LiDAR-Visual-Inertial Odometry Based on Optimized Visual Point-Line Features

Xuan He\textsuperscript{1,2}, Wang Gao\textsuperscript{1,2,*}, Chuanzhen Sheng\textsuperscript{3,4}, Ziteng Zhang\textsuperscript{3,4}, Shuguo Pan\textsuperscript{1,2}, Lijin Duan\textsuperscript{5}, Hui Zhang\textsuperscript{1,2} and Xinyu Lu\textsuperscript{1,2}

\textsuperscript{1} School of Instrument Science and Engineering, Southeast University, Nanjing 210096, China; hexuan@seu.edu.cn (X.H.); psg@seu.edu.cn (S.P.); amzhanghui@seu.edu.cn (H.Z.); 220213597@seu.edu.cn (X.L.)
\textsuperscript{2} Key Laboratory of Micro-Inertial Instrument and Advanced Navigation Technology, Southeast University, Nanjing 210096, China
\textsuperscript{3} State Key Laboratory of Satellite Navigation System and Equipment Technology, Shijiazhuang 050081, China; shengchuanzhen@163.com (C.S.); zzteng54@163.com (Z.Z.)
\textsuperscript{4} The 54th Research Institute of China Electronics Technology Group Corporation, Shijiazhuang 050081, China
\textsuperscript{5} Linzi District Transportation Service Center, Zibo 255400, China; lzqjtysqjd@zb.shandong.cn

\textsuperscript{*} Correspondence: gaow@seu.edu.cn

Abstract: This study presents a LiDAR-Visual-Inertial Odometry (LVIO) based on optimized visual point-line features, which can effectively compensate for the limitations of a single sensor in real-time localization and mapping. Firstly, an improved line feature extraction in scale space and constraint matching strategy, using the least square method, is proposed to provide a richer visual feature for the front-end of LVIO. Secondly, multi-frame LiDAR point clouds were projected into the visual frame for feature depth correlation. Thirdly, the initial estimation results of Visual-Inertial Odometry (VIO) were carried out to optimize the scanning matching accuracy of LiDAR. Finally, a factor graph based on Bayesian network is proposed to build the LVIO fusion system, in which GNSS factor and loop factor are introduced to constrain LVIO globally. The evaluations on indoor and outdoor datasets show that the proposed algorithm is superior to other state-of-the-art algorithms in real-time efficiency, positioning accuracy, and mapping effect. Specifically, the average RMSE of absolute trajectory in the indoor environment is 0.075 m and that in the outdoor environment is 3.77 m. These experimental results can prove that the proposed algorithm can effectively solve the problem of line feature mismatching and the accumulated error of local sensors in mobile carrier positioning.

Keywords: multi-sensor fusion; visual point and line feature; SLAM; LiDAR-visual-inertial odometry

1. Introduction

Multi-sensor fusion localization technology based on Simultaneous Localization and Mapping (SLAM) is a fundamental technology in the field of high-precision localization of mobile carriers [1]. The SLAM-based multi-sensor fusion system applied to mobile carriers can be divided into two core parts: the front-end, and the back-end. The function of the front-end is used to analyze the environmental fingerprint information collected by the sensors, in order to estimate the positional information of the mobile carrier in time. In addition, the change in the surrounding environment with the movement of the carrier is restored. The function of the back-end is used to obtain the final positioning results by iteratively optimizing the position estimates obtained from the front-end analysis. Depending on the sensors used in the front-end, it can be divided into methods mainly based on LiDAR and vision [2,3]. Engineers and researchers in related fields have conducted a lot of research in both directions and produced a series of research-worthy results.

The main vision-based SLAM approach, namely visual odometry (VO), has long dominated the SLAM technology field due to the lower cost of the camera compared with...
LiDAR. However, pure monocular visual SLAM systems cannot recover metric scales. Thus, there is a growing trend to utilize low-cost inertial measurement units to assist monocular vision systems, which is called visual-inertial odometry (VIO). Monocular VIO provides high-quality self-motion simulation by using monocular cameras and inertial measurement unit (IMU) measurements, which has significant advantages in terms of size, cost, and power. Based on the method of feature association, visual SLAM can be classified into feature point method and direct method. The feature point-based method VIO accomplishes the inter-frame feature constraint by extracting and matching image feature points [4–6]. Therefore, rich environmental texture is required to ensure that the threshold of the number of effective feature points required for feature tracking is reached. Tracking loss of feature points is prone to occur in weak texture environments such as parking lots and tunnels, which in turn affects localization accuracy and real-time performance. The theoretical basis of the direct method-based VIO is the assumption of constant grayscale [7,8]. It only needs to capture environmental features by the changes in the grayscale image to establish constraints, which has a better real-time performance. Nevertheless, the tracking accuracy is greatly affected by environmental illumination changes. Therefore, stable and rich line feature models are required to be introduced into the front-end to provide stable and accurate feature constraints for visual back-end state estimation. In 2018, He et al. proposed PL-VIO based on point-line feature fusion, but too many optimization factors greatly limited the real-time performance in practical tests [9]. In 2020, Wen et al. proposed PLS-VIO to optimize the 6-DOF pose by minimizing the objective function and improving the line feature matching filtering strategy to reduce the probability of mismatching [10]. Although the VIO based on point-line features has a positive effect on the number of features [11,12], it still cannot solve the scale uncertainty problem of monocular cameras. The development of VIO in practical applications still has certain limitations.

As another important technical means of SLAM-based localization technology, SLAM mainly based on LiDAR is also widely used in the industry for its high resolution, high accuracy, and high utilization of spatial features. In 2016, Google proposed Cartographer, a 2D LiDAR based on particle filtering and graph optimization. In 2017, Zhang et al. proposed the LOAM for the first time, which uses the curvature of the LiDAR point cloud to register the effective point cloud features as planar points and edge points [13]. In 2018, Shan et al. proposed LeGO-LOAM based on LOAM, which uses the ground plane feature point cloud to further filter outliers from the scanned point cloud and improve the LOAM frame [2]. In 2020, Shan et al. further introduced the LIO-SAM algorithm based on the previous work, which uses IMU pre-integrated measurements to provide initial pose estimation for laser odometry [14]. In addition, a Bayesian network-based factor graph optimization framework is proposed, in which the global position is constrained by adding GPS factors, and an incremental smooth global voxel map is established. These schemes provide technical feasibility for the high-precision positioning by fusing LiDAR with other sensors.

However, due to the inherent shortcomings of the main sensing sensors, such as the limited scanning angle of LiDAR and the sensitivity of the mainly vision-based methods to light variations, these methods can hardly show excellent robustness in real-world applications. To further improve the localization performance, LiDAR-Visual-Inertial Odometry, as a multi-sensor fusion localization method, has become a research focus of SLAM with its advantages of multi-sensor heterogeneity and complementarity.

