2021 IEEE International Conference on Robotics and Automation (ICRA 2021)
May 31 - June 4, 2021, Xi'an, China

Avoiding Degeneracy for Monocular Visual SLAM with Point and Line Features

Hyunjun Lim[1], Yeeun Kim[1], Kwangik Jung[2], Sumin Hu[1], and Hyun Myung*, *Senior Member, IEEE*

***Abstract*— In this paper, a degeneracy avoidance method for a point and line based visual SLAM algorithm is proposed. Visual SLAM predominantly uses point features. However, point features lack robustness in low texture and illuminance variant environments. Therefore, line features are used to compensate the weaknesses of point features. In addition, point features are poor in representing discernable features for the naked eye, meaning mapped point features cannot be recognized. To overcome the limitations above, line features were actively employed in previous studies. However, since degeneracy arises in the process of using line features, this paper attempts to solve this problem. First, a simple method to identify degenerate lines is presented. In addition, a novel structural constraint is proposed to avoid the degeneracy problem. At last, a point and line based monocular SLAM system using a robust optical-flow based lien tracking method is implemented. The results are verified using experiments with the EuRoC dataset and compared with other state-of-the-art algorithms. It is proven that our method yields more accurate localization as well as mapping results.**

I. Introduction

As a result of the advancement of unmanned mobile robots, Simultaneous Localization and Mapping (SLAM) has been extensively studied. Because GPS typically does not function in indoor environments, SLAM based on cameras, LiDARs, and Inertial Measurement Units (IMUs) have become attractive research topics. Among these sensors, cameras are the most widely used ones as they have become low-cost and easily accessible [1], [2]. Thus, visual SLAM have emerged as an essential technology in robotics, augmented/virtual reality, and autonomous driving.

Nowadays, Visual SLAM algorithms are predominantly feature-based method due to its fast performance. However, since most feature-based algorithms are based on point features, there is a high probability that descriptors may change and tracking fails in situations with large illumination variances or low-texture [3], [4]. In addition, as most point-based algorithms are sparse, the identification of the environment structure in the mapping result is challenging.

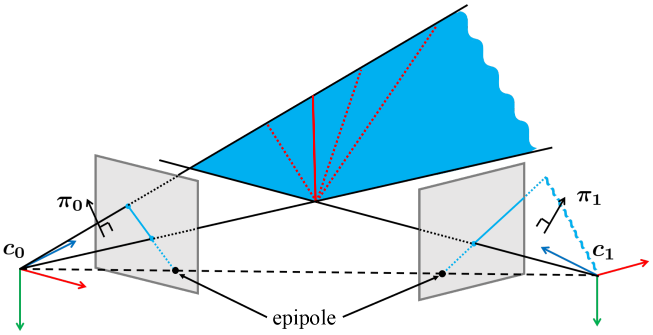


Fig. 1. Illustration of degeneracy. Degeneracy occurs when line is in proximity to the epipole. In the triangulation process, π_0 and π_1 formed by each camera center and the observed line feature, coincide with the epipolar plane. Then, the estimated 3D line should be on a plane colored in blue. Although the actual 3D line is a red solid line, triangulation can mis-determine the estimated line as one of the infinitely many dotted lines on the plane.

Since such shortcomings inherently exist for point features, some studies have shown that line features are better for visual SLAM. A line feature consists of multiple points, making it resilient against environmental changes. Also, it serves as an excellent landmark even in environments with low texture such as corridors. Furthermore, mapping with line features reveal the geometry of the surrounding, becoming easily discernible to the human eye.

However, under degeneracy conditions, line features are no longer robust. Degeneracy refers to a phenomenon in which a line feature does not manifest as a single 3D line during triangulation. This occurs when the lines observed in both images pass through the epipoles. In this case, as shown in Fig. 1, the planes π_0 and π_1 formed by each camera center and the observed line feature, coincide with the epipolar plane. Then, the estimated 3D line should be on a plane colored in blue. Although the actual 3D line is a red solid line, triangulation can mis-determine the estimated line as one of the infinitely many dotted lines on the plane. As a result, the 3D line feature cannot be properly determined, resulting in large reconstruction errors. The degeneracy problem occurs more often when using line features than point features. This is because degeneracy of a point feature occurs only when the point is close to the epipole, whereas in the case of a line feature, the distances of all points included in the line feature must be distant from the epipole. As degeneracy affects the localization accuracy and the mapping result, it must be addressed in the line feature-based SLAM.

In this paper, we propose a variation of visual SLAM that

[1] H. Lim, Y. Kim, and S. Hu are with School of Electrical Engineering, Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Republic of Korea {tp02134, yeeunk, 2minus1}@kaist.ac.kr

[2] K. Jung is with Department of Civil and Environmental Engineering of KAIST, Daejeon, Republic of Korea ankh88324@kaist.ac.kr

*H. Myung is with School of Electrical Engineering, KI-AI, and KI-R, KAIST, Daejeon, Republic of Korea hmyung@kaist.ac.kr

This work was supported by the Defense Challengeable Future Technology Program of Agency for Defense Development, Republic of Korea. The students are supported by Korea Ministry of Land, Infrastructure and Transport (MOLIT) as "Innovative Talent Education Program for Smart City" and BK21 FOUR.

978-1-7281-9077-8/21/$31.00 ©2021 IEEE

can avoids the degeneracy problem of line features stated above. The main contributions of this study are as follows:

- A simple method to identify degenerate lines in pure translation is presented.
- A novel structural constraint is proposed to avoid the degeneracy problem.
- A point-and-line-based monocular SLAM system using a robust optical-flow-based line tracking method is proposed.

