

Received December 6, 2020, accepted December 31, 2020, date of publication January 8, 2021, date of current version January 19, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3049801

PL-GM:RGB-D SLAM With a Novel 2D and 3D Geometric Constraint Model of Point and Line Features

CHENYANG ZHANG 

School of Earth Sciences and Engineering, Hohai University, Nanjing 211100, China

e-mail: zcynj@hhu.edu.cn

This work was supported in part by the Postgraduate Research and Practice Innovation Program of Jiangsu Province under Grant KYCX20_0485, and in part by the Fundamental Research Funds for the Central Universities under Grant B200203106.

ABSTRACT In the study of RGB-D SLAM (Simultaneous Localization and Mapping), two types of primary visual features, point and line features, have been widely utilized to calculate the camera pose. As an RGB-D camera can capture RGB and depth information simultaneously, most RGB-D SLAM methods only utilize the 2D information within the point and line features. To obtain a higher accuracy camera pose and utilize the 2D and 3D information within points and lines better, a novel geometric constraint model of points and lines (PL-GM) using an RGB-D camera is proposed in this paper. Our contributions are threefold. Firstly, the 3D points and lines generated by an RGB-D camera combining with 2D point and line features are utilized to establish the PL-GM, which is different from most models of point-line SLAM (PL-SLAM). Secondly, in addition to the 2D re-projection error of point and line features, the constraint errors of 3D points and lines are constructed and minimized likewise, and then a unified optimization model based on PL-GM is extended to the bundle adjustment model (BA). Finally, extensive experiments have been performed on two public benchmark RGB-D datasets and a real scenario sequence. These experimental results demonstrate that our method achieves a comparable or better performance than the state-of-the-art SLAM methods based on point and line features, and point features.


INDEX TERMS Visual SLAM, RGB-D, PL-GM, bundle adjustment, point features, line features.

I. INTRODUCTION

Simultaneous Localization and Mapping (SLAM) [1] was first proposed by researchers in 1986 and used to estimate the pose of the mobile robot and construct an incremental map in unknown scenes simultaneously [2], [3]. SLAM is the key technology for robots navigating in unknown environments, and it has become a hotspot of the robotic research field [4]. Currently, various sensors (e.g., radar, ultrasonic, laser) have been applied widely in SLAM to help the robots to perceive the scene information [5], [6]. In comparison to the sensors, the vision camera with advantages of smaller size, less power consumption, and acquiring the texture information, can provide abundant information for robots, especially in these GNSS-denied (Global Navigation Satellite System) environments such as the lunar, Martian surface, and the underground. The SLAM

system which uses the visual camera as data input is called the visual SLAM. Over the past few decades, coincided with the development of computer science technology, visual SLAM has attracted extensive focuses in the current SLAM research community [7], [8] and has been applied to planetary exploration missions successfully, for instance, NASA's Mars Exploration Rover 2003 (MER) mission [9], China's Chan' E-3 mission [10], and Chang' E-4 mission [11].

RGB-D camera is a new type of vision sensor with the ability to provide RGB and depth images simultaneously. Kinect-1, the first RGB-D camera, was released by Microsoft in 2009 [12] and was applied in kinds of fields widely. The advantages of the RGB-D camera make it favorable to become another feasible alternative to the monocular and stereo cameras in the visual SLAM fields [13]. In general, most visual SLAM systems estimate the camera pose by matching point features of frames. As RGB-D camera can capture the point features and provide RGB and depth images,

The associate editor coordinating the review of this manuscript and approving it for publication was Yongqiang Zhao .

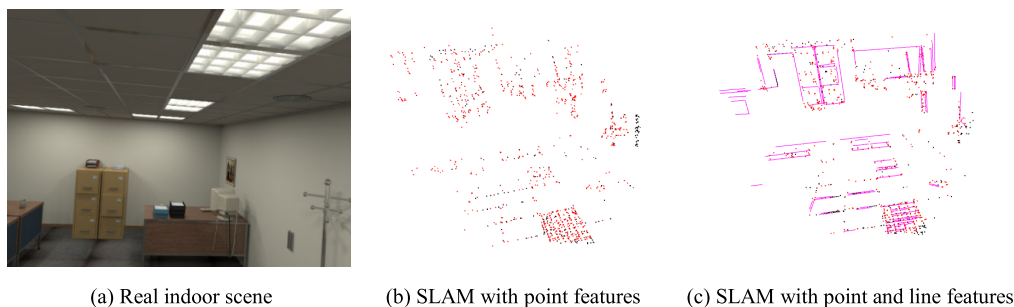


FIGURE 1. The indoor scene visual SLAM is base on point and line features.

many RGB-D SLAM methods based on point features have been released successively. In comparison with point features, as shown in Fig.1, line features are more robust in the low-texture scenes and can represent the structure information of scenes intuitively [14], while the sparse points cannot. Therefore, some related researches on RGB-D SLAM with point and line features have attracted extensive attention and methods have been proposed as well.

