2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

ESO-SLAM: Tightly-Coupled and Simultaneous Estimation of Self and Multi-Object Pose via Sensor Fusion

Wu Li[1], Yunzhou Zhang[1*], Yuezhang Lv[1], Tingting Wang[1], Sizhan Wang[1], Guiyuan Wang[2]

***Abstract*— Simultaneous Localization and Mapping (SLAM) is widely used in applications such as robotics and autonomous driving, with methods involving multi-sensor fusion demonstrating excellent performance. However, they simply reject dynamic features and ignore the mutual benefits of self and dynamic objects, which greatly limits their application in actual high-dynamic scenes. To address this issue, we propose ESO-SLAM, a tightly-coupled system for simultaneous self and multi-object pose estimation achieved through sensor fusion. This system employs a multi-probability fusion tracker based on filter to establish more robust object-level data association. Building upon this, we introduce a method that combines 3D Kalman filter velocity priors and camera optical flow decoupling for dynamic point cloud removal, aiming at improving the accuracy of self-pose estimation in odometry. Finally, we jointly refine the poses of the robot and objects using multiple constraint factors within our proposed framework. Experimental results on the KITTI raw dataset demonstrate that our approach achieves better pose accuracy for both self and tracked objects compared to baseline and state-of-the-art techniques. Furthermore, the proposed method exhibits feasibility in real-time performance to ensure its practical application value.**

I. Introduction

Simultaneous Localization and Mapping (SLAM) is a crucial technology for autonomous mobile robots, primarily for real-time 3D reconstruction and localization in unknown environments. In recent years, some SLAM systems [1], [2], [3], [4] have achieved high accuracy, but most are based on the assumption of static environments and reject observations of dynamic objects during the optimization process. This limitation makes SLAM systems challenging to apply in actual high-dynamic scenes. Additionally, sensing information about the surrounding motion is required for safety and information interaction in some applications, such as autonomous driving and advanced augmented reality (AR).

Recent studies [5], [6], [7] integrate SLAM and Multiple Object Tracking (MOT) to simultaneously estimate the poses of both the robot itself and surrounding moving objects, demonstrating impressive performance and highlighting the benefits of dynamic objects for self-pose estimation. In addition, it is particularly essential to estimate the poses of the neighboring objects while ensuring accuracy in self-pose estimation, which can help robots in various goal-oriented tasks such as path planning, human-robot interaction and behavior prediction. However, most current algorithms rely on single measurement sensors [8], [9], [10], making it challenging to achieve accurate and robust estimates of both the robot and moving object poses in complex real-world environments where robots increasingly demand perception of adjacent objects. And multi-sensor fusion is a reliable solution to the presence problem.

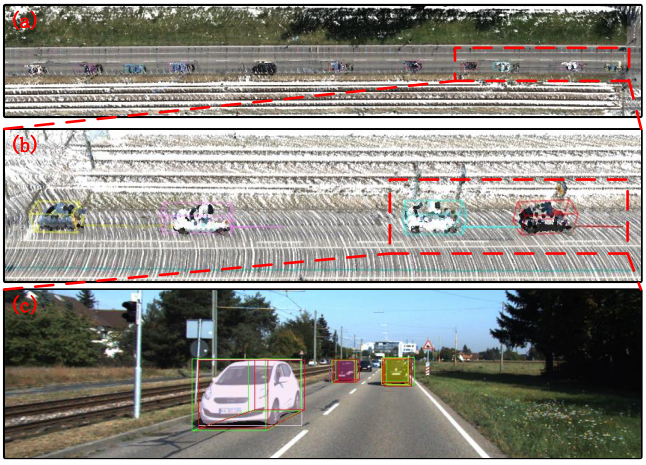

Fig. 1. **The Results of ESO-SLAM under the KITTI raw sequence 0926-0015.** (a) Trajectories and RGB-colored point cloud generated by the system. (b) Close-up of tracked objects. The object trajectory exhibits a high level of smoothness across the global scope. (c) Visualization of 3D object proposals projected onto images. The ground truth and estimated bounding boxes for objects are shown in red and green, respectively. Other colors represent bounding boxes and masks extracted by the detection and segmentation network.

Furthermore, a significant amount of current research focuses on developing precise 3D detectors [11], [12], [13] to estimate object poses, which heavily rely on the performance of the detectors. They aim to enhance detection accuracy using single-frame information but overlook information related to historical observations of both the self and surrounding objects. To retain more valuable observation information during object tracking, object data association plays a vital role. Nevertheless, existing object data association mostly relies on visual features [14], [15] or intersections of 2D or 3D detection box [16], [17], which cannot wholly avoid tracking or optimization failures such as occlusion due to rapid object movement or detector false positives. Additionally, some studies [18], [19], [20] focus on estimating object poses based on specific objects or model priors, which are often designed for particular scenarios and are not easily scalable to general objects.

To address the above issues, we propose a unified multi-

*The corresponding author of this paper
[1]Wu Li, Yunzhou Zhang, Yuezhang Lv, Tingting Wang and Sizhan Wang are with College of Information Science and Engineering, Northeastern University, Shenyang, China. zhangyunzhou@mail.neu.edu.cn
[2]Guiyuan Wang is with Jiangsu Shuguang Optoelectronics Co., Ltd., Yangzhou, China.
This work was supported by National Natural Science Foundation of China (No. 61973066) and Major Science and Technology Projects of Liaoning Province(No. 2021JH1/10400049).

object tracking framework that integrates LiDAR, camera, and IMU data to estimate the poses of both the ego vehicle and dynamic objects accurately. In terms of object perception, we only utilize the pre-trained model of Mask R-CNN [21] without any further processing, ensuring minimal cost and adaptability to different types of LiDAR sensors. We establish continuous and stable object-level association of 2D detected objects and initial state estimation of objects using a multi-threaded fusion tracker based on Kalman filter to handle occlusion, loss, etc., and ensure long-term tracking. Unlike most visual multi-object tracking methods [9], [22], [23], we do not need to extract additional image features or recover depth. Instead, we achieve 2D optical flow tracking by projecting LiDAR points onto the image and further refine the poses of the ego vehicle and objects in a sliding window to ensure global consistency in our 3D reconstruction and final trajectory, as shown in Fig. 1. The main contributions of this work are summarized as follows:

- A multi-probability tracker for optical flow tracking and 2D intersection over union (IoU) fusion association is proposed, and the tracking points and 3D bounding box are constantly updated to improve the robustness and accuracy of object association and tracking.
- A method combining the 3D Kalman filter with velocity prior and camera optical flow is proposed to decouple object velocity, realizing the removal of dynamic points to enhance the accuracy of ego-motion estimation.
- A compact LiDAR-Inertia-Camera fusion multi-object tracking framework is proposed. The state optimization factor graph under tightly-coupled multi-constraints is constructed in a sliding window to enhance robustness and local consistency and realize joint estimation of self and object poses.
- Extensively validates the proposed system on a publicly available KITTI Raw Data dataset. The experimental results show that our method achieves superior accuracy and real-time operation.