The remainder of this paper is organized as follows. Section II provides an overview of related works. Section IV describes the proposed method in detail. Section V presents the experimental results. Finally, Section VI summarizes our contributions and discusses future work.

II. Related works

Until recently, methods using point features were predominant in SLAM [5], [6] and Visual-Inertial System (VINS) [7]–[11] algorithms. The reasons were clear. Point features can be extracted from any image. In addition, the comparison process is simple because it can be expressed in 2D coordinate on the image. However, point features have poor tracking performance in environments with low texture or illumination variance. Furthermore, the sparsity of the estimated 3D point features makes them inappropriate for mapping. To improve the performance, some studies focused on line features instead.

Among the many studies with line features, Bartoli *et al.* [12] laid the foundations on how to use line features in SLAM. They proposed a method using the line feature in structure-from-motion. Since line features have a higher degree of freedom than point features, a parameterization method was required. They used Plücker coordinates and orthonormal representations. Also, the re-projection error of the line feature was proposed for bundle adjustment. Zhang *et al.* [13] went further by implementing a filtering-based stereo SLAM with line features using the method above.

Through the studies above, existing point feature-based algorithm have been extended to incorporate line features. First, Zheng *et al.* [14] and Yang *et al.* [15] used line features based on MSCKF [9]. Next, various algorithms [16]–[19] that add line features based on ORB-SLAM [6] have been proposed. Lastly, He *et al.* [20] and Fu *et al.* [21] applied line feature based on VINS-Mono [11]. Through the studies above, it was possible to improve localization performance. In addition, structural characteristics of the surrounding environment could be obtained through the mapping of line features.

However, degenerate lines appeared in line-based SLAM. The commonly used re-projection error was vulnerable to degeneracy. A line is defined by its normal and direction vectors in the Plücker coordinates [12]. However, the reprojection error is calculated only by a line's normal vector, meaning that the optimization process can only affect this vector. In degenerate cases, lines have anomalous direction vectors which cannot be corrected with the reprojection error alone.

The degeneracy of the line feature is much more severe than that of the point feature [22]. Therefore, there were studies [23], [24] that analyzed the degeneracy occurring in the process of applying line features to visual SLAM. In addition, the MSCKF-based approach proposed by Yang *et al.* [15] analyzed the degenerate motions in two different line triangulation methods. In such studies, when line degeneracy occurs, all the corresponding lines were removed. Therefore, the loss of features was a limitation to the method. As a method to solve the degeneracy of lines, Ok *et al.* [25] proposed to reconstruct 3D line segment with artificial points if the image lines are nearly-aligned with the epipolar line. However, this method has limited applications in that it can only be employed when degenerate lines are in contact with other lines.

III. Preliminaries

A point feature in 3D can be intuitively represented as (x, y, z). In contrast, a line feature has 4-DoF, which can be represented in numerous manners [12]. In this study, we use two different methods of representation: Plücker coordinates and orthonormal representation.

A. Plücker Coordinates

Plücker coordinates are represented by 6-DoF consisting of a direction and a normal vector, as shown in Fig. 4(a). It can intuitively represent line features and is advantageous in obtaining the 3D line by triangulation or projecting the line to a 2D plane. With the 3D direction vector as $\boldsymbol{d}$ and the normal vector as $\boldsymbol{n}$, the mathematical representation of a 3D line is as follows:

$$\boldsymbol{L}(\boldsymbol{n}^\top, \boldsymbol{d}^\top)^\top \in \mathbb{R}^6. \tag{1}$$

B. Orthonormal Representation

Orthonormal representation is a minimal 4-DoF representation of a 3D line. It does not suffer from over-parameterization as Plücker coordinates do during the optimization step. The 3D line's orthonormal representation is as follows:

$$\boldsymbol{o}_j = [\psi_1, \psi_2, \psi_3, \phi] \tag{2}$$

where ψ_j is the j-th column of the 3D line's rotation matrix with respect to the camera coordinate system in Euler angles, and ϕ is the parameter of minimal distance from the camera center to a point on the line.

To optimize the 3D line features, we convert the over-parameterized Plücker coordinates representation to a minimal orthonormal representation. First, the Plücker coordinates are decomposed by QR decomposition [12], as follows:

$$\begin{bmatrix}\boldsymbol{n} & \boldsymbol{d}\end{bmatrix} = \begin{bmatrix}\frac{\boldsymbol{n}}{\|\boldsymbol{n}\|} & \frac{\boldsymbol{d}}{\|\boldsymbol{d}\|} & \frac{\boldsymbol{n}\times\boldsymbol{d}}{\|\boldsymbol{n}\times\boldsymbol{d}\|}\end{bmatrix} \begin{bmatrix} \| \boldsymbol{n} \| & 0 \\ 0 & \| \boldsymbol{d} \| \\ 0 & 0 \end{bmatrix}. \tag{3}$$

The first component on the right side in (3) represents the rotation matrix $\boldsymbol{U}$, which is a function of the Euler angle $\boldsymbol{\psi}$, as follows:

$$\boldsymbol{U} = \boldsymbol{R}(\boldsymbol{\psi}) = \begin{bmatrix}\frac{\boldsymbol{n}}{\|\boldsymbol{n}\|} & \frac{\boldsymbol{d}}{\|\boldsymbol{d}\|} & \frac{\boldsymbol{n}\times\boldsymbol{d}}{\|\boldsymbol{n}\times\boldsymbol{d}\|}\end{bmatrix}. \tag{4}$$