A. RELATED WORKS

In general, visual SLAM is carried out based on the visual camera such as the monocular, stereo camera, or RGB-D camera. For the monocular camera, it is necessary to apply triangulation to recover three-dimension information of scenes, which increases the computational complexity. The stereo camera acquires the 3D scene through stereo matching of frames, and it may not work well in some low-texture scenes. Since RGB-D camera can provide RGB and depth images, the 3D information has been acquired easily and some relevant RGB-D SLAM systems have been raised too. In this paper, we divide these RGB-D SLAM methods into two classes: point-based methods, point-line methods.

Point-based methods use point features to construct and solve the optimization function for pose estimation. Visual SLAM estimates camera pose by matching the point features of frames normally. These algorithms for detecting points including the well-known SIFT [20], SURF [21], and ORB [15] have been applied in the visual SLAM widely, which improved the accuracy and robustness of the pose estimation [22], [23]. The process of solving camera pose is implemented by minimizing the 2D re-projection error of point features. The calculating procedure for minimizing the re-projection error is actually an optimization process, which is carried out using the general graph optimization algorithm of g2o [24]. Henry *et al.* [25] firstly proposed an RGB-D SLAM algorithm based on the points, in which the RGB-D camera was utilized as the sensor input to perform robot navigation and mapping. Huang *et al.* [26] also developed an RGB-D SLAM in which the sparse bundle adjustment (SBA) was applied for global consistency by optimizing the 2D re-projection errors of point features of adjacent frames. A similar constraint model could be found and adopted in [27], [28]. The depth measurement captured

by the RGB-D camera is a new type of measurement data, while the depth information can not be employed fully. Subsequently, Kerl *et al.* [29] proposed a dense RGB-D SLAM approach though minimizing the photometric and depth error of point features between frames. Di *et al.* [30] considered the depth information as a type of observation and constructed the optimization model based on point features by minimizing the re-projection error of the point features and per-pixel depth. Kerl *et al.* [29] and Di *et al.* [30] extended their adjustment model to the bundle adjustment (BA) model to improve the positioning accuracy and performance of RGB-D SLAM somewhat. Compared to the RGB-D SLAM methods above, the ORB-SLAM2 [31], an opening source visual SLAM project, is an outstanding representative of the visual SLAM method which can support a monocular, stereo, or RGB-D camera. However, the mentioned methods are suitable to calculate a camera pose for the scene containing dense point features. If the amount of extraction points in the low-texture environments is smaller, it will lead to the computed camera pose unreliable.

Expect for these visual SLAM methods based on point feature, the line feature, another type of primary visual feature, has been applied in visual SLAM either. Point-line features based SLAM methods applied the constraint of point and line features to construct and optimize the cost function for camera pose estimation. Line feature detection algorithms mainly include LSD [16], Hough Transform [32], and EDLines [33]. It is noticeable that the accuracy and robustness of visual SLAM based on point features often degenerate in the low-texture scenes. Generally, the number of line features is smaller than that of point features, and the pose estimation solved by the line features only is less reliable than point features [34]. Accordingly, to obtain robust and comparable performance of SLAM, researchers proposed some SLAM methods based on point and line features. PL-SLAM, the first opening source visual SLAM project using point and line features [34], [35] (<https://github.com/rubegoj/pl-slam>), achieves robust performance applying a stereo camera. Expect for the SLAM systems in [34], [35], there is another visual SLAM based on point and line features with a monocular camera [14], whose pose calculation model is similar to that of PL-SLAM. To improve the robustness and accuracy of SLAM significantly, Wang *et al.* [19]

improved the optimization model by adding the angle constraint between the re-projected and extracted line features. His work was contributed to improving the location accuracy of PL-SLAM, but he failed to extend it to the BA model. Zhao *et al.* [36] came up with a method named “Good Line Cutting” on basis of the source project of PL-SLAM [35] and could obtain a significant improvement than the primal PL-SLAM. Zhang *et al.* [37] raised to construct a 3D line-based SLAM in his work in which he utilized two different representations for the lines: Pluecker coordinate was utilized for 3D line initialization and representation. Unfortunately, neither the open-source code is available nor the employed datasets contain any ground-truth. Subsequently, Pluecker coordinates were utilized to denote the 3D lines and applied in some point-line SLAM systems [38]. Moreover, the inertial measurement unit (IMU), which can capture a motion platform pose in real-time, was also combined with point and line features to estimate camera pose [39]–[41]. The above-mentioned visual SLAM methods are based on a monocular or stereo camera. Besides, another SLAM systems based on point and line features using the RGB-D camera were proposed too. After analyzing the uncertainties of depth information, Yu and Song [42] applied the RANSAC algorithm [43] to filter out wrong line features and achieved robust RGB-D odometry based on point and line features, Fu *et al.* [44] proposed an RGB-D SLAM system with point and line features for the low scenes in which a line-based refinement algorithm was utilized to achieve the robustness performances. Moreover, to improve the location accuracy and robustness of RGB-D SLAM with point-line features, a new method aiming at detecting and matching line features in some weak-matching scenes is released [45]. To achieve the optimal pose solutions, a two-step optimization algorithm of RGB-D SLAM [46] based on point and line features was raised too. It is mentioned that the visual SLAM with point features is easy to fail in low textured scenes. As it is quite convenient to use Pluecker coordinates to represent lines, a robust RGB-D SLAM using point-line features for low scenes was proposed [47] well. Although RGB-D camera can obtain depth information, these RGB-D SLAM systems above only utilized 2D points and lines (i.e., the 2D re-projection error of point and line features), and the 3D points and lines acquired from depth image are not used for estimation pose. Besides, the depth information either is directly used or not taken into full consideration in the SLAM systems [44], [47].