The existing LVIO multi-sensor fusion strategy can be described from the front-end and back-end perspectives. First, the front-end fusion strategy of LVIO is introduced. Generally, LiDAR acts as a feature depth provider for monocular VO as a way to improve the scale ambiguity of visual features. Meanwhile, VO performs state estimation from the extracted visual features, which is provided as the initial state for LiDAR scan matching. Therefore, the quantity and quality of visual features are closely related to the precision of state estimation of the fusion system. In existing fusion systems, the features extracted
by camera are mainly point features [15,16]. Xiang et al. proposed a combination of fisheye camera and LiDAR based on a semantic segmentation model, which improved the confidence of the depth of visual features in the driving environment of unmanned vehicles [15]. Chen et al. proposed a method to construct a loopback constraint for LiDAR-visual odometry by using the Distributed bag of Words (DboWs) model in the visual subsystem, although, without introducing IMU sensors to assist in the initial positional estimation [16]. In 2021, Lin et al. proposed R2LIVE to incorporate IMU into the fused localization system, in which the LiDAR odometry is used to establish depth constraints for VIO [17]. Although the above-mentioned algorithms exhibit superior performance to the VIO based on point features, it is still difficult to extract rich and effective features in weak texture environments, which leads to the failure in LiDAR scan matching. Therefore, additional feature constraints on the LiDAR need to be added with line features that are more robust to environmental texture and luminosity variations. Visual SLAM based on point-line features has been studied but not widely applied to LVIO systems in recent years [18,19]. In 2020, Huang et al. first proposed a LVIO based on a robust point and line depth extraction method, which greatly reduces the three-dimensional ambiguity of features [18]. Zhou et al. introduced line features in the direct method-based VIO to establish data association [19]. The above-mentioned algorithms provide technical feasibility for LVIO based on point-line features.

From the perspective of the back-end fusion strategy, LVIO can be classified into two categories based on different optimization algorithms: filter-based methods and factor graph methods. Although the filtering method is a traditional technology to realize multi-sensor fusion, its principle defect of frequent reconstruction of increasing or decreasing sensors limits its application in LVIO [20]. As an emerging method in recent years, the factor graph method can effectively improve the robustness of SLAM system when a single sensor fails because of its plug-and-play characteristics. Therefore, it is widely applied to deal with such heterogeneous aperiodic data fusion problems [21]. In addition, since LVIO is in the local frame, there are inherent defects such as accumulated errors. Thus GNSS measurements need to be introduced for global correction [22–24] to realize local accuracy and global drift-free position estimation, which makes full use of their complementarity [24]. The research on adding GNSS global constraints into the local sensor fusion framework are as follows: Lin et al. modified the extended Kalman filter to realize a loose coupling between GPS measurements and LiDAR state estimation, but there is a large single linearization error to be solved [17]. In 2019, Qin et al. proposed VINS-Fusion, which uses nonlinear optimization strategies to support Camera, IMU, and GNSS [25], but it assumes that GNSS is continuous and globally convergent, which is inconsistent with reality. In any case, these strategies presented above provide numerous reliable ideas.

Generally speaking, we can conclude that the existing LVIO fusion system has two problems that deserve further exploration. First, on the premise of ensuring the real-time performance, more abundant feature constraints are needed to improve the pose estimation accuracy of LVIO. Secondly, global constraints are needed to globally optimize the LVIO local pose estimation results. To address these issues, this study presents a LiDAR-Visual-Inertial Odometry based on optimized visual point-line features. First of all, an improved line feature extraction in scale space and constraint matching strategy based on the square method are proposed, which provides richer visual feature for the front-end of LVIO. Secondly, multi-frame LiDAR point clouds were projected into the visual frame for feature depth correlation, which improves the confidence of monocular visual depth estimation. At the same time, the initial visual state estimation can be used to optimize the scan matching of LiDAR. Finally, a factor graph based on the Bayesian network was used to build the LVIO fusion system, in which the GNSS factor and loop factor are introduced to constrain LVIO globally, to achieve locally accurate and globally drift-free position estimation in the complex environment.
2. System Overview

The general framework of the LiDAR-Visual-Inertial Odometry based on optimized visual point-line features proposed in this study is shown in Figure 1. The system consists of the front-end of LiDAR-Visual-Inertial Odometry tight combination and the back-end of factor graph optimization.

Figure 1. Overall algorithm framework, system inputs include IMU, camera, lidar and optional GNSS. IMU provides initial state correction for VIO subsystem and LiDAR-inertial odometry (LIO) subsystem, VIO and LIO systems use each other’s information to improve the positioning accuracy, and GNSS signals are optionally added to the back-end to provide global constraints.

In the front-end of our algorithm, the visual odometry not only extracts point features, but also further extracts line features in the improved scale space and performs geometric constraint matching on them, which improves the number of features in the weak texture environment. Then, the feature depth provided by LiDAR point clouds performed a role in correlating the depth of monocular visual features. IMU pre-integration provides all necessary initial values, including attitude, velocity, acceleration bias, gyroscope bias, and three-dimensional feature position, for completing the initial state estimation after time alignment with a camera. If VIO initialization fails, the IMU pre-integration value is used as the initial assumption to improve the robustness of the fusion system in the texture-free environment.

After the front-end initialization is successfully realized, the back-end optimizes the factor graph by using the estimated residual of each sensor’s state. IMU pre-integration, visual residual and lidar residual were added to the factor graph as local state factors for maximum a posteriori estimation. In order to further correct the cumulative error of local state estimation, the residual of GNSS single-point positioning measurements was used as the global positioning factor to add to the factor graph. Besides, when the system detects the path loop, the loop factor will be added to the factor graph to participate in the nonlinear optimization and obtain the optimal global pose estimation.

3. Front-End: Feature Extraction and Matching Tracking

3.1. Line Feature Extraction

Commonly used line feature extraction algorithms include Hough [26], LSWMS [27], EDLine [28], and LSD [29]. Weighing factors such as accuracy, real-time performance, and the need for parameter adjustment, we chose LSD to extract line features. According to the bottom parameter optimization strategy, we modified an improved LSD algorithm, and a minimum geometric constraint method to realize line feature constraint matching.
Given an N-layer Gaussian pyramid as the scale space of LSD line features, the scale ratio of images in each layer is defined to reduce or eliminate the sawtooth effect in images. After scaling the image \( s \) times, a downsampling was performed, and then the gradient was calculated for all pixels in the new image obtained after downsampling. By traversing the image and getting the gradient values of all pixels, the pixel gradient rectangle can be merged according to the density of same-sex points to obtain a rectangle-like line segment \( l \). The density \( d \) of homogeneous points in the rectangle can be expressed as:

\[
d = \frac{k \cdot \text{length}(l) \cdot \text{width}(l)}{d \leq D}
\]

where \( k \) is defined as the total number of pixels in the rectangle, and \( D \) is the density threshold of parity points. Different from the hypothesis in [12], a low co-location density threshold in the outdoor complex texture environment will extract a large number of invalid line features. Therefore, it is necessary to re-optimize the strategy according to the underlying parameters and select the following combinations near the original parameters \((s = 0.8, D = 0.7)\), for real-time and accuracy experiments.

We measured the positioning accuracy by the root mean square error of absolute trajectory error (APE_RMS). The accuracy and real-time performance of different values of \( s \) and \( D \) on the Hong Kong 0428 dataset are shown in Figure 2. The Monte Carlo method was used in this experiment. Within the parameter range that ensures the stable operation of the line feature extraction algorithm, we conducted three experiments. First of all, as shown in Figure 2a, under the premise that the original scaling times \( s = 0.8 \), 100 random numbers were selected in the range of \( D \in (0.3, 0.9) \) to carry out the experiment of density threshold selection. Secondly, as shown in Figure 2b, we kept the original density threshold \( D = 0.7 \), and then selected 100 random numbers in the range of \( s \in (0.4, 0.9) \), which is to select the appropriate range of scaling times \( s \). Finally, as shown in Figure 2c, within the appropriate parameter range obtained in the previous experiments, 100 groups of parameter combinations were randomly selected for line feature extraction to obtain the optimal value.

