

Enhancing Dynamic SLAM with Point-Line Feature Fusion and Object-Level Tracking

Hao Chen

Sanya Institute of Hunan University of Science and Technology

Chuanguang Zhu

zhuchuanguang@163.com

Sanya Institute of Hunan University of Science and Technology

LiuHai Liao

Sanya Institute of Hunan University of Science and Technology

Research Article

Keywords: Visual SLAM, Dynamic SLAM, Object Tracking, Object Reconstruction, Point-Line Feature Fusion

Posted Date: June 12th, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-8504186/v1>

License:   This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

Enhancing Dynamic SLAM with Point-Line Feature Fusion and Object-Level Tracking.

Hao Chen^{1,2}, Chuanguang Zhu^{1,2}, Liuhai Liao^{1,2}

¹Sanya Institute of Hunan University of Science and Technology, Sanya 572024, China

²School of Earth Sciences and Spatial Information Engineering, Hunan University of Science and Technology, Xiangtan 411201, China

E-mail: zhuchuanguang@163.com

Abstract: Visual Simultaneous Localization and Mapping (SLAM) is essential for autonomous robots to perceive and navigate their environments. However, traditional Visual SLAM systems exhibit degraded performance in weakly textured or highly dynamic scenes. To address these limitations, this study proposes PLO-SLAM, an object-level dynamic Visual SLAM system that integrates point-line feature fusion within the ORB-SLAM3 framework. The system employs optical flow and epipolar constraints to distinguish dynamic objects from static backgrounds, and enhances the DeepSORT algorithm with an ellipsoidal reprojection mechanism to achieve stable cross-frame object associations. Extensive experiments conducted on the TUM RGB-D, BONN, EuRoC, and self-collected datasets demonstrate that PLO-SLAM exhibits strong robustness to illumination variations and texture degradation, achieving state-of-the-art localization accuracy in both static and dynamic environments. The source code is publicly available on GitHub with comprehensive usage guidelines to facilitate replication and evaluation.

Keywords—Visual SLAM, Dynamic SLAM, Object Tracking, Object Reconstruction, Point-Line Feature Fusion

1 Introduction

Simultaneous localization and mapping (SLAM) is a core technology that enables autonomous robots to perceive their surroundings and effectively navigate [1]. Visual SLAM, which uses cameras as its sole sensory modality, provides benefits including lower cost, lighter weight, and richer semantic information, and has consequently been widely adopted in mobile platforms such as unmanned aerial vehicles and service robots[2]. However, conventional Visual SLAM systems generally assume that the scene is static [3-5], which presents significant challenges when applied to dynamic environments containing moving objects. Dynamic elements can disrupt feature tracking and pose estimation, leading to degraded mapping accuracy and reduced robustness in real-world scenarios [6]. Furthermore, issues such as the dependence on prior knowledge and the inherent conflict between feature extraction and moving object removal in weak-texture scenes remain critical challenges that have yet to be fully addressed for Visual SLAM in complex dynamic scenarios.

Most object-based SLAM approaches primarily treat moving objects as sources of interference, overlooking the valuable spatiotemporal information they carry. However, the state and trajectory of moving objects are crucial for path planning, behavioral decision-making, and environment–robot interaction [7]. Ignoring this information embedded in moving objects constrains SLAM’s capability to achieve a comprehensive understanding of the environment and its capacity for high-level semantic mapping.

Strategies for handling dynamic scenes can be generally categorized into two classes. The first category is state-augmentation approaches, which jointly estimate the camera and moving object states to enhance the adaptability to dynamic environments. Although these approaches can achieve high accuracy, they typically incur significant computational loads, rendering them unsuitable for real-time applications. The second category is outlier-rejection approaches, which combine geometric constraints with semantic cues to eliminate dynamic features. These methods

are computationally efficient but often mistakenly discard valid features in highly dynamic or weak-texture scenes, thereby undermining system stability [8, 9].

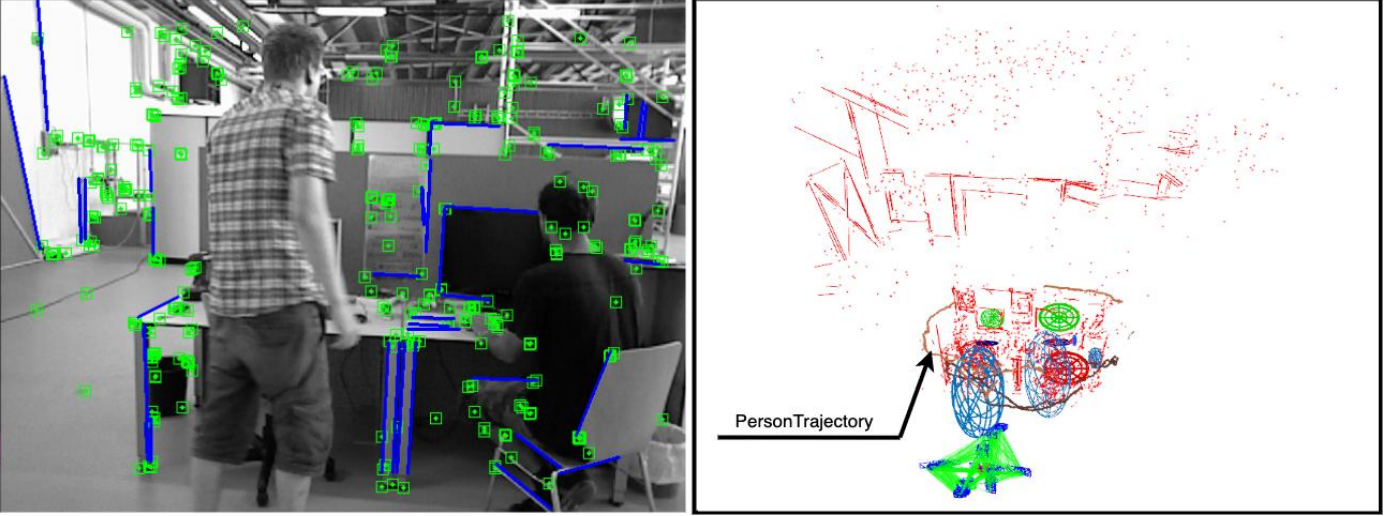


Fig. 1. Left: Point and Line feature culling results; Right: final output of our system on the TUM walking_xyz sequence.

To address these issues, this study presents a dynamic visual SLAM system that integrates scene perception with point–line feature fusion, referred to as PLO-SLAM (Point–Line feature fusion for Object-Based SLAM). An overview of the final output of the proposed system is shown in Fig. 1. The main contributions of this study are summarized as follows:

1. Refining DeepSORT by integrating instance segmentation with ellipsoid reprojection, enabling robust and continuous multi-object tracking in scenarios with simultaneous camera and object motion.
2. Fusing point and line features to enhance the system’s robustness to illumination changes and texture degradation, while improving the map’s capability to represent environmental structures.
3. Proposing a differentiated mapping strategy for static and dynamic targets, constructing an object-level sparse map with semantic and dynamic information using ellipsoid parameter modeling.

2 Related works

2.1 SLAM in Dynamic Scenes

Traditional visual SLAM systems, whether feature-based (e.g., ORB, SIFT) or direct, perform well in static environments but are highly susceptible to disturbances by dynamic elements such as pedestrians or vehicles. The major challenge is detecting and rejecting dynamic features.

Early approaches relied on geometric consistency, with the epipolar constraint serving as the key criterion: points that deviate significantly from the epipolar line are treated were regarded as dynamic and subsequently discarded[10]. Tan et al[11]. extended this idea with a RANSAC-based method that adapts adjusts parameters to scene dynamics, improving rejection accuracy and pose robustness. However, relying on a single motion model, these methods can wrongly remove static features, especially in complex scenes with nonlinear motion or multiple moving objects.

Deep learning models such as SegNet, Mask R-CNN, and YOLO enable the detection of dynamic regions, and when combined with geometric consistency constraints, they provide more reliable moving object detection and improve SLAM performance in dynamic environments. As a result, Dynamic SLAM has shifted from purely geometry-based methods to multi-modal perception frameworks enhanced by deep learning. DS-SLAM [12] is one of the earliest attempts to remove dynamic feature using deep learning by combing semantic segmentation and inter-frame motion consistency. DynaSLAM [13], which integrates Mask R-CNN with multi-view geometric verification and region growing, further improves dynamic region detection. Although its real-time performance is limited, DynaSLAM

achieves strong results in dense mapping and occlusion handling, establishing a foundation for subsequent dynamic SLAM research. RDS-SLAM [14] decouples tracking from semantic segmentation and introduces mobility probability model for dynamic feature detection, thereby significantly enhancing real-time performance. Other methods such as YOLO-SLAM[15], Crowd-SLAM[16], and SG-SLAM[17] also integrate semantic segmentation with geometric consistency verification to improve the performance of SLAM in dynamic environments. Besides, some methods isolate the static regions using semantic masks to estimate the fundamental matrix and adopt epipolar constraints to detect and remove dynamic features, thus improving the pose estimation accuracy. Representative works include SOF-SLAM[18] and OVD-SLAM[19], the latter introducing chi-squared thresholds and depth statistics to segment foreground and background. By checking motion consistency via optical flow, these methods overcome the limitations of predefined object motion assumptions.

2.2 Hybrid Feature Fusion for Robust Perception ,

Visual SLAM faces significant challenges in low-texture environments and under varying illumination conditions, which can degrade both accuracy and stability. To address the problem of lacking of features in low-texture regions, researchers fused multiple feature types to improve the accuracy of localization and mapping. Zuo et al [20]. pointed that line features exhibit natural invariance to lighting and viewpoint changes, and then combined points and lines to provide more geometric constraints. PL-SLAM [21] extends ORB-SLAM by incorporating line features and learning a bag-of-words model of lines features to facilitate loop closure. Building on PL-SLAM, Alamanos et al., [22] proposed a method which combines the point and line features using the inverse of the covariance matrix as a weighting scheme. Zeng et al [23]., proposed the LK-RG which combines Lucas-Kanade optical flow with LSD region growing to achieve efficient and reliable line tracking . Another research direction aims to improve feature matching in low-texture regions using learning-based methods. SuperGlue[24] employs a graph neural network with attention to achieve robust correspondences, while Zhao et al. [25] progressively densify sparse keypoints to obtain dense matches. Although effective, these methods require high computational resources and may struggle to generalize. In contrast, our method exploits complementary geometric properties of point and line features without additional learning, making it more suitable for real-time SLAM.