B. MOTIVATION

At present, many point-line SLAM methods are proposed by using monocular, stereo, or RGB-D camera. However, for a monocular camera, it is inevitable to apply triangulation to obtain the 3D information, which increases the computational complexity. The stereo camera cannot recover 3D scene information well in some low texture scenes. The RGB-D camera can obtain the depth information and is utilized generally in the visual SLAM community. Currently, most

RGB-D SLAM methods do not regard depth information of frames as one kind of measurement constraint. Furthermore, the 3D information constraint with points and lines are not utilized fully in these released SLAM using point and line features.

In this paper, we apply an RGB-D camera and establish a new geometric constraint model based on point and line features to obtain the camera pose with higher accuracy and stronger robustness.

C. CONTRIBUTION AND OUTLINE

We proposed a new geometric constraint model of RGB-D SLAM using point and line features. We detected and matched point and line features of frames by applying ORB [15] and LSD (Line segment Detector) algorithm [16]. With the aid of point-line features and depth images, the 3D points and lines were recovered and then refined according to the knowledge of space geometry property within points and lines. Furthermore, we combined these point and line features with 3D points and lines to construct the geometric constraint model. In our constraint model, apart from the 2D re-projection errors related to point and line features, the constraint errors of 3D points and lines are appended and constituted the overall optimization model with points and lines. Finally, our geometric constraints are extended to the BA model and verified on two public TUM [17], ICL-NUIM [18] RGB-D datasets, and a real scene sequence, respectively. Comprehensive experiments demonstrate that our method performs comparably or better than those state-of-the-art visual SLAM using points merely or point and line features together. The main contributions of our work are summarized as follows:

- 1) We proposed a new RGB-D SLAM based on 2D and 3D points and lines, which is different from the RGB-D SLAM based on point and line features.
- 2) We established the new 2D and 3D geometric constraint model with point and line features and extended the geometric constraint model to the BA model.
- 3) We tested our RGB-D SLAM method on two public RGB-D datasets and a real scene, and our RGB-D SLAM achieved accurate or comparable performances than the classical SLAM systems.

The remainder of this paper is organized as follows. First, our geometric constraint model of points and lines is presented in Section II. Next, we verify our SLAM and show the experimental results in Section III. The discussion is given in Section IV. Finally, conclusions and future works are given in Section V.

II. PL-GM: RGB-D SLAM

A. SYSTEM OVERVIEW

In this paper, we pay close attention to a higher accurate and robust camera pose estimation in the indoor scenes. Our proposed method differs from the visual SLAM methods based on the 2D re-projection error of point and line features. The

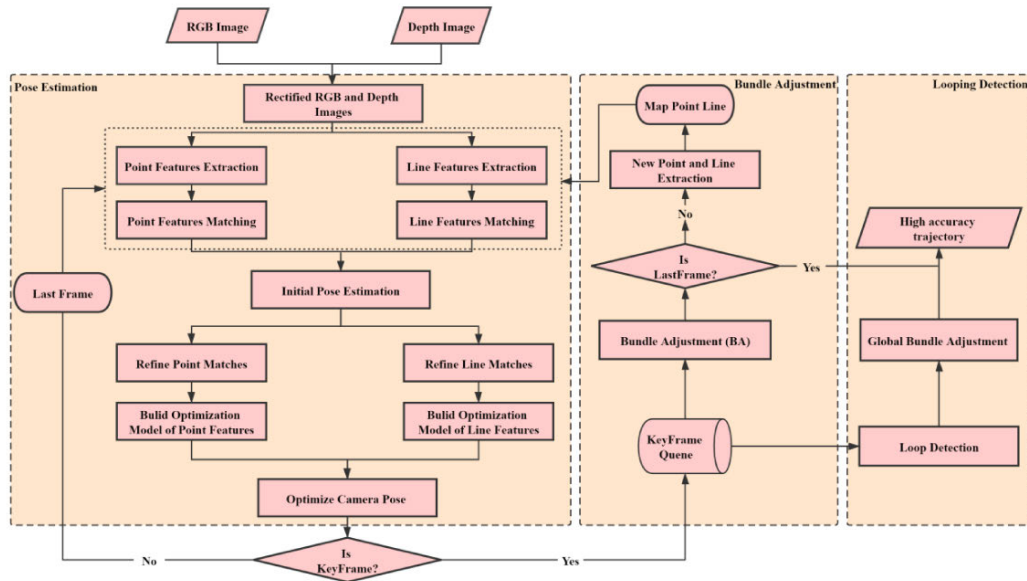


FIGURE 2. An overview of our point-line-based RGB-D SLAM system.