In addition, $\boldsymbol{W}$ calculated from normal and direction vector is used to calculate the 1-DoF ϕ. It contains information about the minimal distance, $\frac{\|\boldsymbol{d}\|}{\|\boldsymbol{n}\|}$, between the camera coordinate and a 3D line feature, as follows:

$$\begin{aligned}\boldsymbol{W} &= \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix} \\ &= \frac{1}{\sqrt{\| \boldsymbol{n} \|^2 + \| \boldsymbol{d} \|^2}} \begin{bmatrix} \| \boldsymbol{n} \| & -\| \boldsymbol{d} \| \\ \| \boldsymbol{d} \| & \| \boldsymbol{n} \| \end{bmatrix}. \end{aligned} \tag{5}$$

IV. The Proposed Method

The proposed algorithm is an extension of VINS-Mono to line features that overcome degeneracy problems. Because this algorithm has roots in VINS-Mono, IMU measurements use the pre-integration method [26]; optimization uses two-way marginalization with Schur complement [27]; and point feature manipulation uses optical-flow, triangulation, and re-projection error. The additional part of our work consists of Section III.*A*, degeneracy analysis, Section III.*B*, degeneracy avoidance using structural constraint, and Section III.*C*, line-based SLAM.

A. Degeneracy Identification

In general, degeneracy can be determined by calculating the distance to the epipole. To determine degeneracy, the position of the epipole must be calculated during the triangulation process. The equation for the position of the epipole is as follows:

$$\boldsymbol{p}_{epipole} = \boldsymbol{R}_{c_j}^{w\top}(\boldsymbol{t}_{c_i}^{w} - \boldsymbol{t}_{c_j}^{w}) \tag{6}$$

where $\boldsymbol{p}_{epipole}$ is the epipole of the i-th camera coordinate c_i with respect to the j-th camera coordinate c_j. $\boldsymbol{p}_{epipole}$ is calculated by rotation matrix $\boldsymbol{R}_{c_j}^{w}$ and translation $\boldsymbol{t}_{c}^{w}$ of camera with respect to world coordinate. As shown in (6), the position of the epipole is affected by the camera motion.

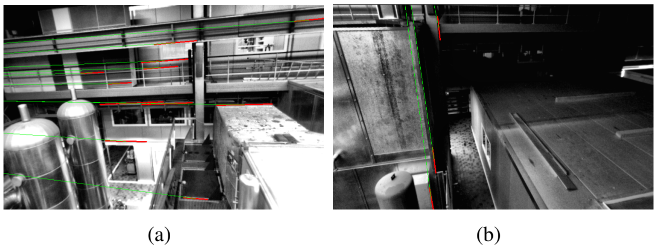
Particularly in pure translation, degeneracy conditions for line features are more susceptable. If the camera moves in pure translation, the epipole is located at infinity in the direction of the camera's movement. In this situation, when 2D lines are considered straight lines rather than line segments, all lines prallel to camera's movement can be considered to pass through the epipole. For example, in Fig. 2 (a) and (b), the camera's movement is on the x and y-axis directions, respectively. In this paper, when the camera moves in pure translation, line features parallel to the camera motion are determined which is when degeneracy occurs.

B. Degeneracy Avoidance Using Structural Constraint

When degeneracy of a line feature occurs, an erroneous direction vector is obtained during the mapping process. Through the line triangulation process (10), the direction vector of the line, $\boldsymbol{d}$, is obtained as follows:

$$\boldsymbol{d} = \begin{bmatrix} \pi_{0z}\pi_{1y} - \pi_{0y}\pi_{1z} \\ \pi_{0x}\pi_{1z} - \pi_{0z}\pi_{1x} \\ \pi_{0y}\pi_{1x} - \pi_{0x}\pi_{1y} \end{bmatrix}. \tag{7}$$

When the epipole is close to the line feature, the two plane vectors, $\boldsymbol{\pi}_0(\pi_{0x}, \pi_{0y}, \pi_{0z})$ and $\boldsymbol{\pi}_1(\pi_{1x}, \pi_{1y}, \pi_{1z})$ are parallel as shown in Fig. 1. Therefore, in theory, with each element of $\boldsymbol{d}$ in (7) as 0, $\boldsymbol{d}$ becomes a zero matrix. However, because the measurement noise of this line feature might exist, an erroneous direction vector might be calculated. Hence, when degeneracy occurs in this way, the direction vector of Plücker coordinates are affected and anomalous. Finally, A poor mapping result is obtained due to degenerate lines.


(a)

(b)

Fig. 2. Extracted lines of the image where degeneracy occurs in the pure translation of a camera in the EuRoC dataset [28]. The red lines are the extracted lines, and the green lines are the lines connecting the epipole and midpoint of the extracted line. (a) The camera moves along the x-axis, and degeneracy occurs in all lines along the x-axis direction. (b) The camera moves along the y-axis, and degeneracy occurs in all lines along the y-axis direction.

However, the line's re-projection error in (12) mainly used in the optimization process cannot solve the degeneracy. Because line's re-projectoin error uses only the normal vector, it cannot correct the direction vector's error. Existing algorithms resolved the degeneracy problem by removing degenerate lines. However, this approach leads to a loss of valuable line feature information. Therefore, there is a need to correct the line direction vector without removing the degenerate line features.