According to Figure 2c it can be seen that the operation time is shorter when the value of \((s, D)\) is around \((0.5, 0.6)\) or around \((0.6, 0.6)\). Furthermore, we compared the accuracy of the above two groups of parameters. It can be concluded that the accuracy of line feature extraction of the former group is slightly higher than that of the latter group. Considering
the accuracy and real-time, we chose \( s = 0.5, D = 0.6 \) as the parameter combination for our system.

### 3.2. Inter-Frame Feature Constraint Matching

Different from the neighboring line merging of different line features within the same frame in feature extraction, the least square method-based line feature constraint matching is for the same line feature pair whose angle and distance change between two consecutive frames. Considering the angle and translation changes in the same line feature pair during the carrier movement, a minimized sparse matrix model can be constructed to ensure the minimum total error in matching the line features extracted between the front and back frames.

Given a line \( l^W = [n^W, v^W]^T \in \mathbb{R}^6 \) extracted from the world coordinate system, where \( n^W, v^W \in \mathbb{R}^3 \) is the normal vector and direction vector, respectively, then \( l^W \) can be expressed in Plücker coordinates within the camera frame as:

\[
l^C = \begin{bmatrix} n^C \\ v^C \end{bmatrix} = T_{WC}^W l^W = \begin{bmatrix} R_{WC}^W & t_{WC}^W \\ 0 & 1 \end{bmatrix} \begin{bmatrix} n^W \\ v^W \end{bmatrix} \in \mathbb{R}^6
\]

(2)

It can be seen that the matching of line feature pairs in the camera frame is a 6-DOF parametric matching problem. In order to improve the accuracy and simplify the line feature matching problem, it can be simplified as a 4-DOF parameter matching optimization problem. Let all the line feature pairs obtained by matching between two consecutive frames in the camera frame be:

\[
F_{ij} = \{ (l_i, l_j) \mid j \in [1, n] \}
\]

(3)

where \( l_i \) and \( l_i \) are certain line features extracted in the previous frame and subsequent frame, respectively, \( n \) is the total number of line features in the subsequent frame.

According to the variation in the inter-frame line characteristics shown in Figure 3, the parameter matrix can be set as \( e_{ij} = [\theta_{ij}, \mu_{ij}, \rho_{ij}, d_{ij}]^T \), \( \theta_{ij} \) and \( d_{ij} \) are the included angle and translation distance between two consecutive frames, respectively, \( \mu_{ij} \) and \( \rho_{ij} \) are the projection ratio and length ratio of the front-to-back interframe line features. Constructing the parameter matrix may establish a linear constraint matrix \( A_i = [e_{i1}, \ldots, e_{ij}, \ldots, e_{in}] \) of the subsequent keyframe for \( l_i \). The target vector of the matching judgment of \( l_i \) is \( m_i = [m_{i1}, \ldots, m_{ij}, \ldots, m_{in}]^T \). The value of each component is determined by the result of feature matching, where matching is 1 and non-matching is 0. If \( \sum m_{in} = 1 \), the linear constraint \( A_i m_i = t \) will be satisfied. Therefore, the line feature matching problem can be optimized into a constrained matching equation based on least squares:

\[
\min_{m_i} \lambda \|m_i\|_1 + \frac{1}{2} \|A_i m_i - t\|_2
\]

(4)

where \( \lambda \) is the weight coefficient and \( t = [0, 1, 1, 0]^T \) is the constraint target vector.

![Figure 3. Deviation of a line feature during the movement of the carrier. (a) Parallel offset (b) Angular offset.](image-url)
3.3. LiDAR-Aided Depth Correlation of Visual Features

LiDAR-aided depth correlation of visual features can effectively improve the scale ambiguity of monocular cameras. Since the LiDAR resolution is much lower than that of the camera, the use of only a single frame of sparse point cloud depth correlation will result in a large number of visual feature depth deletions [30]. Therefore, this study proposes a strategy of superimposing multi-frame sparse point cloud to obtain the depth value of the point cloud, which is used to establish the depth correlation with the visual features.

As shown in Figure 4, $f^V_1$ is a feature point in the visual frame $\{V\}$, and $\{d^L_1, \ldots, d^L_m\}$ is a group of depth point clouds in the lidar frame $\{L\}$. Projecting $d^L_n$ onto a unit spherical surface $\{V_g\}$ with $f^V_1$ as the spherical center to obtain a projection point $d^V_{ng}$:

$$d^V_{ng} = R^V_g d^L_n + p^V_g \quad n \in [1, m]$$

(5)

where $R^V_g$ and $p^V_g$ are the rotation matrix and external parameter matrix of $\{L\}$ to $\{V_g\}$, respectively. Taking $f^V_1$ as the root node to establish KD tree to search for the three closest depth points $d_1, d_2, d_3$ on the sphere. Then, connecting $f^V_1$ with the camera center $O$ and intersecting $\Delta d_1d_2d_3$ with $Od$, we can obtain the characteristic depth of $f^V_1$ as $f^V_1O_d$.

Figure 4. Association of visual feature depth.

4. Back-End: LVIO-GNSS Fusion Framework Based on Factor Graph

4.1. Construction of Factor Graph Optimization Framework

The framework of factor graph optimization based on the Bayesian network proposed in this study is shown in Figure 5. The state vector in that world frame construct according to the constraint factor shown in the figure is:

$$\mathbf{X} = [x_1, x_2, \ldots, x_i, \lambda_1, \lambda_2, \ldots, \lambda_p, 1, 2, \ldots, i, d^p_1, d^p_2, \ldots, d^p_k, d^d_1, d^d_2, \ldots, d^d_k]$$

(6)

where $x_n = [p_n, q_n, v_n, b_a, b_g]$ represents the IMU state at the nth time, which includes the carrier position $p_n$, the rotation quaternion $q_n$, and the velocity $v_n$ obtained by IMU pre-integration in the world frame, $b_a$ and $b_g$ stand for the acceleration bias and the gyroscope bias in IMU body frame, respectively, $\lambda_p$ represents the inverse depth of the visual point feature in the visual frame from its initial observation in the first frame, $\lambda_i$ represents the orthogonal frame of the visual line feature, $d^d_k$ and $d^p_k$ stand for the distances between the LiDAR feature points and its corresponding edge or plane feature point cloud, respectively.
Figure 5. Factor graph optimization framework of our system. Constraints of factor graph on the keyframe maintenance include three local constraints and two global constraints.

Therefore, the Gaussian–Newton method can be used to minimize all cost functions to construct a maximum a posteriori estimation problem, to perform nonlinear optimization on the state vectors in the sliding window:

$$
\min_{\chi} \left\{ \|r_p - J_p \chi\|^2 + \sum_{k \in B} \|r_B(z_{k+1}^k, \chi)\|^2_{p_i} + \sum_{(i,j) \in F} \rho \left( \|r_f(z_i^j, \chi)\|^2_{p_i} + \sum_{d_k^e} \right) + \sum_{(i,j) \in L} \rho \left( \|r_l(z_i^j, \chi)\|^2_{p_i} + \sum_{d_k^l} \right) \right\}
$$ (7)

where \(\{r_p, J_p\}\) contains the prior states after the marginalization in the sliding window, and \(J_p\) is the Jacobian matrix, \(r_B(z_{k+1}^k, \chi)\) represents the IMU residuals, and \(p_i\) is the IMU covariance matrix; \(r_f(z_i^j, \chi)\) and \(r_l(z_i^j, \chi)\) represent the re-projection errors of visual point and line features, \(p_c\) is the visual covariance matrix, and \(\rho\) represents Huber norm, with specific values as follows:

$$
\rho(e(s)) = \begin{cases} 
\frac{1}{2} e_1(s)^2 & e(s) = e_1(s), |e_1(s)| \leq \delta \\
\delta |e_2(s)| - \frac{1}{2} \delta^2 & e(s) = e_2(s), |e_2(s)| > \delta 
\end{cases}
$$ (8)

The specific meaning of each sensor cost function in Formula (6) is as follows.

