# DynPL-SVO: A Robust Stereo Visual Odometry for Dynamic Scenes

Baosheng Zhang, Xiaoguang Ma, Hong-Jun Ma and Chunbo Luo



***Abstract*—Most feature-based stereo visual odometry (SVO) approaches estimate the motion of mobile robots by matching and tracking point features along a sequence of stereo images. However, in dynamic scenes mainly comprising moving pedestrians, vehicles, etc., there are insufficient robust static point features to enable accurate motion estimation, causing failures when reconstructing robotic motion. In this paper, we proposed DynPL-SVO, a complete dynamic SVO method that integrated united cost functions containing information between matched point features and re-projection errors perpendicular and parallel to the direction of the line features. Additionally, we introduced a *dynamic grid* algorithm to enhance its performance in dynamic scenes. The stereo camera motion was estimated through Levenberg-Marquard minimization of the re-projection errors of both point and line features. Comprehensive experimental results on KITTI and EuRoC MAV datasets showed that accuracy of the DynPL-SVO was improved by over 20% on average compared to other state-of-the-art SVO systems, especially in dynamic scenes.**

***Index Terms*—Stereo visual odometry(SVO), dynamic scenes, motion estimation, line features.**

## I. Introduction

VISUAL odometry (VO) is a popular research topic in the fields of robotics, autonomous driving, and augmented reality. It uses various types of cameras to estimate its mobile motion and reconstruct surrounding map [1] [2]. The stereo visual odometry (SVO) has attracted more attention recently due to its low cost, robustness, and wide applicability for both indoor and outdoor scenes [3] [4].

Most VO systems rely solely on point features for motion estimation, as they are easy to detect, track, and handle [5] [6]. However, point features have poor anti-interference ability and are sensitive to lighting variations, occlusion, and rapid motion, limiting the performance of point-feature-based SVO. Introducing line features can effectively address these issue by providing more constraints, improving the accuracy and stability of camera pose estimation. Moreover, in dynamic scenes, it is difficult to extract sufficient static point features to estimate pose of robots, and line features extracted from static regions in the scene can complement the deficiencies of point features in dynamic scenes.

Baosheng Zhang and Xiaoguang Ma are with the College of Information Science and Engineering, Northeastern University, 110819 Shenyang, China (Xiaoguang Ma is the corresponding author, e-mail: maxg@mail.neu.edu.cn).

Hongjun Ma is with the School of Automation Science and Engineering, South China University of Technology, Guangzhou 510641, China (e-mail: mahongjun@scut.edu.cn).

Chunbo Luo is with the School of Information and School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China (e-mail: c.luo@uestc.edu.cn).

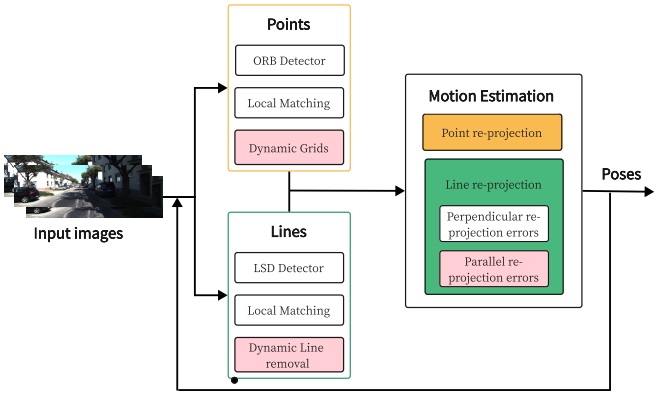



Fig. 1. Overview of the DynPL-SVO.

Dealing with dynamic scenes is a significant challenge for traditional VO methods, as they often fail to achieve accurate inter-frame matching, causing low motion estimation accuracy. This was commonly solved by accurately removing dynamic outliers in images before pose optimization, meaning that the system must remove features introduced by the dynamic objects and rely solely on trusted static features for the motion estimation. Therefore, accurately extracting and removing the dynamic features is critical for improving the performance of VO systems in the dynamic scenes.

The robust constraints and random sample consensus (RANSAC) had been applied to remove outliers (dynamic points) [7], and Mask R-CNN is an excellent semantic segmentation network in detecting dynamic objects in scenes [8]. However, the above methods have serious efficiency shortcomings, limiting their application in computer vision problems that require real-time performance. As an online camera pose estimation system, VO estimates the camera's current pose based on the previous frame motion models, and geometric constraints can be used to identify dynamic regions in the scene. In this paper, we proposed *dynamic grid* approach which used spatial geometric constraints as a prerequisite for mitigating the impact of dynamic scenes on VO, and did not require additional depth information or an explicit understanding of the scene.

In this paper, we proposed DynPL-SVO, a robust and complete SVO system to fully use the structural and geometric information of images to improve accuracy, especially for dynamic scenes.

The main contributions of this paper were listed below:

- We proposed a complete SVO system, named DynPL-SVO, which utilized both point and line features and was

capable of effectively coping with dynamic scenes using only stereo RGB images from stereo cameras.
- The re-projection errors of the point features and re-projection errors perpendicular and parallel to line features were jointly used to construct a unified cost function for pose optimization, resulting in superior robustness compared to conventional feature-based approaches.
- A *dynamic grid* approach was carefully designed to identify dynamic regions and remove dynamic features, and improvement of 13.6% and 2.3% on absolute pose error (APE) and relative pose error (RPE), respectively, were made on highly dynamic scenes, such as KITTI-01, 05, and 09.
- Extensive comparative experiments were conducted on both the KITTI and the EuRoC MAV dataset to validate performance of the DynPL-SVO. The results showed that, the translation RMSE drifts of the DynPL-SVO were improved by 13.8%, 30.0%, and 24.8% as compared to those of PL_SLAM front end, ORB_SLAM2 front end, and ORB_Line SLAM front end, respectively.

## II. Related Work

VO methods could be divided into direct-based [9] [10] and feature-based [5] according to how visual measurements were processed. Direct-based methods used intensity of each pixel to compute camera's motion by minimizing photometric errors, without detecting and matching specific features. However, most VO systems used feature-based methods due to their high robustness and estimation accuracy, wherein feature descriptors were used to detect and track point features and estimate motion by minimizing the re-projection errors between detected features and their corresponding projected features from frames. To ensure real-time performance and reliability of the VO system, many researchers used ORB [5] [11], which provides robust and accurate motion estimation in rich-texture scenes even with only point features. However, the accuracy of motion estimation with insufficient point features can greatly degrade in poor-texture scenes. As a result, many researchers introduced line features to improve robustness and accuracy [11] [12] [13], and several methods have achieved satisfactory results in detecting straight-line features, such as FLD [14], EDLine [15], and LSD [16].

