Computers, Materials & Continua

Tech Science Press

DOI: 10.32604/cmc.2024.058028

ARTICLE

PPS-SLAM: Dynamic Visual SLAM with a Precise Pruning Strategy

Jiansheng Peng[1,2,3,4,*], Wei Qian[1] and Hongyu Zhang[1]

[1]College of Automation, Guangxi University of Science and Technology, Liuzhou, 545000, China

[2]Department of Artificial Intelligence and Manufacturing, Hechi University, Hechi, 546300, China

[3]Key Laboratory of AI and Information Processing, Education Department of Guangxi Zhuang Autonomous Region, Hechi, 546300, China

[4]Guangxi Key Laboratory of Sericulture Ecology and Applied Intelligent Technology, School of Chemistry and Bioengineering, Hechi University, Hechi, 546300, China

*Corresponding Author: Jiansheng Peng. Email: pengjs@hcnu.edu.cn

Received: 02 September 2024 Accepted: 28 October 2024 Published: 17 February 2025

ABSTRACT

Dynamic visual SLAM (Simultaneous Localization and Mapping) is an important research area, but existing methods struggle to balance real-time performance and accuracy in removing dynamic feature points, especially when semantic information is missing. This paper presents a novel dynamic SLAM system that uses optical flow tracking and epipolar geometry to identify dynamic feature points and applies a regional dynamic probability method to improve removal accuracy. We developed two innovative algorithms for precise pruning of dynamic regions: first, using optical flow and epipolar geometry to identify and prune dynamic areas while preserving static regions on stationary dynamic objects to optimize tracking performance; second, propagating dynamic probabilities across frames to mitigate the impact of semantic information loss in some frames. Experiments show that our system significantly reduces trajectory and pose errors in dynamic scenes, achieving dynamic feature point removal accuracy close to that of semantic segmentation methods, while maintaining high real-time performance. Our system performs exceptionally well in highly dynamic environments, especially where complex dynamic objects are present, demonstrating its advantage in handling dynamic scenarios. The experiments also show that while traditional methods may fail in tracking when semantic information is lost, our approach effectively reduces the misidentification of dynamic regions caused by such loss, thus improving system robustness and accuracy.

KEYWORDS

Visual SLAM; dynamic SLAM; YOLOv8

1 Introduction

Visual SLAM (Simultaneous Localization and Mapping) is an important research direction in the SLAM field. Its low cost and rich visual information offer many research opportunities. Dynamic environments are very harmful to any SLAM system. Thus, dynamic SLAM research has become one of the most important areas in visual SLAM in recent years [1].

The main solution for dynamic SLAM is to directly remove feature points on dynamic objects or their corresponding dynamic map points. This addresses the problem at its root. Identifying which feature points are dynamic has become the main improvement method. In most works, authors use deep learning and machine vision to get semantic information from the current frame. This method has proven to have two obvious problems [2].

First, in Crowd-SLAM [3], the authors state that too much removal or incomplete removal can degrade performance. This issue is mentioned in many papers, including CDS-SLAM [4]. In the newer DFD-SLAM [5], more precise removal strategies are proven effective. Currently, the problem of excessive and imprecise removal still has not improved much. The biggest issue is that achieving better removal usually requires precise semantic segmentation. Currently, object detection-based visual SLAM performance is average. The detection boxes for dynamic objects are often much larger than the objects themselves, causing severe over-removal. Using semantic segmentation increases system time consumption. Therefore, we aim for an object detection-based SLAM system that can achieve very precise removal.

In dynamic SLAM, ensuring real-time performance requires using smaller weight files or lightweight models for object detection, which often degrades detection or segmentation quality. This unreliable semantic information can severely impact system performance due to unremoved dynamic feature points. Methods like RDS-SLAM and NGD-SLAM attempt to address this issue [6,7], with NGD-SLAM even rotating frames for better semantic information. Clearly, obtaining high-quality semantic information is crucial for maintaining dynamic SLAM performance.

To address these issues, we propose PPS-SLAM (Dynamic Visual SLAM with a Precise Pruning Strategy). By adding a semantic thread to ORB-SLAM3 [8], we use YOLOv8 with TensorRT for accelerated inference to obtain semantic information [9,10]. Our method employs two innovative algorithms for the precise pruning of dynamic regions. The first algorithm uses an optical flow-epipolar approach to identify and prune dynamic areas while retaining static regions on dynamic objects when stationary to optimize tracking. The second algorithm transmits dynamic probabilities between frames, allowing for the propagation of semantic information and mitigating the impact of missing semantic data in some frames. Our specific contributions are as follows:

1. We built a complete real-time dynamic visual SLAM system using YOLOv8 with TensorRT for inference. Thanks to our excellent pruning strategy, we use the lightweight YOLOv8n for detection, ensuring real-time performance.

2. We developed a novel pruning strategy based on object detection semantic information. This allows the system to accurately find dynamic areas within the detection box using the optical flow-epipolar method. It avoids excessive or missed removal of feature points.

3. We calculate dynamic probabilities for each region of the frame based on semantic information. This allows dynamic semantic information to be transmitted between frames, compensating for frames with failed semantic information.

Our paper consists of five main chapters. Section 2 introduces related dynamic SLAM works. In Section 3, we explain in detail the two main methods of our precise pruning strategy. Section 4 presents detailed tests of our system on the high dynamic sequences of the TUM-RGBD dataset [11]. We compare our results with well-known and recent visual SLAM works. We also conduct comprehensive ablation experiments and real-time analysis. Section 5 provides a complete summary.

2 Related Works

Up to now, many high-performance visual SLAM systems have been developed. Early systems like MONO-SLAM and PTAM utilized monocular cameras as input sensors for real-time localization and mapping [12,13]. Among these, PTAM is regarded as the classic technique for monocular SLAM, considered the first system capable of performing real-time SLAM. Many of its ideas have been adopted in subsequent visual SLAM systems. However, monocular SLAM fundamentally relies on calculating depth information by matching corresponding pixel points across consecutive frames. This depth information is not directly obtained, leading to generally lower accuracy. The subsequent work on S-PTAM supporting stereo input allowed the system to compute relatively direct depth information [14], proving its superior performance over mono-based PTAM. Meanwhile, RGB-D cameras have made significant advancements in the SLAM field. RGB-D SLAM utilizes the RGB-D camera to acquire RGB images along with depth maps [15], thus directly obtaining the corresponding pixel depth information. However, RGB-D cameras have a limited working distance and are significantly affected by lighting conditions, making them suitable primarily for indoor environments. In 2017, VINS-MONO was released [16], employing both visual information and IMU data for pose estimation. This system fully leveraged the nonlinear optimization techniques of visual SLAM, achieving tight coupling of visual and IMU data. The inclusion of IMU enables the system to directly obtain pose information, while the integration of visual data aids in mapping and correcting IMU drift during optimization. In extensive localization and mapping tasks or when the system undergoes complex motions, IMU enhances the robustness and accuracy of visual SLAM systems. The ORB-SLAM3 used in this study is the latest member of the ORB-SLAM family [17], and most current dynamic visual SLAM systems are adaptations derived from this series due to its outstanding real-time performance and tracking accuracy. Typical dynamic SLAM systems, such as the classic DynaSLAM and DS-SLAM [18,19], are improvements based on ORB-SLAM2 [20]. Generally, dynamic SLAM works utilize RGB-D cameras as input and test dynamic performance in indoor environments. This is likely because indoor dynamic SLAM datasets are more plentiful, and dynamic SLAM focuses on improvements specific to dynamic environments. The use of RGB-D cameras can achieve relatively high accuracy in indoor settings, thereby thoroughly validating the effectiveness of the proposed enhancements.