input of our RGB-D SLAM are RGB and depth images, and the output is the camera pose. The camera pose is estimated by means of the 2D and 3D points-lines. We exploit these point and line constraints to extend the BA model. The loop closure is constantly checking for loops and adjusting the camera pose with the BA strategy. The loop closure module of our SLAM system is similar to that of visual SLAM systems. The overall structure of RGB-D SLAM is shown in Fig.2. The pipeline is built by referring to other state-of-the-art visual SLAM (e.g., PTAM [22], ORB-SLAM2 [31]), and it is integrated with three main parts listed as follow:

- 1) Pose Estimation: in the module, firstly, point and line features are detected and matched, and the initial pose is calculated by applying space resection. Then, these matched points and lines are refined. The geometric constraint model is established and utilized to optimize the camera pose initially.
- 2) Bundle Adjustment: in the module, the built geometric constraint model is extended to the BA module. After optimizing the pose of each frame, the frame is determined whether it is a keyframe not. If so, the frame will be inserted into the keyframe sequences. Subsequently, the extended BA model is employed to accomplish the optimization and reduce the accumulated pose errors of frames.
- 3) Looping Detection: Our system checks the correct looping closure to reduce the drifting and accumulated errors globally. If the current frame is the last one, the high-accuracy pose estimation of all frames will be output.

B. EXTRACTION AND MATCHING OF POINT AND LINE FEATURES

When considering some factors within feature extraction algorithms (e.g., SIFT, SURF) such as velocity, stability,

rotation invariance and so on [48], the ORB algorithm [15] is selected to extract and match point features by using the OpenCV library. By combining the RANSAC [43] algorithm with a fundamental matrix constraint, we reject the mismatches and obtain an initial set of matching points. The pixel distortion of the point feature is corrected by the distortion correction model [49]. The depth values of the extraction point features are acquired from the depth image. The 3D coordinates of extraction point features in the camera coordinate system are computed by using (1). d denotes the depth measurement, and (u, v) is the pixel coordinate, (f_x, f_y) is the intrinsic parameter of RGB camera, (X, Y, Z) denotes a 3D coordinate point in the camera coordinate system. To refine and obtain more accurate matching points, the pose of adjacent frames is calculated using the space resection, and the 3D matching points from adjacent frames are converted to the unified coordinate frame. The Euclidean distance of 3D matching points is calculated, and these 3D points are sorted in ascending order according to the Euclidean distances. Finally, 25% of the total matching 3D points will be eliminated, and the remained 3D and 2D points are used to calculate the camera pose. In the process of line features extraction, the LSD [16] is used to extract line features, which can provide high precision and repeatability. The LBD, a line band descriptor [50] is often utilized to match the line features in the consecutive frames. For line features matching, the FLANN algorithm is utilized to match the line features, and then the candidate line features are checked whether they are the right matching pairs or not. Then the geometry information within the matching line features is utilized fully to filter out those matching pairs with different orientations and lengths [35], and the matching result is shown in Fig.3. The approach of rectifying the pixel distortions of line features is similar to that of rectifying the point features. Using the depth image likewise, the depth measurement of each

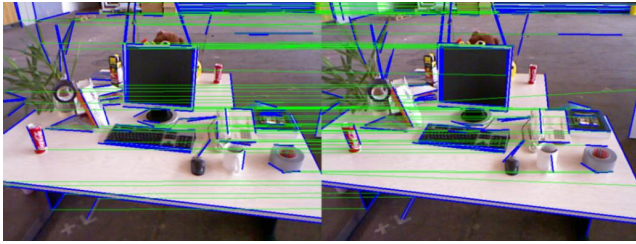


FIGURE 3. Extraction (blue line) and matching line features (green line) using LSD algorithm.

endpoint of line feature is acquired and the 3D coordinates of line endpoints are computed by using (1) as well.

$$\begin{cases} X = \frac{u - c_x}{f_x} d \\ Y = \frac{v - c_y}{f_y} d \\ Z = d \end{cases} \quad (1)$$

As shown in Fig.4, as the line features exist at the edges of objects generally, the depth value of line endpoints may not be the real depth probably. If this case is not considered, the pose solution with points and lines will be inaccurate. As a result, to ensure the accuracy of the pose solution, the 3D lines which do not conform to space geometry property need to be filtered out. In this paper, on the basis of the space geometry knowledge, 3D lines should conform that the 3D coordinates of its midpoint (X_m, Y_m, Z_m) are equal to half of the sum of coordinates of the 3D line endpoints. In effect, each depth measurement has errors, a threshold ratio is preset to reject the inconsistent space lines using (2) where (X_s, Y_s, Z_s) and (X_e, Y_e, Z_e) denote the 3D coordinates of line endpoints. According to our actual experimental results, the threshold is set as 0.99.

$$ratio = \frac{\text{Min}_{(Z_m, Z_s+Z_e)}(2 \times Z_m, Z_s + Z_e)}{\text{Max}_{(Z_m, Z_s+Z_e)}(2 \times Z_m, Z_s + Z_e)} \quad (2)$$

If the *ratio* is less than the threshold, the 3D lines will be rejected. On the contrary, those will be reserved and the pruning results are shown in Fig.4. The related pseudo-code of refining 3D lines is summarized as follows:

Algorithm 1 Pseudo Code: 3D Line Pruning

Input: extraction line features set (L), depth image (D), single line feature (l)

Output: 3D line with consistent coordinates (L')

```

for  $l_i \in L$ 
  Compute  $(X_s, Y_s, Z_s)$ ,  $(X_e, Y_e, Z_e)$ , and  $(X_m, Y_m, Z_m)$ 
  using  $D$  and Equation (2);
   $ratio = \text{Min}(2 \times Z_m, Z_s + Z_e) / \text{Max}(2 \times Z_m, Z_s + Z_e)$ ;
  if ( $ratio >$  threshold)
    Accept;
  else
    Remove;

```

Return L'

C. GEOMETRIC CONSTRAINT OPTIMIZATION MODELS OF 2D AND 3D POIN-LINE

In this section, our geometric constraint model based on 2D and 3D points is introduced firstly. And then, the details of our geometric constraint optimization model of 2D and 3D lines will be explained.

1) GEOMETRIC CONSTRAINT MODEL OF POINT FEATURES

For point features, the space resection is used to calculate the initial camera pose between frames. Each depth information can be obtained from the depth images, and it is considered as an observation in this paper. Different from most RGB-D SLAM methods based on points, in addition to the 2D re-projection error of point features, the constraint term of per-depth is also added in our geometric constraint model. For per-depth measurement, the constraint condition is considered that the depth of matching points should be equal to each other between frames in a unified coordinate frame. Based on the premise, the geometric constraint model of 2D and 3D points can be established as follows: First, the 2D and 3D matching point pairs between frames can be obtained. Then, in the light of the 2D matched point features, error functions based on the 2D re-projection error of point features are constructed. As the per-depth measurement of extraction points features is equal to that of the re-projected 3D matched points in the prior frame, the constraint of 3D points is established. Using the 2D and 3D points, our new geometric constraint model of points is established and shown in Fig.5. In Fig.5, P denotes a matching point, *depth*, and *depth'* represent the depth information of point features. $O-X_w Y_w Z_w$ denotes the world coordinate system, and p and p' are pixel coordinates of extraction point features in the adjacent frames. $o-xyz$ and $o'-x'y'z'$ denote the reference frame and current frame coordinate systems.

In our geometric constraint model of points, the content related to the error equation of points can be described as follows:

$$\begin{cases} err_p = K \cdot (R \cdot P_{world} + t) - p \\ err_dep = dep - P_c(Z) \end{cases} \quad (3)$$

where R and t represent the rotation matrix and translation vector converting from the world coordinate P_{world} or reference coordinate system to the current frame coordinate P_c . To obtain the derivative of rotation matrix and translation, (3) is rewritten using the format of Lie algebra and expressed in the following (4):

$$\begin{cases} err_p = K \cdot T(\zeta) \cdot P_{world} - p \\ err_dep = dep - P_c(Z) \end{cases} \quad (4)$$

where ζ is a six-dimensional vector of Lie algebras that represents a camera pose, and $T(\zeta)$ represents the transformation matrix from world coordinate system to current frame coordinate framework. K represents the intrinsic matrix of the RGB camera and p is the pixel coordinate of extraction point feature. *dep* is the corresponding depth information of

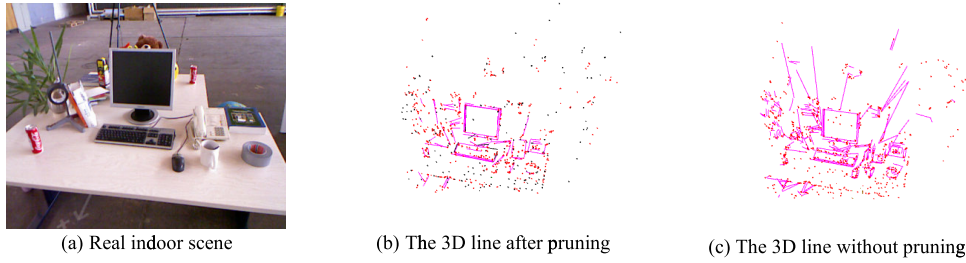


FIGURE 4. The 3D line pruning.

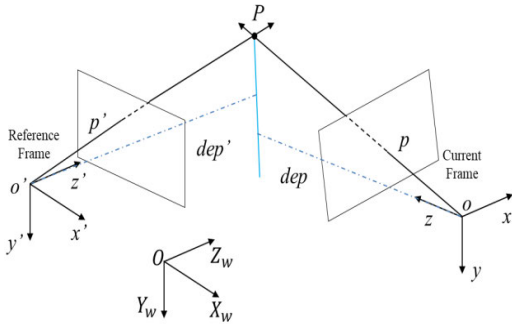


FIGURE 5. 2D and 3D geometric constraint model of point features.

the point feature. $P_c(Z)$ represents the Z component of the 3D point in the current frame coordinate framework. err_p is the re-projection error of 2D points features, and err_dep is the depth constraint error of points between the reference frame and current frame.