Through the degeneracy analysis, it was found that degeneracy often occurs in camera motion of pure translation. In addition, we consider the fact that the epipole and the vanishing point have the same position in pure translational camera motion [22] as shown in Fig. 3. Thus, degenerate lines always have vanishing points in the baseline direction. Therefore, lines with the same vanishing point can be cor-

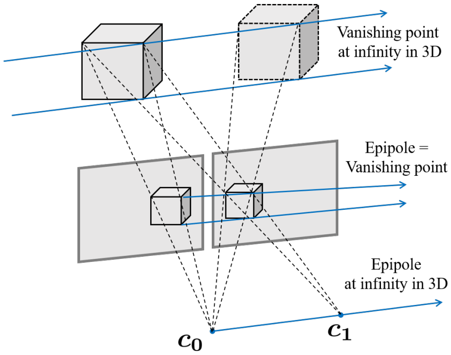
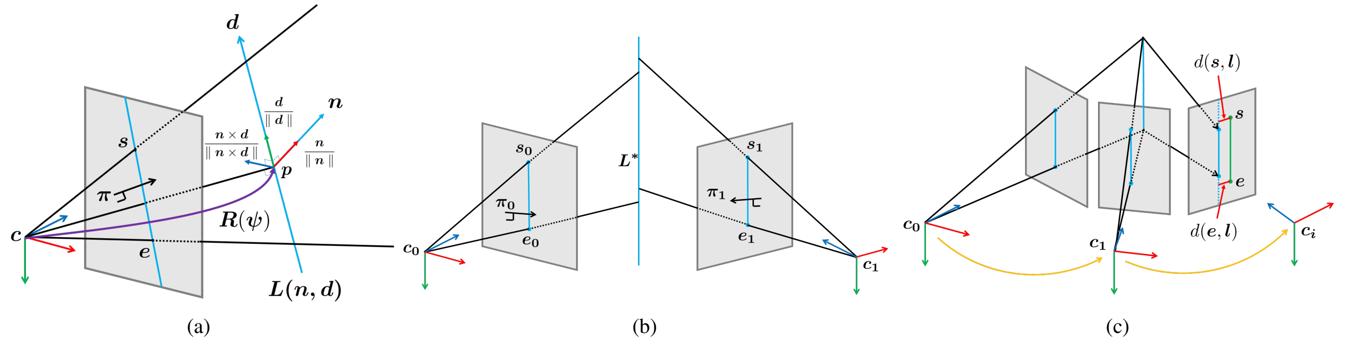


Fig. 3. Illustration showing that the vanishing point and the epipole are the same in the pure translation along x-axis. The object indicated by the dotted line is the translation of the camera expressed as its movement. As straight lines passing through the same point of an object are parallel, they have a common vanishing point at infinity in 3D. In addition, as it is the pure translation, the epipole is located at infinity in 3D. Therefore, the vanishing point and the epipole are identical when projected onto the 2D image.



(a) (b) (c)

Fig. 4. Illustration of line-based SLAM. (a) Plücker coordinates consist of normal and direction vectors. In addition, orthonormal representation consists of Euler angles of the line's rotation matrix and parameter of distance. (b) Each plane can be obtained through the camera position and line endpoints. The intersection of two planes creates a 3D line. (c) The re-projection of the line is used to define the cost function used in the optimization process. The distance between the re-projected line endpoints and the observed line is used as a residual.

rected for lines with degeneracy using the property that they are parallel in 3D. We define this as a structural constraint, and the residual for the parallel condition is:

$$\mathbf{r}_{\mathcal{S}}(\mathbf{z}_{ij}, \mathcal{X}) = \boldsymbol{d}_i \times \boldsymbol{d}_j \tag{8}$$

where $\boldsymbol{d}_i$ and $\boldsymbol{d}_j$ denote the direction vectors of the i-th and j-th lines, respectively. Thus, because a constraint is created by pairing all lines where degeneracy occurs, the accuracy of mapping can be improved by creating parallel lines. Compared to other methods [29], [30], such as using a J-linkage [31], [32] to find parallel lines to the vanishing point, this method is more efficient

C. Line-based SLAM

1) State Definition: The state vector used in our system is as follows:

$$\begin{aligned}
\mathcal{X} &= [\boldsymbol{x}_0, \boldsymbol{x}_1, \cdots, \boldsymbol{x}_{k-1}, \\
&\quad\ \lambda_0, \lambda_1, \cdots, \lambda_{m-1}, \\
&\quad\ \boldsymbol{o}_0, \boldsymbol{o}_1, \cdots, \boldsymbol{o}_{n-1}], \\
\boldsymbol{x}_i &= [\boldsymbol{p}^w_{b_i}, \boldsymbol{q}^w_{b_i}, \boldsymbol{v}^w_{b_i}, \boldsymbol{b}_a, \boldsymbol{b}_g], \\
\boldsymbol{o}_j &= [\boldsymbol{\psi}_j, \phi_j]
\end{aligned} \tag{9}$$

where $\mathcal{X}$ is the whole state and $\boldsymbol{x}_i$ is the state in the i-th sliding window and is composed of position, quaternion, velocity, and biases of the accelerometer and gyroscope in the appearing order. In addition, the whole state includes the inverse depths of point features represented as λ. Moreover, orthonormal representations of the lines, $\boldsymbol{o}$, which are newly added, are included.

2) Triangulation: When two lines from two different frames are matched, a 3D line feature can be estimated using triangulation, as shown in Fig. 4(b). A 2D line feature and a camera coordinate system can define a 3D plane as follows:

$$\begin{gathered}
\boldsymbol{\pi}_i = [\pi_{ix}, \pi_{iy}, \pi_{iz}, \pi_{iw}]^\top, \\
\begin{bmatrix} \pi_{ix} \\ \pi_{iy} \\ \pi_{iz} \end{bmatrix} = [\boldsymbol{s}_i]_\times \boldsymbol{e}_i, \\
\pi_{ix} c_{ix} + \pi_{iy} c_{iy} + \pi_{iz} c_{iz} + \pi_{iw} = 0
\end{gathered} \tag{10}$$