2.3 Object-based SLAM

Object-based SLAM represents the environment with objects as landmarks, modeling them as ellipsoids, cuboids, or dual quadrics, and estimating camera poses by jointly optimizing object and point-line features [26]. In QuadricSLAM [27], objects are modeled as quadrics (ellipsoids), and camera poses are optimized together with quadric parameters via a factor graph to generate high-level object maps. OA-SLAM [28] introduces an object association strategy that combines class constraints, bounding box overlap, and point-level matching: short-term 2D bounding box matching enables initial tracking, followed by long-term association through projections of reconstructed objects and keypoint matching. OA-SLAM also proposes an object-based relocalization strategy, i.e., pairing ellipses and ellipsoids by class and estimating the camera pose using P3P from three object pairs, selecting the pose that minimizes the IoU cost between projected ellipsoid and detected boxes. With these advantages, OA-SLAM has become an important and serves as a baseline for subsequent research, including ours. Ellipsoid-SLAM[29] extends object-level SLAM by incorporating dynamic object parameterization and tracking using ellipsoidal representations, enabling the joint optimization of camera poses with static and dynamic objects. To better handle dynamic environments, Ellipsoid-SLAM filters out features associated with human objects, thereby reducing the impact of non-rigid motion on pose estimation and map consistency.

Earlier algorithms were designed under static environments and are thus unsuitable for dynamic scenes. To address this limitation, Cubeslam [30] reconstruct objects as cuboids based on 2D detection boxes and vanishing points, and jointly optimize both static and moving cuboids together with camera poses and map landmarks. VDO-SLAM[7]

classifies dynamic and static objects using scene flow and tracks objects via optical flow; while the initial camera and object motions are estimated from inliers using a constant-velocity model or P3P with RANSAC, and then jointly

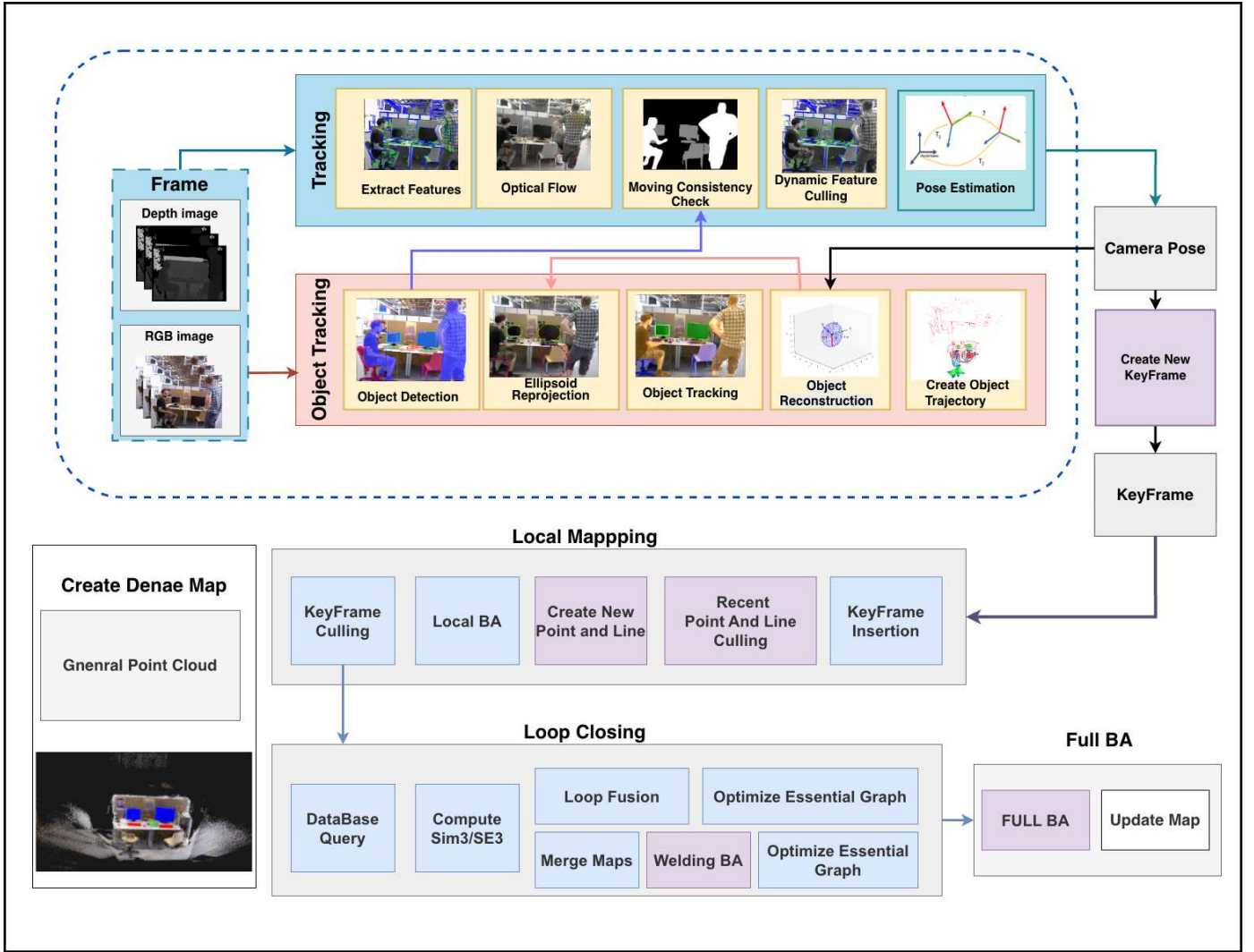


Fig. 2. Overview of our system's framework.

optimized in a factor graph. However, both Cubeslam and VDO-SLAM can only handle rigid objects, and are unable to model non-rigid motion such as human bodies. PLMOT-SLAM[31] integrates moving object tracking into a point-line visual SLAM framework to construct semantic dense point cloud maps and estimate object trajectories, but the heavy computational cost of dense point cloud processing limits its efficiency.

3 System Overview

Our system is built by augmenting, optimizing, and introducing additional modules into the ORB-SLAM3 framework. First, an object tracking module (emphasized with orange background in Fig. 2) is added to detect, segment and track objects, thereby enabling the reconstruction of objects as ellipsoids and the generation of dynamic object trajectories. Second, our system incorporates line features, the frontend processing pipeline of which follows a procedure similar to that applied to point features. Additionally, the bundle adjustment (BA) objective function is modified to achieve joint optimization of point and line features. The overall system architecture is shown in Fig. 2.

3.1 Object Detection and Tracking

The reliable extraction of bounding boxes and masks is fundamental to enabling algorithms like dynamic SLAM and object-based SLAM to achieve robust scene modeling and subsequent object tracking. Thus, our system adopts

YOLOv8-seg as the primary detection and segmentation model, with inference accelerated by TensorRT to ensure real-time performance. YOLOv8-seg extends the YOLO series by integrating a CSP-based backbone, a PAN-FPN feature

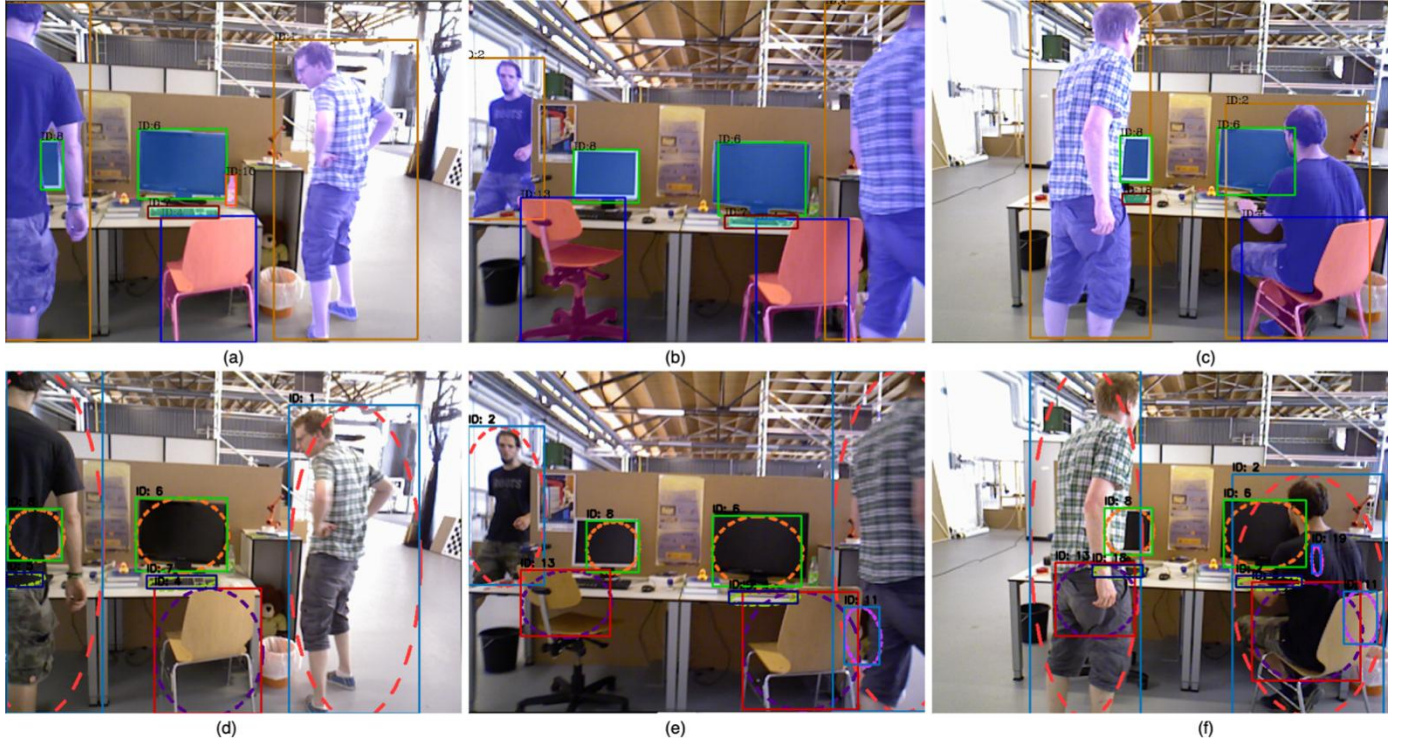


Fig. 3 illustrates the object tracking results and the reprojection of the corresponding ellipsoids. Specifically, Figs. (a)–(c) demonstrate the improved tracking performance, where stable object IDs are consistently maintained even under large camera motions and temporary object disappearance. Figs. (d)–(f) show the reprojection ellipses of the associated 3D ellipsoids, indicating that the proposed ellipsoid-based representation effectively handles occlusions and objects moving out of the camera’s field of view.

fusion network, and an anchor-free detection head. In addition, it introduces a dedicated mask branch that generates instance-level segmentation through prototype masks and coefficient prediction, thereby striking a balance between detection accuracy and efficiency. The model is trained on the COCO2017 dataset and is available in multiple parameter-scale variants, ranging from lightweight to large.

Traditional multi-object tracking methods generally assume that the camera remains static or undergoes only minor motion. Under significant camera movement, however, the trajectories of detection boxes deviate from linear models, while human motion often leads to occlusions, reduced matching accuracy, and frequent ID switches. To overcome these limitations, we extend DeepSORT and design a robust multi-object tracking approach tailored for dynamic SLAM scenarios, where both the camera and objects are simultaneously in motion.