When applying the Levenberg-Marquardt (LM) algorithm to optimize the camera pose iteratively, our error equations need linearization. Based on the matrix derivation rule, the Jacobian matrices of (4) are expressed as follows:

$$\frac{\partial (err_p)}{\partial (\zeta)} = \begin{bmatrix} f_x \frac{1}{Z_c}, 0, -f_x \frac{X_c}{Z_c^2}, -f_x \frac{Y_c X_c}{Z_c^2}, f_x + f_x \frac{X_c^2}{Z_c^2}, -f_x \frac{Y_c}{Z_c} \\ 0, f_y \frac{1}{Z_c}, -f_y \frac{Y_c}{Z_c^2}, -(f_y + f_y \frac{Y_c^2}{Z_c^2}), f_y \frac{X_c Y_c}{Z_c^2}, f_y \frac{X_c}{Z_c} \end{bmatrix} \quad (5)$$

$$\frac{\partial (err_dep)}{\partial (\zeta)} = [-Y_c, X_c, 0, 0, 0, 1] \quad (6)$$

(X_c, Y_c, Z_c) represents the 3D point in the camera coordinate system. f_x and f_y are the camera calibration parameters. According to the iteration residuals of err_p and err_dep , the weights of point features and the corresponding per-depth value are defined using robust kernel functions, such as the Cauchy function. When finishing the above-mentioned steps, our geometric constraint with 2D and 3D points is established.

2) GEOMETRIC CONSTRAINT MODEL OF 2D AND 3D LINES

For the line features, our proposed geometric constraint model includes three constraints. The first one is the traditional 2D line re-projection error and is shown in Fig.6. For

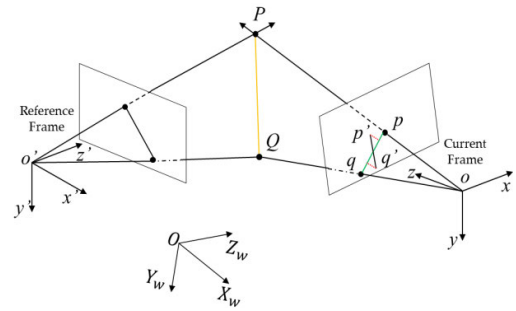


FIGURE 6. The illustration of re-projection errors of 2D line features.

the geometric constraint of 3D lines, the plane normal and the distance from the camera optical center to a 3D line are appended. Firstly, the cross product of the normal vector of a plane that contains the 3D line in consecutive frames should be close to a zero vector in a unified coordinate framework. Meanwhile, the distance from the camera optical center to a 3D line should be equal to each other in the adjacent frames in the unified coordinate framework, and the two constraint conditions are shown in Fig.7. Finally, the constraint conditions of 2D and 3D lines constitute our new geometric constraint model of lines. Next, the details of our proposed geometric constraint model of 2D and 3D lines are introduced systematically.

For the first constraint condition, we still adopt the minimization of the distances from the re-projected endpoints of the 3D line to the 2D line features. The geometric constraint model is shown in Fig. 6. Here, P and Q are endpoints of the 3D line, while p and q are pixel coordinates of line features. p' and q' are the re-projection of the matching line features. $o-xyz$ and $o'-x'y'z'$ denote the current frame and reference frame coordinate system, and $O-X_wY_wZ_w$ represents the world coordinate system. The red dotted line denotes the distance between the endpoints of projected lines and the original line features. The constraint model of line features is established as follows: Firstly, the 2D line equation in image space is computed by using pixel coordinates of line features. The line equation is solved using (7):

$$\lambda_{1 \times 3} = \frac{[p]^\wedge \cdot q}{|[p]^\wedge \cdot q|} \quad (7)$$

Here, $\lambda_i (i = 0, 1, 2)$ is the coefficients of the line equation, and λ is a unit vector. p and q are the homogeneous coordinates consist of line features endpoints, and $[\cdot]^\wedge$ is the

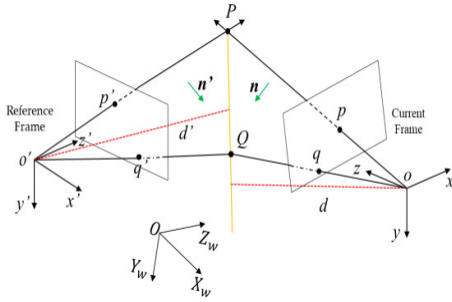


FIGURE 7. The illustration of 3D re-projection errors of 3D line.

skew-symmetric matrix of a three-dimensional vector. On the grounds of the computed line equation in the image plane coordinate system and the pixel coordinate of re-projection line features, the error equation is established using (8):

$$\begin{cases} err_s = \lambda_0 x_p + \lambda_1 y_p + \lambda_2 \\ err_e = \lambda_0 x_q + \lambda_1 y_q + \lambda_2 \end{cases} \quad (8)$$

