Montiel, and J.D. Tardós, ORB-SLAM: A versatile and accurate monocular SLAM system, *IEEE Transactions on Robotics*, *31*(5), 2015, 1147–1163.

[5] 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, 33*(5), 2017, 1255–1262.

[6] T. Taketomi, H. Uchiyama, and S. Ikeda, Visual SLAM algorithms: A survey from 2010 to 2016, *IPSJ Transaction on Computer Vision Applications*, *9*(1), 2017, 1–11.

[7] Z. Chang, H. Wu, Y. Sun, and C. Li, RGB-D visual SLAM based on Yolov4-tiny in indoor dynamic environment, *Micromachines*, *13*(2), 2022, 230.

[8] X. Gao, X. Shi, Q. Ge, and K. Chen, An overview of visual SLAM for dynamic object scenes, *Robotics,* 2021.

[9] T. Diwan, G. Anirudh, and J.V. Tembhurne, Object detection using YOLO: Challenges, architectural successors, datasets and applications, *Multimedia Tools and Applications*, *82,* 2022, 9243– 9275.

[10] S. Minaee, Y. Boykov, F. Porikli, A. Plaza, N. Kehtarnavaz, and D. Terzopoulos, Image segmentation using deep learning: A survey, *IEEE Transactions on Pattern Analysis and Machine Intelligence, 44*(7), 2022, 3523–3542.

[11] L. Kenye and R. Kala, Improving RGB-D SLAM in dynamic environments using semantic aided segmentation, *Robotica, 40*(6), 2022, 2065–2090.

[12] L. Cui and C. Ma, SOF-SLAM: A semantic visual SLAM for dynamic environments, *IEEE Access*, *7*, 2019, 166528–166539.

[13] B. Bescos, J.M. Facil, J. Civera, and J. Neira, DynaSLAM: Tracking, mapping, and inpainting in dynamic scenes, *IEEE Robotics and Automation Letter*, *3*(4), 2018, 4076–4083.

[14] F. Zhong, S. Wang, Z. Zhang, C. Chen, and Y. Wang, Detect-SLAM: Making object detection and SLAM mutually beneficial, *Proc. IEEE Winter Conf. on Applications of Computer Vision (WACV),* Lake Tahoe*,* NV*,* 2018, 1001–1010.

[15] 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*, *117*, 2019, 1–16.

[16] W. Chen, M. Fang, Y.-H. Liu, and L. Li, Monocular semantic SLAM in dynamic street scene based on multiple object tracking, *Proc. IEEE International Conf. on Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM)*, Ningbo, 2017, 599–604.

[17] Y. Hu, S. Ma, B. Li, M. Wang, and Y. Wang, Dynamic modelling of reconfigurable robots with independent locomotion and manipulation ability, *International Journal of Robotics and Automation*, *32*(3), 2017, 206–4381.

[18] G. Yang, Z. Chen, Y. Li, and Z. Su, Rapid relocation method for mobile robot based on improved ORB-SLAM2 algorithm, *Remote Sensing*, *11*(2), 2019, 149

[19] 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, *Proc. IEEE/RSJ International Conf. on Intelligent Robots and Systems (IROS), Madrid,* Oct. 2018, 1168–1174.

[20] K. He, G. Gkioxari, P. Dollár, and R. Girshick, Mask R-CNN, *Proc. of the IEEE International Conf. on Computer Vision,* Venice, 2017, 2961–2969.

[21] V. Badrinarayanan, A. Kendall, and R. Cipolla, Segnet: A deep convolutional encoder-decoder architecture for image segmentation, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*(12)*,* 2017, 2481–2495.

[22] B. Triggs, P.F. McLauchlan, R.I. Hartley, and A.W. Fitzgibbon, Bundle adjustment—A modern synthesis, *Proc. International Workshop on Vision Algorithms*, Corfu, 1999, 298–372.

[23] C.–F. Tsai, Bag-of-words representation in image annotation: A review, *ISRN Artificial Intelligence, 2012*, 2012, 1–19.

[24] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, and I. Polosukhin, Attention is all you need*,* Dec. 05, 2017, *arXiv:1706.03762*.

[25] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, End-to-end object detection with transformers, 2020, *arXiv:2005.12872*.

[26] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, Deformable DETR: Deformable transformers for end-to-end object detection, 2021, *arXiv:2010.04159*.

[27] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, A benchmark for the evaluation of RGB-D SLAM systems, *Proc. IEEE/RSJ International Conf. on Intelligent Robots and Systems*, Vilamoura-Algarve, 2012, 573–580.

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

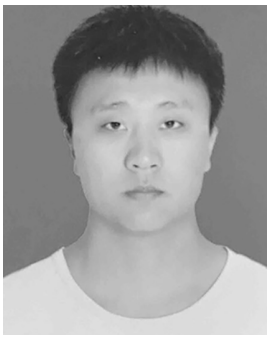
Biographies

Feng Li received the Ph.D. degree in automation and intelligent monitoring from Southwest Jiaotong University, Chengdu, Sichuan, China, in 1998. He is currently a Professor with the School of Computer Science and Technology, Donghua University, Shanghai, China. His current research interests include image processing, artificial intelligence, embedded systems, body sense networks, and intelligent hardware.

Yuanyuan Liu received the B.S. degree from the Henan University of Chinese Medicine, in 2020, She is currently pursuing the M.S. degree with Donghua University. Her research interests include Visual SLAM and machine vision.

Kelong Zhang was born on December 1991. He received the B.S. degree from the School of Automation, Heilongjiang University of Science and Technology. He is currently an Engineer and with the National Innovation Center of Advanced Dyeing and Finishing Technology, Shandong, China.

Zhengpeng Hu was born on April 1980. He received the B.S. degree from the School of Information-Technology. He is an Engineer with the National Innovation Center of Advanced Dyeing and Finishing Technology, Shandong, China. His research interests focus on fabric defect detection and computer vision.



Guozheng Zhang was born on February 1997. He received the B.S. degree from the School of Automation. He is an Engineer and with the National Innovation Center of Advanced Dyeing and Finishing Technology, Shandong, China. His research interests focus on fabric defect detection, image retrieval, and computervision.