DynaSLAM, a classic representative of dynamic visual SLAM, builds upon the ORB-SLAM2 system by integrating Mask R-CNN for semantic segmentation of the current frame [21], allowing for the removal of dynamic feature points. Even in 2024, DynaSLAM continues to demonstrate exceptional tracking accuracy. However, this also raises certain issues, such as the poor real-time performance resulting from the segmentation model inferred with PyTorch. Other works have begun to place greater emphasis on the precise removal of dynamic points. DynaSLAM, a classic representative of dynamic visual SLAM, builds upon the ORB-SLAM2 system by integrating Mask R-CNN for semantic segmentation of the current frame, allowing for the removal of dynamic feature points. Even in 2024, DynaSLAM continues to demonstrate exceptional tracking accuracy. However, this also raises certain issues, such as the poor real-time performance resulting from the segmentation model inferred with PyTorch. Other works have begun to place greater emphasis on the precise removal of dynamic points. DS-SLAM, from the same period as DynaSLAM, adds optical flow epipolar geometry to help remove dynamic objects. This method influenced many later dynamic SLAM works. In Crowd-SLAM, the authors state that the quality of removed dynamic points affects tracking accuracy. Many later works aim to improve the removal effect. In 2021, the ORB-SLAM series was updated to the third generation. Many dynamic SLAM systems began to improve based on ORB-SLAM3. Recently, most dynamic SLAM systems use high-performance inference frameworks for fast model inference to achieve good real-time performance. In SG-SLAM, the authors use the NCNN framework for target

detection network inference, achieving excellent real-time performance [22,23]. They propose a new removal strategy based on probability calculation. However, this method does not solve the problem of "false positives" in dynamic SLAM based on target detection. This can lead to tracking loss when dynamic objects occupy a large part of the frame. In CDS-SLAM, the authors optimize for seated and standing humans. They propose a more detailed removal strategy. This system uses TensorRT for model inference, achieving excellent tracking accuracy and real-time performance. In the latest DFD-SLAM, the authors propose a dual-model parallel inference dynamic SLAM system. This system's regional precise removal strategy offers a novel removal method, achieving better tracking accuracy. Its real-time performance shows that even with per-frame semantic information, excellent real-time performance is possible with high-performance inference methods. Precise removal of dynamic feature points is a key research direction in dynamic SLAM.

At the same time, some authors found the problem of losing semantic information. In RDS-SLAM, the authors proposed a method based on the ROS framework [24]. They get semantic information only for keyframes and use dynamic probability propagation to remove dynamic points. This method greatly improves the system's real-time performance but loses some removal accuracy. This results in average tracking accuracy. However, updating the dynamic probability of map points helps to some extent. It compensates for the loss of semantic information and improves system accuracy. In the updated NGD-SLAM, the authors rotate and crop frames that might lose semantic information and then detect them. They found that detection fails when people in frames are tilted, but when the frame is smaller and upright, detection performance recovers. Compensating for the loss of semantic information is very important. Often, the system's accuracy drops significantly during a sequence with rotating scenes, as semantic information is lost in those parts.

3 Methods

3.1 System Architecture

Our system enhances ORB-SLAM3 with two modules: a semantic module and a precise region removal module. The semantic module runs on a separate thread using YOLOv8 and TensorRT for real-time object detection. The precise region removal module uses the optical flow-epipolar method to verify feature points and accurately identify dynamic regions for removal. Our system framework is illustrated in Fig. 1. It should be noted that the input resolution of the system can be adjusted without affecting its performance. In our subsequent experiments, we used the TUM-RGBD dataset, which has a standard size of 640 $\times$ 480. The input RGB images first go through the semantic thread to obtain semantic information, while feature extraction converts them to grayscale for tracking. Our keyframe settings, bundle adjustment optimization frequency, loop closure, and relocalization remain unchanged, following the default parameters of the RGBD mode in the open-source ORB-SLAM3 framework.

We are using YOLOv8 as our semantic module. Our input dimensions are 640 $\times$ 480, and inference is performed frame by frame. The confidence threshold is set at 0.6, which is relatively high but necessary, as it helps reduce incorrect detections to some extent. This is particularly important when the probability propagation method is enabled, as incorrect detections can have a certain impact. However, with this threshold, our method works together smoothly. It's important to note that the quality of the semantic information obtained by the object detection network is not the most critical factor. We primarily use the object detection network to identify dynamic regions, which allows our strategy to perform effectively. Therefore, most mainstream object detection networks can be paired with this geometric strategy to achieve excellent tracking performance.

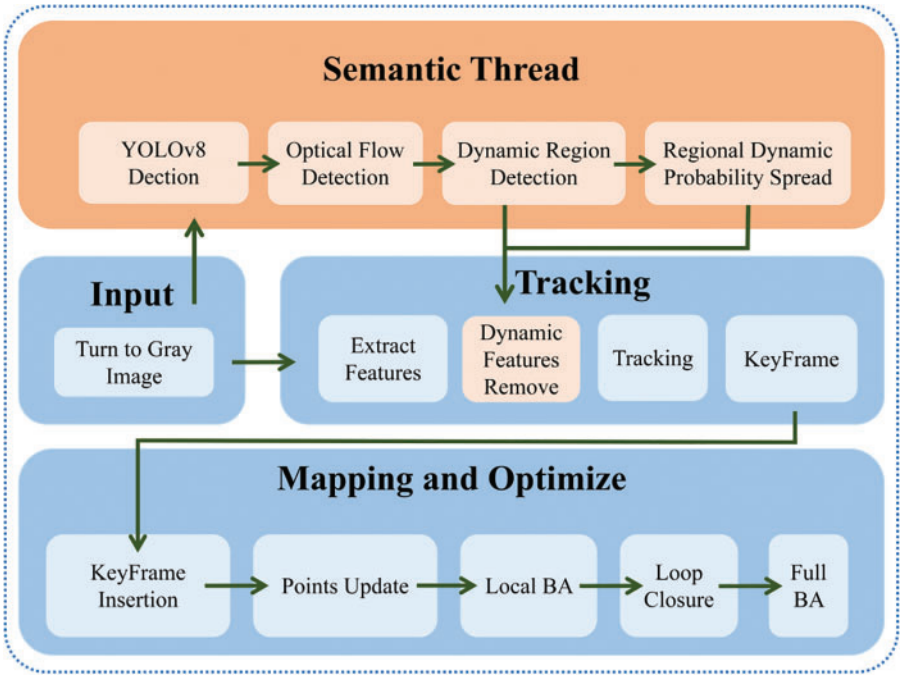


Figure 1: System architecture

In the TUM-RGBD dataset, most dynamic objects are easily detected, as is the case in most SLAM scenarios. Thus, the semantic accuracy of the object detection network is not that important, especially with the assistance of the region dynamic probability propagation method. The difference in semantic accuracy caused by network rotation is further minimized. In our tests, YOLOv9 and YOLOv10 did not show any explainable difference compared to YOLOv8. They all performed well enough in detecting dynamic objects in TUM-RGBD. Therefore, we chose the more mature YOLOv8 for faster inference in our experiments. However, it is worth noting that newer networks typically have fewer parameters and lower computational costs, which might be advantageous on some embedded platforms. However, in our experimental platform, when using smaller weights, the real-time performance differences were minimal.