II. Related Works

In this section, we mainly introduce algorithms for tracking and estimating dynamic objects instead of simply rejecting dynamic feature points or masking. Early dynamic perception SLAM primarily relies on loosely coupled methods based on vision, typically tracking and reconstructing dynamic objects based on semantic information from images. In dynamic and static object tracking, the groundbreaking work SLAMMOT [24] utilizes a filtering-based mathematical framework to estimate the motion of the camera and multiple objects separately. Qiu *et al*. [25] develops a method for estimating the 3D motion of dynamic objects based on monocular visual-inertial sensors, solving object scale ambiguity in monocular vision through geometric analysis based on motion correlation without requiring prior knowledge of object scale. ClusterVO [14] clusters landmarks and performs multi-level probabilistic association using a constructed heterogeneous conditional random field to solve the poses of both the camera and dynamic objects.

As dynamic objects are proven beneficial for self-pose estimation, the tightly-coupled methods of multi-object tracking and visual SLAM have received increasing attention. Specifically, CubeSLAM [18] recovers the 3D vertex coordinates of cubes using 2D object detection information between adjacent frames and combines with multi-view bundle adjustment (BA) to jointly optimize poses of cameras, objects and feature points. VDO-SLAM [8] performs object data association and tracking using dense optical flow fields of object instances, showing more refined estimation results. Still, the introduction of multiple networks significantly limits real-time performance. DynaSLAM II [7] implements object data association by employing instance segmentation and visual features and jointly optimizes scene structure, camera pose, and object trajectory within a fixed temporal window. Twist-SLAM [15] creates point sets based on semantic classes and uses optical flow tracking to achieve data association. Then, it models mechanical joints between clusters to construct inter-cluster constraints, but it does not adequately address the 6-DoF estimation of objects. Recently, VIMOT [17] utilizes 3D bounding box generated by the object detector in images to represent and correlate object instances, integrating multi-object tracking into VINS to optimize camera and object poses jointly.

Compared to the extensive research on vision-based methods, Tian *et al*. [5] proposed DL-SLOT, a dynamic LiDAR SLAM that tightly couples the ego and object states within a sliding window. However, this method filters out all potential movable objects without selection, which can lead to sparse features. Recently, Lin *et al*. [6] introduces LIO-SEGMOT, which implements an asynchronous state updates with odometry and object tracking, and designs a hierarchical criterion to prevent system instability caused by poor 3D object detection. Nevertheless, LIO-SEGMOT does not mitigate the influence of dynamic points on ego-motion estimation. Furthermore, the aforementioned LiDAR-based methods rely on the accuracy of 3D object detection and do not utilize LiDAR observation information to refine object trajectory further.

In contrast, we utilize geometric methods to obtain object features based on perceptual results. What sets us apart from other works is that we use the interframe optical flow to decouple the self and object-level motion to remove the influence of dynamic points on the odometry. At the same time, we conduct kinematic modeling of the object to ensure the overall smoothness of the object estimation. Finally, we combine multiple constraint factors in the sliding window to jointly estimate the poses of self and the surrounding moving objects to achieve the most advanced performance.

III. System Overview

A. Notation

In this section, we will provide a detailed description of the proposed system. The symbols are defined as illustrated in Fig. 2. We denote the IMU, camera, LiDAR, objects, and global reference frame as $\{I\}$, $\{C\}$, $\{L\}$, $\{O\}$, and $\{W\}$, respectively. The pose of frame $\{A\}$ at the time t_k

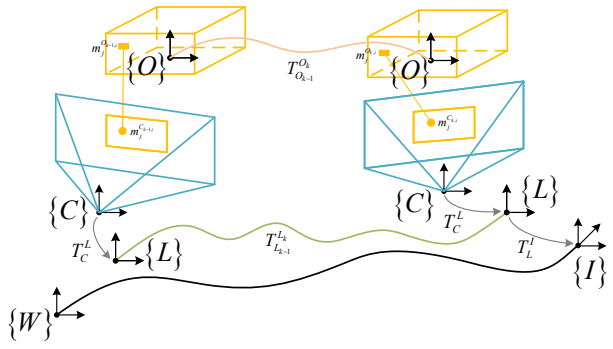

Fig. 2. Symbol representation in different reference frames.

is specified by A_k and this pose in frame $\{B\}$ is represented by $T^B_{A_k} \in \mathrm{SE}(3)$. The transformation of frame $\{A\}$ from the previous moment at t_{k-1} to the current moment at t_k is denoted by $T^{A_k}_{A_{k-1}} \in \mathrm{SE}(3)$. Besides, we denote the pose and velocity of the i-th object at the moment t_k of the reference frame $\{A\}$ as $T^A_{O_{k,i}}$, $T^A_{V_{O_{k,i}}} \in \mathrm{SE}(3)$, and the relative poses and velocities from the previous moment t_{k-1} to the current moment t_k are represented by ${}_A T^{O_k}_{O_{k-1,i}} \in \mathrm{SE}(3)$, ${}_A V^{O_k}_{O_{k-1,i}} \in \mathbb{R}^3$, respectively. The transformation relationship of the object pose is as follows:

$$ {}_W T^{O_k}_{O_{k-1,i}} \cdot T^W_{O_{k-1,i}} = T^W_{O_{k,i}} \tag{1} $$

Let the j-th point on the i-th object at the moment t_k be $m_j^{A_{k,i}} = [m_{j_x}^{A_{k,i}}, m_{j_y}^{A_{k,i}}, m_{j_z}^{A_{k,i}}, 1]^T$ under the reference frame $\{A\}$. Note that for a point $m_j^{O_{k,i}} = m_j^{O_{k-1,i}}$ of a rigid body object, the transformation of the object point in the world coordinate frame is represented as:

$$ T^W_{O_{k,i}} \cdot m_j^{O_{k,i}} = T^W_{C_k} \cdot T^C_{O_{k,i}} \cdot m_j^{O_{k,i}} = T^W_{C_k} \cdot T^{C_k}_{C_{k-1}} \cdot T^C_{O_{k-1,i}} \cdot m_j^{O_{k-1,i}} \tag{2} $$