where $\boldsymbol{\pi}_i$ is the i-th plane vector in sliding windows. $\boldsymbol{\pi}_i$ consists of values (π_{ix}, π_{iy}, π_{iz}) that can be calculated using the cross product of $\boldsymbol{s}_i$ and $\boldsymbol{e}_i$, which are the two endpoints of a detected line segment, and π_{iw} that can be calculated using the camera center, $\boldsymbol{c}_i = (c_{ix}, c_{iy}, c_{iz})$. The two planes created by two camera coordinate systems, $\boldsymbol{c}_0$ and $\boldsymbol{c}_1$, intersect at a single 3D line, which becomes the 3D line feature. The dual Plücker matrix to determine a line intersecting two planes can be represented as follows:

$$\boldsymbol{L}^* = \begin{bmatrix} [\boldsymbol{d}]_\times & \boldsymbol{n} \\ -\boldsymbol{n}^\top & 0 \end{bmatrix} = \boldsymbol{\pi}_0 \boldsymbol{\pi}_1^\top - \boldsymbol{\pi}_1 \boldsymbol{\pi}_0^\top. \tag{11}$$

3) Measurement Model: The Plücker coordinates are transformed into an orthonormal representation by optimally finding the value that minimizes the cost function. For this reason, the 2D projection to an image plane should be obtained using the intrinsic parameter, as follows:

$$\boldsymbol{l} = \begin{bmatrix} l_1 \\ l_2 \\ l_3 \end{bmatrix} = \boldsymbol{K}\boldsymbol{n}^c = \begin{bmatrix} f_y & 0 & 0 \\ 0 & f_x & 0 \\ -f_y c_x & -f_x c_y & f_x f_y \end{bmatrix} \boldsymbol{n}^c \tag{12}$$

where $\boldsymbol{l}$ represents the re-projected line, $\boldsymbol{K}$ represents the camera's intrinsic parameter, and $\boldsymbol{n}^c$ represents re-projected line's normal vector. As this study uses normalized images, the intrinsic parameter is an identity matrix. Therefore, the normal of the 3D line feature is equal to the re-projected line.

In the case of Fig. 4(c), the projected 3D line feature does not have an end point. Therefore, the cost function, or the residual is calculated by finding the distance between the detected 2D line feature endpoints and the 3D line feature, as follow:

$$\begin{gathered}
\mathbf{r}_{\mathcal{L}}(\mathbf{z}^c_l, \mathcal{X}) = \begin{bmatrix} d(\boldsymbol{s}^c, \boldsymbol{l}^c) \\ d(\boldsymbol{e}^c, \boldsymbol{l}^c) \end{bmatrix}, \\
d(\boldsymbol{p}, \boldsymbol{l}) = \frac{\boldsymbol{p}^\top \boldsymbol{l}}{\sqrt{l_1^2 + l_2^2}}
\end{gathered} \tag{13}$$

where $\boldsymbol{r}_{\mathcal{L}}$ is the residual of the line feature and $\mathbf{z}^c_l$ is the measurement of detected line $\boldsymbol{l}$ with respect to camera coordinate c. And $\boldsymbol{r}_{\mathcal{L}}$ consists of the distance between $\boldsymbol{l}$ and endpoint $\boldsymbol{p}$ ($\boldsymbol{s}^c$ or $\boldsymbol{e}^c$).

Using the state defined in (9), the overall cost function for optimization is set as follows:

$$\min_{\mathcal{X}} \Bigg\{ \| \mathbf{r}_p - \mathbf{H}_p \mathcal{X} \|^2 + \sum_{k \in \mathcal{B}} \| \mathbf{r}_{\mathcal{B}}(\hat{\mathbf{z}}^{b_k}_{b_{k+1}}, \mathcal{X}) \|^2_{\mathbf{P}^{b_k}_{b_{k+1}}} + \sum_{(i,j) \in \mathcal{P}} \| \mathbf{r}_{\mathcal{P}}(\hat{\mathbf{z}}^{c_i}_{p_j}, \mathcal{X}) \|^2_{\mathbf{P}^{c_i}_{p_j}} + \sum_{(i,j) \in \mathcal{L}} \| \mathbf{r}_{\mathcal{L}}(\hat{\mathbf{z}}^{c_i}_{l_j}, \mathcal{X}) \|^2_{\mathbf{P}^{c_i}_{l_j}} + \sum_{i \in \mathcal{L}^*} \sum_{j \in \mathcal{L}^*} \| \mathbf{r}_{\mathcal{S}}(\hat{\mathbf{z}}_{ij}, \mathcal{X}) \|^2 \Bigg\} \tag{14}$$

where $\mathbf{r}_p$, $\mathbf{r}_{\mathcal{B}}$, and $\mathbf{r}_{\mathcal{P}}$ represent marginalization, IMU, and point measurement factor, respectively. These are the same with VINS-Mono. The newly added factors in the cost function are $\mathbf{r}_{\mathcal{L}}$ and $\mathbf{r}_{\mathcal{S}}$, which represent line measurement and structural constraints, respectively. In this case, $\mathcal{L}^*$ means parallel lines. In the optimization process, Ceres Solver [33] was used.

4) Optical-flow-based Line Tracking: To utilize line features in SLAM, it is necessary to extract the lines and match them with lines extracted from the previous frame. Conventional SLAM algorithms, adopting line features, employ a Line Segment Detector (LSD) [34], [35] for their line extraction process. Line Band Descriptor (LBD) [36] was applied to match the extracted line features with the previous ones. In the previous studies [15], [17], [20], LSD and LBD were used to filter out incorrectly matched lines through information such as orientation, length, and endpoint positions. However, there are limitations to this method. First, a single line can be split into multiple lines as the frame progresses. In addition, lines at the boundary of the image plane can be cut off. And those lines are considered as different lines and it causes tracking performance degradation.