4.2. IMU Factor

The IMU state of the \(k\)th frame and the \(k+1\)th frame in the global coordinate system can be defined as:

$$
x_k = \left[ p_{x_k}^G, q_{x_k}^G, v_{x_k}^G, b_{ak}, b_{gk} \right]
$$

$$
x_{k+1} = \left[ p_{x_{k+1}}^G, q_{x_{k+1}}^G, v_{x_{k+1}}^G, b_{ak+1}, b_{gk+1} \right]
$$ (9)

Take the IMU state of the \(k\)th frame, \(x_k\), as an example, which includes position \(p_{x_k}^G\), rotation \(q_{x_k}^G\), velocity \(v_{x_k}^G\), accelerometer bias \(b_{ak}\) and gyroscope bias \(b_{gk}\).
Next, the IMU residual equation can be constructed, which is defined as:

\[
    r_B \left( \hat{z}^k_{k+1}, \mathcal{X} \right) = \begin{bmatrix} r_p \\ r_q \\ r_v \\ r_{ba} \\ r_{bg} \end{bmatrix} = \begin{bmatrix} R^G_B \left( p^G_{b_{k+1}} - p^G_{b_k} + \frac{1}{2} g \Delta t^2_k - v^G_{b_k} \Delta t_k \right) - \hat{p}^k_{k+1} \\ 2 \left[ q^G_{b_k} \otimes \hat{q}^G_{b_{k+1}} \otimes \hat{q}^G_{k+1} \right]_{xyz} \\ R^G_B \left( v^G_{k+1} + g \Delta t_k - \hat{v}^G_k \right) - \hat{v}^k_{k+1} \\ b_{ak+1} - b_{ak} \\ b_{gk+1} - b_{gk} \end{bmatrix}
\]  

(10)

where \( \begin{bmatrix} r_p, r_q, r_v, r_{ba}, r_{bg} \end{bmatrix}^T \) represents the observation residual of IMU state between two consecutive keyframes in the sliding window, including the residual of position, rotation, velocity, accelerometer bias and gyroscope bias, \( R^G_B \) represents the pose conversion matrix of the \( k \)th frame from the IMU coordinate system to GNSS global coordinate system, and \( \left[ \hat{p}^k_{k+1}, \hat{q}^k_{k+1}, \hat{v}^k_{k+1} \right] \) represents the IMU pre-integration value of two keyframes in the sliding window within \( \Delta t_k \).

4.3. Visual Feature Factor

The visual feature factor is essentially the re-projection error of the visual feature, that is, the difference between the theoretical value projected on the image plane and the actual observation value. In order to unify the coordinate system in Section 3.3, we provide the definition of re-projection error on the unit sphere instead of the generalized image plane. Specific schematic diagrams are shown in Figures 6 and 7.

![Figure 6. Re-projection error of visual point features.](image1)

![Figure 7. Re-projection error of visual line features.](image2)
4.3.1. Visual Point Feature Factor

In this study, the visual feature factors are built with reference to VINS-Mono [5]. As shown in Figure 6, the re-projection error of visual point features can be defined as the difference between the projection point on the unit spherical surface and the observation value after distortion correction. Given the $i$th normalized projection point $f_i^j = \left[ u_i^j, v_i^j, 1 \right]^T$ and observation point $f_i^j = \left[ u_i^j, v_i^j, 1 \right]^T$ in the $j$th frame, we use the first observation value $f_i^j = \left[ u_i^j, v_i^j, 1 \right]^T$ in the $j$th frame to define the visual point feature factor as:

$$
\begin{align*}
rf \left( z_i^j, \chi \right) &= \left[ \begin{array}{c} a_i^j - u_i^j \\ \varphi_i - v_i^j \end{array} \right] \\
&= R_b^j \left( R_C^j \left( R_B^j \left[ u_i^j, v_i^j, 1 \right] + p_{i0}^j \right) + p_{i1}^j - p_{i0}^j \right) - p_B^j
\end{align*}
$$

where $R_b^j$ represents the external parameter matrix between camera and IMU, which is obtained by calibration, $R_C^j$ represents the pose conversion matrix from the IMU observation in the $j$th frame to the global coordinate system, $R_B^j$ represents the pose conversion matrix from the global coordinate system to the initial IMU observation, $\chi$ stands for the inverse depth of $f_i^j$, $p_{i0}^j$ represents the displacement from the IMU coordinate system to the camera coordinate system. Finally, $p_{i1}^j$ and $p_{i2}^j$ represent the displacement of the first and the $i$th IMU observation in the global coordinate system, respectively.

4.3.2. Visual Line Feature Factor

As shown in Figure 7, similar to the visual point feature, the definition of the re-projection error of the visual line feature is as follows: Given the characteristics of a visual line in space, the end point of a line segment is the center of the sphere to construct a unit sphere. Therefore, the reprojection error is the difference between the projection line on the unit spherical surface and the observation value. According to Equation (2), given the observed value of the characteristic factor of the $i$th line in the $j$th frame in the camera coordinate system as $l_{ci}^j = \left[ n_{ci}^j, v_{ci}^j \right]^T$, the projection line is obtained by projecting it onto the unit sphere, and can be expressed as:

$$
\hat{p}_{ci}^j = \left[ \begin{array}{c} l_1^j \\ l_2^j \\ l_3^j \end{array} \right] = K n_{ci}^j \in R^6
$$

where $K$ is the camera internal reference projection matrix. It can be seen from Equation (12) that the spatial coordinates of the line features projected onto the unit sphere are only related to $n_c$. The two end points of the observation line are $a_i^j$ and $b_i^j$, then the re-projection error of the line feature can be expressed by the dotted distance from the two end points of the observation line feature to the projection line feature:

$$
\begin{align*}
rf \left( z_i^j, \chi \right) &= \left[ d \left( a_i^j, \hat{p}_{ci}^j \right), d \left( b_i^j, \hat{p}_{ci}^j \right) \right]^T \\
d \left( a_i^j, \hat{p}_{ci}^j \right) &= \frac{a_i^j \hat{p}_{ci}^j}{\sqrt{a_i^j \hat{p}_{ci}^j}} \\
d \left( b_i^j, \hat{p}_{ci}^j \right) &= \frac{b_i^j \hat{p}_{ci}^j}{\sqrt{b_i^j \hat{p}_{ci}^j}}
\end{align*}
$$