B. System Architecture

Fig. 3 depicts the framework of our system, which takes monocular images, LiDAR point cloud, and IMU data as input, performs 2D inference tasks on each image frame and then acquires the optical flow features of the object by LiDAR projection onto the image. In the object data association section, the Hungarian algorithm [26] solves the association matrix constructed by 2D IoU and optical flow inner points. For objects successfully tracked, we predict the initial state of the object using the 3D Kalman filter or the Perspective-n-Point (PNP) estimation. To ensure the correctness of the 3D Kalman filter prediction, we decouple the 2D optical flow of object features and estimate the object's current velocity combined with velocity priors to precisely reject dynamic points while also re-estimating the object's 3D bounding box and updating the optical flow. Then, the Kalman filter judges and creates, deletes, or updates object trajectories. To refine the states of self and tracking objects, we feed them into a tightly-coupled factor graph for joint optimization to obtain accurate localization and maintain a globally consistent map of multi-object trajectories and RGB-color point cloud in real-time.

IV. PROPOSED METHOD

A. Preprocessing

For object perception, we use Mask R-CNN [21] to provide 2D bounding boxes and corresponding masks for 3D objects in each image frame, which serve as inputs to the tracking module. Some algorithms rely on precise 3D detection or more complex detection combinations to obtain object information, which can be challenging to port to a broader range of sensors or require higher costs, posing a significant challenge for implementing low-power devices and ensuring real-time performance, which is not desired. Since the detection module is not our focus, we can explore replacing it with lighter detection models in the future, which often involve a trade-off between accuracy and speed.

Due to differences in detection accuracy and viewing angle, an object's edge information is unreliable, which may result in the loss of some information or cause depth errors in the projected point cloud. Additionally, some LiDAR point cloud may be occluded and invisible in the camera. Simply projecting all LiDAR points onto objects for tracking may lead to incorrect calculations and introduce additional errors that will affect the optimization results. We employ three simple and effective methods to address this issue to obtain correct object points.

- We project LiDAR points into the image through LiDAR-camera transformation, retaining only the points closest to each pixel to obtain the camera-visible points.
- We use geometric clustering to remove object points with inconsistent depths quickly.
- Within clusters that may contain object points, we use the nearest neighbor search for rapid iteration to find potentially missing object points.

B. Object Association and Management

To ensure the performance and accuracy of object association, we design a multi-probability tracker. In addition to projecting LiDAR point cloud onto the image to achieve Lucas-Kanade (LK) sparse optical flow tracking of object features, we utilize a 2D Kalman filter to maintain only the 2D bounding boxes of objects. We perform the Hungarian algorithm [26] to match the 2D bounding boxes predicted by the constant model with the detected 2D bounding boxes. Apart from the 2D IoU, the cost function includes an additional metric based on the optical flow field: the number of inliers to handle associations in more complex scenes, such as when occlusion occurs. We predict the positions of the projected LiDAR points using optical flow and calculate the number of these points within the detected 2D bounding boxes of objects in the current image frame. Therefore, the overall cost function is as follows:

$$ \sigma_{k,i} = \left(1 - \frac{N\left(O^{2d}_{k,i}, \hat{p}_{k,i}\right)}{N\left(O^{2d}_{k-1,i}, p_{k-1,i}\right)}\right) \cdot \left(1 - \frac{\hat{O}^{2d}_{k,i} \cap O^{2d}_{k,i}}{\hat{O}^{2d}_{k,i} \cup O^{2d}_{k,i}}\right) \tag{3} $$

where $N(\cdot)$ is the number of features within the object's 2D bounding box, p and O are the positions in the image for the