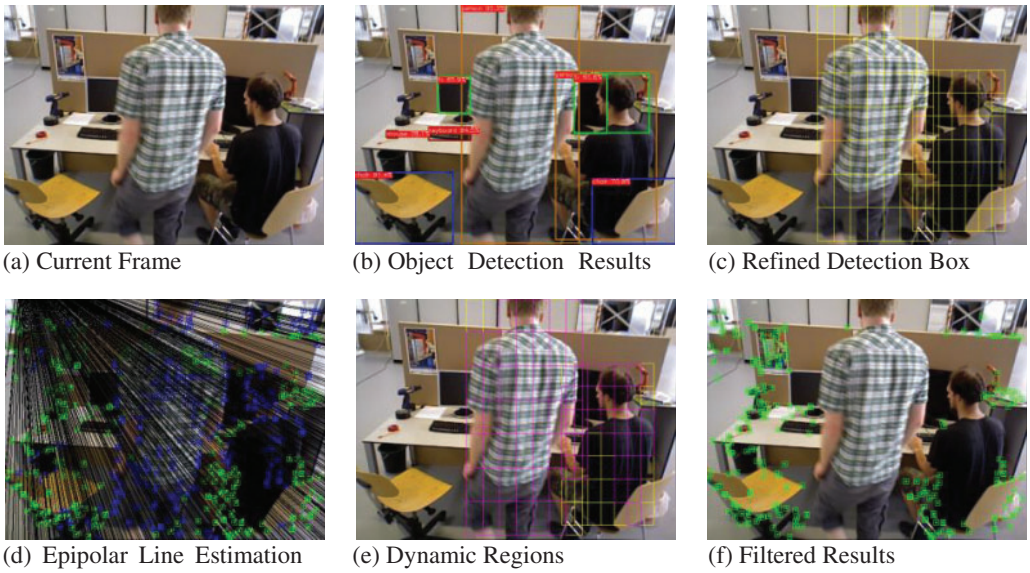
In the precise region removal module, we adopt two strategies to achieve accurate removal: the Precise Dynamic Region Search Strategy and the Dynamic Region Probability Propagation Strategy. The Precise Dynamic Region Search Strategy is designed to address the issue of dynamic SLAM based on object detection, which lacks accurate removal and can easily lead to tracking loss or missed removals. Compared to the related algorithm CDS-SLAM, this method can more accurately locate dynamic regions in the current frame, rather than roughly splitting the object detection box in two for coarse judgment. More precise dynamic region determination directly leads to more accurate tracking performance, as confirmed in our experiments.

The Dynamic Region Probability Propagation Strategy is designed to address the issue of missing semantic information. Dynamic objects in some frames may not be effectively detected by the object detection network, leading to a loss of semantic information between frames. This can cause the entire system to be disturbed by dynamic feature points in dynamic environments, reducing accuracy and even resulting in tracking loss. This type of method is used in RDS-SLAM. However, unlike our

method, RDS-SLAM propagates dynamic probabilities based on map points, while our approach directly propagates dynamic probabilities within frame regions. Our method updates dynamic probabilities faster, though the duration for which the probability of each region is maintained is shorter. The updated SG-SLAM uses thresholds to update dynamic values of detection regions, a concept similar to ours. In subsequent experiments, we found that SG-SLAM's method caused direct tracking loss due to insufficient removal precision, while our removal method did not have this issue.

3.2 Precise Dynamic Region Search Strategy

Our method can accurately remove dynamic object boxes and retain static areas. First, we use the optical flow-epipolar method to track all feature points from the previous frame and project them onto the current frame. We use LK optical flow (Lucas-Kanade Optical Flow) for tracking [25]. The effect after optical flow tracking is shown in the first row and first column in Fig. 2.



(a) Current Frame (b) Object Detection Results (c) Refined Detection Box

(d) Epipolar Line Estimation (e) Dynamic Regions (f) Filtered Results

Figure 2: Complete culling process

Next, we use the semantic thread to obtain semantic information for the current frame, identifying potential dynamic objects. We consider all animals and vehicles as potential dynamic objects. The detection results are shown in the first row and second column in Fig. 2. After obtaining the optical flow vectors, we follow Algorithm 1 to perform the specific removal process. It should be noted that our *Distancethreshold* is set to 2, meaning a distance of 2 pixels.

Algorithm 1: Identify dynamic regions based on optical flow

Input: Optical flow vectors FV, Endpoint $point1$, $point2$,
Distance threshold *Distancethreshold*, Rect of dynamic objects RO

Output: Dynamic regions DR

(Continued)

Algorithm 1 (continued)

Extract coordinates from *FV* to get *coords*1
Compute optical flow to get *good_new* and *good_old*
Compute Fundamental Matrix *F* using RANSAC and epilines for *good_old*
for each (*new*, *old*) in (*good_new*, *good_old*) **do**
    Calculate distance *dist* from *new* to epiline
    Mark *new* as blue if *dist* > *Distancethreshold*, else green
endfor
Initialize *filtered_rects*
for each *rect* in *all_rects* **do**
  if then
      Add *rect* to *Dynamicposbility* (*rect*) ≤ 0.5 *filtered_rects*
  endif
endfor
Return *filtered_rects* as dynamic regions *DR*

Algorithm 1 describes in detail the method and process for calculating dynamic regions. I will now elaborate on the specific computational details of this algorithm. Firstly, Lines 1 through 7 of Algorithm 1 employ the classic optical flow-epipolar line method to detect dynamic points within the frame. This approach tracks the dynamic feature points extracted from the previous frame using LK optical flow and applies RANSAC (Random Sample Consensus) to filter out optical flow vectors, yielding the tracking results for the current frame [26]. Then, the calculated fundamental matrix is used to compute the epipolar lines and project them onto the current frame. The tracking results are evaluated by measuring the distance to the corresponding epipolar lines to determine if they are dynamic. It should be clearly stated that the RANSAC used here is not the RANSAC algorithm for feature point matching. In this case, RANSAC is used to remove optical flow vectors that deviate significantly from the motion model, thereby obtaining a relatively accurate set of optical flow vectors for the subsequent calculation of a more precise fundamental matrix. This method can filter out some dynamic feature points and dynamic regions, but relying solely on such geometric methods is not accurate enough. Therefore, this method must be combined with semantic information. As a result, systems like DS-SLAM, RDS-SLAM, and CDS-SLAM all adopt a similar approach of combining the optical flow-epipolar method with semantic information for more precise removal. It is also important to clarify that we did not use the method employed in DS-SLAM and CDS-SLAM for this step. DS-SLAM extracts new feature points from the previous frame and performs tracking, but our testing showed suboptimal results with this approach. Therefore, we utilized the precise feature points extracted from the previous frame to carry out this operation, resulting in a more effective differentiation of dynamic points. The second row and first column in Fig. 2 shows the final removal effect and the epipolar lines.

Now, suppose we subdivide all the object detection bounding boxes to obtain sub-bounding boxes. The total set of all sub-bounding boxes is referred to as the potential dynamic regions, which we represent using Eq. (1):

$$D = \{R_1, R_2, \ldots, R_n\}. \tag{1}$$

Here, D represents the total set of all potential dynamic sub-bounding boxes. We can retrieve the potential tracked points in a sub-region by using the optical flow-epipolar line tracking results

mentioned earlier. We refer to these points as P, defined by Eq. (2):

$$P = \{p_1, p_2, \ldots, p_n\}. \tag{2}$$

The distance D between a pixel in the current frame and its corresponding epipolar line is defined by Eq. (3). If this distance exceeds a certain threshold, the point is flagged as a potential dynamic point.

In this case, the dynamic and static tracked points within the region can be distinguished by the distance to the epipolar line, denoted as d_i, as shown in Eq. (3). Here, p_s represents the set of static tracked points within the sub-region, and p_d represents the set of dynamic tracked points within the region.

$$p_s = \{p_i \in P | d_i \leq \tau\}, p_d = \{p_i \in P | d_i > \tau\}. \tag{3}$$

The collection of points within the sub-region can be expressed in the Eq. (4):

$$R_i = \{p_{d1}, p_{d2}, \ldots, p_{dk}, p_{s1}, p_{s2}, \ldots, p_{sm}\}. \tag{4}$$

We determine whether an area is likely to be dynamic by analyzing the proportion of points within that area, as illustrated by the Eq. (5):

$$DP_i = \frac{m}{k}, \tag{5}$$

where DP_i is the probability of the area being dynamic, m is the number of dynamic tracking points within the area, and k is the total number of tracking points in the area. We set θ as the probability threshold, with a value of 0.5, to determine whether the area is dynamic. The decision is made using the Eq. (6). The final effect is shown in the second row and second column in Fig. 2.