4.4. LiDAR Factor

As mentioned in Section 3.3, after the LiDAR-assisted monocular visual depth correlation, the VIO will provide the LiDAR with visual initial positional estimates to correct
the motion distortion of the LiDAR point cloud and improve the scan matching accuracy. The scanning matching error between adjacent keyframes of LiDAR involved in this study can be expressed by the distance from the feature point to the matched edge line and feature plane as:

\[
\begin{aligned}
    d_k^e &= \left| \frac{\left( X_{(k+1,i)}^e - X_{(k,a)}^e \right) \times \left( X_{(k+1,i)}^e - X_{(k,b)}^e \right)}{\left( X_{(k+1,i)}^e - X_{(k,b)}^e \right) \times \left( X_{(k+1,i)}^e - X_{(k,b)}^e \right)} \right| \\
    d_k^p &= \left| \frac{\left( X_{(k,a)}^p - X_{(k,b)}^p \right) \times \left( X_{(k,a)}^p - X_{(k,c)}^p \right)}{\left( X_{(k,a)}^p - X_{(k,c)}^p \right) \times \left( X_{(k,a)}^p - X_{(k,c)}^p \right)} \right|
\end{aligned}
\]  

(14)

where \(X_{(k+1,i)}^e\) represents the edge feature point at the \(k+1\)th time, \(X_{(k,a)}^e\) and \(X_{(k,b)}^e\) are the endpoint of the edge line matched with the feature point at the \(k\)th time, \(X_{(k+1,i)}^p\) represents the plane feature point at the \(k+1\)th time, and the feature surface matched with it at the \(k\)th time can be represented by three points \(X_{(k,a)}^p\), \(X_{(k,b)}^p\) and \(X_{(k,c)}^p\).

4.5. GNSS Factor and Loop Factor

When the carrier moves to a GNSS signal trusted environment, GNSS factors can be added to optimize with local sensors. The time interval of two frames of GNSS observations is \(\Delta t\), and given the GNSS measurements \(p_k^{G^s}\) in the global frame and \(p_k^{V^s}\) representing the observation of LVIO in the global frame, the GNSS factor can be expressed by the following observation residuals:

\[
r_G\left( z_{k+1}^k, \mathcal{X} \right) = p_k^{V^s} - p_k^{G^s}
\]

(15)

Different from the assumption in [14] that GNSS factors are added to the system only when the GNSS measurement covariance is smaller than the LVIO measurement covariance, we noticed that the accuracy of outdoor GNSS positioning results is much higher than the LVIO local positioning results. The covariance threshold size for judging whether to add GNSS factors has little impact on the positioning accuracy. Therefore, we present that once the GNSS signal is detected by the system, the GNSS factor is added to the factor graph. In this way, even if the mobile carrier enters the GNSS rejection environment (such as the indoor parking lot or tunnel), it can also provide a more accurate initial observation value after GNSS correction. The fusion strategy of GNSS and LVIO is shown in Figure 8.

Further, considering the possible overlap of the mobile carrier travel area, i.e., the mobile carrier travels to the same position again after a period of time, we also added a loopback detection link to establish the loopback constraint that exists between non-adjacent frames. Unlike introducing another sensor (GNSS) for global correction of the local sensor (LVIO), the loopback factor establishes the correlation between the current observed frames and the historical data by the local sensor itself to obtain a globally consistent estimate. The conditions for adding the loopback factor are similar to those of GNSS. Once the carrier motion trajectory is detected to travel to the environment passed by the history, the loop factor is added to the factor graph. By registering with the point cloud of the prior map, the historical trajectory is corrected, and the global pose estimation result with higher accuracy is obtained.
Figure 8. Fusion strategy of GNSS and LVIO. The initial rotation $R_{L}^{G}$ of LVIO in the local frame and the global frame is set to identity matrix. GNSS provides global constraints to LVIO to correct the global position of LVIO and update $R_{L}^{G}$, and the new $R_{L}^{G}$ is used for the next frame of LVIO.

5. Experimental Results

5.1. Real-Time Performance

5.1.1. Indoor Environment

For evaluating the real-time performance of our algorithm, we randomly selected the MH_01_easy dataset for indoor experiments. Since the strategy of adding line feature constraints to the VIO subsystem of our algorithm is referenced to PL-VIO, the time consumption of several threads involving line features of PL-VIO and this algorithm is compared. As shown in Figure 9, the appropriate selection of hidden parameters and the least-squares-based geometric constraint matching strategy have positive effects on real-time performance. The time cost of the line feature extraction and matching process and the line feature tracking process of the proposed algorithm is about one-third that of similar algorithms.

The time consumption of the line feature matching process is shown in Figure 9a. In the period of (110 s, 170 s), the carrier passes through the well-lit factory wall duct area. The number of line features extracted by both algorithms increases, and the corresponding time cost of line feature matching also increases with the number of line features. However, unlike PL-VIO which is significantly affected by the increase in the number of line features, the line feature matching the process time of our algorithm remains relatively stable within 1 ms. The reason is that the number of invalid line features is reduced due to the geometric constraint-based line feature matching strategy, which improves the accuracy of line feature matching between the front and current frames of the image. In the time-consuming of line feature tracking process shown in Figure 9b, it can be seen that in the initial stage (0 s, 5 s) of the visual subsystem, the line feature tracking process of the two systems takes longer. The reason is the UAV is at rest during this time and the VIO subsystem does not receive sufficient motion excitation, which leads to its incomplete initialization. After 5 seconds of initialization, the PL-VIO line feature tracking time remains stable at about 125 ms, while the time consumption of our algorithm is about 4/5 less than that of PL-VIO, about 25 ms. It has a strong positive effect on the real-time performance of the fusion system in the actual operating environment.

Although as shown in Figure 9c, the time-consuming cost of the line feature residual optimization process increases by about 10 ms, the time-consuming of the line feature tracking process is significantly reduced. Thus, the proposed method leads to a decrease in the total time cost of the three line feature-related processes in the fusion system, which still has a better real-time performance overall than before the improvement.
5.1.2. Outdoor Environment

Since the distribution characteristics of line features are different in indoor and outdoor environments, in order to fully evaluate the superior performance of this algorithm in terms of real-time, we selected the Hong Kong 0428 dataset for outdoor experiments. The experimental results are shown in Figure 10.

Different from the indoor environment, the outdoor environment has more complex conditions of light refraction and reflection, and the dynamic interference such as pedestrians and vehicles in the driving process of moving vehicles. The time consumption of the line feature matching process in the outdoor environment is shown in Figure 10a. It can be seen that the line feature matching time of PL-VIO in the outdoor environment is about 10 ms on average, and our algorithm still maintains the same good real-time characteristics as the indoor environment. In the line feature tracking process shown in Figure 10b, it can be seen that the line feature tracking process in the initialization phase (0 s, 5 s) of the visual subsystem is abnormally high for both systems. The same reason is that the VIO system is not provided sufficient motion excitation at the beginning of the vehicle stationary phase. It can be concluded that it is more difficult to match and track
visual line features in the outdoor environment, and the time consumed for line feature tracking rises about 3–4 times compared with the indoor environment. However, the time consumed by our algorithm is still greatly shortened compared with similar algorithms, leaving more time for the optimization of a multi-sensor fusion at the back-end.

![Figure 10](image_url)

Figure 10. Real-time comparison experiment of Hong Kong 0428 dataset. (a) Line feature extraction and matching process. (b) Line feature tracking process. (c) Line feature residual optimization process.

In addition, as shown in Figure 10c, the time-consuming cost of the line feature residual optimization process is not much different from that of PL-VIO. Combining the above three time-consuming threads, it can be proved that our algorithm can achieve better real-time performance in different environments.