The original DeepSORT framework performs multi-object tracking by combining motion and appearance cues. Candidate detections are first processed to extract appearance embeddings, while a Kalman filter predicts the state of each track. Matches are initially constrained by motion using the Mahalanobis distance, and the cosine distance between embeddings is used to associate candidates that satisfy this constraint. Finally, unmatched detections and tracks are matched based on bounding box IoU, ensuring complete data association.

Traditional Mahalanobis distance gating becomes unreliable under rapid camera motion. To mitigate this, we replace it with a class-aware constraint: detections whose classes do not match the associated track are effectively excluded by assigning an infinite appearance cost. For more precise matching, each track maintains a running average of its feature embeddings over recent frames, and the cosine distance between a detection’s embedding and this average is used as the association metric.

To maintain global consistency and handle temporary disappearance of detections due to occlusions or rapid

camera motion, we propose a second-stage association strategy based on 3D ellipsoid projections. Each object is approximated as an ellipsoid and projected onto the image plane to form an ellipse. A bounding rectangle is then

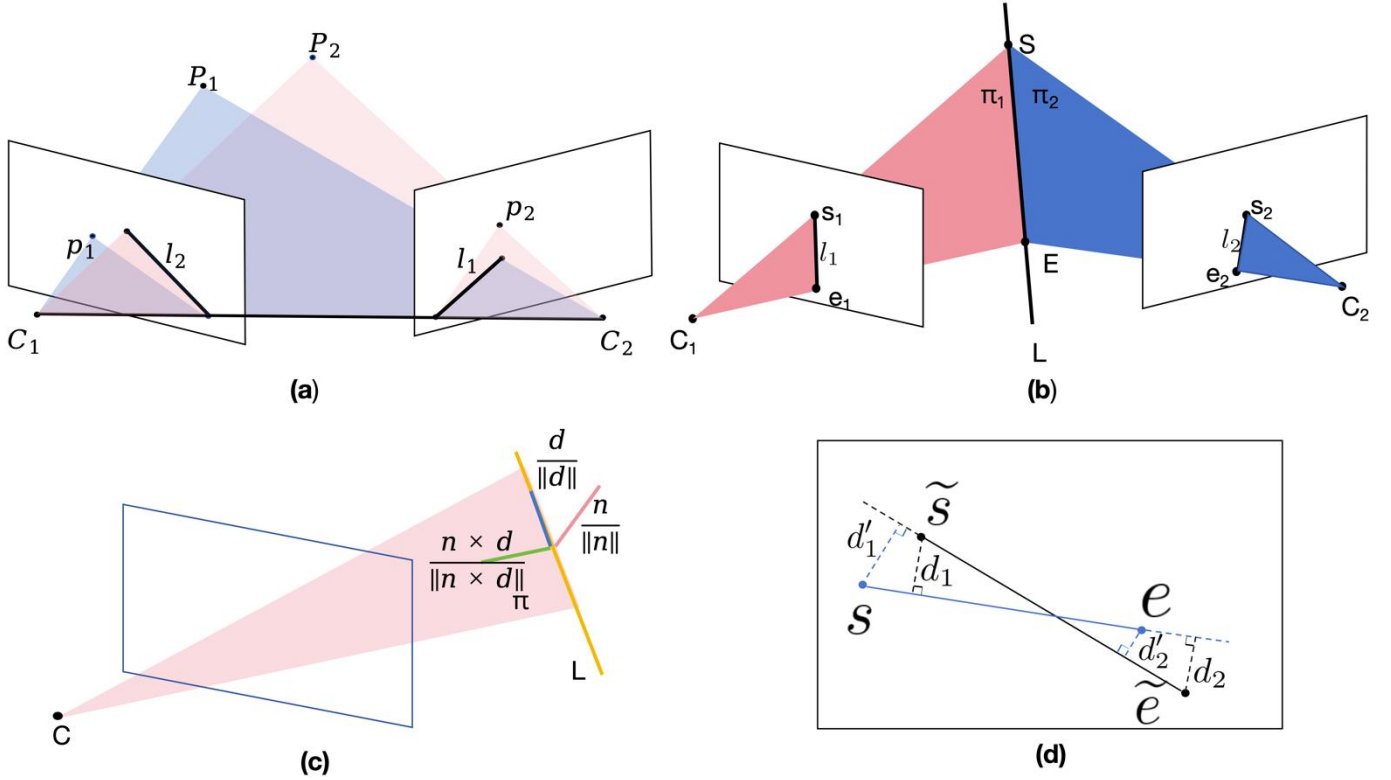


Fig. 4: Geometric constraints and the associated reprojection errors for point and line features.

computed from the ellipse's center and its major and minor axes, which is used to calculate IoU with unmatched detections, facilitating robust global data association. The ellipsoid projection is defined as follows:

$$PQ^*P^T = C^* \quad (1)$$

where $P = K[R|t]$ is the projection matrix, $Q^* \in \mathbb{R}^{3 \times 3}$ is the dual matrix of the object ellipsoid, and $C^* \in \mathbb{R}^{3 \times 3}$ is the dual matrix of the ellipse on the image plane. The reconstruction of the ellipsoid Q^* is detailed in Section 3.3 and the geometric parameters of the ellipse can be derived from C^* as described in [32].

For temporally adjacent frames, the motion of static objects remains approximately linear, rendering Kalman filtering effective. Following the initial and global matching stages, any remaining unmatched detections and trajectories are associated using the IoU between predicted Kalman states and detections. All successfully matched trajectories update their Kalman states, whereas unmatched detections initialize new trajectories. Fig. 3 illustrates the tracking results and ellipsoid reprojections.

3.2 Moving Consistency Check

To perform motion consistency checking, the system first establishes feature correspondences between consecutive frames using optical flow, based on the principle of photometric consistency. Semantic masks are employed to differentiate between foreground and background regions. Subsequently, background correspondences are combined with RANSAC to estimate the fundamental matrix F . As illustrated in Fig. 4(a), let P_1 represent a dynamic point at a given moment, and P_2 denotes its corresponding location in another frame with image coordinates p_1 and p_2 , respectively. According to the epipolar constraint, the corresponding epipolar lines are expressed as:

$$l_1 = Fp_1, l_2 = F^T p_2 \quad (2)$$

where $l_1 = [A_1, B_1, C_1]$, $l_2 = [A_2, B_2, C_2]$ are the epipolar lines of points p_1 and p_2 in the other image. The geometric distance from a point to its epipolar line and the symmetric epipolar error are defined as:

$$d(p, l) = \frac{|Ax + By + C|}{\sqrt{A^2 + B^2}}, \quad e = \frac{d(p_1, l_2) + d(p_2, l_1)}{2} \quad (3)$$

The system employs e^2 as the measurement and substitutes it into a chi-squared distribution with 2 degrees of freedom

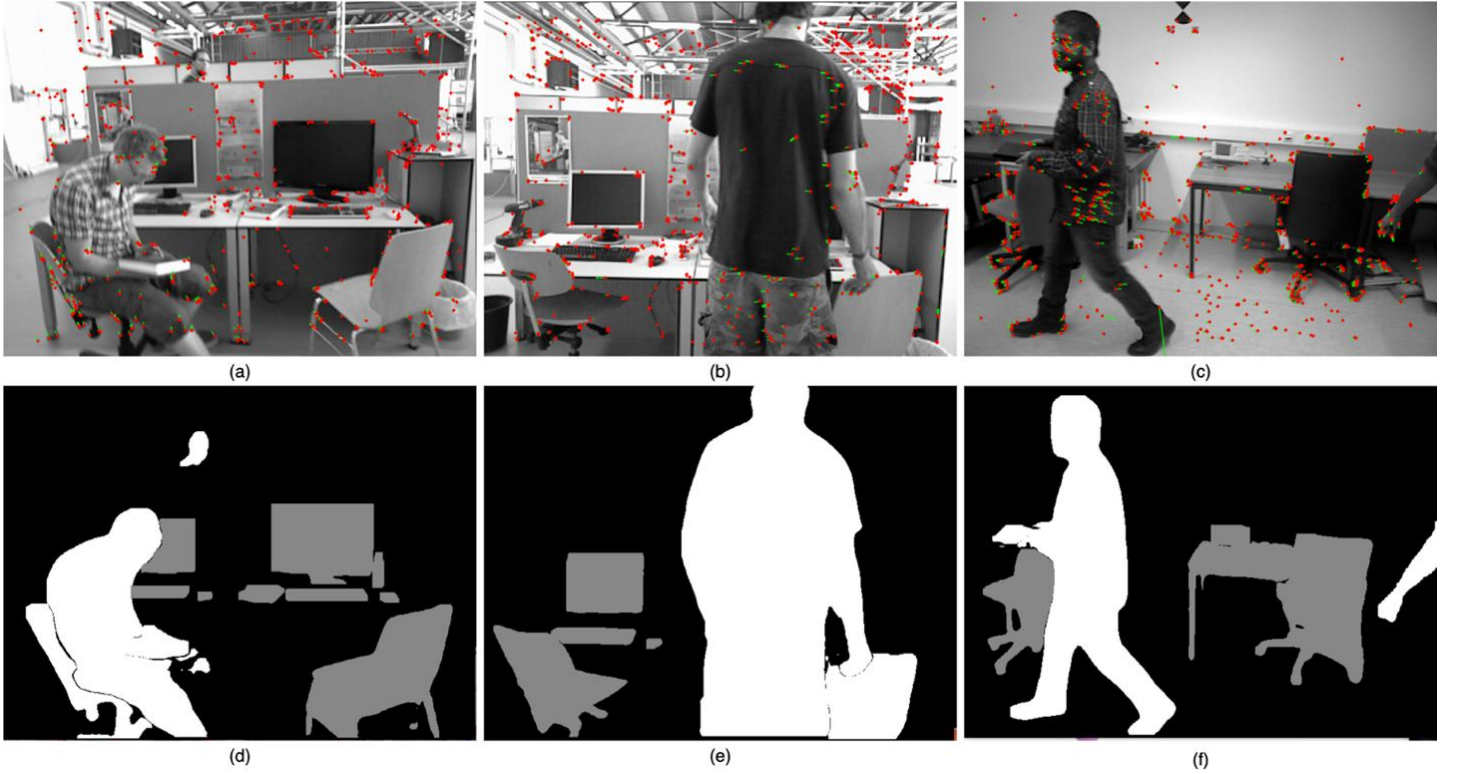


Fig. 5: Optical flow matching and mask evaluation in the walking_xyz and crowd sequences

to compute the probability that a point is static:

$$\alpha_i = 1 - F_{\chi^2_2}(e_i^2) \quad (4)$$

where $F_{\chi^2_2}(\cdot)$ denotes the cumulative distribution function (CDF) of the chi-squared distribution with 2 degrees of freedom.

Feature points with very low confidence (below 0.05) are considered unreliable and removed. At the object level, the mean confidence of all points within a mask is computed, and objects with an average below 0.9 are classified as dynamic, leading to the exclusion of all their associated features. This two-tier filtering strategy effectively reduces the impact of dynamic features on camera pose estimation. Fig. 5 illustrates the optical flow matches and the resulting dynamic object masks, with white regions representing dynamic objects and gray regions representing static ones.