One common and concise way to represent line features is to use two endpoints to model a 3D line [17]. To avoid optimization constraints brought by the over-parameterized representation of line features, researchers [11] [18] applied orthogonal representation [19] and $Plücker$ coordinate [13] to transform and optimize line features, respectively.

Optimization of line features required various representations with corresponding cost functions. Koletschka [17] used the Euclidean distance sum of equally spaced sampling points on the line segments as cost functions for the line features. In most methods [11, 13, 18, 20], the line re-projection errors were defined as the Euclidean distance from the endpoints of the detected line features in a current frame to their projections in a previous frame. All these cost functions could be easily computed using $Plücker$ coordinates and the distance formula between points and lines. However, previous studies only considered structural information perpendicular to the direction of line features, and implementing a cost function that integrated re-projection errors perpendicular and parallel to line features could substantially enhance SVO performance.

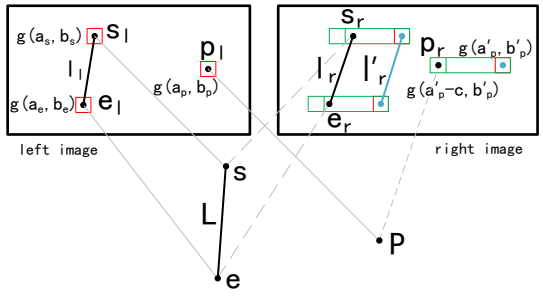



Fig. 2. The feature matching process between the left and right images in a stereo frame. We assumed that the point $p_l$ detected in the left image in the $grid(a_p, b_p)$ (red square), followed the epipolar constraint and imaging principle of the stereo camera. The point (such as $p_r$) to be matched in the right image should be located at the range from $grid(a'_p - c, b'_p)$ to $grid(a'_p, b'_p)$ in the right image, i.e., the green rectangular area. Similar to point matching, if the endpoints of the line $l_r$ detected in the right image met the matching rule, it would be a candidate line of $l_l$.

Estimating motion in dynamic scenes poses a significant challenge to VO systems. Several works [21, 22] have addressed this issue by leveraging vision information to fuse other sensors, such as IMU and wheel odometry. Additionally, numerous methods [23] used RGB-D information to identify dynamic areas where unstable point features were removed, allowing only stable static point features to be retained during optimization. More recently, deep learning-based solutions were combined with VO systems to provide feasible solutions to handle dynamic scenes [24]. However, all these methods impose strict requirements on scenario conditions and computer resources, and a novel SVO approach that could achieve accurate motion estimation in dynamic scenes without relying on image depth information or assistance from other sensors would be highly desirable.

## III. Detection and Pre-Processing

### A. Feature detection and matching

*1) Point features:* After detecting an ORB point feature $p_l(u_l, v_l)$ in the left image and its corresponding feature $p_r(u_r, v_r)$ in the right image of a stereo frame, as illustrated in Figure 2, the next step was to match these features between the two images. To achieve this, we evenly divided each image into 64×48 grids and stored the point features according to their grid positions within the images. It is essential to note that the feature matching must follow the epipolar constraint, i.e., $v_l = v_r$, and comply with the imaging principle, i.e., $u_l > u_r$. The point features required for matching in the right image were limited to horizontal grids, ranging from $grid(a'_p - c, b'_p)$ to $grid(a'_p, b'_p)$. Additionally, a mutual consistency check was performed, meaning that only matches with corresponding best-left and best-right matches were deemed valid.

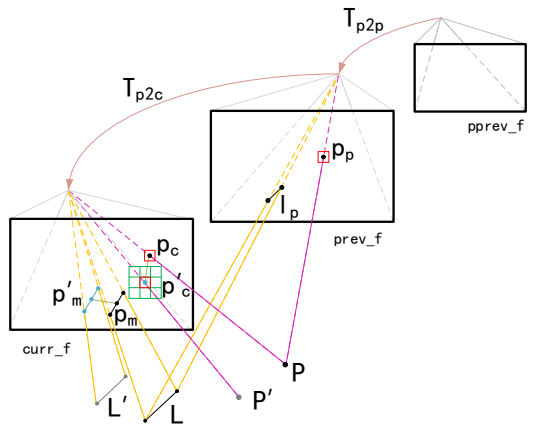



Fig. 3. The feature tracking process between adjacent frames. $T_{p2p}$ denoted pose transformation between the previous frame $prev\_f$ and its preceding frame $pprev\_f$. $T_{p2c}$ denoted the pose transformation between $prev\_f$ and the current frame $curr\_f$. The dynamic spatial point $P$ and spatial line $L$ moved to $P'$ and $L'$ during sampling time, respectively. For line features $L$, we defined their re-projection errors in a similar way with point features by utilizing the Euclidean distance between midpoints $p'_m$ and $p_m$ of matched and estimated line features. Based on this, we classify a feature as dynamic or not.

Figure 3 illustrated the feature tracking process between adjacent frames. To mitigate the impact of dynamic object features on our estimation results, we utilized a motion model to estimate the locations of the matched point features in the current frame $curr\_f$. Specifically, we employed a uniform motion model represented by a pose transformation matrix $T_{p2p}$ between the previous frame $prev\_f$ and its preceding frame $pprev\_f$ as the initial state of the motion model. Given that dynamic objects exhibited abnormal motion relative to static scenes, the dynamic spatial point $P$ moved to $P'$ during sampling time, resulting in larger re-projection errors for dynamic features $p'_c$ compared to estimated features $p_c$ predicted by the motion model. To identify *dynamic grid*s, we computed and averaged the sum of squared Euclidean distances between matched point features and estimated features in all relevant grids. If this value exceeded a threshold, we classified the grid and its surrounding eight grids in green as *dynamic grid*s, and the point features within these grids were identified as dynamic point features. This enabled us to accurately track and estimate feature locations in complex and dynamic scenes.

Algorithm 1 showed detailed process of proposed dynamic region marking based on *dynamic grid* algorithm.

*2) Line features:* In this work, line features were extracted from images using LSD and represented using LBD. Similar to the point feature matching process between the left and right images as shown in Figure 2, we grouped the line features in the right image that passed through the same grids. We assumed that the endpoints of the line feature $l_l$ in the left image were located in $(a_s, b_s)$ and $(a_e, b_e)$. In the right image, only lines within corresponding grids were considered as candidate matches. To ensure the accuracy of the matches, we applied a line matching rule and performed a mutual consistency check, as illustrated in Figure 2. Only matches that satisfy these requirements were selected for further processing.