5.2. Positioning Accuracy

5.2.1. Indoor Environment

In this study, the EuROC dataset was used to compare and verify the positioning accuracy of each algorithm in the indoor environment. The experimental environment was in a factory with complex signal refraction and reflection conditions. LiDAR frequently fails in the experimental environment, so no comparison was made. The comparison of the point-line feature results extracted by PL-VIO and our algorithm in the experimental environment is shown in Figure 11.
Figure 11. Comparison of point-line feature extraction results in poor lighting conditions and weak texture environment. (a) Point-line feature extraction results of PL-VIO. (b) Point-line feature extraction results of our algorithm.

As seen in Figures 12 and 13 and Table 1, the introducing line features in the image frames to add additional feature constraints can reduce the positioning error of the system to some extent, especially in areas with dim light and poor textures. For example, during the (160 s, 240 s) time, the UAV flight area is nearly full of darkness. Thus it is difficult for Harris corner point detection method to extract the corner points with large grayscale difference from the surrounding pixel blocks. The reduction in the number of effective feature points directly leads to poor feature tracking accuracy. Therefore, the absolute trajectory error of VINS-Mono based on point features is larger in this interval (as shown in Figure 13a). In contrast, PL-VIO based on point-line features and the present algorithm are less negatively affected by illumination, and the absolute trajectory error remains within 0.6 m. In a longitudinal comparison of similar algorithms based on point and line features, the accuracy of our algorithm is significantly improved over PL-VIO. These results are attributed to the high quality of matching by the geometric constraint strategy, which avoids the missegmentation of long-line features and then misclassification as invalid matches. The experimental results demonstrate the robustness and accuracy of this algorithm in the case of single system failure, which is important for localization in complex indoor environments.

Table 1. Motion estimation errors of each algorithm in indoor dataset.

| Sequence       | Vins_Mono (w/o loop) | Vins_Mono (w/ loop) | PL-VIO  | LVI-SAM | Purposed |
|----------------|----------------------|---------------------|---------|---------|----------|
|                | ATE_RMSE(m)/Mean Error(m) |                     |         |         |          |
| MH_01_easy     | 0.213/0.189          | 0.188/0.158         | 0.093/0.081 | 0.181/0.147 | 0.073/0.062 |
| MH_02_easy     | 0.235/0.193          | 0.188/0.157         | 0.072/0.062 | 0.182/0.167 | 0.045/0.039 |
| MH_03_medium   | 0.399/0.321          | 0.402/0.315         | 0.260/0.234 | 0.400/0.308 | 0.056/0.050 |
| MH_04_difficult| 0.476/0.423          | 0.422/0.348         | 0.364/0.349 | 0.398/0.399 | 0.079/0.075 |
| MH_05_difficult| 0.426/0.384          | 0.370/0.309         | 0.251/0.238 | 0.380/0.287 | 0.139/0.127 |
| V1_01_easy     | 0.157/0.137          | 0.145/0.121         | 0.078/0.067 | 0.142/0.119 | 0.040/0.037 |
| V1_03_difficult| 0.314/0.275          | 0.329/0.289         | 0.205/0.179 | 0.322/0.283 | 0.077/0.069 |
| V2_01_easy     | 0.133/0.115          | 0.120/0.108         | 0.086/0.072 | 0.121/0.110 | 0.056/0.048 |
| V2_02_medium   | 0.287/0.244          | 0.293/0.255         | 0.150/0.097 | 0.291/0.250 | 0.089/0.078 |
| V2_03_difficult| 0.343/0.299          | 0.351/0.315         | 0.273/0.249 | 0.351/0.308 | 0.098/0.092 |
Figure 12. Comparison of trajectory fitting curve of each algorithm in the indoor dataset. (a) Global trajectory fitting curve. (b) Details of local trajectory. (c) Details of local trajectory.

Figure 13. Comparison of positioning results of each algorithm in the indoor dataset. (a) APE_RMSE error fitting curve. (b) Comparison of index of absolute trajectory error.
5.2.2. Outdoor Environment

To evaluate the performance of the algorithm we conducted in the outdoor environment, the Hong Kong dataset was used for performance evaluation and it was compared with other similar advanced algorithms. The experimental equipment and environment are shown in Figure 14. The sensor models are as follows: the camera is BFLY-U3-23S6C-C, the LiDAR is HDL 32E Velodyne, IMU is Xsens Mti 10, and the GNSS receiver is u-blox M8T. In addition, we utilized the high-grade RTK GNSS/INS integrated navigation system, NovAtel SPAN-CPT, as the ground truth.

![Experimental equipment and environment](image-url)

To verify the superior performance of each aspect of our system, we performed ablation experiments, constructed without GNSS global correction (*), without visual line features (#), and our complete system (proposed), respectively. The experimental results are shown in Figures 15 and 16 and Table 2.

| Sequence               | Hong Kong 0428          | Hong Kong 0314          |
|------------------------|-------------------------|-------------------------|
| Vins_Mono (w/o loop)   | 101.735/89.470          | 40.651/35.035           |
| Vins_Mono (w/ loop)    | 76.179/67.535           | 19.191/15.617           |
| LIO-SAM                | 7.181/6.787             | 41.933/39.672           |
| LVI-SAM                | 9.764/9.061             | 3.065/2.557             |
| Purposed(*)            | 9.475/8.884             | 2.842/2.456             |
| Purposed(#)            | 5.808/5.436             | 2.595/2.041             |
| Purposed               | 5.299/4.955             | 2.249/1.880             |

Figure 14. Experimental equipment and environment. (a) The experimental vehicle and sensors setup. (b) Image of experimental environment.

Table 2. Motion estimation errors of each algorithm on outdoor dataset.
Figure 15. Comparison of trajectory fitting curve of each algorithm in the indoor dataset. (a) Global trajectory fitting curve. (b) Details of local trajectory. (c) Details of local trajectory.

Figure 16. Comparison of positioning results of each algorithm on outdoor dataset. (a) APE_RMSE error fitting curve. (b) Comparison of index of absolute trajectory error.
From Figure 15, it can be seen that VIO and LIO, which are mainly based on a single sensor, each have different defects. First of all, VIO (VINS-Mono) is introduced. Before starting the movement, the moving carrier stopped at the roadside parking position for about 10 seconds. VIO was not given a large motion excitation during this period, which led to the VIO not being initialized properly. Secondly, the cumulative error caused by the scale uncertainty of the monocular camera increased significantly over time, and a large-scale estimation error was already generated at the second lap. Although the scale drift of LIO (LIO-SAM) is not large, it will immediately fail and keep restarting in the complex area of signal fold reflection. After LiDAR resumes operation, the translation and rotation of the current frame will be accumulated based on the positional estimation at the last frame that did not fail, resulting in the misjudgment of stopping the motion at the carrier motion to (50 m, 150 m). When the carrier moves to the corner, LIO re-estimates the position and attitude. It was misjudged that the carrier stopped at (50 m, 150 m) for a while and then began to turn, so it lost the estimated position and attitude for a period of time, which led to a large positioning error.

In a longitudinal comparison with the other LVIO system (LVI-SAM), we can conclude that our complete algorithm maintains a lower drift rate and localization integrity, which benefits from the extra constraint of line features and the global correction of GNSS. In conclusion, even in complex outdoor environments, our algorithm still outperforms other advanced algorithms.