Here, err_s and err_e denote 2D re-projection error of line feature. (x_p, y_p) and (x_q, y_q) are the pixel coordinate of re-projected line endpoints. Based on (8) and the rule of chain derivation, the Jacobian matrices are expressed as (9):

$$\begin{cases} \frac{\partial(err_s)}{\partial(\xi)} = \lambda_0 \frac{\partial(err_s)}{\partial(x_p)} \frac{\partial(x_p)}{\partial(\xi)} + \lambda_1 \frac{\partial(err_s)}{\partial(y_p)} \frac{\partial(y_p)}{\partial(\xi)} \\ \frac{\partial(err_e)}{\partial(\xi)} = \lambda_0 \frac{\partial(err_e)}{\partial(x_p)} \frac{\partial(x_p)}{\partial(\xi)} + \lambda_1 \frac{\partial(err_e)}{\partial(y_p)} \frac{\partial(y_p)}{\partial(\xi)} \end{cases} \quad (9)$$

By expanding and rearranging the formula (9), the concrete form of two Jacobian matrices are expressed in (10), as shown at the bottom of the next page:

where (f_x, f_y) is the camera parameters and $\lambda_i (i = 0, 1, 2)$ is coefficients of the line equation. (X_s, Y_s, Z_s) and (X_e, Y_e, Z_e) are the 3D coordinates of line endpoints in the camera coordinate system.

Next, another two constraint conditions are introduced and shown in Fig.7. In Fig.7, P and Q are endpoints of the 3D lines, while p and q are endpoints of the 2D line feature. p' and q' are the extraction line features of the prior frame. $o-xyz$ and $o'-x'y'z'$ are the camera coordinate systems of different frames, and $O-X_w Y_w Z_w$ is the world coordinate system. oPQ is the plane containing the camera optical center and 3D line (PQ) , and $o'PQ$ is another plane that contains the camera optical center and 3D line PQ . n' and n are normal vectors of two planes. d and d' denote the distance from the camera optical center to the 3D line PQ . As described in Fig.7, on the basis of space geometric knowledge, the cross product between the normal n of plane oPQ and n' of plane $o'PQ$ should be a zero vector in a unified coordinate reference framework. The two planes contain the camera optical center and the 3D line. According to the above-mentioned, our proposed geometric constraint model using the 3D line is constructed as follows: Firstly, the normal vector of plane oPQ is calculated

using (11) as follows:

$$\eta_{3 \times 1} = \frac{[X_s]^\wedge \cdot X_e}{|[X_s]^\wedge \cdot X_e|} \quad (11)$$

where the X_s and X_e are the coordinates of 3D line endpoints (PQ) in the current coordinate system, and η is a unit normal vector. $[\cdot]^\wedge$ is the skew-symmetric matrix of a three-dimensional vector. After acquiring the 3D coordinates of the matching line (PQ) in the prior coordinate frame, the geometric constraint related to the plane normal of 3D line is established. In other words, the plane containing the camera optical center (o') and re-projected 3D line (PQ) and another plane containing the camera optical center (o) and the 3D line (PQ) should conform to the geometric coplanar condition under the unified coordinate system. The relevant error function is expressed using (12). P_s and P_e are 3D endpoint coordinates of a 3D line in the reference frame coordinate system. X_p and X_q represent the 3D coordinate of line endpoints transformed from reference coordinate system or world coordinate system to the current camera coordinate system. The err_n is a three-dimensional vector, and $[\cdot]^\wedge$ is the operator that converts a three-dimensional vector to a skew-symmetric matrix.

$$\begin{cases} X_p = T(\xi) \cdot P_s \\ X_q = T(\xi) \cdot P_e \\ err_n_{3 \times 1} = [\eta]^\wedge \cdot ([X_p]^\wedge \cdot X_q) \end{cases} \quad (12)$$

Given (12) and the chain derivation rule, the Jacobian matrix of the error term is presented in (13) as follows:

$$\begin{aligned} \frac{\partial(err_n)}{\partial(\xi)} &= \begin{bmatrix} \frac{\partial(err_n)_1}{\partial(\xi)} \\ \frac{\partial(err_n)_2}{\partial(\xi)} \\ \frac{\partial(err_n)_3}{\partial(\xi)} \end{bmatrix} \\ &= \begin{bmatrix} \frac{\partial(err_n)_1}{\partial(X_p)} \frac{\partial(X_p)}{\partial(\xi)} + \frac{\partial(err_n)_1}{\partial(X_q)} \frac{\partial(X_q)}{\partial(\xi)} \\ \frac{\partial(err_n)_2}{\partial(X_p)} \frac{\partial(X_p)}{\partial(\xi)} + \frac{\partial(err_n)_2}{\partial(X_q)} \frac{\partial(X_q)}{\partial(\xi)} \\ \frac{\partial(err_n)_3}{\partial(X_p)} \frac{\partial(X_p)}{\partial(\xi)} + \frac{\partial(err_n)_3}{\partial(X_q)} \frac{\partial(X_q)}{\partial(\xi)} \end{bmatrix}_{3 \times 6} \end{aligned} \quad (13)$$

where $\partial(err_n)_1 / \partial(\xi)$, $\partial(err_n)_2 / \partial(\xi)$, and $\partial(err_n)_3 / \partial(\xi)$ are the differential element of error vector err_n , respectively. X_p and X_q are 3D coordinates of the 3D line in the current camera coordinate system.