In this study, we propose a method based on optical-flow [37] to robustly track lines for a long time. The overall procedure can be seen in Fig. 5. When the current frame at t is given, 1) first, line features are extracted using LSD. In this process, the short line features are removed for robustness. 2) Extracted line features are matched with the previous line features at the $(t-1)$-th frame. But, as seen in the preliminary matching results, divided, or cut off lines are not matched, even if they should be considered as the same lines. To solve this problem, we add one more step: 3) optical-flow-based line prediction. In this step, the new line features at t are predicted by transferring the lines' endpoints observed at $t-1$ through optical-flow. And it is checked whether the predicted line features from $t-1$ are misidentified lines through matching with the line features at t. Then, these predicted line features are used to supplement the preliminary matching results. Among the line features in the preliminary matching results, there can exist the lines that are not matched but should have been matched. In case of matched lines, the matching scores between the extracted lines and their corresponding lines from optical-flow are compared, and the line features which have higher score are selected for final line features at t. When the lines are not matched, extracted line features are replaced with the corresponding line features from optical-flow. These processes correspond to the 4) merging step. Therefore, more robust line features can be utilized, and the line features can be continuously tracked more robustly.

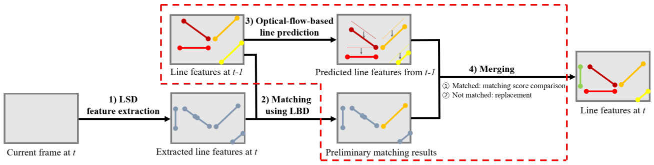

3) Optical-flow-based line prediction
Line features at t-1
Predicted line features from t-1
4) Merging
① Matched: matching score comparison
② Not matched: replacement
Line features at t
1) LSD feature extraction
2) Matching using LBD
Current frame at t
Extracted line features at t
Preliminary matching results

Fig. 5. Flowchart of line tracking. Between adjacent frames, the same lines are represented with the same color. By using LSD and LBD, lines are extracted and matched, respectively. In addition, the predicted lines are made by optical-flow. By comparing the predicted lines and the lines created through matching, it is possible to remove ther cut or duplicate lines.

V. Experimental Results

The experiments were conducted with an Intel Core i7-9700K CPU with 32GB of memory. We evaluated the proposed algorithm using the EuRoC Micro Aerial Vehicle (MAV) dataset [28]. Each dataset has its own level of difficulty depending on factors such as illumination, texture, and MAV speed. Therefore, it was suitable to test the effectiveness of the proposed method. TABLE I summarizes the parameters used in the experiments.

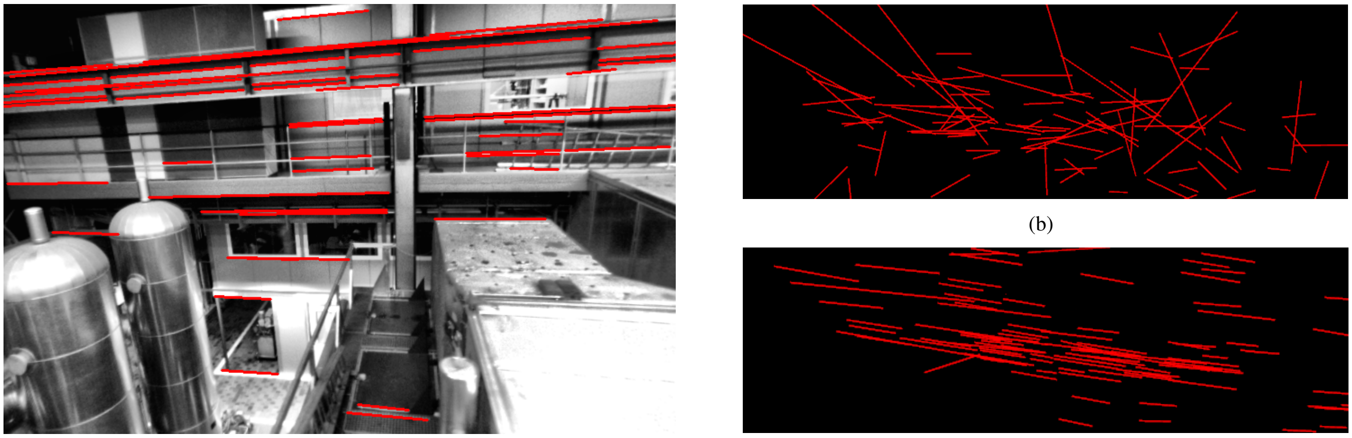
TABLE I
Parameter Settings in Experiments

Parameter	Value
Sliding window size	10
Maximum number of point features	150
Minimum pixel length of line feature	50
Minimum number of tracked frames of line feature	5
Maximum solver time for optimization (s)	0.1
Maximum iterations for optimization	10

To evaluate the performance of line tracking, we exploited only the lines tracked for a long time to reduce the computational cost in the optimization process. Among all lines, the ratio of the number of lines observed in sliding windows of a certain number of frames or more was calculated. In this experiment, only the lines observed for at least five sliding windows were used. As shown in TABLE II, the optical-flow-based line tracking has approximately 49% higher tracking performance than the conventional method with LSD and LBD while the computation time was increased by about 20%.

The accuracy of the proposed algorithm is compared with the accuracy of VINS-Mono without loop-closing. In addition, to prove the efficacy of degeneracy avoidance, we compare the proposed algorithm with PL-VINS [21] which is a state-of-the-art algorithm that uses line features and the code has been released.