5.3. Mapping Performance

As a demonstration of the superiority of our algorithm in building maps, we compared the building results with other advanced algorithms on different datasets. The visual line feature extraction and map building results are shown in Figure 17. Compared with PL-VIO, our algorithm has a great improvement in the number of visual line features extracted, which is attributed to the improved line feature extraction strategy. In a factory environment with complex lighting conditions, the line features in the actual environment will look minutely curved due to the refraction of light. Due to the proper value of the threshold value \( D \) of the density of homogeneous points, the angle tolerance of fitting pixels to approximate rectangles in this environment can be improved, thus increasing the number of line feature extraction. Further, the accuracy of the bit pose estimation is also substantially improved by the combination of the improved line feature extraction and tracking optimization strategies.

![Figure 17](image_url)

Figure 17. Comparison of visual line feature extraction mapping. (a) show the mapping of each subsystem before improvement, and (b) show our algorithm mapping.

Further, comparison of the LiDAR point cloud detail views is shown in Figure 18. The more accurate VIO pose estimation after the line features are added provides a more accurate initial value for LiDAR scan matching, and reduces a large number of point cloud mismatching. Comparison of global point cloud trajectories is shown in Figure 19.
The area marked by circles demonstrates that the data drift caused by cumulative errors is significantly reduced by adding a GNSS factor and loop factor to our algorithm.

![Figure 18. Comparison of LiDAR point cloud map details. (a) show the mapping of each subsystem before improvement, and (b) show our algorithm mapping.](image1)

![Figure 19. Comparison of global point cloud trajectory. (a) show the mapping of each subsystem before improvement, and (b) show our algorithm mapping.](image2)

6. Discussion

Multi-sensor fusion positioning technology based on SLAM provides new opportunities for the high-precision positioning of mobile carriers. In this study, two problems that need to be further explored in the existing LVIO fusion system are proposed. The first problem is that LVIO system needs enough environmental feature information. According to the previous studies of Pumarola et al. [11] and Fu et al. [12], theoretically, the accuracy of the fusion system can be improved by increasing the constraint of visual line features. Huang et al. also proved that the average positioning error of the fusion system based on point-line feature can generally be improved, from the traditional 2.16% to 0.93% [18]. In this study, the steps of increasing visual line feature constraints are further optimized. The Monte Carlo method was used to select the appropriate scaling ratio and the density threshold of homogeneous points, which improves the angle tolerance of pixel fitting to line features. To a certain extent, it reduces the probability that short segments are wrongly judged as invalid features. According to the experiment of parameter selection in Section 3.1, compared with the indoor environment where the angle and translation of line features change little, the movement of line features between consecutive frames is more complicated in the outdoor environment. Therefore, the density threshold of homogeneous points needs to be lowered to reduce the probability that the valid line feature pairs are misjudged as invalid matches when turning sections. The results of outdoor real-time analysis shown in Section 5.1.2. show that the traditional method based on point-line features is difficult to match and track the visual line features in the outdoor environment,
which takes a long time. However, the time consumption of this algorithm maintained a low level, which is beneficial to leave more time for back-end fusion optimization.

To solve the problem of line feature mismatching during the movement of carriers, Zhou et al. established a constraint equation using the 6-DOF Plücker coordinates of line features to perform matching optimization [19]. However, this increases the computational complexity of the fusion system, which is inconsistent with the lightweight requirements of autonomous driving positioning. In this study, the link of line feature constraint matching is simplified, and the original 6-DOF parameters are replaced by 4-DOF which represents the movement of line features for optimization. Thus, it can reduce the computational complexity of the system and effectively improve the inter-frame matching accuracy of line features. To explore the superiority of the proposed algorithm in real-time and positioning accuracy, we compared the precision and the time-consuming of three processes related to line features of this algorithm with several similar advanced algorithms in different environments. The experimental results show that our optimization strategy based on front-end point-line features effectively achieves the positive balance between reducing time consumption and improving accuracy.

The second problem to be solved is the global optimization of LVIO local pose estimation results by introducing global constraints. To further improve the positioning accuracy of local sensors, Qin et al. proposed a GNSS and local sensor fusion method to construct GNSS residual factors to correct the cumulative error of VIO [25]. Further, we propose a factor graph based on Bayesian network, in which GNSS observations are added as global constraint factors. The accumulated errors of LVIO are corrected by using GNSS observations within 0.1 s interval from LVIO keyframes as global constraint. In this study, it is proved that GNSS global constraint factor can effectively correct LVIO positioning error in the outdoor environment. It should be noted that since the coordinate of the current frame of LVIO is calculated from the coordinate of the previous frame, long-term observation or long moving distance will lead to more serious data drift. However, the GNSS observations are in the global coordinate, so long-term observation is not related to the data drift. Therefore, we can reasonably speculate that the longer the algorithm runs, the more obvious the correction effect of GNSS on LVIO will be. More comprehensively, LVIO will continue local positioning in GNSS rejection environment, so the positioning continuity of mobile carriers in different environments can be effectively guaranteed.

7. Conclusions

In this study, a LiDAR-Visual-Inertial Odometry based on optimized visual point-line features is proposed, taking advantage of the heterogeneous complementary characteristics of multiple sensors. First, a visual line feature extraction and matching optimization method is proposed. By improving the line feature extraction in the scale space and selecting the appropriate scaling ratio and same-sex point density threshold, the number of line features extracted in the light complex environment is largely improved to provide richer feature information for the front-end. Meanwhile, the original 6-DOF parameter optimization problem is further improved to a 4-DOF parameter optimization problem by using a least squares-based line feature constrained matching strategy. The complexity of the fusion system is reduced, and more accurate visual pose estimation is effectively accomplished. Second, the LiDAR point cloud is projected into the visual coordinates for depth correlation. Meanwhile, the initial pose estimation provided by the optimized VIO is used to help LiDAR scan matching. Finally, a factor graph method based on Bayesian networks is established. Two global constraint factors are added to the factor graph framework to constrain LVIO globally, which are the global constraint of GNSS factors from external sensors and the loop factor constraint of local sensors. The experimental results show that the algorithm can achieve real-time attitude estimation with good localization and mapping accuracy in different environments.

In the future, we will further improve and refine our work in the following aspects. First, the point cloud alignment algorithm of the loop factor in this study utilizes the
traditional ICP algorithm, which is time-consuming to perform the nearest domain search using the KD tree. Thus, we will consider the improvement of the point cloud alignment algorithm next. Second, the inclusion of the GNSS factor in this study only utilizes the GNSS pseudo range single point positioning result. Although it is relatively simple and feasible on the vehicle platform with only one GNSS receiver, there is still room for improvement in the positioning accuracy of GNSS. A more accurate correction of LVIO by using higher accuracy RTK positioning results will be considered in the next step. Finally, since our proposed fusion system consists of two subsystems with high runtime computational resource requirements, we will work on reducing the resource occupation rate of the algorithm. Further, we will evaluate the positioning accuracy of the algorithm on vehicles with limited computing resources.

**Author Contributions:** Conceptualization, Z.Z., C.S., H.Z. and X.L.; methodology, X.H.; software, X.H.; validation, S.P. and L.D.; formal analysis, X.H.; investigation, W.G. and X.H.; resources, S.P., W.G. and X.H.; writing—original draft preparation, X.H.; writing—original draft preparation, X.H.; writing—review and editing, W.G. and X.H.; supervision, S.P. and W.G.; project administration, W.G.; funding acquisition, S.P. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research study was funded by the Fundamental Research Funds for the Central Universities (2242021R41134) and the Research Fund of the Ministry of Education of China and China Mobile (MCM20200J01).