3.3 Point-line feature processing

3.3.1 3D Line Representation

As shown in Fig. 4(b) 4(c), In the Plücker representation, a spatial line L is expressed as a 6D vector $\mathcal{L} = [n^T \quad d^T] \in \mathbb{R}^6$ where d is the direction vector, and $n = p \times d$ is the moment vector with p being any point on the line. The spatial line is transformed from the camera frame to the world frame via the pose matrix $T_{cw} = \begin{bmatrix} R_{cw} & p_{cw} \\ 0 & 1 \end{bmatrix}$, and the corresponding transformation of the Plücker line coordinates is given in (6).

$$\mathcal{L}^c = \begin{bmatrix} n^c \\ d^c \end{bmatrix} = T_{cw} \mathcal{L}^w = \begin{bmatrix} R_{cw} & [p_{cw}]_{\times} R_{cw} \\ 0 & 1 \end{bmatrix} \mathcal{L}^w \quad (6)$$

Where $\mathcal{L}^w$ and $\mathcal{L}^c$ are the Plücker representations of the line in the world and camera coordinate systems, respectively, and $[\cdot]_{\times}$ denotes the skew-symmetric matrix.

The dual Plücker representation L^* of a line can be computed from planes π_1, π_2 with the computation method given by (7):

$$\mathcal{L}^* = \begin{bmatrix} [d]_{\times} & n \\ n^T & 0 \end{bmatrix} = \pi_1 \pi_2^T - \pi_2 \pi_1^T \in \mathbb{R}^{4 \times 4} \quad (7)$$

The method for plane construction and the conversion from $\mathcal{L}^*$ to the Plücker coordinates $\mathcal{L}$ can be found in reference [32].

Since a spatial line has 4 degrees of freedom, while the Plücker coordinates are 6-dimensional and must satisfy the constraint $n^T d = 0$, direct unconstrained optimization is not suitable. Therefore, an orthogonal parameterization is adopted, representing the line as $L' = [U, W] \in [SO(3), SO(2)]$. Specifically, the line L is factorized using QR decomposition.

$$[n, d] = \begin{bmatrix} n & d & n \times d \\ \frac{n}{\|n\|} & \frac{d}{\|d\|} & \frac{n \times d}{\|n \times d\|} \end{bmatrix} \begin{bmatrix} \|n\| & 0 \\ 0 & \|d\| \\ 0 & 0 \end{bmatrix} \quad (8)$$

$$W = \begin{bmatrix} \sin(\emptyset) & -\cos(\emptyset) \\ \cos(\emptyset) & \sin(\emptyset) \end{bmatrix} = \begin{bmatrix} w_1 & -w_2 \\ w_2 & w_1 \end{bmatrix}, U = R(\psi_x, \psi_y, \psi_z) = \begin{bmatrix} n & d & n \times d \\ \frac{n}{\|n\|} & \frac{d}{\|d\|} & \frac{n \times d}{\|n \times d\|} \end{bmatrix} \quad (9)$$

where the three angles (ψ_x, ψ_y, ψ_z) represent the rotation of the line's coordinate frame around the x-, y-, and z-axes, respectively, while the scalar $\emptyset$ implicitly encodes the distance from the origin to the line.

Finally, the Plücker coordinates can be reconstructed from the orthogonal representation as (10):

$$L' = \begin{bmatrix} w_1 u_1^T \\ w_2 u_2^T \end{bmatrix} = \frac{1}{\sqrt{(\|n\|^2 + \|d\|^2)}} L \quad (10)$$

where u_i is the i th column of matrix U .

3.3.2 Bundle Adjustment with Points and Lines

Based on the ORB-SLAM3 framework, the system formulates a nonlinear optimization model that minimizes reprojection error, jointly optimizing:

Camera poses: $T = \{T_1, T_2, \dots, T_f\}$ where $T_i \in SE(3)$;

MapPoints: $X = \{X_1, X_2, \dots, X_m\}$ where $X_j \in \mathbb{R}^3$;

MapLines: $L = [L_1, L_2 \dots L_n]$ where $L_k \in \mathbb{R}^4$;

The reprojection error term for point features is defined in (13).

$$\begin{cases} e_j^p = \omega [x_j - \pi_p(T_i X_j)] \\ \omega = 1.0 + 0.5 * (1 - e^{-n}) \end{cases} \quad (11)$$

where x_j denotes the keypoint coordinate, X_j is the corresponding 3D map point, $\pi_p(\cdot)$ represents the point projection function, ω is the dynamic weight, and n indicates the number of times the map point appears within dynamic regions.

The projection of a Plücker line in the camera coordinate system is given by:

$$l = K^{-T} n_c = \begin{bmatrix} l_1 \\ l_2 \\ l_3 \end{bmatrix} \quad (12)$$

As shown in Fig. 4(d), for an observed line segment with endpoints $s_k(u_s, v_s)$, $e_k(u_e, v_e)$ the reprojected endpoints $\tilde{s}_k, \tilde{e}_k$ are used to define the following endpoint-to-line distance error term.

$$e_s^k = \frac{l_1 u_s + l_2 v_s + l_3}{\sqrt{l_1^2 + l_2^2}}, e_e^k = \frac{l_1 u_e + l_2 v_e + l_3}{\sqrt{l_1^2 + l_2^2}}, e_k^l = \omega \begin{bmatrix} e_s^k \\ e_e^k \end{bmatrix} \quad (13)$$

where ω is a weighting factor, calculated in the same way as the weight for point features.

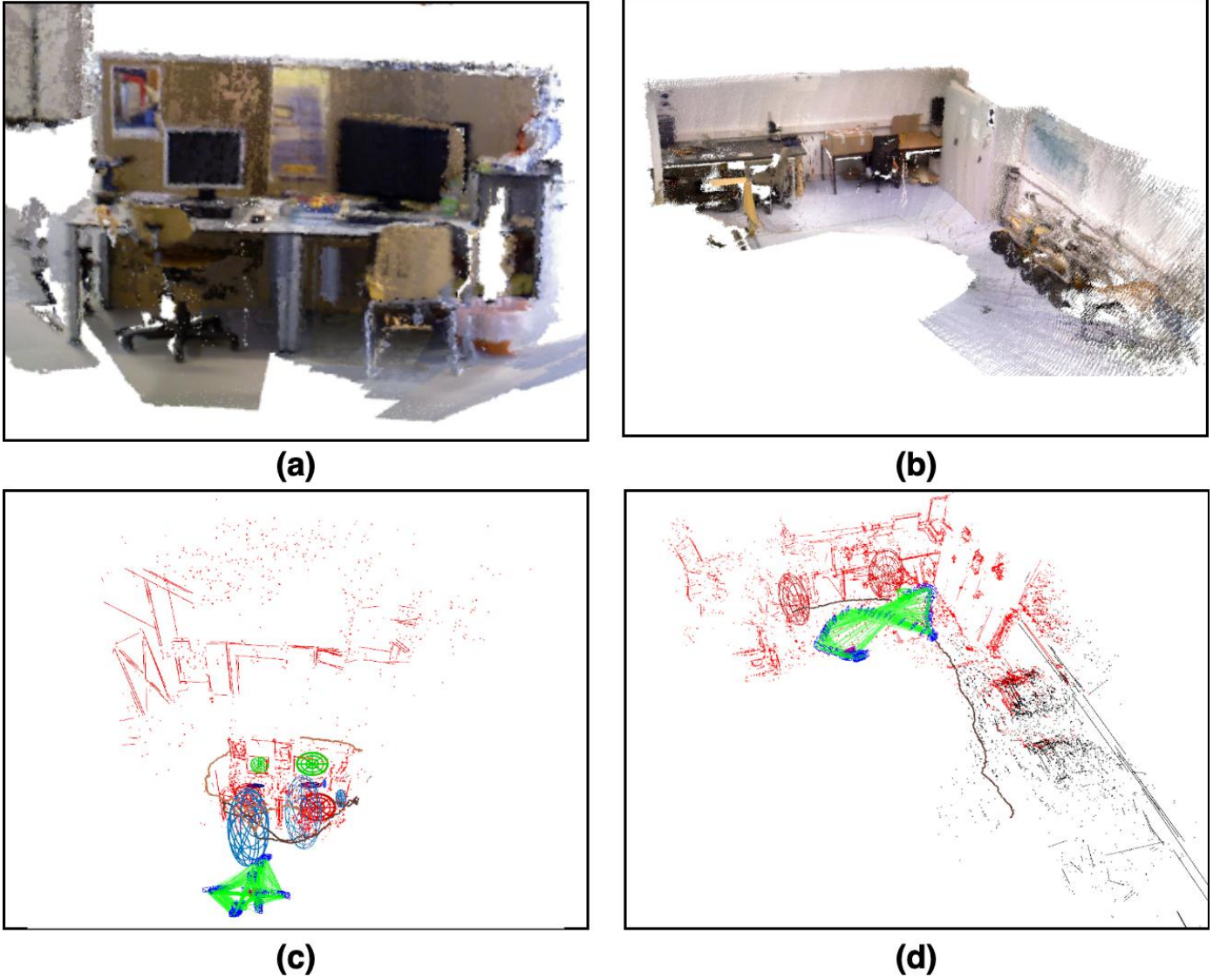


Fig. 6. Dense (a, b) and sparse (c, d) mapping results for the *walking_xyz* and *person_tracking* sequences. In the sparse maps, colored ellipsoids indicate different object categories, and colored curves represent object trajectories.

$$\{T_i\}, \{X_j\}, \{L_k\} = \underset{T_i, X_j, L_k}{\operatorname{argmin}} \left(\sum_{i \in F_l \cup F_k} \left(\sum_{j \in \chi_i} \rho_p \left(\|e_{ij}^p\|_{\Sigma_p}^2 \right) + \sum_{k \in \chi_i} \rho_l \left(\|e_{ik}^l\|_{\Sigma_l}^2 \right) \right) \right) \quad (14)$$

3.4 Object Reconstruction

In our 3D object reconstruction framework, objects are modeled as parameterized ellipsoids. Cross-frame associations are established using object tracking IDs, allowing the mapping of 2D detection boxes into 3D space and the generation of a semantic object map. The reconstruction process proceeds at two levels: in the image plane, the object projection is expressed as an ellipse in the dual 2D space $C^* \in \mathbb{R}^{3 \times 3}$; in 3D space, the object is represented as a dual quadric $Q^* \in \mathbb{R}^{4 \times 4}$, linked through the projection constraint defined in Equation (1).

During initialization, the 3D center of each object is computed from the detection box center $p = (u, v, 1)$ and its mask-averaged depth d . The initial dual quadric of the ellipsoid is defined as follows:

$$Q^* = \begin{bmatrix} R \text{diag}(a^2, b^2, c^2) R - p_w p_w^T & -p_w \\ p_w^T & -1 \end{bmatrix} \quad (15)$$

where $\text{diag}(\cdot)$ denotes a diagonal matrix, R denotes the rotation matrix of the ellipsoid, $p_w = T_{cw}(dK^{-1}p)$.