**Algorithm 1** : Dynamic region marking using the *dynamic grid*.

**Input:** Uniform motion model $T_{p2p}$ of the previous frame $prev\_f$, the matched point feature set between the current frame $curr\_f$ and $prev\_f$;
**Output:** The location of the *dynamic grid*.

```
Divide curr_f evenly into 64×48 grids and only keep n(n <= 8) point features p_j in each grid g_i;
for each g_i ∈ curr_f do
    for each p_j ∈ g_i do
        e_{g_i} += PointErr(p_j, T_p2p)/n
    end for
    if e_{g_i} > ρ then
        GRIDS_LOCATION += {(x_{g_i} − 1, y_{g_i} − 1) ∼ (x_{g_i} + 1, y_{g_i} + 1)}
    end if
end for
return GRIDS_LOCATION
```

In Figure 3, we showed the use of re-projection errors between the estimated line midpoint $p_m$ and matched line midpoint $p'_m$ as the criterion for dynamic line feature tracking between adjacent frames. Once the errors exceed a preset threshold, the line features were identified as dynamic ones and removed. Furthermore, effective *dynamic grid*s were obtained in the scene, as mentioned above.

### B. Representation of the Line Features

We assumed that the homogeneous coordinates of line endpoints were represented by $\bar{X}_s(x_1, y_1, z_1, w_1)$ and $\bar{X}_e(x_2, y_2, z_2, w_2)$, with their inhomogeneous counterparts denoted as $X_s$ and $X_e$. The *Plücker* coordinates of the line $L$ could then be constructed using the following formula:

$$L = \begin{bmatrix} X_s \times X_e \\ w_2 X_s - w_1 X_e \end{bmatrix} = \begin{bmatrix} \mathbf{n} \\ \mathbf{d} \end{bmatrix} \in \mathbb{R}^6 \tag{1}$$

, where $\mathbf{d}$ represented the direction vectors of the line, and $\mathbf{n}$ denoted the normal vectors of the plane determined by the lines and the origin. Specifically, $\mathbf{n}^T \times \mathbf{d} = 0$. The *Plücker* coordinates for $L$ could also be extracted from the dual *Plücker* matrix $T^*$, which was defined as follows:

$$T^* = \begin{bmatrix} \mathbf{d}^\wedge & \mathbf{n} \\ -\mathbf{n}^T & 0 \end{bmatrix} \tag{2}$$

, where $\wedge$ denoted transformation between vectors and anti-symmetric matrices. The representation of the *Plücker* coordinates was chosen due to its convenience for line feature projection, transformation, and Jacobian usage.

## IV. Motion Estimation

### A. Problem Statement

The primary objective of VO systems is to find an optimal transformation that can satisfy the projection constraints for the corresponding features with high accuracy. This can be achieved by solving a non-linear least-squares equation

formed by the projection constraints of corresponding features between the adjacent frames.

Compared to other VO systems that rely solely on the re-projection errors of point features and the re-projection errors perpendicular to the direction of line features. In this paper, we proposed DynPL-SVO, a complete dynamic SVO that integrated cost functions containing information between matched point features and re-projection errors perpendicular and parallel to the direction of the line features, effectively leveraging the rich structural information contained in the endpoints of line features detected by LSD, leading to increased robustness and accuracy. The non-linear least-squares equation for the proposed method was shown in following:

$$\xi^* = arg\min_{\xi} \left[ \sum_{i=1}^{m} e_i^p(\xi)^T \Sigma_{e_i^p}^{-1} e_i^p(\xi) + \sum_{j=1}^{n} e_j^{l_{pe}}(\xi)^T \Sigma_{e_j^{l_{pe}}}^{-1} e_j^{l_{pe}}(\xi) + \sum_{k=1}^{q} e_k^{l_h}(\xi)^T \Sigma_{e_k^{l_{pa}}}^{-1} e_k^{l_{pa}}(\xi) \right] \tag{3}$$

, where $m$, $n$, and $q$ respectively denoted the numbers of points, lines, and the numbers of complete line features extracted within the image. The equation included point re-projection errors ($e_i^p$), re-projection errors perpendicular to the line direction ($e_k^{l_{pe}}$), and re-projection errors parallel the direction of line features ($e_j^{l_{pa}}$). The $\Sigma^{-1}$ matrices in (3) represented the inverse covariance matrices related to the uncertainty of each re-projection error term.

The re-projection errors of point features were defined as the distance between projected point features from the previous frames and the corresponding ones in the current frames:

$$e_i^p = p_i - p_i'(\xi) \tag{4}$$

, where $p_i$ represented points detected in the current frames and $p_i'(\xi)$ represented points projected from the previous frames into the current frames.

In previous SVO systems [13] [18], only the perpendicular re-projection errors of line features were employed in the motion estimation process. This involved calculating the distance from the endpoints of detected line features to the projected infinite line features, expressed as:

$$e_j^{l_{pe}} = \begin{bmatrix} d(l_j, p_{j,s}'(\xi)) \\ d(l_j, p_{j,e}'(\xi)) \end{bmatrix} \tag{5}$$

, where $p_s'$ and $p_e'$ represented the endpoints of line features, and $d(.)$ was the distance function from endpoints to lines.

For translational differences between two lines, we employed their midpoints' re-projection residue in the most concise form. Additionally, using midpoints to represent re-projection residuals could facilitate the reuse of point feature code and improve code readability, ultimately saving system running time. Thus, this work introduced re-projection errors parallel to line features for optimization purposes, using the midpoints of line features, i.e.,

$$e_k^{l_{pa}} = p_{i,m} - p_{i,m}'(\xi) \tag{6}$$

, where $p_{i,m}$ represented the midpoints of the lines detected in the current frames, while $p_{i,m}'(\xi)$ represented the midpoints of the lines projected from the previous frames into the current frames. By utilizing these two constraint conditions, the re-projection errors between lines could be effectively aligned with their actual pose relationships. This was important as relying on one single constraint for line features might sometimes hinder the ability to ensure consistency between re-projection errors and the true pose relationship between lines.

The optimization problem in equation (3) could be solved iteratively using the Levenberg-Marquardt algorithm.