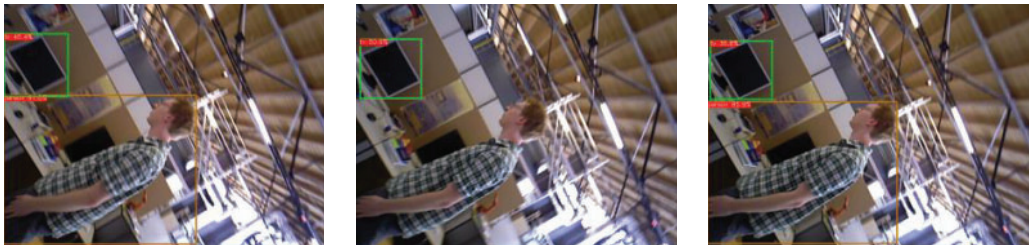
$$D_{dynamic} = \{R_i \in D \mid P_i > \theta\}. \tag{6}$$

This strategy enables us to accurately complete the removal task by effectively balancing precision and efficiency. By doing so, we mitigate the risks of over-removal, which could eliminate essential static points, and under-removal, which might leave dynamic points that could compromise tracking accuracy. Consequently, the system's overall tracking performance is significantly enhanced. Furthermore, we avoid relying on semantic segmentation, which is often computationally intensive and time-consuming, to obtain precise semantic information. Instead, our approach streamlines the process, thereby improving real-time performance. This optimization ensures that the system operates swiftly and efficiently, maintaining high accuracy without sacrificing speed. In essence, our method not only preserves critical static points and eliminates dynamic points effectively but also boosts the system's real-time capabilities, making it more robust and responsive in dynamic environments.

3.3 Dynamic Region Probability Propagation Strategy

Although our precise removal algorithm can accurately and effectively remove dynamic region feature points, it heavily and significantly relies on the semantic information derived from the current frame. In certain frames, the semantic information is completely and utterly lost. As demonstrated and shown in Fig. 3, these three frames are consecutive and successive in the dataset. All three images are of 640 × 480 resolution. However, in this rotating situation or scenario, the object detection results are far from ideal and not up to par. The first and last frames detect dynamic objects quite well and efficiently, but the middle frame fails and does not perform as expected. This greatly and severely

affects the removal of dynamic objects because our Algorithm 1 completely and totally fails. It should be noted that such plane rotation scenarios frequently occur in SLAM environments. The occurrence of plane rotation leads to a decline in the performance of deep learning networks, resulting in the loss of semantic information. Similar reports have been made in both RDS-SLAM and the latest NGD-SLAM. Addressing the issue of missed dynamic region detection caused by the loss of semantic information is an important task.



(a) Previous Frame (b) Semantic Lost Frame (c) Next Frame

Figure 3: Loss of semantic information

Dynamic points on the human body cause mismatches, leading to significant system performance degradation or even complete tracking loss. We need a method or approach to effectively propagate and transfer dynamic information from frames with sufficient semantic information. This will ensure that frames with detection failure still receive some amount of semantic information, thus mitigating and alleviating this problem.

We propose and introduce Algorithm 2. This innovative method utilizes Bayesian Probability to retain and keep some of the semantic information from Algorithm 1 and effectively pass it to the current frame for a second dynamic feature point filter. We firmly believe and hypothesize that each region on the camera's frame is fixed and stable, and that dynamic objects leave some trace of dynamic information in these regions. Probability propagation helps us to accurately capture and continually update this vital information. First and foremost, we need partial information from Algorithm 1. This crucial information includes all the dynamic sub-regions and the comprehensive feature point set of the current frame after it has been filtered by Algorithm 1.

Algorithm 2: Dynamic region probability filtering

Input: feature points after initial dynamic point removal RP,
rectangles from previous algorithm DR,
Probabilities from previous 10 frames P_n
Output: Filtered feature points FP
Divide the current frame into 10 rows and 10 columns, creating 100 *dr_rects*
for each *dr_rect* **do**
    occupied_area ← Calculate the area of *dr_rect* occupied by DR
    if *occupied_area* > *threshold* **then**
        current_probability ← 0.9
    else

(Continued)

Algorithm 2 (continued)

```
7:                    current_probability ← 0.1
8:                endif
9:                // Update Bayesian Probability
10:               prior ← Average of previous 10 frames' probabilities for this dr_rect
11:               likelihood ← current_probability
12:               posterior = Cacluatetheprobabilities(prior,likelihood)
13:               probabilities[dr_rect] ← posterior
14:           endfor
15:           for each dr_rect do
16:               if probabilities[dr_rect] > threshold then
17:                 Remove feature points in this dr_rect
18:               endif
19:            endfor
```

In the first line of Algorithm 2, we divide the current frame into 10 × 10 regions. In each subsequent frame, we will calculate the dynamic probability for each region. In Lines 3 to 8 of Algorithm 2, we calculate the intersection area between each region and the dynamic sub-regions from Algorithm One. The threshold in the fourth line is set to 0.65. Setting this value too low can cause the method to fail, while setting it too high can lead to incorrect judgments of dynamic areas, resulting in the removal of too many feature points. If the intersection area is more than half of the region's area, the region is considered a potential dynamic region. The region in the current frame is assigned a value of 0.9. In Lines 10 to 13 of Algorithm 2, we use Bayesian Probability to calculate the potential dynamic probability of the current frame with the dynamic probabilities of the same regions from the previous ten frames. This is the number we believe is most suitable for the testing situation. We tried using 15 frames, which improved the system's performance when faced with semantic information loss but also led to the issue of residual dynamic feature points in ordinary situations. This method is similar to adding damping to all dynamic points, making them less responsive to external changes. We recommend increasing this value when experiencing significant semantic information loss while setting it to a value lower than 5 in regular circumstances. If there is no loss of semantic information at all, this method may be considered unnecessary. In our algorithm, the formula for updating the probability value is shown in Eq. (7).

$$P\,(\text{posterior}) = \frac{P\,(\text{likelihood}) \cdot P\,(\text{prior})}{P\,(\text{evidence})}. \tag{7}$$

However, in the dynamic region probability removal, the actual application is simpler. Since P (evidence) is a normalization constant, it can be ignored. We only need to compare the relative posterior probabilities. The formula is shown in Eqs. (8)–(10).

$$P\,(\text{posterior}) = \frac{P\,(A)}{P\,(A) + P\,(B)}, \tag{8}$$

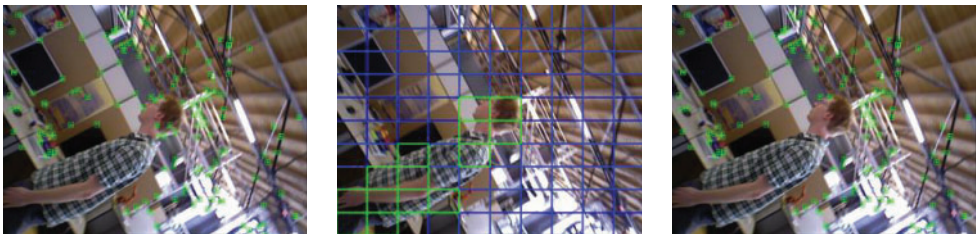
$$P\,(A) = P\,(\text{likelihood}) \cdot P\,(\text{prior}), \tag{9}$$

$$P\,(B) = (1 - P\,(\text{likelihood})) \cdot (1 - P\,(\text{prior})). \tag{10}$$

In Lines 15 to 19 of Algorithm 2, we rigorously compare the calculated P (posterior) with a set threshold to accurately identify and find the dynamic regions. Subsequently, we proceed to remove

the dynamic points in these identified regions. This method effectively supplements and compensates for the areas missed by Algorithm 1. Because our probability update method relies heavily on prior semantic information, the regions found and identified by Algorithm 2 usually match the dynamic regions observed in the previous frames, thereby compensating for the loss of semantic information.