For accurate algorithm evaluation, we used the rpg trajectory evaluation tool suggested by Zhang *et al.* [38] TABLE III shows the translational root mean square error



(b)

(a)

(c)

Fig. 6. Line mapping result of VINS with a structural constraint. (a) Image with degenerate lines represented in red for *machine hall* in EuRoC dataset. Parallel lines on the wall cause degeneracy during triangulation. (b) Top view of line mapping result without structural constraint. (c) Top view of line mapping result with structural constraint.

TABLE II

TRACKING PERFORMANCE OF LINE FEATURES WITH THE EUROC DATASET. PERCENTAGE (%) OF THE NUMBER OF LINES TRACKED IN 5 OR MORE SLIDING WINDOWS.

	LSD+LBD (%)	Proposed (%)
MH_01_easy	23.48	**49.76**
MH_02_easy	28.35	**51.30**
MH_03_medium	25.24	**43.35**
MH_04_difficult	30.58	**51.92**
MH_05_difficult	31.46	**53.82**
V1_01_easy	34.51	**45.21**
V1_02_medium	24.95	**33.43**
V1_03_difficult	19.96	**27.32**
V2_01_easy	37.56	**47.59**
V2_02_medium	24.15	**28.69**
V2_03_difficult	19.16	**22.60**

TABLE III

TRANSLATIONAL RMSE AND MAXIMUM ERROR FOR THE EUROC DATASET (UNIT: M)

	VINS-Mono		PL-VINS		Proposed	
Translation	RMSE	Max.	RMSE	Max.	RMSE	Max.
MH_01_easy	0.164	0.383	0.283	0.582	**0.142**	**0.270**
MH_02_easy	0.140	0.381	0.195	0.546	**0.126**	**0.346**
MH_03_medium	0.225	0.575	**0.187**	**0.377**	0.198	0.453
MH_04_difficult	0.408	0.610	0.335	0.588	**0.301**	**0.448**
MH_05_difficult	0.312	0.512	0.347	0.560	**0.293**	**0.481**
V1_01_easy	0.094	0.209	**0.071**	**0.150**	0.087	0.200
V1_02_medium	0.115	0.416	0.086	0.232	**0.072**	**0.137**
V1_03_difficult	0.203	0.401	**0.152**	**0.348**	0.156	0.375
V2_01_easy	0.099	**0.236**	**0.095**	0.251	0.098	0.245
V2_02_medium	0.161	0.569	0.120	0.287	**0.103**	**0.211**
V2_03_difficult	0.341	0.748	0.278	**0.505**	**0.277**	0.571

(RMSE) and maximum error for the EuRoC dataset. PL-VINS and the proposed algorithm have better performance than VINS-Mono overall. In particular, PL-VINS shows about 6.3% smaller error than the proposed algorithm in four datasets out of 11 datasets. On the contrary, the proposed algorithm shows 25% smaller error than PL-VINS in the remaining 7 datasets. In the proposed algorithm, more accurate results can be obtained because degenerate lines can be corrected through structural constraints while PL-VINS does nothing with degenerate lines. The state estimation rate of the proposed algorithm is about 20Hz, which is somewhat slower than that of VINS-Mono running at about 28Hz, but it is not a problem since the IMU is updated very fast.

As shown in Fig. 6(a), for the *machine hall* in the EuRoC dataset, degeneracy occurred in all lines facing the x-axis of the camera motion. Figs. 6(b) and (c) show the line mapping results of the proposed algorithm with and without structural constraint, respectively. Through structural constraints, the degenerate lines are optimized as parallel lines.

VI. CONCLUSION

In summary, we developed a novel SLAM that utilizes line features efficiently. For robust tracking, the optical-flow-based line tracking method was proposed in line matching. Finally, we analyzed and resolved the degeneracy problem that must be considered when employing line features in SLAM. Experiments were performed with the EuRoC MAV dataset, from easy to difficult situations, to prove the validity of the proposed algorithm. The proposed method has better localization accuracy than VINS-Mono and PL-VINS. As a result, line tracking performance was increased by about 20%, and translation RMSE was decreased by about 15% compared to the base algorithm. It was also possible to obtain a more accurate and visually discernable line map by solving the degeneracy problem. In future work, we will apply an adaptive algorithm that finds the proper optimization factors between point and line features in SLAM.

REFERENCES

[1] D. Scaramuzza and Z. Zhang, “Visual-inertial odometry of aerial robots,” *arXiv preprint arXiv:1906.03289*, 2019.

[2] G. Huang, “Visual-inertial navigation: A concise review,” in *Proc. International Conference on Robotics and Automation (ICRA)*, 2019, pp. 9572–9582.

[3] D. G. Kottas and S. I. Roumeliotis, “Efficient and consistent vision-aided inertial navigation using line observations,” in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 2013, pp. 1540–1547.

[4] X. Kong, W. Wu, L. Zhang, and Y. Wang, “Tightly-coupled stereo visual-inertial navigation using point and line features,” *Sensors*, vol. 15, no. 6, pp. 12 816–12 833, 2015.

[5] G. Klein and D. Murray, “Parallel tracking and mapping on a camera phone,” in *2009 8th IEEE International Symposium on Mixed and Augmented Reality*, 2009, pp. 83–86.

[6] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, “ORB-SLAM: A versatile and accurate monocular SLAM system,” *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[7] E. S. Jones and S. Soatto, “Visual-inertial navigation, mapping and localization: A scalable real-time causal approach,” *The International Journal of Robotics Research*, vol. 30, no. 4, pp. 407–430, 2011.

[8] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, “Robust visual inertial odometry using a direct EKF-based approach,” in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 298–304.