### *B. Jacobian Matrix of the Re-Projection Errors of Points and Lines*

*1) Jacobian of the point re-projection errors:* We used six-dimensional vectors $\xi \in \mathfrak{se}(3)$ to represent the pose transformation matrix $T \in SE(3)$, and the Jacobian of point features was expressed as follows:

$$J_p = \frac{\partial e_i^p}{\partial \delta \xi} = \frac{\partial e_i^p}{\partial P'} \frac{\partial P'}{\partial \delta \xi} \tag{7}$$

, where $P'$ represented the 3D point of matched point feature from the previous frame to the current camera frame. The Jacobian could be divided into two parts using the chain rule. The first part could be expressed by the camera projection principle as follows:

$$\frac{\partial e_i^p}{\partial P'} = -\begin{bmatrix} \frac{\partial u}{\partial X'} & \frac{\partial u}{\partial Y'} & \frac{\partial u}{\partial Z'} \\ \frac{\partial v}{\partial X'} & \frac{\partial v}{\partial Y'} & \frac{\partial v}{\partial Z'} \end{bmatrix} = -\begin{bmatrix} \frac{f_x}{Z'} & 0 & -\frac{f_x X'}{Z'^2} \\ 0 & \frac{f_y}{Z'} & -\frac{f_y Y'}{Z'^2} \end{bmatrix} \tag{8}$$

and the second part could be obtained through the Lie algebra perturbation model:

$$\frac{\partial P'}{\partial \delta \xi} = \frac{\partial TP}{\partial \delta \xi} \Rightarrow \begin{bmatrix} I & -P'^{\wedge} \end{bmatrix} \tag{9}$$

, where $[.]^{\wedge}$ denoted skew-symmetric matrix of a vector. The Jacobian of point re-projection errors could be rewritten as:

$$\begin{aligned} J_p &= \frac{\partial e_i^p}{\partial \delta \xi} = \frac{\partial e_i^p}{\partial P'} \frac{\partial P'}{\partial \delta \xi} \\ &= -\begin{bmatrix} \frac{f_x}{Z'} & 0 & -\frac{f_x X'}{Z'^2} & -\frac{f_x X' Y'}{Z'^2} & f_x + \frac{f_x X'^2}{Z'^2} & -\frac{f_x Y'}{Z'} \\ 0 & \frac{f_y}{Z'} & -\frac{f_y Y'}{Z'^2} & -f_y - \frac{f_y Y'^2}{Z'^2} & \frac{f_y X' Y'}{Z'^2} & \frac{f_y X'}{Z'} \end{bmatrix} \end{aligned} \tag{10}$$

Detailed mathematical derivation could be found in [11] [18] .

*2) Jacobian of the re-projection errors perpendicular to line features:* The re-projection errors perpendicular to line features were similar to the expression presented in [20] [25], and were defined in equation (5). Firstly, we converted the 3D line $L_w$ from the world frame to the current camera frame using the following procedure:

$$L_c = \begin{bmatrix} \mathbf{n_c} \\ \mathbf{d_c} \end{bmatrix} = T_{cw} L_w = \begin{bmatrix} R_{cw} & (t_{cw})^{\wedge} R_{cw} \\ 0 & R_{cw} \end{bmatrix} L_w \tag{11}$$

, where $R_{cw}$ and $t_{cw}$ represented the rotation matrix and translation vector, respectively. Next, the 3D line $L_c$ was projected onto normalized image planes and represented as $l'$, using known intrinsic parameter matrix of cameras, i.e.,

$$l' = \mathcal{K} L_c = \begin{bmatrix} f_y & 0 & 0 \\ 0 & f_x & 0 \\ -f_y c_x & -f_x c_y & f_x f_y \end{bmatrix} \mathbf{n_c} = \begin{bmatrix} l_1 \\ l_2 \\ l_3 \end{bmatrix} \tag{12}$$

, where $\mathcal{K}$ represented the projection matrix of the line. Since we projected the line features onto the image plane as an infinite line, only the normal component $\mathbf{n_c}$ in the $Plücker$ coordinates $L_c$ provided meaningful information during projection. As mentioned in Section IV-A, the re-projection errors perpendicular to line features could be expressed as follows:

$$e_j^{l_{pe}} = \begin{bmatrix} d_s \\ d_e \end{bmatrix} = \begin{bmatrix} \frac{p_s^T l'}{\sqrt{l_1^2+l_2^2}} \\ \frac{p_e^T l'}{\sqrt{l_1^2+l_2^2}} \end{bmatrix} \tag{13}$$

We assumed $l = \sqrt{l_1^2 + l_2^2}$ and $d = \frac{\mathbf{p^T} l'}{l}$, and the Jacobian of perpendicular line re-projection errors could be expressed as:

$$\begin{aligned} \frac{\partial d}{\partial \delta\xi} = \frac{\partial \frac{\mathbf{p^T} l'}{l}}{\partial \delta\xi} &= \frac{\partial (u\, l_1 + v\, l_2 + l_3)}{\partial \delta\xi} \frac{1}{l} \\ &= \begin{bmatrix} l1 & l2 \end{bmatrix} \frac{\partial \begin{bmatrix} u \\ v \end{bmatrix}}{\partial \delta\xi} \frac{1}{l} \end{aligned} \tag{14}$$

Equation below could be obtained using the chain rule:

$$\frac{\partial \begin{bmatrix} u \\ v \end{bmatrix}}{\partial \delta\xi} = \frac{\partial \begin{bmatrix} u \\ v \end{bmatrix}}{\partial P'} \frac{\partial P'}{\partial \delta\xi} \tag{15}$$

Referring to equation (10), the Jacobian of perpendicular line re-projection errors could be expressed as follows:

$$\begin{aligned} \frac{\partial d}{\partial \delta\xi} = &- \begin{bmatrix} l1 & l2 \end{bmatrix} \\ &\begin{bmatrix} \frac{f_x}{Z'} & 0 & -\frac{f_x X'}{Z'^2} & -\frac{f_x X' Y'}{Z'^2} & f_x + \frac{f_x X'^2}{Z'^2} & -\frac{f_x Y'}{Z'} \\ 0 & \frac{f_y}{Z'} & -\frac{f_y Y'}{Z'^2} & -f_y - \frac{f_y Y'^2}{Z'^2} & \frac{f_y X' Y'}{Z'^2} & \frac{f_y X'}{Z'} \end{bmatrix} \frac{1}{l} \end{aligned} \tag{16}$$

$$J_{l_{pe}} = d_s \frac{\partial d_s}{\partial \delta\xi} + d_e \frac{\partial d_e}{\partial \delta\xi} \tag{17}$$

In comparison to methods of directly deriving the Lie algebra of the transformation, as presented in [11] [18], obtaining the Jacobian of re-projection errors perpendicular to line features was more efficient and convenient over ones using derivation results from line endpoints.