The Fig. 4 provides a visual representation of the corrected removal for a frame where semantic acquisition has failed. In this specific frame, semantic information is completely and entirely lost, leading to the retention of feature points on the person's head. It is important to note that the person is moving in the surrounding frames. The leftmost image illustrates the failed removal due to the lost semantic information. The middle image shows the dynamic probability regions as identified by Algorithm 2 using information from previous frames. The rightmost image depicts the completed and successful removal. This clearly proves and demonstrates the effectiveness of our method in handling situations where semantic information is not available or has failed.



(a) Initial Filtering (b) Probability-Based Dynamic Region Prediction (c) Further Filtering

Figure 4: Complete culling process. The leftmost image is the result after removal by Algorithm 1, where the failure of semantic information has led to removal failure. The middle image shows the potential dynamic regions identified by Algorithm 2 in this frame. The rightmost image is the result after removal

4 Experiments

4.1 Experiment Introduction

In this experiment, we rigorously tested dynamic SLAM on the widely recognized and extensively used TUM-RGBD dataset. The TUM-RGBD dataset is commonly and frequently used for SLAM and 3D reconstruction research, making it a standard benchmark in the field. It includes data from a diverse array of various scenes and movement patterns, providing a comprehensive testing ground. We specifically and particularly focused on two highly dynamic scenes: W/RPY and W/HALF. These specific scenes contain dynamic objects and complex motion patterns that pose significant challenges to SLAM systems, thereby making them our key and primary test subjects for evaluation and analysis.

In the TUM-RGBD dataset we used, the most important tests are conducted under the W/RPY and W/HALF sequences. These two sequences take place in a highly dynamic environment. In an indoor setting, two people transition back and forth between walking, standing, and sitting positions, and the camera experiences significant rotations. Dynamic SLAM systems with average performance often struggle under these conditions, leading to direct loss of tracking.

In contrast, the other sequences in the dataset feature more controlled camera movements, with less activity from the individuals, primarily remaining seated. However, they still fall within the category of highly dynamic sequences. This particular scenario effectively simulates a stringent dynamic environment, often proving to be more challenging than many real-world SLAM scenarios.

Given these characteristics, we chose this dataset for our testing. It provides valuable insights into how dynamic SLAM systems perform under realistic yet demanding conditions, ensuring that our evaluations are rigorous and meaningful. By focusing on these sequences, we aim to assess the robustness and reliability of our approach in scenarios that closely mimic the complexities encountered in actual applications.

To thoroughly evaluate and assess the performance of dynamic SLAM algorithms, we employed and utilized two main error metrics: Absolute Trajectory Error (ATE) and Relative Pose Error (RPE). ATE measures and quantifies the global error between the estimated trajectory and the true, actual trajectory. On the other hand, RPE evaluates and determines the error in the estimated pose changes over short time intervals compared to the actual, real changes. For a more accurate, detailed, and robust performance evaluation, we used Root Mean Square Error (RMSE) to statistically analyze these error metrics. We meticulously designed comprehensive ablation and test experiments to rigorously assess and evaluate the effectiveness and efficiency of each individual module in our dynamic SLAM system. It is worth noting that almost all SLAM systems use ATE and RPE as metrics for system evaluation in research papers. Lower ATE and RPE values indicate better system performance. The evaluation methods of existing common visual SLAM systems also rely on these metrics for result calculation. Using these metrics to test on public datasets is a widely accepted practice in the field and forms the basis for conducting fair evaluations of systems as much as possible.

We thoroughly compare our dynamic SLAM system with several prominent and widely recognized systems, including ORB-SLAM3, DynaSLAM, DS-SLAM, CDS-SLAM, Lccrf-SLAM [27], Crowd-SLAM, Blitz-SLAM [28], and SG-SLAM, utilizing ATE and RPE data for a comprehensive performance evaluation. ORB-SLAM3 serves as our baseline for comparison. DynaSLAM is considered a classic dynamic SLAM system and still ranks very high in terms of tracking accuracy. DS-SLAM combines the use of epipolar optical flow with semantic SLAM techniques. CDS-SLAM and SG-SLAM are the latest and most advanced object detection-based SLAM systems, featuring novel and innovative removal strategies to ensure accurate feature point removal. Crowd-SLAM introduced the concept of precise removal into the field. Lccrf-SLAM and Blitz-SLAM are also regarded as classic works in the domain of dynamic SLAM. Our test hardware platform is comprised of an Intel I7-12700h CPU, a GTX-1070TI GPU, and 16 GB of RAM.

4.2 Result

We conducted detailed and thorough tests on the TUM-RGBD dataset and performed comprehensive and extensive ablation experiments. Additionally, we compared our dynamic SLAM system with the most advanced and cutting-edge works in the field. Our test data, which are presented and illustrated in Tables 1–4, provide a clear overview of the performance metrics. In several highly dynamic scenes within the TUM-RGBD dataset, our dynamic SLAM system significantly and markedly reduced both trajectory error (ATE) and pose error (RPE), thereby demonstrating excellent and outstanding performance. This remarkable performance is attributed to our precise and meticulous strategy for removing dynamic objects, which allows the system to effectively handle dynamic interference and subsequently improve trajectory estimation accuracy. Fig. 5 provides a detailed illustration of how our system processes frames in a meticulous and detailed manner. The final removal effect achieved by our system is very close to the high precision obtained through semantic segmentation, showcasing and highlighting its high level of accuracy. One of the key advantages and strengths of our approach lies in the fact that traditional segmentation-based dynamic SLAM systems are heavily and significantly reliant on the quality of the segmentation process, which can be a limiting factor.

Table 1: ATE of TUM-RGBD 1

Scence	O3		Dyna		DS		CDS		OURS	
	rmse	std	rmse	std	rmse	std	rmse	std	rmse	std
half	0.424	0.346	0.029	0.015	0.030	0.026	0.019	0.010	0.020	0.012
rpy	0.726	0.311	0.035	0.019	0.044	0.378	0.053	0.031	0.032	0.016
static	0.022	0.017	0.006	0.003	0.008	0.007	0.005	0.002	0.005	0.003
xyz	0.825	0.575	0.016	0.008	0.024	0.019	0.013	0.008	0.010	0.007

Table 2: ATE of TUM-RGBD 2

Scence	Lccrf		Crowd		Blitz		SG		OURS	
	rmse	std	rmse	std	rmse	std	rmse	std	rmse	std
half	0.028	0.015	0.026	–	0.026	0.013	0.027	0.013	0.020	0.012
rpy	0.046	0.034	0.044	–	0.036	0.022	0.032	0.019	0.032	0.016
static	0.011	0.008	0.007	–	0.010	0.005	0.007	0.003	0.005	0.003
xyz	0.016	0.011	0.020	–	0.015	0.008	0.015	0.008	0.010	0.007

Table 3: RPE of TUM-RGBD 1

Scence	O3		Dyna		DS		CDS		OURS	
	rmse	std	rmse	std	rmse	std	rmse	std	rmse	std
half	0.143	0.739	0.028	0.014	0.030	0.026	0.018	0.009	0.017	0.013
rpy	0.124	0.533	0.044	0.026	0.150	0.094	0.035	0.024	0.026	0.022
static	0.119	0.619	0.008	0.004	0.010	0.009	0.006	0.003	0.007	0.004
xyz	0.087	0.054	0.021	0.011	0.033	0.024	0.017	0.011	0.012	0.010