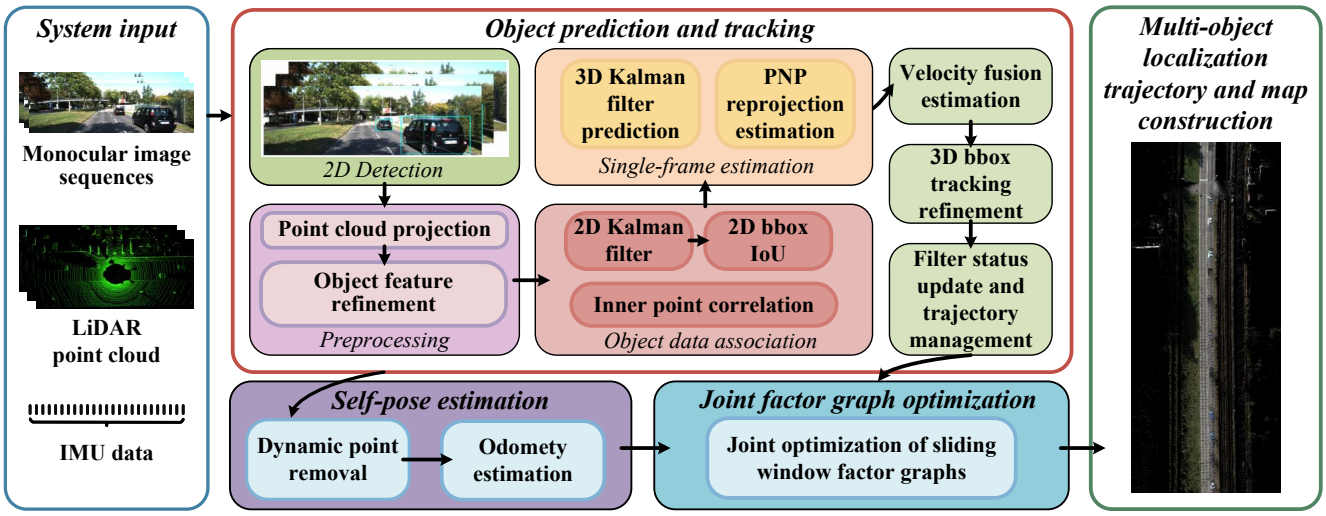

Fig. 3. **The framework of our proposed system.** We first obtain object features by LiDAR points projection onto the image over the object perception result to avoid additional feature extraction or depth recovery (See IV-A). Then we construct an association matrix based on optical flow tracking and 2D bounding box(bbox) IoU fusion to realize object association (See IV-B). To ensure good initial estimation of the object, we fuse the velocity prior and camera optical flow in the 3D Kalman filter to estimate the object velocity and predict the initial state of the object (See IV-C). We use the velocity estimation results to cull the dynamic point cloud to improve odometry accuracy (See IV-D), and then perform joint state estimation in a sliding window for self and multiple objects (See IV-E). Finally we publish globally consistent object trajectories and RGB-colored point cloud (See IV-F).

tracked point of the optical flow and the 2D bounding box, respectively. $\hat{(\cdot)}$ represents the tracked and predicted position.

Object-matching between neighboring frames can be achieved by solving the cost function. Objects will inherit their matched trajectory ID and undergo further optimization if a successful matching relationship is obtained. Additionally, we employ different association periods to address inconsistencies in object information due to occlusion, misidentification or loss. If an object has never been tracked, we assign it a new ID and delete it when the α frames are consecutively exceeded without a successful match. If an object has been tracked for a certain period and is not successfully matched by any new detections in the subsequent λ frames, we delete it. This approach allows us to have some tolerance for the detection module, thus ensuring continuous object tracking in the system.

C. Initial Estimation

To ensure the optimization effectiveness of objects, we use a 3D Kalman filter to predict the object's state at the current frame and update the object's 3D bounding box. We continuously refine the 3D bounding boxes in the filter for each frame of object points to ensure that they are all inside. We define the state vector $T \in \mathbb{R}^{15}$ of the object as follows:

$$T = \left[t_{O_i}^W, A_{O_i}^W, L_{O_i}, V_{O_i}, \omega_{O_i}\right] \tag{4}$$

where $A_{O_i}^W$ represents the angles of rotation around the three axes for the i-th object, $L_{O_i} \in \mathbb{R}^3$ are the length, width, and height of the object, V_{O_i} and $\omega_{O_i} \in \mathbb{R}^3$ represent the velocity and angular velocity of the object, respectively.

In addition, we perform forward and backward optical flow algorithms to retain more accurate feature matching points. Based on the PNP or 3D Kalman filter, we obtain the predicted state of the object and retain the one with the maximum number of inliers as the initial result for sliding window optimization. Since we use the constant velocity model to approximate object motion, for a more accurate object prediction, we decouple the motion state of the object based on the 2D optical flow of the camera and integrate the velocity prior of the Kalman filter to improve the accuracy of the velocity estimation. Given a pair of tracked 2D optical flow points, we find the optimal estimate $[V, \omega] \in \mathbb{R}^{2\times 3}$ that minimizes the distance error:

$$[V,\omega] = \underset{[V,\omega]\in\mathbb{R}^{2\times 3}}{\arg\min} \sum_{p\in O_i} \left\| \theta\left(d(p)' - d\left(\hat{p} - p\right)\right)\right\|_{\Sigma} \tag{5}$$

where $d(\cdot)$ denotes the optical flow distance of the object point moving on the image, $\theta(\cdot)$ is the Huber norm, Σ is the covariance matrix, $(\cdot)'$ represents the estimated motion distance of the object, which can be defined as:

$$\begin{aligned}\hat{d}(p) = \begin{bmatrix} p_u \\ p_v \end{bmatrix} &= \begin{bmatrix} -f_x/Z & 0 & x/Z \\ 0 & -f_y/Z & y/Z \end{bmatrix} \begin{bmatrix} V_x \\ V_y \\ V_z \end{bmatrix} \\ &+ \begin{bmatrix} xy/f_x & -\left(f_x + x^2/f_x\right) & y \\ f_y + y^2/f_y & -xy/f_y & -x \end{bmatrix} \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}\end{aligned} \tag{6}$$

As the optical flow field combines self and object motion, we transform the 3D points from the LiDAR frame to the object frame based on the rigid assumption. We use Ceres optimizer to solve this least squares problem and decouple the object's motion state. For the tracked object, we can also obtain the object's velocity state prior $[V,\omega]_{KF}$ and covariance matrix $\sum_{KF}$ from the 3D Kalman filter. By fusing the velocity and covariance decoupled from the optical flow estimation, the updated velocity is represented as:

$$K = \Sigma_{KF} \cdot \left(\Sigma_{KF} + \Sigma\right)^{-1} \tag{7}$$

$$[V,\omega]_{\text{update}} = [V,\omega]_{KF} + K \cdot \left([V,\omega] - [V,\omega]_{KF}\right) \tag{8}$$

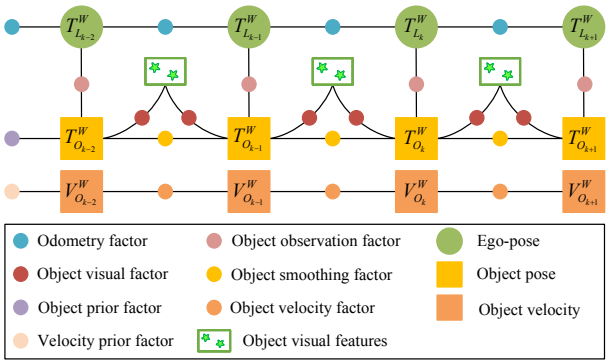

Fig. 4. Illustrates the description of the factor graph framework proposed for joint optimization, where all factors in the figure collectively impose constraints to enhance the accuracy of state estimation of ego vehicle and surrounding objects.