*3) Jacobian of the re-projection errors parallel to line features:* Similar to previous section, we defined the re-projection errors of matched and projected line midpoints as the cost function for the re-projection errors parallel to line features, i.e.,

$$\frac{\partial e_k^{l_{pa}}}{\partial \delta\xi} = \frac{\partial e_k^{l_{pa}}}{\partial P'_m} \frac{\partial P'_m}{\partial \delta\xi} \tag{18}$$

, where $P'_m$ represented the 3D midpoint of the matched line feature from the previous frame to the current frame. Referring to (10), we could obtain Jacobian of the re-projection errors parallel to line features as follows:

$$\begin{aligned} J_{l_{pa}} &= \frac{\partial e_k^{l_{pa}}}{\partial \delta\xi} \\ &= - \begin{bmatrix} \frac{f_x}{Z'_m} & 0 & -\frac{f_x X'_m}{Z'^2_m} & -\frac{f_x X'_m Y'_m}{Z'^2_m} & f_x + \frac{f_x X'^2_m}{Z'^2_m} & -\frac{f_x Y'_m}{Z'_m} \\ 0 & \frac{f_y}{Z'_m} & -\frac{f_y Y'_m}{Z'^2_m} & -f_y - \frac{f_y Y'^2_m}{Z'^2_m} & \frac{f_y X'_m Y'_m}{Z'^2_m} & \frac{f_y X'_m}{Z'_m} \end{bmatrix} \end{aligned} \tag{19}$$

Thus far, we have obtained the Jacobians for the re-projection errors of all three types of features. The pose transformation between adjacent frames could be achieved by solving the non-linear least-square equation (3) using the Levenberg-Marquardt algorithm.

## V. Experimental Results and Analysis

For fair comparison, we used only the front end of PL_SLAM, ORB_SLAM2, and ORB_Line SLAM to form VO systems to avoid possible influences caused by loop closure detection in these systems. All experiments were conducted on an Intel Core i5-4210U CPU @ 1.70GHz × 4 and 16GB RAM without GPU acceleration.

### A. Performance on KITTI

We tested the DynPL-SVO on the KITTI dataset, which provided ground truth trajectories based on a 64-channel Velodyne LiDAR sensor and GPS localization. Additionally, the presence of dynamic scenes containing moving objects such as cars and pedestrians in some sequences had a significant impact on the performance of VO/vSLAM systems.

We presented the absolute pose error (APE) in Table I along with three other benchmark systems, wherein we listed the root-mean-square error (RMSE) of absolute translation and rotation errors for all methods. It showed that the DynPL-SVO outperformed other methods in 9 sequences, demonstrating its superior accuracy in motion estimation across most sequences, especially those that were highly dynamic, such as sequences 01, 05, and 09. The translation RMSE drifts of the DynPL-SVO were improved by 13.8%, 30.0%, and 24.8% averagely compared to those of PL_SLAM front end, ORB_SLAM2 front end, and ORB_Line SLAM front end, respectively, indicating its superior efficiency.

Table II presented the relative pose error (RPE) comparisons, where the DynPL-SVO achieved better translation and rotation accuracy over other comparative systems in most scenes, with an average improvement of 14.8% and 2.1%, respectively, further confirming its superior performance. Figure 4 depicted the reconstructed paths of the four SVO systems on several sequences of the KITTI dataset, wherein the ones from benchmark methods had larger deviations over the DynPL-SVO in all three sequences, particularly around corners with large viewpoint changes. This further confirmed that the DynPL-SVO outperformed other SVO systems, specifically in dealing with dynamic environments.

In addition, we conducted an ablation study of the DynPL-SVO. Table III showed that the DynPL-SVO with only re-projection errors parallel to line features achieved better accuracy in 10 sequences compared to the system without line re-projection, and improved accuracy by 30.9% as compared to those using only the re-projection errors perpendicular to line features in terms of RPE on the KITTI dataset, illustrating importance of considering them in SVO systems.

### B. Performance on EuRoC MAV

We compared the DynPL-SVO and PL_SLAM front end and performed an ablation study on the EuRoC MAV dataset

TABLE I
MEAN ABSOLUTE RMSE IN THE KITTI DATASET, WITH THE DASH INDICATING FAILED EXPERIMENT.

| Seq. | DynPL-SVO | | PL_SLAM front end | | ORB_SLAM2 front end | | ORB_Line SLAM front end | |
|---|---|---|---|---|---|---|---|---|
| | $t_m$ | $R_{deg}$ | $t_m$ | $R_{deg}$ | $t_m$ | $R_{deg}$ | $t_m$ | $R_{deg}$ |
| 00 | **6.691** | 1.788 | 7.426 | 2.105 | 14.076 | 3.461 | 7.551 | **1.519** |
| 01 | **172.502** | **8.910** | 371.245 | 12.212 | - | - | - | - |
| 02 | 21.653 | 4.423 | **8.167** | **1.505** | 14.187 | 2.615 | 11.276 | 2.550 |
| 03 | 6.077 | 4.308 | 6.030 | 3.310 | **2.317** | 1.511 | 3.018 | **1.466** |
| 04 | **2.100** | **29.944** | 2.216 | 34.067 | 2.655 | 49.025 | 2.550 | 38.861 |
| 05 | **4.097** | **1.598** | 6.506 | 2.695 | 11.755 | 4.217 | 8.750 | 3.641 |
| 06 | **4.113** | 2.927 | 5.564 | 6.305 | 4.219 | **1.518** | 4.120 | 2.364 |
| 07 | 5.216 | **1.957** | **3.028** | 2.165 | 14.155 | 5.698 | 15.512 | 6.957 |
| 08 | **7.202** | **3.019** | 10.054 | 3.254 | 24.796 | 6.134 | 17.134 | 3.302 |
| 09 | **4.729** | **1.065** | 12.205 | 2.454 | 18.387 | 3.718 | 24.154 | 4.288 |
| 10 | **2.064** | 1.831 | 2.649 | **1.155** | 3.823 | 2.081 | 4.992 | 1.823 |

$t_m$:average translational RMSE drift(meter).
$R_{deg}$:average rotational RMSE drift (°).