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Not applicable.

**Conflicts of Interest:** The authors declare no conflict of interest.

**References**

1. Qian, Q.; Bai, T.M.; Bi, Y.F.; Qiao, C.Y.; Xiang, Z.Y. Monocular Simultaneous Localization and Mapping Initialization Method Based on Point and Line Features. *Acta Opt. Sin.* 2021, 41, 1215002.
2. Shan, T.; Englot, B. LeGO-LOAM: Lightweight and Ground-Optimized Lidar Odometry and Mapping on Variable Terrain. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 4758–4765.
3. Zuo, X.; Xie, X.; Liu, Y.; Huang, G. Robust visual SLAM with point and line features. In Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada, 24–28 September 2017; pp. 1775–1782.
4. Zhang, J.; Singh, S. Laser-Visual-Inertial Odometry and Mapping with High Robustness and Low Drift. *J. Field Robot.* 2018, 35, 1242–1264. [CrossRef]
5. Qin, T.; Li, P.; Shen, S. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Trans. Robot.* 2018, 34, 1004–1020. [CrossRef]
6. Mur-Artal, R.; Tardós, J.D. ORB-SLAM2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Trans. Robot.* 2018, 33, 1255–1262. [CrossRef]
7. Forster, C.; Carlone, L.; Dellaert, F.; Scaramuzza, D. On-manifold preintegration for real-time visual–inertial odometry. *IEEE Trans. Robot.* 2017, 33, 1–21. [CrossRef]
8. Forster, C.; Zhang, Z.; Gassner, M.; Werlberger, M.; Scaramuzza, D. SVO: Semi-direct visual odometry for monocular and multicamera systems. *IEEE Trans. Robot.* 2016, 33, 249–265. [CrossRef]
9. He, Y.; Zhao, J.; Guo, Y.; He, W.H.; Yuan, K. Pl-vio: Tightly-coupled monocular visual–inertial odometry using point and line features. *Sensors* 2018, 18, 1159. [CrossRef] [PubMed]
10. Wen, H.; Tian, J.; Li, D. PLS-VIO: Stereo Vision-inertial Odometry Based on Point and Line Features. In Proceedings of the 2020 International Conference on High Performance Big Data and Intelligent Systems (HPBD&IS), Shenzhen, China, 23 May 2020.
11. Pumarola, A.; Vakhitov, A.; Agudo, A.; Sanfelice, A.; Moreno-Noguer, F. PL-SLAM: Real-time monocular visual SLAM with points and lines. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; pp. 4503–4508.
12. Fu, Q.; Wang, J.; Yu, H.; Ali, I.; Zhang, H. PL-VINS: Real-Time Monocular Visual-Inertial SLAM with Point and Line. [DB/OL]. Available online: https://arxiv.org/abs/2009.07462v1 (accessed on 27 November 2021).
13. Zhang, J.; Singh, S. LOAM: Lidar Odometry and Mapping in Real-time. In Proceedings of the 2014 Robotics: Science and Systems, Berkeley, CA, USA, 12–16 July 2014; pp. 9–17.
14. Shan, T.; Englot, B.; Meyers, D.; Wang, W.; Rus, D. LIO-SAM: Tightly-Coupled Lidar Inertial Odometry via Smoothing and Mapping. [DB/OL]. Available online: https://arxiv.org/abs/2007.00258v3 (accessed on 27 November 2021).
15. Xiang, Z.; Yu, J.; Li, J.; Su, J. ViLiVO: Virtual LiDAR-Visual Odometry for an Autonomous Vehicle with a Multi-Camera System. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macao, China, 3–8 November 2019; pp. 2486–2492.
16. Chen, S.; Zhou, B.; Jiang, C.; Xue, W.; Li, Q. A LiDAR/Visual SLAM Backend with Loop Closure Detection and Graph Optimization. Remote Sens. 2021, 13, 2720. [CrossRef]
17. Lin, J.; Zheng, C.; Xu, W.; Zhang, F. R2LIVE: A Robust, Real-Time, LiDAR-Inertial-Visual Tightly-Coupled State Estimator and Mapping. [DB/OL]. Available online: https://arxiv.org/abs/2102.12400 (accessed on 27 November 2021).
18. Huang, S.; Ma, Z.; Mu, T.; Fu, H.; Hu, S. Lidar-Monocular Visual Odometry using Point and Line Features. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation(ICRA), Online, 1–15 June 2020; pp. 1091–1097.
19. Zhou, L.; Wang, S.; Kaess, M. DPLVO: Direct Point-Line Monocular Visual Odometry. IEEE Robot. Autom. Lett. 2021, 6, 7113–7120. [CrossRef]
20. He, X.; Pan, S.G.; Tan, Y.; Gao, W.; Zhang, H. VIO-GNSS Location Algorithm Based on Point-Line Feature in Outdoor Scene. Laser Optoelectron. Prog. 2022, 56, 1815002.
21. Silva, V.D.; Roche, J.; Kondoz, A. Fusion of LiDAR and camera sensor data for environment sensing in driverless vehicles. arXiv 2018, arXiv:1710.06230.
22. Liu, X.; Li, D.; Shi, J.; Li, A.; Jiang, L. A framework for low-cost Fusion Positioning with Single Frequency RTK/MEMS-IMU/VIO. J. Phys. Conf. Ser. 2021, 1738, 012007. [CrossRef]
23. Mascaro, R.; Teixeira, L.; Hinzmann, T.; Siegwart, R.; Gomsf, M.C. Graph-optimization based multi-sensor fusion for robust uav pose estimation. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation(ICRA), Brisbane, Australia, 21–24 May 2018; pp. 1421–1428.
24. Woosik, L.; Eckenhoff, K.; Geneva, P.; Huang, G.Q. Intermittent GPS-aided VIO: Online Initialization and Calibration. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation(ICRA), Paris, France, 31 May–6 June 2020; pp. 5724–5731.
25. Qiu, T.; Cao, S.; Pan, J.; Shen, S. A General Optimization-Based Framework for Global Pose Estimation with Multiple Sensors. [DB/OL]. Available online: https://arxiv.org/abs/1901.03642 (accessed on 27 November 2021).
26. Fernandes, L.A.F.; Oliveira, M.M. Real-time line detection through an improved Hough transform voting scheme. Pattern Recognit. 2008, 41, 299–314. [CrossRef]
27. Nieto, M.; Cuevas, C.; Salgado, L.; Narciso, G. Line segment detection using weighted mean shift procedures on a 2D slice sampling strategy. Pattern Anal. Appl. 2011, 14, 149–163. [CrossRef]
28. Akinlar, C.; Topal, C. EDLines: A real-time line segment detector with a false detection control. Pattern Recognit. Lett. 2011, 32, 1633–1642. [CrossRef]
29. Gioi, R.; Jakubowicz, J.; Morel, J.M.; Randall, G. LSD: A fast line segment detector with a false detection control. IEEE Trans. Pattern Anal. Mach. Intell. 2010, 32, 722–732. [CrossRef] [PubMed]
30. Shan, T.; Englot, B.; Ratti, C.; Rus, D. LVI-SAM: Tightly-Coupled Lidar-Visual-Inertial Odometry via Smoothing and Mapping. [DB/OL]. Available online: https://arxiv.org/abs/2104.10831 (accessed on 27 November 2021).