D. Self-pose Estimation

We do not focus much on odometry work but instead implemented self-pose estimation within the framework of an Iterated Error State Kalman Filter based on Fast-LIO2 [27]. The Odometry module primarily utilizes IMU forward propagation for state prediction and performs motion compensation on point cloud based on backward propagation. The point cloud in the local map are organized and maintained by the ikd-Tree [28]. Finally, the pose is updated within a tightly coupled iterative Kalman filter framework, showing promising results across various autonomous driving datasets. We integrate this work into our system and combine it with multi-object tracking to achieve a unified framework for joint estimation. Furthermore, before odometry iteration updates, we utilize the velocity above fusion estimation (See IV-C) to remove LiDAR points from dynamic objects. In theory, we can replace the current odometry with a more accurate, arbitrary multi-sensor fusion framework containing LiDAR while improving the robustness and accuracy of the system in highly dynamic scenes.

E. Joint Factor Graph Estimation in Sliding Window

If objects are continuously tracked successfully, we employ sliding window techniques to ensure the local consistency of object states and the real-time performance of the system, enabling batch optimization of frames for continuous object tracking. The factor graph optimization framework is shown in Fig. 4, which integrates the state nodes of the ego-vehicle and surrounding objects and the constraint edges provided by various factors to achieve a tightly coupled joint estimation. Initially, we acquire odometry results (See IV-D) and register the localization of dynamic objects. Within the sliding window, we introduce odometry factors, object observation factors, visual factors, object smoothing factors, object velocity factors, and prior factors for joint optimization, effectively constraining the poses of both the ego vehicle and objects.

The odometry factors constrain the ego vehicle's pose; object observation factors constrain the relative transformation of the pose between the ego vehicle and the object; visual factors constrain the object's pose; object smoothing factors constrain the pose and velocity of the object, and object velocity factors constrain the velocity consistency of the object between two frames. In order to maintain the consistency of the sliding window and to make the system fast to execute, we choose to marginalize the state of the oldest frame instead of adding the Hessian block of dense landmarks that is also used to constrain the first pose of the batch data optimization framework. Given the ego vehicle's pose at the time t_k and the pose at the previous time t_{k-1}, the odometry factor is defined as:

$$\gamma_o\left(T^W_{L_{k-1}}, T^W_{L_k}\right) = \mathrm{Log}\left(\left(T^{W\ -1}_{L_{k-1}} \cdot T^W_{L_k}\right) \cdot T^{L_k}_{L_{k-1}}\right) \tag{9}$$

where $\mathrm{Log}(\cdot)$ is a mapping from the Lie group in $\mathrm{SE}(3)$ to its Lie algebra in $\mathfrak{se}(3)$.

After the initial estimation of the object is obtained, the object's observation factor at the current time is denoted as:

$$\gamma_c\left(T^W_{L_k}, T^W_{O_{k,i}}\right) = \mathrm{Log}\left(\left(T^{W\ -1}_{L_k} \cdot T^W_{O_{k,i}}\right) \cdot T^{L\ -1}_{O_{k,i}}\right) \tag{10}$$

For the visual features of dynamic objects, we apply the rigid body assumption to transform the points of the object in the world coordinate frame to static points in the object coordinate frame and project them onto the image. Then, the image visual factor for the j-th dynamic feature of the i-th object at the time t_k can be represented as:

$$\gamma_p\left(T^W_{O_{k,i,j}}\right) = p^C_{O_{k,i,j}} - \pi\left(T^{W\ -1}_{C_k} \cdot T^W_{O_{k,i}} \cdot m^{O_{k,i}}_j\right) \tag{11}$$

where $\pi(\cdot)$ represents the camera projection model, and $p^C_{O_{k,i,j}} \in \mathbb{R}^2$ denotes the projected point's actual pixel location on the image plane.

Similar to [6], we also employ the Constant Linear and Angular Velocity (CLAV) model to characterize the motion of moving objects. The smoothness factor is defined as:

$$\begin{aligned}\gamma_s\left(T^W_{O_{k-1,i}}, T^W_{O_{k,i}}, T^W_{V_{O_{k-1,i}}}\right) = \\ \mathrm{Log}\left(T^W_{O_{k-1,i}} \cdot \mathrm{Exp}\left(T^W_{V_{O_{k-1,i}}} \cdot \delta^{t_k}_{t_{k-1}}\right) \cdot T^{W\ -1}_{O_{k,i}}\right)\end{aligned} \tag{12}$$

where $T^W_{V_{O_{k-1,i}}}$ consists of the linear and angular velocity of the object, $\delta^{t_k}_{t_{k-1}}$ denotes the timestamp difference between two frames, and $\mathrm{Exp}(\cdot)$ is the inverse operation of $\mathrm{Log}(\cdot)$.

To ensure the continuity of the tracked object trajectory, we assume that the dynamic object moves at a constant velocity over a short period. The definition of the continuous velocity factor for continuously tracked objects is as follows:

$$\gamma_v\left(T^W_{V_{O_{k-1,i}}}, T^W_{V_{O_{k,i}}}\right) = \mathrm{Log}\left(T^W_{V_{O_{k-1,i}}} \cdot T^{W\ -1}_{V_{O_{k,i}}}\right) \tag{13}$$

The final optimization problem of the system is defined as Eq. 14, where the best estimates of the self-vehicle and the tracked object are obtained by minimizing the sum of the prior and the Mahalanobis norm of all measurement residuals. Where χ is the set of all variables, $\gamma_{prior}(\chi)$ is the marginalized prior residual term, and $\Sigma_O, \Sigma_C, \Sigma_P, \Sigma_S, \Sigma_V$ are the covariance matrices of the corresponding factors. Each frame of dynamic objects set $O_{k,i}$ and observed object points $m^{O_{k,i}}_j$ are closely related to the pose of ego vehicle, ensuring the robustness of the system in highly dynamic scenes and satisfactory accuracy of both self and object trajectories.

$$\chi^* = \arg\min_{\chi}\Bigg\{ \left\|\gamma_{\text{prior}}(\chi)\right\|^2 + \sum_{k\in O}\left\|\gamma_o\left(T^W_{L_{k-1}}, T^W_{L_k}\right)\right\|^2_{\Sigma_O} + \sum_{k,i\in C}\left\|\gamma_c\left(T^W_{L_k}, T^W_{O_{k,i}}\right)\right\|^2_{\Sigma_C} + \sum_{k,i,j\in P}\left\|\theta\left(\gamma_p\left(T^W_{O_{k,i,j}}\right)\right)\right\|^2_{\Sigma_P} + \sum_{k,i\in S}\left\|\gamma_s\left(T^w_{O_{k-1,i}}, T^W_{O_{k,i}}, T^W_{V_{O_{k-1,i}}}\right)\right\|^2_{\Sigma_S} + \sum_{k,i\in V}\left\|\gamma_v\left(T^W_{V_{O_{k-1,i}}}, T^W_{V_{O_{k,i}}}\right)\right\|^2_{\Sigma_V}\Bigg\} \tag{14}$$