TABLE II
MEAN RELATIVE POSE ERRORS ON THE KITTI DATASET, WITH THE DASH INDICATING FAILED EXPERIMENT.

| Seq. | DynPL-SVO | | PL_SLAM front end | | ORB_SLAM2 front end | | ORB_Line SLAM front end | |
|---|---|---|---|---|---|---|---|---|
| | $t_{\%}$ | $R_{deg/100m}$ | $t_{\%}$ | $R_{deg/100m}$ | $t_{\%}$ | $R_{deg/100m}$ | $t_{\%}$ | $R_{deg/100m}$ |
| 00 | 1.569 | 0.443 | 1.607 | 0.405 | 1.424 | 0.590 | **1.137** | **0.366** |
| 01 | **21.339** | **1.402** | 43.723 | 1.939 | - | - | - | - |
| 02 | 1.733 | 0.517 | 1.738 | **0.344** | 1.621 | 0.508 | **1.581** | 0.433 |
| 03 | 3.604 | 1.637 | 3.467 | 1.260 | **1.964** | 0.712 | 2.140 | **0.639** |
| 04 | **1.907** | 0.424 | 2.023 | **0.282** | 2.447 | 0.498 | 3.398 | 0.511 |
| 05 | **1.007** | **0.375** | 1.412 | 0.493 | 2.649 | 0.754 | 2.405 | 0.609 |
| 06 | 1.898 | 0.582 | 2.372 | **0.506** | 1.974 | 0.524 | **1.761** | 0.567 |
| 07 | 2.051 | **0.879** | **1.725** | 1.047 | 4.658 | 2.138 | 5.067 | 2.542 |
| 08 | **1.279** | **0.453** | 1.670 | 0.468 | 3.121 | 0.959 | 2.642 | 0.608 |
| 09 | **1.486** | **0.353** | 2.129 | 0.465 | 3.422 | 1.044 | 4.388 | 0.933 |
| 10 | 1.204 | 0.572 | **1.032** | **0.337** | 1.693 | 0.716 | 1.903 | 0.566 |

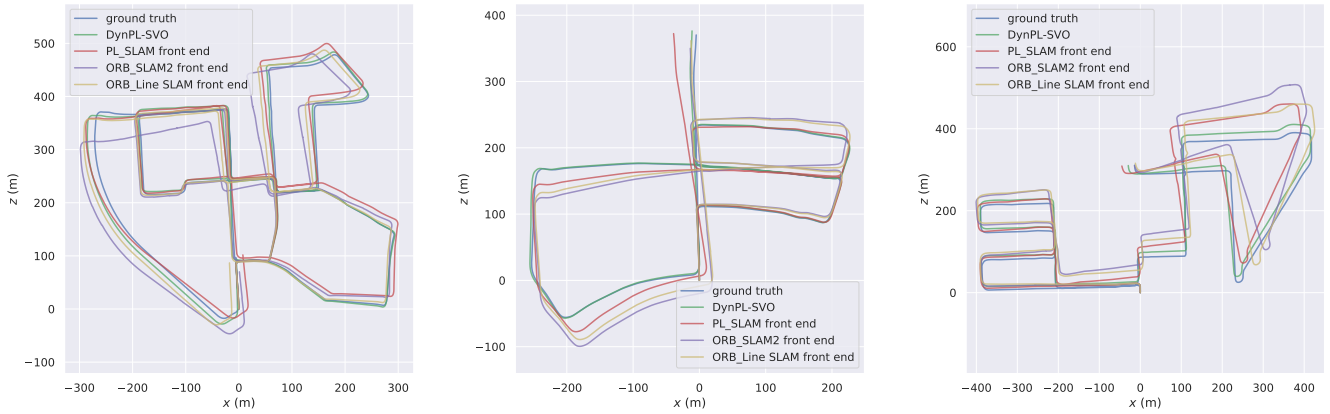
$t_{\%}$:average translational RMSE drift(%).
$R_{deg/100m}$:average rotational RMSE drift (°/100m).

[26] to validate the effects of various line re-projection errors on estimation accuracy. In Table IV, we showed that re-projection errors parallel to line features were more important over re-projection errors perpendicular to line features in most sequences, with an 8.6% improvement overall. The DynPL-SVO outperformed PL_SLAM front end in several sequences, including MH_01_easy and MH_04_different, due to the presence of many short but complete line features in these sequences, wherein the introduction of the re-projection errors parallel to line features in the DynPL-SVO helped improve its performance. It is worth noting that the introduction of re-projection errors perpendicular to line features had a negative impact on three sequences. This could be attributed to irregular drone motions and numerous short line features in the scenes, leading to misalignment and mismatches.

### *C. Evaluating the capability of dynamic grids in dealing with dynamic scenes*

We conducted a comparative analysis in order to evaluate the effectiveness of the *dynamic grid*s. As shown in Figure 5(a) and 5(b), *dynamic grid*s could identify vehicles traveling at high speeds in different directions in the KITTI dataset, with or without rich structural information. Additionally, for slow-moving objects such as cyclists and pedestrians, *dynamic grid*s could eliminate dynamic features, as demonstrated in Figure 5(c) and 5(d), indicating that the *dynamic grid*s could accurately identify dynamic regions, reducing the influence of dynamic features on accuracy, and making the DynPL-SVO more robust when operating in dynamic scenes.

Table V presented a quantitative analysis of the *dynamic grid* approaches in the KITTI dataset. Results illustrated that the DynPL-SVO with the *dynamic grid* method provided more accurate estimation in 8 sequences, improving APE and RPE by about 13.6% and 2.3%, respectively, compared to




(a) Top view of sequence 00 (b) Top view of sequence 05 (c) Top view of sequence 08

Fig. 4. Reconstruction of the path from DynPL-SVO, PL_SLAM front end, ORB_SLAM2 front end, and ORB_Line SLAM front end. The blue lines represented the ground truth trajectory, the green lines corresponded to the estimation of DynPL-SVO, while the red lines represented the estimation of PL_SLAM front end. The paths provided by ORB_SLAM2 and ORB_Line SLAM were plotted with purple and brown lines, respectively.