For static objects, the initial ellipsoid axes are set as $a, b, c = \left(\frac{w}{f_x} + \frac{h}{f_y}\right) * \frac{d}{2}$, with rotation $R = I$. For dynamic objects,

axes are derived from the bounding box dimensions using back-projection: $a = \frac{w*d}{2f_x}, b = \frac{h*d}{2f_y}, c = \frac{1}{2}(a + b)$, with rotation $R = R_{wc}$ corresponding to the camera rotation in the world frame.

After initialization, static object ellipsoids are refined using reprojection errors across consecutive frames. The

projected ellipse C is treated as a 2D Gaussian $\mathcal{N}(\mu, \Sigma)$, with $\mu = \begin{bmatrix} c_x \\ c_y \end{bmatrix}, \Sigma^{-1} = R(\theta) \begin{bmatrix} \frac{1}{\alpha^2} & 0 \\ 0 & \frac{1}{\beta^2} \end{bmatrix} R(\theta)^T$ where

(c_x, c_y) is the ellipse center in pixel coordinates, α, β are ellipse axes lengths, and $R(\theta)$ is the 2D rotation matrix. The Wasserstein distance between two 2D Gaussians $\mathcal{N}_1(\mu_1, \Sigma_1)$ and $\mathcal{N}_2(\mu_2, \Sigma_2)$ is defined as:

$$\mathcal{W}_2^2(\mathcal{N}_1, \mathcal{N}_2) = \|\mu_1 - \mu_2\|_2^2 + \text{Tr}(\Sigma_1 + \Sigma_2 - 2(\Sigma_1^{\frac{1}{2}} \Sigma_2 \Sigma_1^{\frac{1}{2}})^{\frac{1}{2}}) \quad (16)$$

Reprojection errors are weighted according to the temporal distance between frames, with observations from more distant frames assigned higher weights. The objective function for the i -th object is given in Eq. (17).

$$\frac{1}{N} \sum_{j=0}^N \rho_j \mathcal{W}_2^2(E_{ij} P_j Q_i^* P_j^T), \rho_j = \frac{1 + \lambda d_j}{\sigma_j} \quad (17)$$

Where E_{ij} is the inscribed ellipse of the i -th detection, σ_j the detection confidence, and d_j denotes the temporal distance between the j -th detection and the current frame.

For dynamic objects, the 3D position and shape are updated with each new observation. In each frame, the ellipsoid center and axes are updated based on Kalman-filtered detection boxes and depth measurements, aligned with the camera rotation in the world frame ($R = R_{wc}$) and the updated center is appended to the trajectory to construct the 3D motion path. By aggregating observations across frames, the system maintains real-time 3D representations and motion states of dynamic objects, thereby achieving continuous reconstruction of both object shapes and trajectories. Fig. 6(c) and 6(d) show the results on the walking_xyz and person_tracking sequences.

4 Experiment Result

We evaluated the proposed system using multiple benchmark datasets and real-world recordings. Performance in dynamic environments was assessed with subsets of the Bonn RGB-D Dynamic Dataset and the TUM RGB-D Dataset[33, 34], while the EuRoC MAV Dataset [35] was employed to test robustness under challenging conditions such as illumination variation and texture degradation. To further verify real-world applicability, we collected additional data using an Intel RealSense D435i camera. To ensure fairness, each sequence was repeated five times, and the evaluation metrics (ATE and RPE) were obtained using the official scripts from the TUM RGB-D benchmark[34]. The global consistency is evaluated using the absolute trajectory error (ATE) which represents the distance between the estimated and the ground truth trajectories, while the local consistency is evaluated using relative pose error (RPE) and relative rotation error (RRE). All experiments were conducted on a workstation running Ubuntu 20.04 LTS, equipped with an Intel Core i5-14600K CPU, 32 GB DDR5 memory, and an NVIDIA RTX 4060 Ti GPU.

4.1 Evaluation in dynamic environments

To evaluate dynamic SLAM accuracy, we adopted two widely used benchmarks: Bonn RGB-D and TUM RGB-D.

From the Bonn dataset, we selected two representative sequences, person_tracking2 for complex human motion and

crowd for dense dynamic disturbances. From the TUM dataset, five sequences were chosen: sitting_xyz to capture subtle human activity, and four sequences reflecting different camera motion patterns—static (stationary), xyz (linear), half (hemispherical), and rpy (rotational). Together, these sequences cover a broad spectrum of motion

Table I. RMSE and Standard Deviation of Absolute Trajectory Error (ATE) [m] on dynamic sequences

Seq.	ORB-SLAM3		RDS-SLAM		SG-SLAM		OVD-SLAM		PLO-SLAM(P)		PLO-SLAM(P+L)	
	RMSE	S.D.	RMSE	S.D.	RMSE	S.D.	RMSE	S.D.	RMSE	S.D.	RMSE	S.D.
fr3_sitting_xyz	0.0136	0.0059	0.0218	0.0146	0.0135	0.0094	0.0124	0.0062	0.0148	0.0098	0.0119	0.0049
fr3_walking_static	0.2286	0.1717	0.0096	0.0044	0.0075	0.0037	0.0058	0.0030	0.0064	0.0038	0.0053	0.0031
fr3_walking_xyz	0.5099	0.1951	0.0210	0.0107	0.0226	0.0123	0.0178	0.0092	0.0182	0.0087	0.0159	0.0085
fr3_walking_half	0.2870	0.1335	0.0306	0.0146	0.0446	0.0237	0.0244	0.0119	0.0256	0.0133	0.0226	0.0116
fr3_walking_rpy	0.8756	0.4513	Lost	Lost	0.0411	0.0277	0.0404(92%)	0.0219	0.0394(93%)	0.0229	0.0356	0.0198
person_tracking	0.7641	0.4107	0.0433	0.0189	0.0445	0.0143	0.0514	0.0205	0.0480	0.0178	0.0359	0.0200
person_tracking2	0.9095	0.5007	0.0596	0.0229	0.6514	0.3664	0.0497	0.0218	0.0418	0.0222	<u>0.0431</u>	0.0178
crowd	1.4570	0.4924	0.0123(93%)	0.0073	0.0208	0.0100	0.0172	0.0097	0.0222	0.0119	<u>0.0146</u>	0.0060

Table II. RMSE of Relative Translation Error (RTE) and Relative Rotation Error (RRE) on dynamic sequences.

Seq.	ORB-SLAM3		RDS-SLAM		SG-SLAM		OVD-SLAM		PLO-SLAM(P)		PLO-SLAM(P+L)	
	RTE	RRE	RTE	RRE	RTE	RRE	RTE	RRE	RTE	RRE	RTE	RRE
fr3_sitting_xyz	0.0160	0.5237	0.0155	0.9222	0.0185	0.4865	0.0142	0.4516	0.0196	0.4411	0.0118	0.4628
fr3_walking_static	0.0664	1.1872	0.0114	0.2536	0.0093	0.2450	0.0070	0.2228	0.0081	0.2405	0.0063	0.1955
fr3_walking_xyz	0.0774	1.3201	0.0184	0.4662	0.0207	0.6328	0.0179	0.4595	0.0152	0.5085	0.0179	0.4862
fr3_walking_half	0.0416	0.9200	0.0192	0.5043	0.0356	0.8812	0.0187	0.4786	0.0191	0.5282	0.0176	0.5005
fr3_walking_rpy	0.0564	1.1782	Lost	Lost	0.0489	1.1101	0.0288	0.6825	0.0417	1.0047	0.0297	0.7273
person_tracking	0.4027	23.5465	0.2211	21.8187	0.3816	29.3568	0.3037	28.5772	0.2643	24.4889	0.1463	15.1855
person_tracking2	0.2826	17.7050	0.1610	15.1935	0.2735	18.7712	0.4021	32.3830	0.2413	21.3376	0.2179	20.2972
crowd	1.0003	24.5489	0.2043	18.2866	0.1096	9.1936	0.1207	9.6535	0.1359	9.0222	0.1103	9.8555

states, from weak to strong and from simple to complex, enabling a comprehensive evaluation of the robustness and generalization capability of our system.

Based on the datasets, we conducted a comparative analysis of ORB-SLAM3, RDS-SLAM, SG-SLAM, OVD-SLAM, and the proposed PLO-SLAM—both P (point features only) and P+L (point-line fusion). The experimental results are summarized in Tables I–II. The experiments demonstrate that our improved algorithm significantly enhances tracking stability. On the TUM dataset, PLO-SLAM consistently maintained stable tracking of both static and dynamic objects, regardless of whether the camera was stationary or in motion. On the Bonn dataset, the system was able to re-acquire targets even after camera rotations exceeding 180°. The ellipsoid representation not only encodes the 3D state of each object but also allows for the generation of bounding boxes.

The results indicate that the improved method significantly enhances target tracking stability. In the TUM dataset, both static and dynamic targets were stably tracked regardless of camera motion. In the Bonn dataset, the system was able to re-capture targets even after camera rotations exceeding 180° or temporary target occlusions. This robustness stems from cross-frame associations and ellipsoid-based target modeling. The ellipsoids represent each target’s 3D state, generate bounding boxes to mitigate tracking failures from occlusion or camera motion, and enrich the sparse map with semantic information to support navigation and path planning.

As shown in Table 1, the motion consistency check in PLO-SLAM effectively determines the motion state of objects and filters out dynamic features, thereby improving localization and mapping accuracy. In contrast, RDS-SLAM and SG-SLAM tend to misclassify some object features as dynamic. Depth noise at the edges of moving objects

causes OVD-SLAM's foreground-background segmentation, based on depth statistics and the Chi-square test, to fail in these regions, resulting in incomplete removal of dynamic features. Experiments on the walking_rpy sequence show that incorporating line features significantly reduces both RMSE and SD, indicating that line features help mitigate the

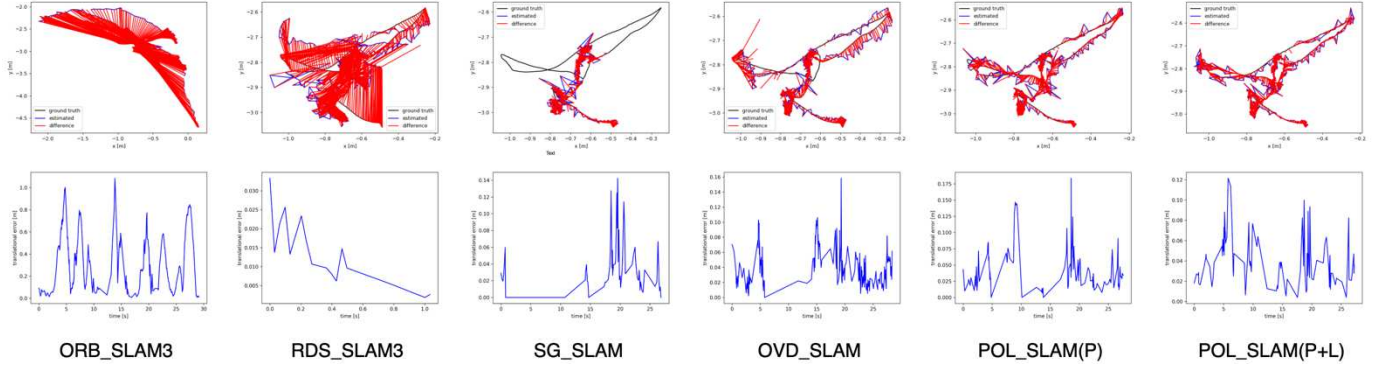


Fig. 7. Comparison of absolute trajectory error (ATE) and relative pose error (RPE) among different algorithms on the TUM fr3_w_rpy sequence.