F. Map Data Publication

To ensure the system's efficient operation, we open a new thread of parallel processing for map publishing. Due to the removal of invisible points in different viewing angles, we can obtain accurate color information of the point cloud from the image through the LiDAR to camera extrinsic parameter transformation and the proper estimation of self-pose to reconstruct a precise global and consistent RGB-color point cloud map. For the tracked object, we distinguish between dynamic and static to publish the colored point cloud of either the last frame or all frames and all the motion trajectories.

V. Experience

We evaluate the performance of the proposed method on self and object state estimation on the KITTI raw dataset [29]. In the experiment, the sliding window size, α, and λ parameters are set to 8, 5, and 2, respectively. All experiments are completed on a laptop computer with an Intel i7-12700 CPU, 16GB RAM, and a GeForce RTX 3070 8GB graphics card.

A. Evaluation Details

The KITTI raw dataset [29] contains the ground truth of the camera poses and sensor extrinsic params, as well as manually annotated of some object poses, which effectively validate the performance of our proposed method. In addition, it provides complete LiDAR scans, camera, and IMU data for evaluating multi-object tracking approach through sensor fusion. We assess the accuracy of self-pose estimation using the Root Mean Square Error (RMSE) of the Absolute Trajectory Error ($\text{ATE}_\text{T}[\text{m}]$ and $\text{ATE}_\text{R}[\text{rad}]$) [30], and for multi-object tracking accuracy assessment, we additionally report the RMSE of the Relative Positional Error ($\text{RPE}_\text{T}[\text{m/m}]$ and $\text{RPE}_\text{R}[^\circ/\text{m}]$).

B. Self-pose Estimate Evaluation

We quantitatively evaluate the self-pose estimation of our system in dynamic scenes. To further validate the performance of our proposed methods, when the system is loosely coupled, the Multi-Object Tracking (MOT) is performed separately from the odometry, and only the proposed dynamic feature point removal (See IV-D) is executed, this method is referred to as "Ours(DR)". Furthermore, the method without dynamic point removal and joint optimization is referred to as "Ours(Initial)".

As shown in Tab. I, the proposed method outperforms the no extra processing odometry benchmark regarding translation and rotation errors. Compared with the process without removing the dynamic points, the pose estimation accuracy using only the static points is also better, which verifies the common phenomenon that most odometry has poor accuracy in dynamic scenes. On the other hand, we observe that the absolute pose error of self also exceeds that of the loosely coupled method when jointly optimized, proving that the simultaneous estimation of the poses of self and the dynamic object can be mutually beneficial. It's worth noting that the accuracy improvement of "Ours(DR)" is mainly reflected in scenes containing lots of dynamic objects, such as sequences 0018 and 0020, as filtering methods are less sensitive to low-dynamic scenes.

TABLE I

Comparison of self-pose estimation on the KITTI raw dataset. Bold and underlined text indicates the best and the suboptimal results. ATE_T is in m, ATE_R in rad.

Sequence	Ours(Initial)		Ours(DR)		Ours	
	ATE_T	ATE_R	ATE_T	ATE_R	ATE_T	ATE_R
0003	<u>0.247</u>	<u>0.070</u>	**0.246**	<u>0.070</u>	0.249	**0.068**
0005	0.496	0.093	<u>0.494</u>	<u>0.092</u>	**0.476**	**0.090**
0010	0.536	**0.140**	<u>0.533</u>	**0.140**	**0.524**	**0.140**
0011	0.287	0.038	**0.274**	**0.031**	<u>0.277</u>	<u>0.033</u>
0018	0.419	0.202	<u>0.402</u>	**0.200**	**0.394**	<u>0.201</u>
0019	<u>1.784</u>	<u>0.066</u>	<u>1.784</u>	<u>0.066</u>	**1.783**	**0.064**
0020	1.853	0.189	<u>1.768</u>	<u>0.175</u>	**1.732**	**0.171**
Mean	0.803	0.114	<u>0.786</u>	<u>0.111</u>	**0.776**	**0.110**

C. Object-Motion Evaluation

To investigate the effectiveness of our proposed multi-object tracking approach, we evaluate the pose accuracy of objects in different sequences. Tab. II presents the comparison results with other state-of-the-art algorithms, where the last column is the number of consecutive optimized frames for objects. We snub some frames with fewer object points to prevent optimization errors. The results for DynaSLAM II [7], VIMOT [17], and TwistSLAM [15] are directly obtained from their respective papers.

As can be seen from Tab. II, our method achieves better object pose accuracy, with optimal trajectory errors of 0.26m, 0.08m/m, and 0.37°/m for ATE, RPE_T, and RPE_R, respectively, which are significantly improved 72.04%, 86.89%, and 96.87% compared to DynaSLAM II. This is because DynaSLAM II is affected by occlusion or rapid movement and does not have enough feature points to model the object. Although VIMOT and TwistSLAM attempt to address this issue using additional methods like optical flow, they require significant time for feature extraction. In addition, TwistSLAM constrains the 3-DoF pose of the object through mechanical joints between dynamics and specific static clusters, avoiding the error caused by extra optimization variable constraints. In contrast, we employ the multi-probability fusion tracker (See IV-B) and good initial estimation (See IV-C) to robustly track the 6-DoF pose of the

TABLE II

COMPARISON OF OBJECT POSE ESTIMATION ON THE KITTI RAW DATASET. BOLD AND UNDERLINED TEXT INDICATES THE BEST AND THE SUBOPTIMAL RESULTS. ATE IS IN M, RPE_T IN M/M, RPE_R IN °/M.