TABLE III
MEAN ABSOLUTE AND RELATIVE RMSE ERRORS OF THE DYNPL-SVO ON THE KITTI DATASET.

| Seq. | wo/line error | | w/re-projection errors perpendicular to line features | | w/re-projection errors parallel to line features | |
|---|---|---|---|---|---|---|
| | APE | RPE | APE | RPE | APE | RPE |
| 00 | 7.3728 | 0.0323 | 11.9514 | 0.0391 | **6.6818** | **0.0322** |
| 01 | **52.5071** | **0.7188** | 166.6740 | 1.0059 | 89.8455 | 0.7788 |
| 02 | 20.6018 | 0.0348 | 23.6016 | 0.0559 | **19.5024** | **0.0346** |
| 03 | 6.2301 | **0.0317** | **6.0842** | 0.0322 | 6.2207 | **0.0317** |
| 04 | 2.7768 | 0.0374 | **2.2843** | 0.0554 | 2.7410 | **0.0365** |
| 05 | 3.9010 | 0.0182 | 4.3099 | 0.0196 | **3.8735** | **0.0181** |
| 06 | 5.0703 | 0.0326 | **4.6390** | 0.0448 | 4.9705 | **0.0322** |
| 07 | **1.5365** | 0.0178 | 5.4844 | 0.0634 | 1.8679 | **0.0174** |
| 08 | 5.4780 | 0.0390 | 7.3161 | 0.0441 | **4.8398** | **0.0389** |
| 09 | 4.5855 | 0.0248 | **4.4380** | 0.0674 | 4.5637 | **0.0245** |
| 10 | 2.0184 | 0.0198 | 2.2578 | 0.0342 | **2.0173** | **0.0197** |

TABLE IV
MEAN RELATIVE RMSE OF THE DYNPL-SVO ON THE EUROC MAV DATASET.

| Seq. | PL_SLAM front end | DynPL-SVO w/only re-projection errors perpendicular to line features | DynPL-SVO w/only re-projection errors parallel to line features |
|---|---|---|---|
| MH_01_easy | 0.033349 | **0.033294** | 0.033358 |
| MH_02_easy | 0.032896 | 0.032501 | **0.032016** |
| MH_03_med | 0.073692 | 0.071908 | **0.071196** |
| MH_04_dif | 0.103936 | **0.103346** | 0.103473 |
| MH_05_dif | 0.095603 | 0.094640 | **0.094608** |
| V1_01_easy | **0.048642** | 0.049011 | 0.049427 |
| V1_02_med | 0.102017 | 0.102260 | **0.101900** |
| V1_03_dif | 0.098576 | 0.101670 | **0.097804** |
| V2_01_easy | 0.037155 | 0.032655 | **0.032629** |
| V2_02_med | 0.074001 | 0.073592 | **0.071348** |

The sequence V2_03_difficult contained an unequal number of left and right images, rendering it unsuitable for evaluating stereo systems, and therefore, it was not included in the table presented above.

those without the *dynamic grid*s. It was noteworthy that the *dynamic grid* method worked well, particularly in sequences

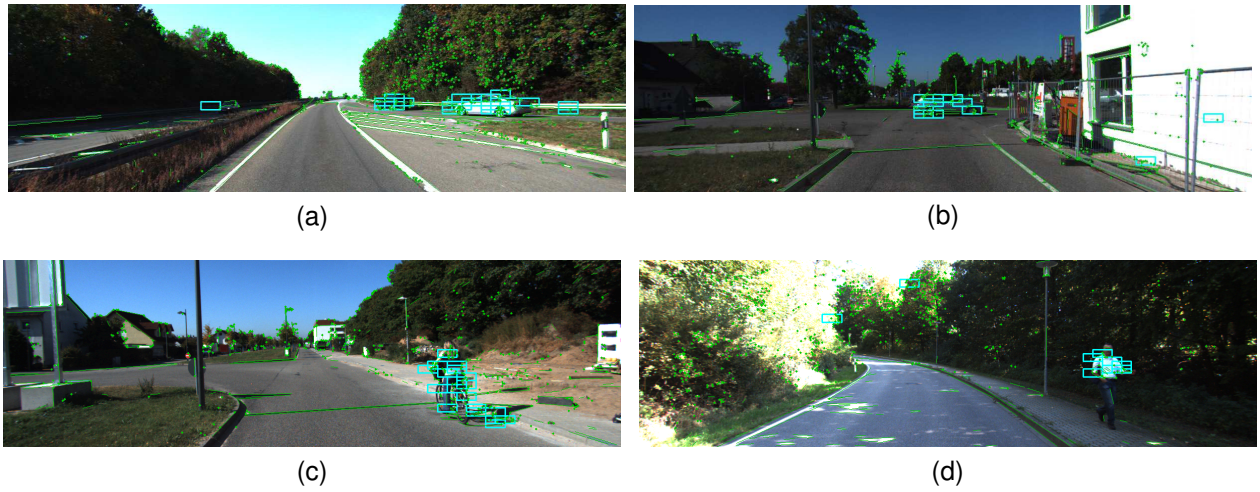



Fig. 5. Dynamic scenes in the *KITTI* dataset and the impact of *dynamic grid*s on dynamic regions. In each subfigure, cyan boxes represented the *dynamic grid*s. Figure 5(a) and 5(b) showed moving cars in *KITTI-01* and *KITTI-06*, respectively. Figure 5(c) showcased a person riding bike in *KITTI-06*, while Figure 5(d) depicted a pedestrian in *KITTI-09*.

TABLE V
MEAN ABSOLUTE AND RELATIVE RMSE ERRORS OF THE DYNPL-SVO IN KITTI DATASET.

| Seq. | DynPL-SVO w/*dynamic grid* | | DynPL-SVO wo/*dynamic grid* | |
|---|---|---|---|---|
| | APE | RPE | APE | RPE |
| 00 | **11.221715** | 0.040752 | 13.236462 | **0.038351** |
| 01 | **182.531449** | **1.019784** | 319.707988 | 1.306922 |
| 02 | 21.654051 | 0.064572 | **10.960957** | **0.055898** |
| 03 | 6.094158 | 0.032278 | **5.946504** | **0.031802** |
| 04 | 2.207623 | 0.054315 | **2.015523** | **0.052209** |
| 05 | **4.403628** | 0.019534 | 7.060361 | **0.019145** |
| 06 | **4.428359** | **0.044869** | 4.440736 | 0.056673 |
| 07 | 5.216922 | **0.062928** | **4.725139** | 0.064078 |
| 08 | **7.443639** | 0.043877 | 12.509229 | **0.042731** |
| 09 | **4.823906** | **0.066494** | 13.612393 | 0.070549 |
| 10 | **2.282632** | **0.032286** | 2.786268 | 0.033558 |

with highly dynamic scenes such as KITTI-01, 05, and 09, wherein SVO accuracy was improved by over 30%.