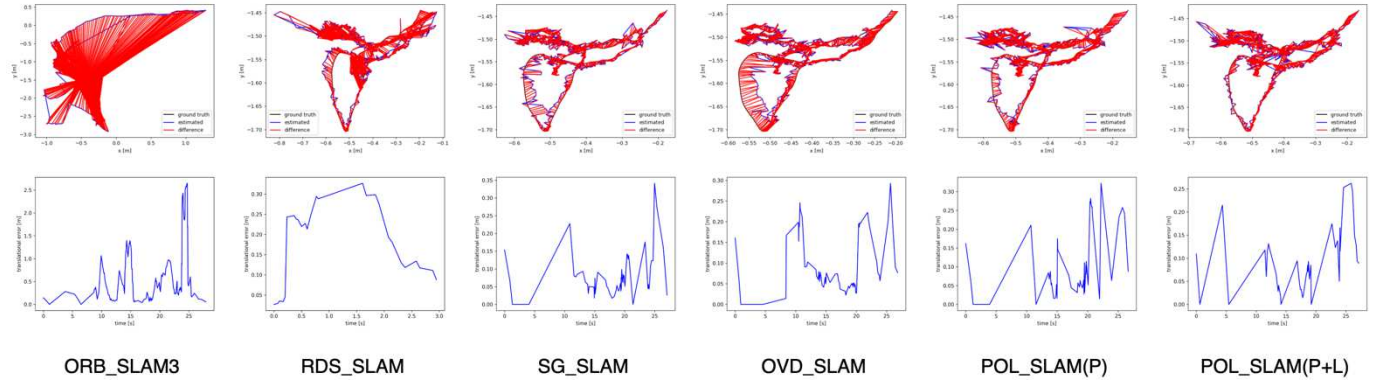
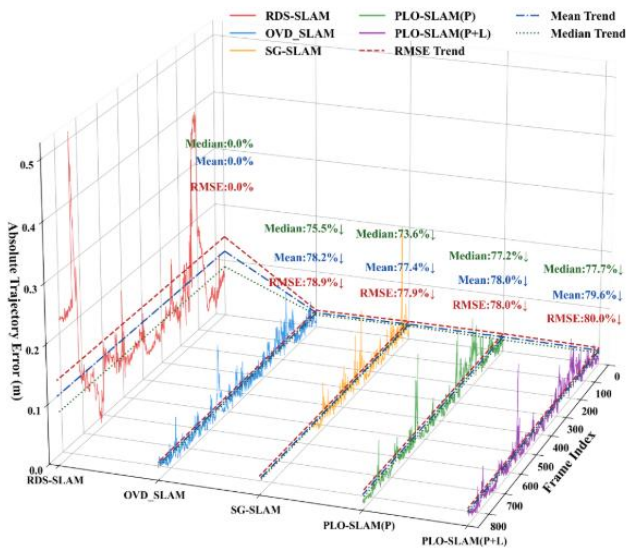
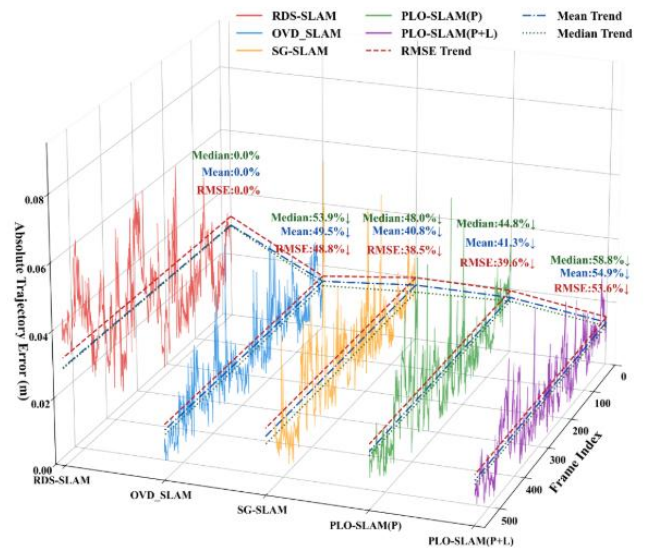


Fig. 8. Comparison of absolute trajectory error (ATE) and relative pose error (RPE) among different algorithms on the Bonn crowd sequence.



(a)



(b)

Fig. 9. Comparison of absolute trajectory error (ATE) for different dynamic SLAM algorithms on dynamic scenes: (a) TUM dataset *walking_rpy*; (b) Bonn dataset *crowd*.

issues of insufficient valid feature points caused by large camera rotations and featureless walls.

Table IV. Absolute Trajectory Error (ATE) [m] and ours method Improvements vs ORB_SLAM3. in in EuRoc MAV dataset.

Seq.	ORB-SLAM3(V)		ORB-SLAM3(V+I)		ORB-LINE-SLAM		PLO-SLAM(P+L)		Improvements.	
	RMSE	S.D.	RMSE	S.D.	RMSE	S.D.	RMSE	S.D.	RMSE	S.D.
MH01_easy	0.0671	0.0375	0.0344	0.0127	0.0409	0.0216	0.0375	0.0189	44.1%	49.6%
MH02_easy	0.0713	0.0270	0.0185	0.0096	0.0411	0.0184	0.0354	0.0151	50.3%	44.1%
MH03_medium	0.0570	0.0230	0.0265	0.0125	0.0460	0.0179	0.0365	0.0155	36.0%	32.6%
MH04_difficult	0.1395	0.0626	0.0421	0.0229	0.1017	0.0432	0.0592	0.0309	57.6%	50.6%
MH05_difficult	0.0962	0.0454	0.0707	0.0352	0.0599	0.0324	0.0468	0.0173	51.4%	61.9%
V101_easy	0.0881	0.0296	0.0376	0.0141	0.0872	0.0309	0.0877	0.0301	0.0%	-0.0%
V102_medium	0.0741	0.0291	0.0185	0.0075	0.0646	0.0196	0.0617	0.0183	16.7%	37.2%
V103_difficult	0.1987	0.1307	0.0257	0.0114	0.0647	0.0245	0.0850	0.0421	57.2%	67.9%
V201_easy	0.0743	0.0315	0.0144	0.0056	0.0844	0.0623	0.0606	0.0233	18.4%	26.0%
V202_medium	0.0880	0.0442	0.0118	0.0053	0.0643	0.0261	0.0721	0.0306	18.1%	30.1%
V203_difficult	0.4962(81%)	0.1678	0.0267(100%)	0.0121	0.2187(91%)	0.0939	0.2055(88%)	0.0965	52.5%	42.5%

Table V. RMSE of Relative Translation Error (RTE) and Relative Rotation Error (RRE) in EuRoc MAV dataset.

Seq.	ORB-SLAM3(V)		ORB-SLAM3(V+I)		ORB-LINE-SLAM		PLO-SLAM(P+L)		Improvements.	
	RTE	RRE.	RTE	RRE	RTE	RRE	RTE	RRE	RTE (%)	RRE (%)
MH01_easy	0.9274	1.8706	0.0198	0.0321	0.9317	1.8779	0.9340	1.8842	-0.71%	-0.73
MH02_easy	0.8492	1.9606	0.0136	0.0303	0.8534	1.9774	0.8528	1.9741	-0.42%	-0.69%
MH03_medium	1.1925	2.7193	0.0284	0.0278	1.1928	2.7104	1.1912	2.7115	0.11%	0.29
MH04_difficult	1.2316	2.0998	0.0280	0.0230	1.2363	2.0852	1.2393	2.0923	-0.63%	0.36%
MH05_difficult	1.1594	2.0864	0.0276	0.0265	1.1633	2.0699	1.1621	2.0746	-0.23%	0.57%
V101_easy	0.9093	2.1396	0.0841	0.0587	0.9242	2.1273	0.9289	2.1275	-2.16%	0.57%
V102_medium	0.9990	3.4914	0.0349	0.0668	1.0066	3.4842	1.0050	3.4967	-0.60%	-0.15%
V103_difficult	0.8975	4.2887	0.0331	0.0659	0.9027	4.3080	0.9164	4.3148	-2.11%	-0.61%
V201_easy	0.7699	2.5221	0.0168	0.0456	0.7596	2.6423	0.7842	2.5252	-1.86%	-0.12%
V202_medium	0.9128	4.3336	0.0128	0.0425	0.9214	4.3304	0.8933	4.3349	2.14%	-0.03%
V203_difficult	1.1653	6.5681	0.0203	0.1549	1.0263	6.1772	1.0307	6.1952	11.93%	5.68

As shown in Table II, the RTE and RRE computed on the Bonn dataset are significantly higher than those on the TUM dataset, which is due to the lower frame rate of the Bonn data and the larger relative motion between consecutive frames. Incorporating line features reduces both RTE and RRE, highlighting the advantages of line features in providing wider coverage and higher directional accuracy. Notably, RDS-SLAM achieves relatively low RTE and RRE on the Bonn dataset. This can be attributed to the fact that large camera motions reduce the number of matched feature points, which in turn increases the number of keyframes and indirectly mitigates relative errors.

We also analyzed the runtime of each system module, as summarized in Table III. On average, processing a single frame takes around 52–55 ms. Feature extraction and semantic feature extraction dominate the computational cost. In contrast, object tracking and motion detection impose a much smaller computational burden. Object reconstruction, while relatively expensive, is not performed on every frame, so its overall impact on runtime remains limited.