[9] A. I. Mourikis and S. I. Roumeliotis, “A multi-state constraint Kalman filter for vision-aided inertial navigation,” in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 2007, pp. 3565–3572.

[10] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, “Keyframe-based visual–inertial odometry using nonlinear optimization,” *The International Journal of Robotics Research*, vol. 34, no. 3, pp. 314–334, 2015.

[11] T. Qin, P. Li, and S. Shen, “VINS-mono: A robust and versatile monocular visual-inertial state estimator,” *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[12] A. Bartoli and P. Sturm, “Structure-from-motion using lines: Representation, triangulation, and bundle adjustment,” *Computer Vision and Image Understanding*, vol. 100, no. 3, pp. 416–441, 2005.

[13] G. Zhang, J. H. Lee, J. Lim, and I. H. Suh, “Building a 3-D line-based map using stereo SLAM,” *IEEE Transactions on Robotics*, vol. 31, no. 6, pp. 1364–1377, 2015.

[14] F. Zheng, G. Tsai, Z. Zhang, S. Liu, C.-C. Chu, and H. Hu, “Trifo-VIO: Robust and efficient stereo visual inertial odometry using points and lines,” in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 3686–3693.

[15] Y. Yang, P. Geneva, K. Eckenhoff, and G. Huang, “Visual-inertial odometry with point and line features,” in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 2447–2454.

[16] 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 International Conference on Robotics and Automation (ICRA)*, 2017, pp. 4503–4508.

[17] R. Gomez-Ojeda, F.-A. Moreno, D. Zuñiga-Noël, D. Scaramuzza, and J. Gonzalez-Jimenez, “PL-SLAM: A stereo SLAM system through the combination of points and line segments,” *IEEE Transactions on Robotics*, vol. 35, no. 3, pp. 734–746, 2019.

[18] X. Zuo, X. Xie, Y. Liu, and G. Huang, “Robust visual SLAM with point and line features,” in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 1775–1782.

[19] S. J. Lee and S. S. Hwang, “Elaborate monocular point and line SLAM with robust initialization,” in *Proc. of the IEEE International Conference on Computer Vision*, 2019, pp. 1121–1129.

[20] 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.

[21] Q. Fu, J. Wang, H. Yu, I. Ali, F. Guo, and H. Zhang, “PL-VINS: Real-Time Monocular Visual-Inertial SLAM with Point and Line,” *arXiv preprint arXiv:2009.07462*, 2020.

[22] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[23] T. Sugiura, A. Torii, and M. Okutomi, “3d surface reconstruction from point-and-line cloud,” in *International Conference on 3D Vision*, 2015, pp. 264–272.

[24] H. Zhou, D. Zhou, K. Peng, W. Fan, and Y. Liu, “SLAM-based 3D line reconstruction,” in *Proc. 13th World Congress on Intelligent Control and Automation (WCICA)*, 2018, pp. 1148–1153.

[25] A. Ö. Ok, J. D. Wegner, C. Heipke, F. Rottensteiner, U. Sörgel, and V. Toprak, “Accurate reconstruction of near-epipolar line segments from stereo aerial images,” *Photogrammetrie-Fernerkundung-Geoinformation*, no. 4, pp. 345–358, 2012.

[26] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, “On-manifold preintegration for real-time visual–inertial odometry,” *IEEE Transactions on Robotics*, vol. 33, no. 1, pp. 1–21, 2016.

[27] G. Sibley, L. Matthies, and G. Sukhatme, “Sliding window filter with application to planetary landing,” *Journal of Field Robotics*, vol. 27, no. 5, pp. 587–608, 2010.

[28] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, “The EuRoC micro aerial vehicle datasets,” *The International Journal of Robotics Research*, vol. 35, no. 10, pp. 1157–1163, 2016.

[29] H. Zhou, D. Zou, L. Pei, R. Ying, P. Liu, and W. Yu, “StructSLAM: Visual SLAM with building structure lines,” *IEEE Transactions on Vehicular Technology*, vol. 64, no. 4, pp. 1364–1375, 2015.

[30] D. Zou, Y. Wu, L. Pei, H. Ling, and W. Yu, “StructVIO: visual-inertial odometry with structural regularity of man-made environments,” *IEEE Transactions on Robotics*, vol. 35, no. 4, pp. 999–1013, 2019.

[31] R. Toldo and A. Fusiello, “Robust multiple structures estimation with j-linkage,” in *Proc. European Conference on Computer Vision (ECCV)*. Springer, 2008, pp. 537–547.

[32] J.-P. Tardif, “Non-iterative approach for fast and accurate vanishing point detection,” in *Proc. IEEE 12th International Conference on Computer Vision*, 2009, pp. 1250–1257.

[33] S. Agarwal, K. Mierle, and Others, “Ceres solver,” Accessed on: Oct. 12, 2020. [Online], Available: http://ceres-solver.org.

[34] R. G. Von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall, “LSD: A fast line segment detector with a false detection control,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 4, pp. 722–732, 2008.

[35] ——, “LSD: A line segment detector,” *Image Processing On Line*, vol. 2, pp. 35–55, 2012.

[36] L. Zhang and R. Koch, “An efficient and robust line segment matching approach based on LBD descriptor and pairwise geometric consistency,” *Journal of Visual Communication and Image Representation*, vol. 24, no. 7, pp. 794–805, 2013.

[37] B. D. Lucas, T. Kanade *et al.*, “An iterative image registration technique with an application to stereo vision,” in *Proc. International Joint Conference on Artificial Intelligence*, 1981, pp. 674–679.

[38] Z. Zhang and D. Scaramuzza, “A tutorial on quantitative trajectory evaluation for visual (-inertial) odometry,” in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 7244–7251.