## VI. CONCLUSIONS

In this paper, we proposed a robust SVO method, i.e., DynPL-SVO, that utilized both point and line features to improve motion estimation accuracy in dynamic scenes. The method introduced the re-projection errors parallel to line features into cost functions to make full use of the structural information of line features. The *dynamic grid* method was also introduced to address the reduction of robustness and accuracy of SVO systems caused by moving objects, wherein dynamic regions could be efficiently marked and point features on dynamic objects could be removed without using depth information and other sensors. The performance of the DynPL-SVO was compared with three SOTA SVO systems on two datasets. Comprehensive experimental results showed that the DynPL-SVO achieved more robust and accurate results in most scenes, particularly on highly dynamic scenes.

Future research will focus on introducing features with greater geometric information such as planes and cubes into VO systems to further improve estimation accuracy of SVO systems.

## REFERENCES

[1] S. Cheng, C. Sun, S. Zhang, and D. Zhang, "Sg-slam: A real-time rgb-d visual slam toward dynamic scenes with semantic and geometric information," *IEEE Trans. Instrum. Meas.*, vol. 72, pp. 1–12, 2023.

[2] A. Pumarola, A. Vakhitov, A. Agudo, A. Sanfeliu, and F. Moreno-Noguer, "Pl-slam: Real-time monocular visual slam with points and lines," in *Proc. IEEE Int. Conf. Robot. Autom.*, pp. 4503–4508, Jul. 2017.

[3] R. Miao, J. Qian, Y. Song, R. Ying, and P. Liu, "Univio: Unified direct and feature-based underwater stereo visual-inertial odometry," *IEEE Trans. Instrum. Meas.*, vol. 71, pp. 1–14, 2022.

[4] J. H. Jung, S. Heo, and C. G. Park, "Observability analysis of imu intrinsic parameters in stereo visual–inertial odometry," *IEEE Trans. Instrum. Meas.*, vol. 69, no. 10, pp. 7530–7541, 2020.

[5] R. Mur-Artal and J. D. Tardós, "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, 2017.

[6] B. Kitt, A. Geiger, and H. Lategahn, "Visual odometry based on stereo image sequences with ransac-based outlier rejection scheme," in *Proc. IEEE Int. V. Sym.*, pp. 486–492, 2010.

[7] E. Garcia-Fidalgo and A. Ortiz, "Hierarchical place recognition for topological mapping," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1061–1074, 2017.

[8] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," *IEEE Trans. Pattern. Anal. Mach. Intell.*, 2017.

[9] J. Engel, T. Schöps, and D. Cremers, "Lsd-slam: Large-scale direct monocular slam," in *Proc. IEEE Eur. Conf. Comput. Vis.*, pp. 834–849, Sep. 2014.

[10] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE Trans. Pattern. Anal. Mach. Intell.*, vol. 40, no. 3, pp. 611–625, 2017.

[11] X. Zuo, X. Xie, Y. Liu, and G. Huang, "Robust visual slam with point and line features," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, pp. 1775–1782, 2017.

[12] G. Zhang, J. H. Lee, J. Lim, and I. H. Suh, "Building a 3-d line-based map using stereo slam," *IEEE Trans. Robot.*, vol. 31, no. 6, pp. 1364–1377, 2015.

[13] R. Gomez-Ojeda, J. Briales, and J. Gonzalez-Jimenez, "Pl-svo: Semi-direct monocular visual odometry by combining points and line segments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, pp. 4211–4216, Dec. 2016.

[14] J. H. Lee, S. Lee, G. Zhang, J. Lim, W. K. Chung, and I. H. Suh, "Outdoor place recognition in urban environments using straight lines," in *Proc. IEEE Int. Conf. Robot. Autom.*, pp. 5550–5557, 2014.

[15] C. Akinlar and C. Topal, "Edlines: A real-time line segment detector with a false detection control," *Pat. Rec. Lett.*, vol. 32, no. 13, pp. 1633–1642, 2011.

[16] R. G. Von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall, "Lsd: A fast line segment detector with a false detection control," *IEEE Trans. Pattern. Anal. Mach. Intell.*, vol. 32, no. 4, pp. 722–732, 2008.

[17] T. Koletschka, L. Puig, and K. Daniilidis, "Mevo: Multi-environment stereo visual odometry," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, pp. 4981–4988, 2014.

[18] Y. He, J. Zhao, Y. Guo, W. He, and K. Yuan, "Pl-vio: Tightly-coupled monocular visual–inertial odometry using point and line features," *Sensors*, vol. 18, no. 4, p. 1159, 2018.

[19] A. Bartoli and P. Sturm, "Structure-from-motion using lines: Representation, triangulation, and bundle adjustment," *Comput. Vis. Image. Underst.*, vol. 100, no. 3, pp. 416–441, 2005.

[20] R. Gomez-Ojeda and J. Gonzalez-Jimenez, "Robust stereo visual odometry through a probabilistic combination of points and line segments," in *Proc. IEEE Int. Conf. Robot. Autom.*, pp. 2521–2526, Jun. 2016.

[21] R. Sahdev, B. X. Chen, and J. K. Tsotsos, "Indoor localization in dynamic human environments using visual odometry and global pose refinement," in *Proc. Conf. Comput. Robot. Vis.*, pp. 360–367, May 2018.

[22] M. Ouyang, Z. Cao, P. Guan, Z. Li, C. Zhou, and J. Yu, "Visual-gyroscope-wheel odometry with ground plane constraint for indoor robots in dynamic environment," *IEEE Sens. Lett.*, vol. 5, no. 3, pp. 1–4, 2021.

[23] H. Kim, P. Kim, and H. J. Kim, "Moving object detection for visual odometry in a dynamic environment based on occlusion accumulation," in *Proc. IEEE Int. Conf. Robot. Autom.*, pp. 8658–8664, Sep. 2020.

[24] Q. Sun, Y. Tang, C. Zhang, C. Zhao, F. Qian, and J. Kurths, "Unsupervised estimation of monocular depth and vo in dynamic environments via hybrid masks," *IEEE Trans. Neural. Netw.*, 2021.

[25] R. Gomez-Ojeda, F.-A. Moreno, D. Zuniga-Noël, D. Scaramuzza, and J. Gonzalez-Jimenez, "Pl-slam: A stereo slam system through the combination of points and line segments," *IEEE Trans. Robot.*, vol. 35, pp. 734–746, May 2019.

[26] Z. Liu, D. Shi, R. Li, W. Qin, Y. Zhang, and X. Ren, "Plc-vio: Visual-inertial odometry based on point-line constraints," *IEEE Trans. Autom. Sci. Eng.*, 2021.