In this section, we evaluate the effect of line features on system accuracy using the EuRoC MAV dataset, released jointly by the University of Zurich and ETH Zurich [35]. The dataset was collected in various indoor environments,

including industrial settings and motion capture rooms, and provides stereo images, IMU measurements, and high-precision ground-truth poses obtained via laser tracking and motion capture. It features scenes with rich textures, weak textures, and significant illumination changes, making it suitable for evaluating visual-inertial odometry and 3D

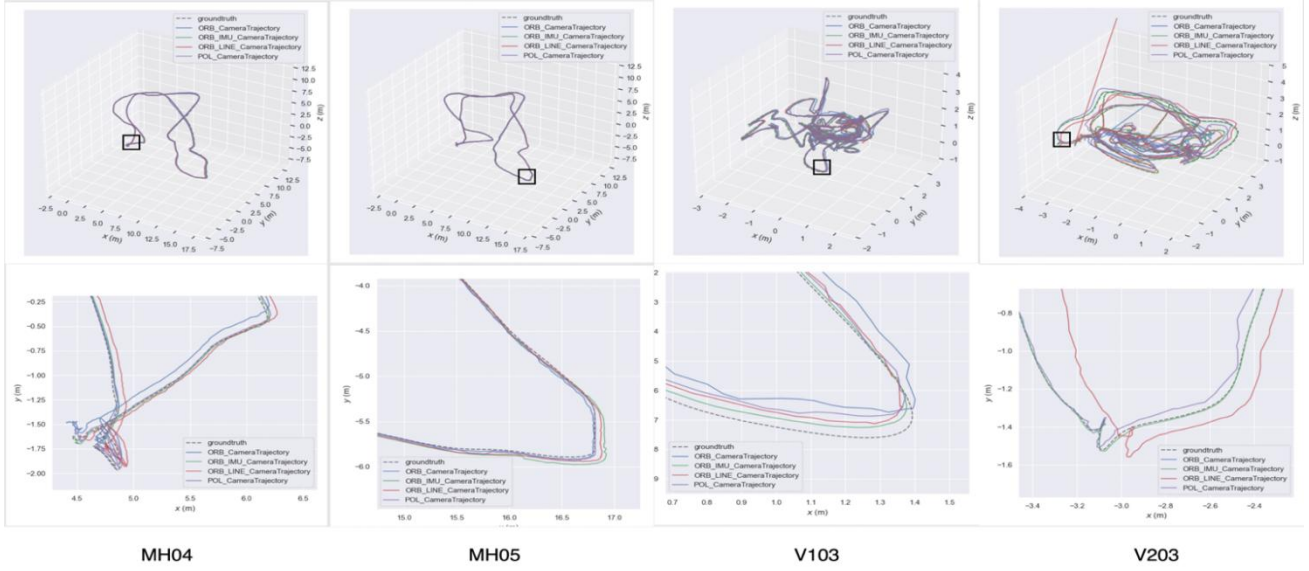


Fig. 10. Trajectories of different algorithms on MH04, MH05, V103, and V203 sequences. Top row: full trajectories; bottom row: zoomed-in views.

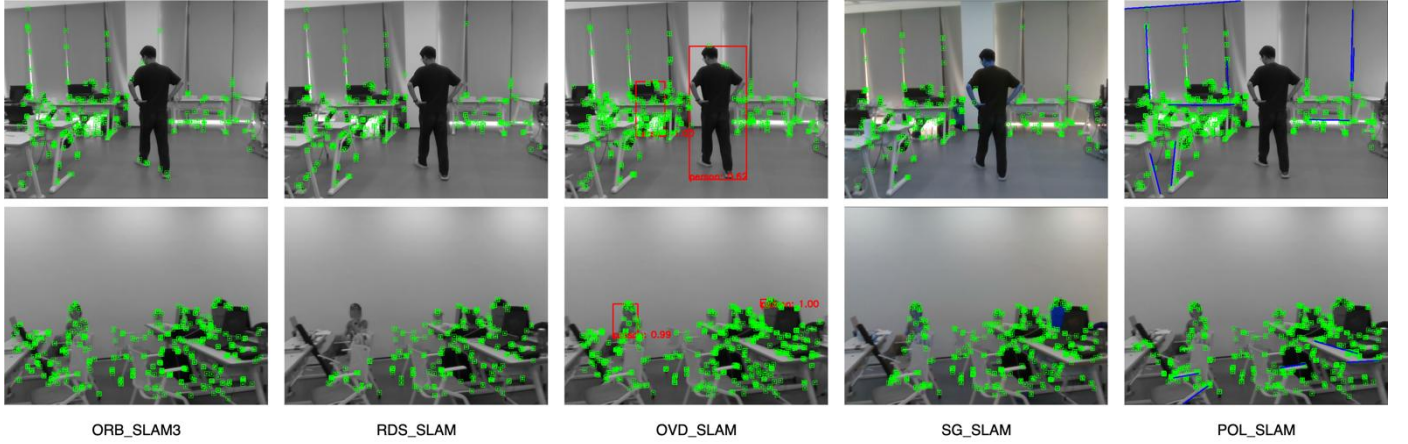


Fig. 11. Comparison of dynamic feature filtering across algorithms.

reconstruction algorithms. Experiments were conducted on 11 flight sequences from the EuRoC MAV dataset, categorized into three difficulty levels: easy, medium, and difficult. Using these sequences, we performed a comparative analysis of ORB-SLAM3 (stereo), ORB-SLAM3 (visual-inertial), ORB-LINE-SLAM, and the proposed POL method. The results are summarized in Tables IV and V.

The experimental results show that incorporating line features greatly reduces the absolute trajectory error (ATE) of PLO-SLAM compared to ORB-SLAM3, with notable improvements on medium and high-difficulty sequences. For instance, on the V203_difficult sequence, the ATE decreases by 52%. However, the improvements in relative trajectory error (RTE) and relative rotation error (RRE) are limited, and in some sequences these metrics even increase. This suggests that line features primarily enhance global localization accuracy rather than the precision of local motion estimation.

The MH sequences feature relatively small rotations, which generally result in lower ATEs across all methods. However, the significant illumination changes in these sequences make tracking challenging. Incorporating line features helps reduce the ATE, and PLO-SLAM even surpasses the visual-inertial ORB-SLAM3 on MH05_difficult, highlighting the robustness of line features against visual disturbances.

By adopting a more selective line feature strategy, our method outperforms ORB-LINE-SLAM in sequences rich in features, achieving lower ATEs. In feature-sparse sequences such as V103_difficult and V203_difficult, however, system stability is slightly compromised. Visual-inertial odometry (VIO), on the other hand, offers significantly higher

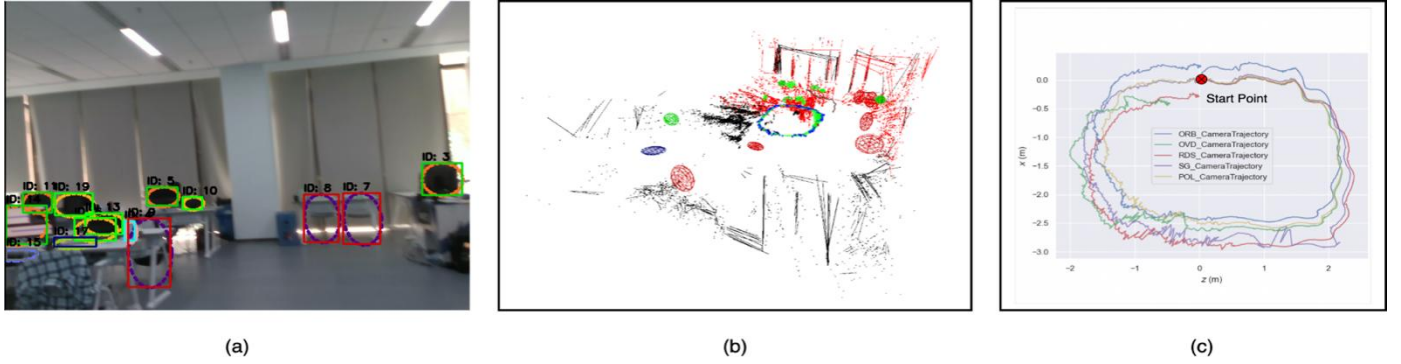


Fig. 12. (a) Object ellipsoid projections; (b) Sparse map by our algorithm; (c) Camera trajectories of various algorithms.

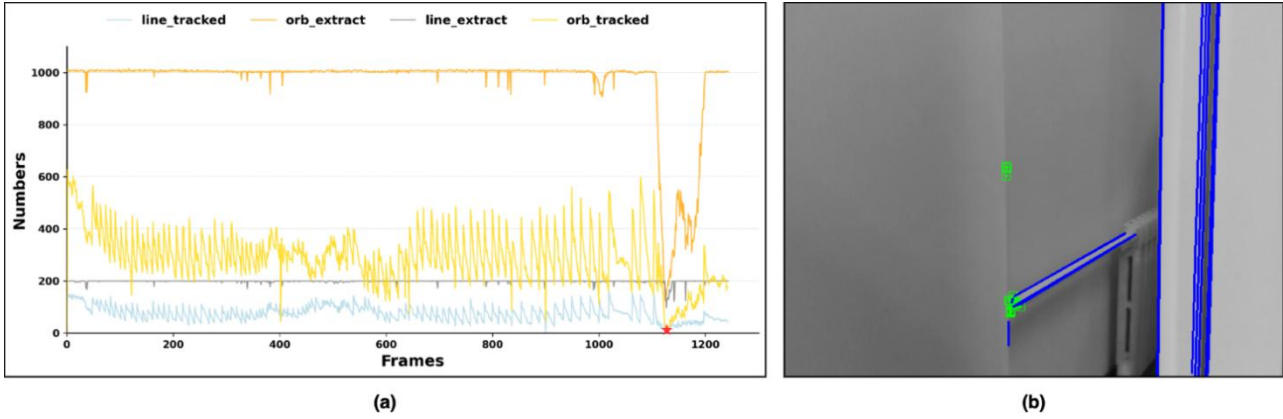


Fig. 13. (a) Number of extracted and matched point and line features; (b) Matches at the red pentagram in (a).

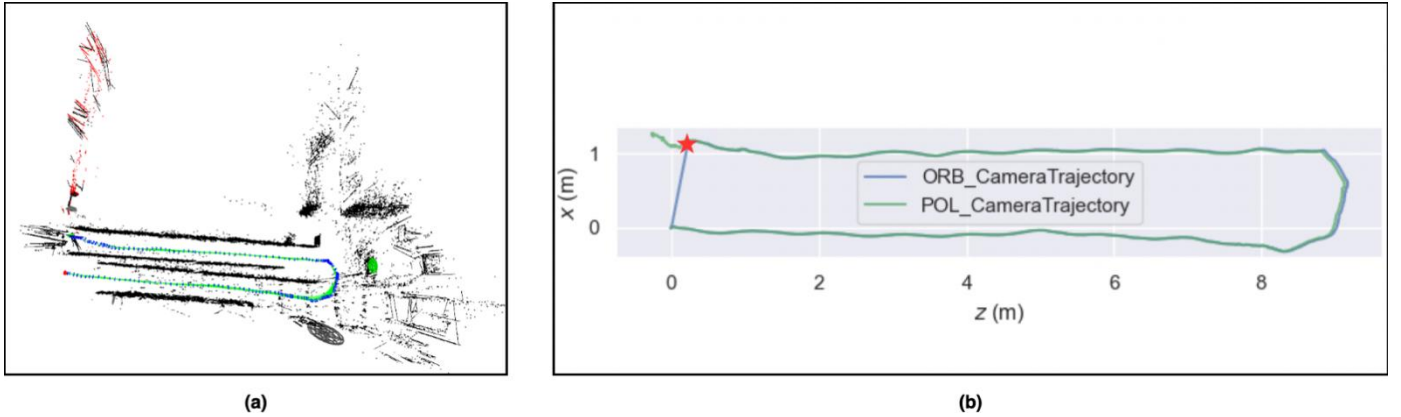


Fig. 14. (a) Sparse map by the algorithm; (b) Trajectory comparison with ORB-SLAM3.

relative pose accuracy compared to purely visual methods and handles camera motion and rotations more robustly. The IMU further reduces cumulative drift, effectively stabilizing long-term pose estimates.