Table 4: RPE of TUM-RGBD 2

Scence	Lccrf		Crowd		Blitz		SG		OURS	
	rmse	std	rmse	std	rmse	std	rmse	std	rmse	std
half	0.035	0.024	0.037	–	0.025	0.012	0.028	0.015	0.017	0.013
rpy	0.050	0.046	0.065	–	0.047	0.028	0.045	0.026	0.026	0.022
static	0.014	0.011	0.010	–	0.013	0.007	0.010	0.005	0.007	0.004
xyz	0.012	0.007	0.025	–	0.020	0.010	0.019	0.010	0.012	0.010

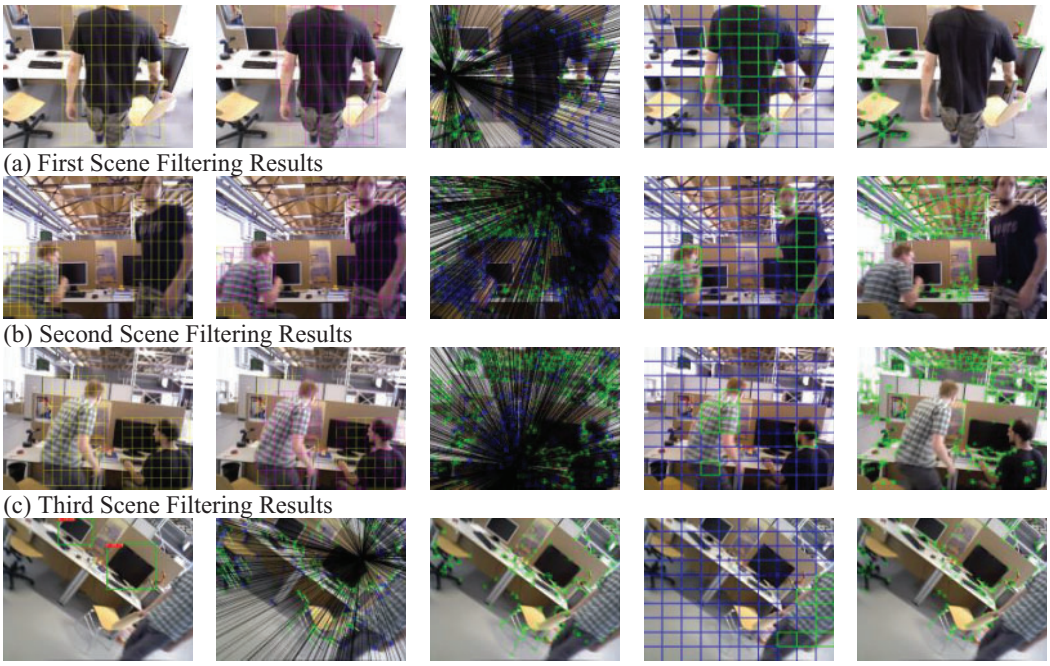


Figure 5: Demonstration of part of the scenario in TUM-RGBD. Each row represents a complete removal process

The last row of the figure shows the case of semantic loss. The current frame's semantic information is completely lost, causing Algorithm 1 removal strategy to fail. This leads to the failure to remove dynamic feature points on the human body. In our tests, such removal failure causes serious issues. For example, the system may lose track. Although the system can use relocalization to recover the lost track, this greatly reduces accuracy. Algorithm 2 uses semantic information from multiple previous frames to update the current frame's dynamic regions. Even without semantic information, this region is still partially recognized as dynamic after incorporating prior probabilities. The third column in the last row shows the removal result of Algorithm 1. The last column shows the removal result of Algorithm 2. Although Algorithm 2's removal ability is limited, it significantly improves the issue of losing dynamic regions due to semantic loss.

In Fig. 5b, c, we provide a detailed demonstration of cases where multiple dynamic objects appear in the scene. Since our algorithm can segment and detect each object detection box, the removal effectiveness remains unaffected. It is important to clarify that Algorithm 1 performs dynamic detection on sub-regions. If there is an overlap of dynamic regions, regardless of whether the overlapping area is labeled as a potential dynamic object, Algorithm 1 will only use the ratio of potential dynamic points to the total number of feature points in the sub-region as the basis for dynamic region classification. Therefore, our Algorithm 1 can handle these dynamic scenarios effectively.

In the W/RPY sequence, such camera rotation still exists and persists. However, due to the complex and multifaceted interference from dynamic objects, our system performs exceptionally well in both ATE and RPE with our effective and efficient dynamic object removal strategies. Especially in scenes with numerous dynamic object interferences, our system can better and more accurately filter out these

dynamic interferences, thereby improving trajectory estimation accuracy significantly. The specific trajectory plots and the ATE visual comparison, as clearly shown and illustrated in Fig. 6, provide concrete evidence that our dynamic SLAM system has significant and substantial advantages in highly dynamic scenes. It demonstrates higher accuracy and robustness, particularly in environments with complex dynamic object interference. This robustness and high accuracy are especially evident in challenging scenarios where many dynamic objects are present, further showcasing the effectiveness of our approach in handling such intricate and dynamic environments. Overall, our system's ability to maintain high performance in these conditions highlights its superiority and advanced capabilities compared to other existing methods.

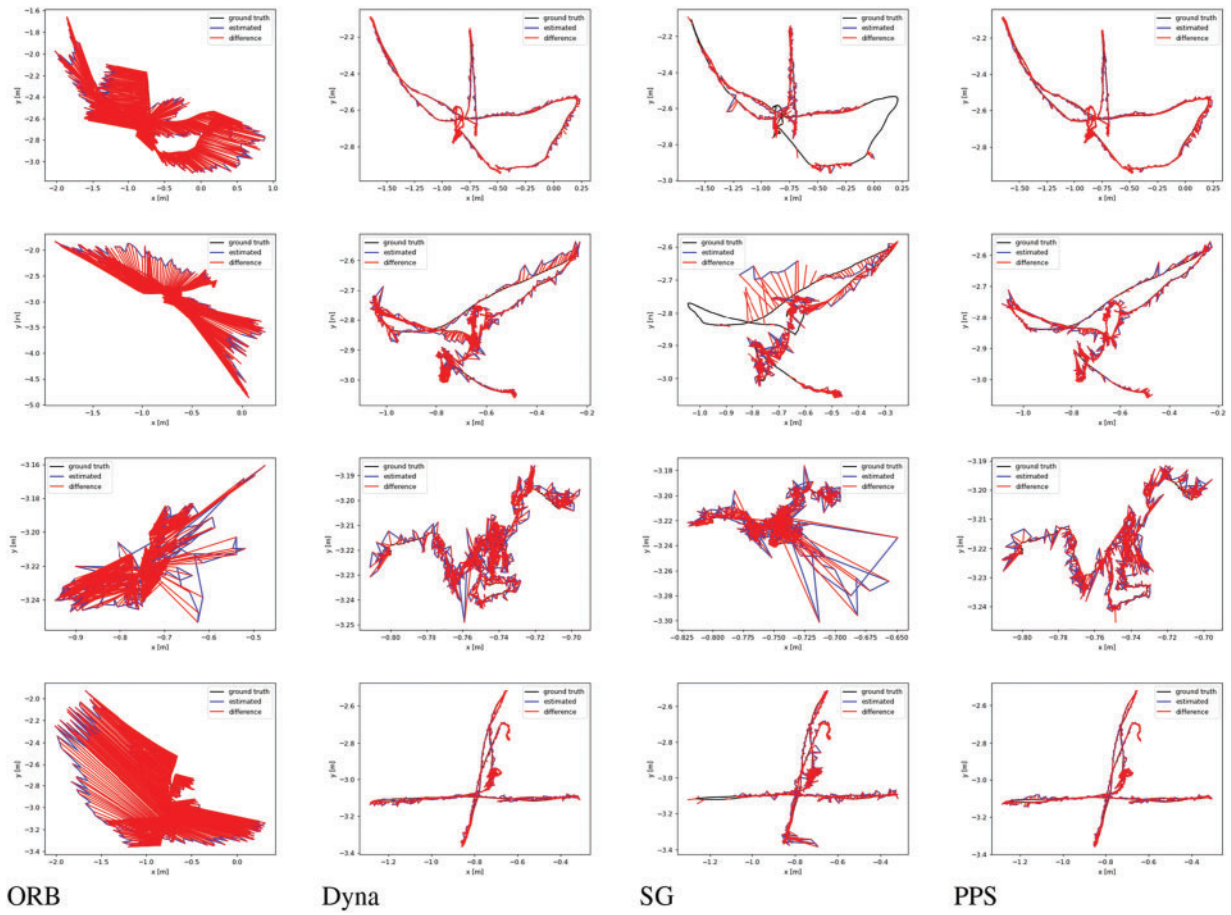


Figure 6: Trajectories of different systems under TUM-RGBD. The trajectory plots for ORB-SLAM3, DynaSLAM, SG-SLAM, and OURS across different sequences from left to right