Sequence-ID	DynaSLAM II [7]			VIMOT [17]			TwistSLAM [15]			Ours			Optimize frame numbers
	ATE	RPE_T	RPE_R	ATE	RPE_T	RPE_R	ATE	RPE_T	RPE_R	ATE	RPE_T	RPE_R	
0003-01	0.69	0.34	1.84	0.63	0.27	0.65	0.31	0.10	**0.28**	**0.15**	**0.06**	0.55	52
0005-31	0.51	0.26	13.50	0.46	0.15	**0.35**	**0.35**	0.19	0.58	0.45	**0.08**	0.46	267
0018-02	1.10	0.30	9.27	0.70	0.19	1.01	**0.21**	0.27	0.66	0.24	**0.06**	**0.27**	226
0018-03	1.13	0.55	20.05	0.43	0.25	**0.30**	**0.15**	0.21	0.56	0.23	**0.10**	0.40	89
0020-00	0.56	0.45	1.30	0.44	0.13	0.23	0.17	0.20	0.70	**0.13**	**0.07**	**0.21**	118
0020-12	1.18	0.40	6.19	0.91	0.25	**0.04**	0.24	0.20	1.54	**0.23**	**0.06**	0.41	240
0020-122	0.87	0.72	5.75	–	–	–	**0.17**	**0.02**	**0.07**	0.42	0.09	0.29	94
0010-00	0.95	0.40	2.84	–	–	–	0.77	0.21	1.98	**0.51**	**0.10**	**0.48**	265
0011-00	1.05	0.43	12.51	–	–	–	**0.17**	0.23	**0.23**	0.35	**0.04**	0.37	268
0011-35	1.25	0.89	16.64	–	–	–	**0.10**	**0.03**	**0.11**	0.13	0.09	0.29	31
0019-63	0.86	1.45	48.80	–	–	–	0.28	2.17	1.08	**0.25**	**0.10**	**0.53**	109
0019-72	0.99	1.12	3.36	–	–	–	0.16	**0.05**	0.34	**0.08**	0.06	**0.17**	119
Mean	0.93	0.61	11.83	0.60	0.21	0.43	**0.26**	0.32	0.68	**0.26**	**0.08**	**0.37**	156.5

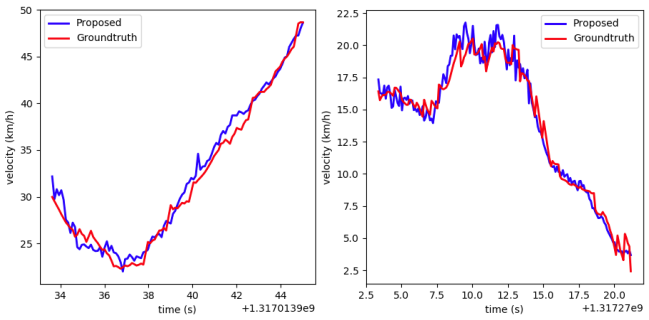

Fig. 5. Comparison of ground truth and estimated fusion velocity. The left column is object ID 31 in sequence 05 and the right column is object ID 02 in sequence 18.

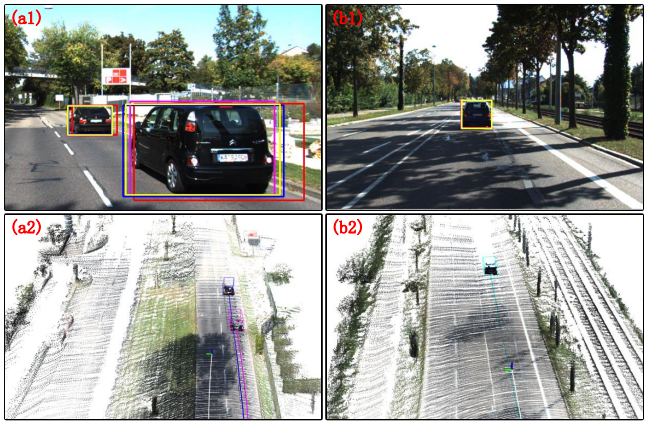

Fig. 6. Qualitative example of partial object tracking on the KITTI raw dataset. (a1) and (b1) are the update results of the 2D bounding boxes of the object. Red, purple, pink and yellow are the 2D bounding boxes of the last frame update, current frame detection, prediction and update of the object respectively. (a2) and (b2) are object tracking results.

object and refine the object state in the sliding window (See IV-E) to achieve optimal accuracy. Furthermore, our system achieves a smoother result in interframe relative object pose estimation, which is reflected in considerable advancements in the RPE_T and RPE_R metrics, mainly due to our well-designed state factor graph in the sliding window. We further compared the fusion velocity estimation results of the first two objects in the evaluation object with tracking of more than 100 frames. Fig. 5 shows the result that the fusion velocity estimation of the object is close to the reference velocity, and the changing trend of the object velocity can be accurately tracked, which also provides reliable constraint information for the pose optimization of the object.

In sequences 0020 and 0019, the ATE metric of self-pose estimation accuracy reaches a high error, while the pose accuracy of moving objects attains the optimal result. Specifically, the accuracy of object estimation depends largely on self-pose accuracy. Since the manual annotations used to evaluate the object accuracy on the KITTI raw dataset are in the camera coordinate frame, the introduction of self-pose errors is avoided. We also show the qualitative results of self and various vehicle movements on different sequences of the KITTI raw dataset in Fig. 6. These examples demonstrate the global consistency and smoothness of object tracking.

D. System Efficiency Analysis

In this section, we evaluate the time efficiency of different modules of the system in two representative sequences. The main time-consuming components and results are shown in Tab. III. Compared to solutions using visual features optimization, our system can maintain a lower computation time between scenes with different degrees of dynamics, because the number of points scanned by the same type of LiDAR is always similar and odometry and dynamic object tracking are complementary. Meanwhile, our system adopts a framework combining filtering and optimization, which enables rapid completion of odometry and object association. In addition, it avoids feature extraction or depth recovery to reduce overall time consumption, which fully guarantees the real-time performance and the ability to optimize the batch data and show a more efficient result.