4.2 Evaluation in real-world environment

To assess the system’s performance in real-world scenarios, we tested two datasets captured with a handheld RealSense D435i RGB-D camera. In the first dataset, the camera moved slowly through a classroom containing both moving and stationary people. We evaluated PLO-SLAM against RDS-SLAM and other methods in terms of feature filtering, trajectory estimation, and sparse map quality. As shown in Fig. 11, ORB-SLAM3 struggled to filter out features associated with moving people and incorrectly removed features from stationary individuals, while SG-SLAM and OVD-SLAM failed to fully eliminate edge features of dynamic objects. By contrast, PLO-SLAM more reliably detects

and removes dynamic features, resulting in cleaner maps and more accurate trajectories. Fig. 12(a) illustrates the results of multi-object tracking. The improved DeepSORT algorithm enables our system to reliably track targets even after the camera completes a 360° rotation. Beyond tracking, dynamic objects can be modeled as ellipsoids (Fig. 12b) and their motion trajectories can be estimated (brown trajectories in Fig. 12b). In Fig. 12c, other methods show noticeable trajectory deviations near walls and corners, whereas our system, by incorporating line features, substantially improves both localization and mapping accuracy.

In the second experiment, the camera navigated through the library, moving along bookshelves where line features are abundant. While no dynamic objects were present, some areas exhibited weak or degraded textures. This setting provides an opportunity to evaluate how line features contribute to the system's robustness under challenging visual conditions. Fig. 13(a) presents the variation in the number of point and line feature matches throughout the sequence. While both feature types exhibit similar overall trends, line features remain more consistent in regions with sparse textures, such as corners (marked with red stars). Fig. 13(b) highlights the successfully matched features in these areas. As shown in Fig. 14, the tracking of ORB_SLAM3 get lost at corners, whereas our system maintains stable tracking with the support of line features, confirming their role in enhancing robustness in weak-texture environments.

5 Conclusion

This paper presents a novel algorithm (i.e., PLO-SLAM) to improve the performance of Visual SLAM in dynamic scene. Object states are determined using the epipolar constraint of optical flow, which mitigates the adverse the effects of dynamic features on localization and mapping. A refined object tracking module ensures reliable cross-frame association, enabling the reconstruction of objects as 3D ellipsoids and the generation of their motion trajectories. Furthermore, line features are incorporated to enhance the representational capacity of the map and to supplement the available feature, thereby improving the robustness of PLO-SLAM in dynamic and texture-degraded scenes. Experimental results show that, compared with other state-of-the-art visual SLAM systems, the proposed system delivers superior performance in both conventional static and dynamic scenarios.

In long-term sequences characterized by limited effective features and rapid camera motion, however, the localization and mapping accuracy of PLO-SLAM still falls short of that achieved by visual-inertial systems. Future work will focus on improving localization and mapping accuracy, as well as developing a unified framework that integrates ellipsoid modeling with multi-sensor data to strengthen system robustness.

Author Contributions: Hao Chen: Conceptualization, Methodology, System implementation, Writing – original draft. Chuanguang Zhu: Experimental support, Data curation, Writing – review & editing. LiuHai Liao: Data acquisition, Writing – review & editing.

Funding: This work was supported in part by the National Natural Science Foundation of China under Grant 41901373, and in part by the Natural Science Foundation of Hunan Province under Grant 2023JJ30240 and Grant 2019JJ50190. (Corresponding author: Chuanguang Zhu.)

Conflicts of Interest: The authors declare no conflicts of interest.

REFERENCES

- [1] Durrant-Whyte H and Bailey T 2006 Simultaneous localization and mapping: part I *IEEE Robotics & Automation Magazine* **13** 99-110
- [2] Fuentes-Pacheco J, Ruiz-Ascencio J and Rendón-Mancha J M 2015 Visual simultaneous localization

and mapping: a survey *Artificial Intelligence Review* **43** 55-81

- [3] Campos C, Elvira R, Rodríguez J J G, Montiel J M M and Tardós J D 2021 ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial, and Multimap SLAM *IEEE Transactions on Robotics* **37** 1874-90
- [4] Wang R, Schwörer M and Cremers D 2017 Stereo DSO: Large-Scale Direct Sparse Visual Odometry with Stereo Cameras. In: *2017 IEEE International Conference on Computer Vision (ICCV)*, pp 3923-31
- [5] Qin T, Li P and Shen S 2018 VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator *IEEE Transactions on Robotics* **34** 1004-20
- [6] Wolf D F and Sukhatme G S 2005 Mobile robot simultaneous localization and mapping in dynamic environments *Autonomous Robots* **19** 53-65
- [7] Zhang J, Henein M, Mahony R and Ila V 2020 VDO-SLAM: A visual dynamic object-aware SLAM system *arXiv preprint arXiv:2005.11052*
- [8] Barfoot T D 2024 *State estimation for robotics*. Cambridge University Press)
- [9] Saputra M R U, Markham A and Trigoni N 2018 Visual SLAM and structure from motion in dynamic environments: A survey *ACM Computing Surveys (CSUR)* **51** 1-36
- [10] Costeira J P and Kanade T 1998 A multibody factorization method for independently moving objects *International Journal of Computer Vision* **29** 159-79
- [11] Tan W, Liu H, Dong Z, Zhang G and Bao H 2013 Robust monocular SLAM in dynamic environments. In: *2013 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*: IEEE) pp 209-18
- [12] Yu C, Liu Z, Liu X-J, Xie F, Yang Y, Wei Q and Fei Q 2018 DS-SLAM: A semantic visual SLAM towards dynamic environments. In: *2018 IEEE/RSJ international conference on intelligent robots and systems (IROS)*: IEEE) pp 1168-74
- [13] Bescos B, Fàcil J M, Civera J and Neira J 2018 DynaSLAM: Tracking, mapping, and inpainting in dynamic scenes *IEEE robotics and automation letters* **3** 4076-83
- [14] Liu Y and Miura J 2021 RDS-SLAM: Real-time dynamic SLAM using semantic segmentation methods *IEEE Access* **9** 23772-85
- [15] Wu W, Guo L, Gao H, You Z, Liu Y and Chen Z 2022 YOLO-SLAM: A semantic SLAM system towards dynamic environment with geometric constraint *Neural Computing and Applications* **34** 6011-26
- [16] Soares J C V, Gattass M and Meggiolaro M A 2021 Crowd-SLAM: visual SLAM towards crowded environments using object detection *Journal of Intelligent & Robotic Systems* **102** 50
- [17] Cheng S, Sun C, Zhang S and Zhang D 2022 SG-SLAM: A real-time RGB-D visual SLAM toward dynamic scenes with semantic and geometric information *IEEE Transactions on Instrumentation and Measurement* **72** 1-12
- [18] Cui L and Ma C 2019 SOF-SLAM: A semantic visual SLAM for dynamic environments *IEEE access* **7** 166528-39
- [19] He J, Li M, Wang Y and Wang H 2023 OVD-SLAM: An online visual SLAM for dynamic environments *IEEE Sensors Journal* **23** 13210-9
- [20] Zuo X, Xie X, Liu Y and Huang G 2017 Robust visual SLAM with point and line features. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*: IEEE) pp 1775-82
- [21] Pumarola A, Vakhitov A, Agudo A, Sanfeliu A and Moreno-Noguer F 2017 PL-SLAM: Real-time monocular visual SLAM with points and lines. In: *2017 IEEE international conference on robotics and automation (ICRA)*: IEEE) pp 4503-8
- [22] Alamanos I and Tzafestas C 2023 ORB-LINE-SLAM: an open-source stereo visual SLAM system with point and line features *Authorea Preprints*

- [23] Zeng D, Liu X, Huang K and Liu J 2024 EPL-VINS: Efficient point-line fusion visual-inertial SLAM with LK-RG line tracking method and 2-DoF line optimization *IEEE Robotics and Automation Letters* **9** 5911-8
- [24] Sarlin P-E, DeTone D, Malisiewicz T and Rabinovich A 2020 Superglue: Learning feature matching with graph neural networks. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp 4938-47
- [25] Zhao Y, Zhang H, Lu P, Li P, Wu E and Sheng B 2022 DSD-MatchingNet: deformable sparse-to-dense feature matching for learning accurate correspondences *Virtual Reality & Intelligent Hardware* **4** 432-43
- [26] McCormac J, Clark R, Bloesch M, Davison A and Leutenegger S 2018 Fusion++: Volumetric object-level slam. In: *2018 international conference on 3D vision (3DV)*: IEEE) pp 32-41
- [27] Nicholson L, Milford M and Sünderhauf N 2018 Quadricslam: Dual quadrics from object detections as landmarks in object-oriented slam *IEEE Robotics and Automation Letters* **4** 1-8
- [28] Zins M, Simon G and Berger M-O 2022 Oa-slam: Leveraging objects for camera relocalization in visual slam. In: *2022 IEEE international symposium on mixed and augmented reality (ISMAR)*: IEEE) pp 720-8
- [29] Zhu H, Bai S, Shi J, Lu J, Zuo X, Huang S and Yao X 2025 Ellipsoid-SLAM: enhancing dynamic scene understanding through ellipsoidal object representation and trajectory tracking *The Visual Computer* 1-16
- [30] Yang S and Scherer S 2019 Cubeslam: Monocular 3-d object slam *IEEE Transactions on Robotics* **35** 925-38
- [31] Wang K, Yao X, Ma N and Ran G 2025 PLMOT-SLAM: a point-line features fusion SLAM system with moving object tracking *The Visual Computer* **41** 4547-65
- [32] Hartley R and Zisserman A 2003 *Multiple view geometry in computer vision*: Cambridge university press)
- [33] Palazzolo E, Behley J, Lottes P, Giguere P and Stachniss C 2019 ReFusion: 3D reconstruction in dynamic environments for RGB-D cameras exploiting residuals. In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*: IEEE) pp 7855-62
- [34] Sturm J, Engelhard N, Endres F, Burgard W and Cremers D 2012 A benchmark for the evaluation of RGB-D SLAM systems. In: *2012 IEEE/RSJ international conference on intelligent robots and systems*: IEEE) pp 573-80
- [35] Burri M, Nikolic J, Gohl P, Schneider T, Rehder J, Omari S, Achtelik M W and Siegwart R 2016 The EuRoC micro aerial vehicle datasets *The International Journal of Robotics Research* **35** 1157-63