Each column in Fig. 6 represents the performance of a visual SLAM system in dynamic environments. The red line indicates the error, while the blue line shows the trajectory obtained by our system. Ideally, the red line should closely match or even completely overlap with the ground truth, indicating minimal error and a trajectory that closely aligns with the ground truth. ORB-SLAM3, which we used as the baseline for improvement, performs poorly in dynamic environments due to its lack of ability to handle dynamic interference. DynaSLAM, a classic dynamic SLAM system based on instance segmentation, generally achieves higher accuracy than object detection-based methods

because of its more precise dynamic region removal. Object detection methods often suffer from severe frame loss or low accuracy. In our tests, SG-SLAM struggled with highly dynamic RPY and HALF sequences. When calculating ATE, we did not include deviations from lost frames, which makes SG-SLAM's results appear better than they actually are. The trajectories we measured are similar to those in the SG-SLAM paper, with significant trajectory loss. For example, in the third column of the first and second rows, SG-SLAM, similar to our method, completely deviates from the original ground truth, resulting in trajectory loss due to imprecise dynamic object removal. Our method, however, achieves more accurate trajectories, and the removal regions selected by our strategy are already very close to the precision achieved by segmentation. Such a precise removal strategy enhances the accuracy of our system.

4.3 Ablation Study

We conducted ablation experiments to evaluate the performance impact of each system module. We focused on testing our system with two designs: one using the complete precise pruning strategy and another removing all feature points within the dynamic object detection box. This was done to determine the impact of the pruning strategy on tracking accuracy. Since our method consistently outperformed ORB-SLAM3 in previous tests, we no longer include ORB-SLAM3 comparisons here.

Table 5 presents the consolidated test data. Results show that without precise pruning, the system loses track in highly dynamic environments and fails to complete sequences. In less dynamic environments, complete removal also causes tracking loss, though the system can relocalize, albeit with significant performance degradation. Thus, our precise pruning method is crucial for maintaining accuracy.

Table 5: Ablation study

Scence	O3	Ours1		Ours2		Ours3	
	rmse	rmse	Improvement	rmse	Improvement	rmse	Improvement
half	0.424	LOST	–	0.021	95.05%	0.020	95.28%
rpy	0.726	LOST	–	0.034	95.32%	0.032	95.59%
static	0.022	0.005	77.27%	0.004	81.82%	0.005	77.27%
xyz	0.825	0.026	96.85%	0.010	98.79%	0.010	98.79%

Additionally, we observed that using only Algorithm 1 led to higher errors in sequences like W/HALF due to challenges in acquiring semantic information during camera rotation, though it still improved accuracy in dynamic scenes. Algorithm 2 provided further improvement, particularly in sequences with rotational dynamics like W/HALF and W/RPY, by enhancing dynamic region handling.

We also found some issues with using Algorithm 2. In sequences without semantic loss, like W/static, introducing Algorithm 2 increased system errors to some extent. We believe this is mainly because the removal regions planned by Algorithm 2 have some delay due to probability calculations in some frames. This leads to extra removal in static regions. In highly dynamic environments, more removal improves performance. However, in less dynamic sequences, performance decreases.

4.4 Real Time Test

Real-time performance is a key metric for SLAM. We also tested the real-time performance of our system's tracking thread. We compared it with some classic or real-time SLAM systems, including DynaSLAM, CDS-SLAM, ORB-SLAM3, DGS-SLAM [29], PR-SLAM [30], Lccrf-SLAM, and Blitz-SLAM. The specific test data are shown in the table. We tested the average tracking time of our system on the TUM-RGBD dataset and compared it with other excellent SLAM systems.

Our baseline, ORB-SLAM3, has a tracking time of 20.92 ms on our computer. Both our system and the advanced CDS-SLAM add a semantic thread to ORB-SLAM3. Our pruning strategy is less time-consuming. Additionally, CDS-SLAM uses OpenCV for feature extraction and corner detection during optical flow-epipolar operations, whereas we track directly from extracted feature points, avoiding redundant operations. Table 6 shows our test data. ORB-SLAM3, DynaSLAM, and CDS-SLAM were tested on our hardware with a unified standard, while other systems use data from their original papers. The most time-consuming part of semantic-based dynamic SLAM is deep learning inference, where our GPU lacks a significant advantage. Nonetheless, our system achieves competitive real-time performance, meeting the standards of leading systems.

Table 6: Real-time test

Systems	Cost	CPU	GPU
DGS-SLAM	38.53 ms	–	RTX 3070
PR-SLAM	50–60 ms	R5-3600	RTX 3070
Lccrf-SLAM	51.73 ms	I9-9900K	–
Blitz-SLAM	81 ms	I5-6500	–
ORB-SLAM3	20.92 ms	Intel i7-12700h	GTX1070TI
DynaSLAM	472.42 ms	Intel i7-12700h	GTX1070TI
CDS-SLAM	42.21 ms	Intel i7-12700h	GTX1070TI
OURS	34.26 ms	Intel i7-12700h	GTX1070TI

We also measured the time taken by each algorithm in our system on the current platform. Using YOLOv8-n for semantic information adds about 6 ms on average. Using Algorithm 1 adds another 6 ms, and Algorithm 2 adds about 3 ms. Since precise pruning relies almost entirely on geometric strategies and semantic information is accelerated with TensorRT, our system maintains excellent real-time performance and stays ahead among many systems.

5 Conclusion

In this paper, we propose a novel dynamic SLAM system. We employ an innovative pruning strategy that allows the system to achieve semantic segmentation-level dynamic feature point removal using only object detection bounding boxes. To address potential semantic information loss, we use a probabilistic approach to propagate and update semantic information across different regions of the frames, ensuring that even frames with missing semantic information can still obtain some semantic context. However, in extreme cases of prolonged semantic information loss, this probabilistic propagation method will also fail. We believe that to further enhance system performance, a more lightweight and powerful model for semantic information extraction is needed to fundamentally solve the problem.

Acknowledgement: The authors are highly thankful to the National Natural Science Foundation of China, to the Guangxi Natural Science Foundation under Grant, to the Innovation Fund of Chinese Universities Industry-University-Research, to the Special Research Project of Hechi University. Guangxi Colleges and Universities Key Laboratory of AI and Information Processing (Hechi University), and to the Central Foreign Trade and Economic Development Special Project, Education Department of Guangxi Zhuang Autonomous Region.

Funding Statement: The authors are highly thankful to the National Natural Science Foundation of China (No. 62063006), to the Guangxi Natural Science Foundation under Grant (Nos. 2023GXNSFAA026025, AA24010001), to the Innovation Fund of Chinese Universities Industry-University-Research (ID: 2023RY018), to the Special Guangxi Industry and Information Technology Department, Textile and Pharmaceutical Division (ID: 2021 No. 231), to the Special Research Project of Hechi University (ID: 2021GCC028), and to the Key Laboratory of AI and Information Processing, Education Department of Guangxi Zhuang Autonomous Region (Hechi University), No. 2024GXZDSY009.