VI. CONCLUSION

This paper presents ESO-SLAM, a dynamic object-aware approach that integrates simultaneous self-pose estimation and multi-object tracking through sensor fusion. We formulate a multi-probability fusion tracker to conduct data association and use the rigid assumption to transform the LiDAR point cloud to the object coordinate frame to execute state decoupling, thereby eliminating dynamic points to im-

TABLE III
AVERAGE RUNTIME OF DIFFERENT COMPONENTS

Components	Time Consume(ms)	
	KITTI 0003	KITTI 0020
Pre-processing	11.66±0.18	18.78±0.30
LiDAR Odometry	42.54±1.03	31.75±0.48
Data Association and Tracking	6.24±0.05	8.10±0.13
Sliding Window Optimization	6.87±0.09	7.29±0.04

Frame rate	DynaSLAM II [7]	VIMOT [17]	Proposed
FPS	10-12	10-11	14-15

prove odometry accuracy. In addition, a compact framework is developed to fuse LiDAR, camera, and IMU data and combine multiple constraint factors in a sliding window to optimize self and multi-object poses jointly. Experimental results exhibit that our proposed method achieves superior accuracy compared to state-of-the-art dynamic SLAM techniques, indicating the mutual benefits of simultaneous self and multi-object estimation. Moreover, our algorithm is devoted to more types of LiDAR and guarantees computational efficiency in a wide range of applications. Noteworthy, when the most challenging scenarios occur where the sparse object point cloud due to object material, viewpoint change, or gradually moving away from the car, how to ensure the optimization accuracy is an exciting idea and our next steps.

REFERENCES

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

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

[3] H. Wei, T. Zhang, and L. Zhang, "A fast analytical two-stage initial-parameters estimation method for monocular-inertial navigation," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–12, 2022.

[4] X. Zhao, C. Wang, and M. H. Ang, "Real-time visual-inertial localization using semantic segmentation towards dynamic environments," *IEEE Access*, vol. 8, pp. 155 047–155 059, 2020.

[5] X. Tian, Z. Zhu, J. Zhao, G. Tian, and C. Ye, "DL-SLOT: Tightly-coupled dynamic lidar slam and 3d object tracking based on collaborative graph optimization," *IEEE Transactions on Intelligent Vehicles*, vol. 9, no. 1, pp. 1017–1027, 2024.

[6] Y.-K. Lin, W.-C. Lin, and C.-C. Wang, "Asynchronous state estimation of simultaneous ego-motion estimation and multiple object tracking for lidar-inertial odometry," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 10 616–10 622.

[7] B. Bescos, C. Campos, J. D. Tardós, and J. Neira, "DynaSLAM II: Tightly-coupled multi-object tracking and slam," *IEEE robotics and automation letters*, vol. 6, no. 3, pp. 5191–5198, 2021.

[8] J. Zhang, M. Henein, R. Mahony, and V. Ila, "VDO-SLAM: A visual dynamic object-aware slam system," *arXiv preprint arXiv:2005.11052*, 2020.

[9] H. Zhang, H. Uchiyama, S. Ono, and H. Kawasaki, "MOTSLAM: Mot-assisted monocular dynamic slam using single-view depth estimation," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 4865–4872.

[10] Y. Qiu, C. Wang, W. Wang, M. Henein, and S. Scherer, "AirDOS: Dynamic slam benefits from articulated objects," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 8047–8053.

[11] H. A. Hoang, D. C. Bui, and M. Yoo, "TSSTDet: Transformation-based 3-d object detection via a spatial shape transformer," *IEEE Sensors Journal*, 2024.

[12] H. Wu, C. Wen, S. Shi, X. Li, and C. Wang, "Virtual sparse convolution for multimodal 3d object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 21 653–21 662.

[13] X. Wu, L. Peng, H. Yang, L. Xie, C. Huang, C. Deng, H. Liu, and D. Cai, "Sparse fuse dense: Towards high quality 3d detection with depth completion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5418–5427.

[14] J. Huang, S. Yang, T.-J. Mu, and S.-M. Hu, "ClusterVO: Clustering moving instances and estimating visual odometry for self and surroundings," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 2168–2177.

[15] M. Gonzalez, E. Marchand, A. Kacete, and J. Royan, "Twistslam: Constrained slam in dynamic environment," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 6846–6853, 2022.

[16] M. Gladkova, N. Korobov, N. Demmel, A. Ošep, L. Leal-Taixé, and D. Cremers, "DirectTracker: 3d multi-object tracking using direct image alignment and photometric bundle adjustment," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 3777–3784.

[17] S. Feng, X. Li, C. Xia, J. Liao, Y. Zhou, S. Li, and X. Hua, "VIMOT: A tightly-coupled estimator for stereo visual-inertial navigation and multi-object tracking," *IEEE Transactions on Instrumentation and Measurement*, 2023.

[18] S. Yang and S. Scherer, "CubeSLAM: Monocular 3-d object slam," *IEEE Transactions on Robotics*, vol. 35, no. 4, pp. 925–938, 2019.

[19] G. B. Nair, S. Daga, R. Sajnani, A. Ramesh, J. A. Ansari, K. M. Jatavallabhula, and K. M. Krishna, "Multi-object monocular slam for dynamic environments," in *2020 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2020, pp. 651–657.

[20] J. K. Murthy, G. S. Krishna, F. Chhaya, and K. M. Krishna, "Reconstructing vehicles from a single image: Shape priors for road scene understanding," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 724–731.

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

[22] M. Shan, Q. Feng, and N. Atanasov, "Orcvio: Object residual constrained visual-inertial odometry," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 5104–5111.

[23] I. A. Bârsan, P. Liu, M. Pollefeys, and A. Geiger, "Robust dense mapping for large-scale dynamic environments," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 7510–7517.

[24] C.-C. Wang, C. Thorpe, S. Thrun, M. Hebert, and H. Durrant-Whyte, "Simultaneous localization, mapping and moving object tracking," *The International Journal of Robotics Research*, vol. 26, no. 9, pp. 889–916, 2007.

[25] K. Qiu, T. Qin, W. Gao, and S. Shen, "Tracking 3-d motion of dynamic objects using monocular visual-inertial sensing," *IEEE Transactions on Robotics*, vol. 35, no. 4, pp. 799–816, 2019.

[26] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[27] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "Fast-lio2: Fast direct lidar-inertial odometry," *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2053–2073, 2022.

[28] Y. Cai, W. Xu, and F. Zhang, "ikd-tree: An incremental kd tree for robotic applications," *arXiv preprint arXiv:2102.10808*, 2021.

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

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