Author Contributions: The authors confirm contribution to the paper as follows: study conception and design: Wei Qian, Jiansheng Peng; data collection: Jiansheng Peng, Hongyu Zhang; analysis and interpretation of results: Jiansheng Peng, Wei Qian; draft manuscript preparation: Wei Qian, Jiansheng Peng. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

[1] A. Kazerouni, L. Fitzgerald, G. Dooly, and D. Toal, "A survey of state-of-the-art on visual SLAM," *Expert Syst. Appl.*, vol. 205, no. 6, 2022, Art. no. 117734. doi: 10.1016/j.eswa.2022.117734.

[2] M. Barros, M. Michel, Y. Moline, G. Corre, and F. Carrel, "A comprehensive survey of visual SLAM algorithms," *Robotics*, vol. 11, no. 1, 2022, Art. no. 24. doi: 10.3390/robotics11010024.

[3] J. C. V. Soares, M. Gattass, and M. A. Meggiolaro, "Crowd-SLAM: Visual SLAM towards crowded environments using object," *J. Intell. Robot. Syst.*, vol. 102, no. 1, 2021, Art. no. 50. doi: 10.1007/s10846-021-01414-1.

[4] Q. Zhang and C. Li, "Semantic SLAM for mobile robots in dynamic environments based on visual camera sensors," *Meas. Sci. Technol.*, vol. 34, no. 8,2023, Art. no. 085202. doi: 10.1088/1361-6501/acd1a4.

[5] W. Qian, J. S. Peng, and H. Y. Zhang, "DFD-SLAM: Visual SLAM with deep features in dynamic environment," *Appl. Sci.*, vol. 14, no. 11, 2024, Art. no. 4949. doi: 10.3390/app14114949.

[6] Y. Liu and J. Miura, "RDS-SLAM: Real-time dynamic SLAM using semantic segmentation methods," *IEEE Access*, vol. 9, pp. 23772–23785, 2021. doi: 10.1109/ACCESS.2021.3050617.

[7] Y. Zhang, "NGD-SLAM: Towards real-time SLAM for dynamic environments without GPU," 2024, *arXiv:2405.07392*.

[8] C. Campos, R. Elvira, J. J. G. Rodriguez, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM3: An accurate open-source library for visual, visual-inertial, and multimap SLAM," *IEEE Trans. Robot.*, vol. 37, no. 6, pp. 1874–1890, 2021. doi: 10.1109/TRO.2021.3075644.

[9] Ultralytics, "YOLOv8," Accessed: Jan. 10, 2023. [Online]. Available: https://github.com/ultralytics/ultralytics
[10] E. Jeong, J. Kim, S. Tan, J. Lee, and S. Ha, "Deep learning inference parallelization on heterogeneous processors with tensorrt," *IEEE Embed. Syst. Lett.*, vol. 14, no. 1, pp. 15–18, 2021. doi: 10.1109/LES.2021.3087707.
[11] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," presented at the IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Vilamoura-Algarve, Portugal, Oct. 7–12, 2012, pp. 573–580. doi: 10.1109/IROS.2012.6385773.
[12] J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-time single camera SLAM," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 6, pp. 1052–1067, 2007. doi: 10.1109/TPAMI.2007.1049.
[13] G. Lein and D. Murray, "Parallel tracking and mapping for small AR workspaces," presented at the 6th IEEE/ACM Int. Symp. Mixed Augmen. Real. (ISMAR), Nara, Japan, Nov. 13–16, 2007, pp. 225–234. doi: 10.1109/ISMAR.2007.4538852.
[14] T. Pire, T. Fischer, G. Castro, P. De Cristóforis, J. Civera and J. J. Berlles, "S-PTAM: Stereo parallel tracking and mapping," *Robot Auton. Syst.*, vol. 94, no. 5, pp. 80–93, 2017. doi: 10.1016/j.robot.2017.03.019.
[15] G. Hu, S. Huang, L. Zhao, A. Alempijevic, and G. Dissanayake, "A robust RGB-D SLAM algorithm," presented at the IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Vilamoura-Algarve, Portugal, Oct. 7–12, 2012, pp. 1714–1719. doi: 10.1109/IROS.2012.6386103.
[16] T. Qin, P. Li, and S. Shen, "VINS-Mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Trans. Robot.*, vol. 34, no. 4, pp. 1004–1020, 2018. doi: 10.1109/TRO.2018.2853729.
[17] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM: A versatile and accurate monocular SLAM system," *IEEE Trans. Robot.*, vol. 31, no. 5, pp. 1147–1163, 2015. doi: 10.1109/TRO.2015.2463671.
[18] B. Bescos, J. M. Facil, J. Civera, and J. Neira, "DynaSLAM: Tracking, mapping, and inpainting in dynamic scenes," *IEEE Robot. Autom. Lett.*, vol. 3, no. 4, pp. 4076–4083, 2018. doi: 10.1109/LRA.2018.2860039.
[19] C. Yu *et al.*, "DS-SLAM: A semantic visual SLAM towards dynamic environments," presented at the IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Madrid, Spain, Oct. 1–5, 2018, pp. 1168–1174. doi: 10.1109/IROS.2018.8593691.
[20] R. Mur-Artal and J. Tardós, "ORB-SLAM2: An open-source slam system for monocular, stereo, and RGB-D cameras," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, 2017. doi: 10.1109/TRO.2017.2705103.
[21] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," presented at the IEEE Int. Conf. Comput. Vision (ICCV), Venice, Italy, Oct. 22–29, 2017, pp. 2961–2969.
[22] S. Cheng, C. Sun, S. Zhang, and D. Zhang, "SG-SLAM: A real-time RGB-D visual SLAM toward dynamic scenes with semantic and geometric information," *IEEE Trans. Instrum. Meas.*, vol. 72, no. 1, pp. 1–12, 2022. doi: 10.1109/TIM.2022.3228006.
[23] Tencent, "NCNN," Accessed: Jul. 24, 2024. [Online]. Available: https://github.com/Tencent/ncnn
[24] M. Quigley, K. Conley, and B. Gerkey, "An open-source robot operating system," presented at the IEEE Int. Conf. Robot. Autom. (ICRA), Kobe, Japan, May 12–17, 2009, vol. 3.2, p. 5.
[25] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proc. DARPA Image Understanding Workshop*, Washington, DC, USA, 1981, pp. 674–679.
[26] M. Fischler and R. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981. doi: 10.1145/358669.358692.
[27] Z. Du, S. Huang, T. Mu, Q. Zhao, R. Martin and K. Xu, "Accurate dynamic SLAM using CRF-based long-term consistency," *IEEE Trans. Vis. Comput. Graph.*, vol. 28, no. 6, pp. 1745–1757, 2020. doi: 10.1109/TVCG.2020.3028218.
[28] Y. Fan, Q. Zhang, Y. Tang, S. Liu, and H. Han, "Blitz-SLAM: A semantic SLAM in dynamic environments," *Pattern Recognit.*, vol. 121, no. 2, 2022, Art. no. 108225. doi: 10.1016/j.patcog.2021.108225.

[29] L. Yan, X. Hu, L. Zhao, Y. Chen, P. Wei and H. Xie, "DGS-SLAM: A fast and robust RGBD SLAM in dynamic environments combined by geometric and semantic information," *Remote Sens.*, vol. 14, no. 3, 2022, Art. no. 795. doi: 10.3390/rs14030795.

[30] H. Zhang, J. Peng, and Q. Yang, "PR-SLAM: parallel real-time dynamic SLAM method based on semantic segmentation," *IEEE Access*, vol. 12, no. 3, pp. 36498–36514, 2024. doi: 10.1109/ACCESS.2024.3373308.