For the third constraint term of lines, in the light of space geometric knowledge again, the distance d between the camera optical center (o) and the 3D line (PQ) is equal to d' calculated from the camera center (o') to re-projected 3D line (PQ) . In the same way, the distance between the camera optical center and the corresponding 3D line is calculated first. As shown in Fig.7, on the grounds of the geometric properties of the cross product of two vectors, the distance d in Fig.7 from the camera center (o) to a 3D line (PQ) is calculated by using (14).

$$\begin{cases} dis_s = |-[X_s]^\wedge \cdot v| \\ dis_e = |-[X_e]^\wedge \cdot v| \end{cases} \quad (14)$$

dis_s and dis_e are calculated by utilizing two endpoints of a 3D line. \mathbf{v} represents the unit direction vector of a 3D line, and $[\cdot]^\wedge$ is also the skew-symmetric matrix of a 3D vector. If the value d in Fig.7 is acquired, the coordinates of the matching 3D lines in the prior frame and the direction \mathbf{v} are used to establish an error equation. The error equation related to distance constraint is established using (15).

$$\begin{cases} err_dis_s = | - [\mathbf{T}(\zeta) \cdot \mathbf{P}_s]^\wedge \cdot \mathbf{v} | - dis_s \\ err_dis_e = | - [\mathbf{T}(\zeta) \cdot \mathbf{P}_e]^\wedge \cdot \mathbf{v} | - dis_e \end{cases} \quad (15)$$

Given (15) and the matrix derivation rule, the corresponding Jacobin matrix can be presented in (16).

$$\begin{bmatrix} \frac{\partial(err_dis_s)}{\partial(\zeta)} \\ \frac{\partial(err_dis_e)}{\partial(\zeta)} \end{bmatrix} = \begin{bmatrix} \frac{\partial(err_dis_s)}{\partial(X_s)} \frac{\partial(X_s)}{\partial(\zeta)} \\ \frac{\partial(err_dis_e)}{\partial(X_e)} \frac{\partial(X_e)}{\partial(\zeta)} \end{bmatrix}_{2 \times 6} \quad (16)$$

X_s and X_e are the coordinates of line endpoints in the current camera coordinate system. After acquiring the Jacobian matrices of error terms, the error Equation (4), (8), (12), and (15) are utilized to establish our new optimization model of points and lines. Based on each iteration of residuals, the weight of each error term related to a line is defined using the robust kernel function such as the Cauchy function.

D. BUNDLE ADJUSTMENT WITH POINT AND LINE

To improve the accuracy and efficiency of SLAM, the keyframes need to be selected. If a new frame is considered as a keyframe, the frame will be inserted into the keyframes storage. In this manuscript, the strategy of determining a keyframe based on entropy [29], [44] is adopted in our SLAM system, which is different from the keyframe strategy existing in ORB-SLAM2. The strategy of determining the keyframe applies the uncertainty of relative pose estimation in the adjacent frames.

Once the frame is confirmed as a keyframe, the pose parameters of keyframes sequences that share the common points and lines in the map are optimized with a BA strategy. In this study, our proposal is extended to the classic BA model. Firstly, for point features, the 2D and 3D error equations of points presented in (4) are appended. Subsequently,

for the line features, the 2D and 3D error equation related to lines based on (8), (12), and (15) are added either. Then, a unified cost equation that contains the point-line is built as follows:

$$\zeta^* = \operatorname{argmin} \sum_i^n (e_{ij}^T \sum_p^{-1} e_{ij}) + \sum_i^m (l_{jk}^T \sum_l^{-1} l_{jk}) \quad (17)$$

Here, e_{ij} is the error term related to 2D and 3D points. l_{jk} is the error term related to 2D and 3D lines. n and m denote the number of point feature and line feature, respectively. Σ_p^{-1} and Σ_l^{-1} are the weight matrices associated with points and lines, and the weight matrices are defined by using each iteration residuals of error terms related to point and line features. Thus, our new geometric constraints are extended to the BA model. When a new frame is decided as a keyframe, the pose of keyframes that share common points and lines will be optimized and the relative pose between the current frame and previous frame is updated. If a frame is the last one of the image sequences, the optimized pose of all frames would be output. If not so, the system would return to continue tracking frames and optimize the camera pose. Meanwhile, when obtaining a new keyframe, the process of looping detection is done to reduce the accumulated errors and drifting. In this paper, point and line features are used for loop detection. Once the loop closure is detected, a global BA strategy is performed to update the pose of all frames.

III. EXPERIMENTAL RESULTS

In this section, to verify our RGB-D SLAM, we have performed a series of experiments on the public datasets with ground-truth. Then we also compared our method with the state-of-the-art SLAM methods, e.g., RGB-D SLAM [45], PTAM [22], LSD-SLAM [23], ORB-SLAM2 [31]. All the experiments are performed on a laptop computer with an Intel®Core™i5-8250U@CPU with 1.6 GHz. The software utilized in our SLAM method include the Linux operating system of Ubuntu 16.04, OpenCV 3.2.0, Eigen 3.0, g2o, and C++ language.

For the quantitative evaluation of visual SLAM, the easy-to-use evaluation tool of Python (<https://vision.in.tum.de/>)

$$\begin{cases} \frac{\partial(err_s)}{\partial(\zeta)} = \begin{bmatrix} -(f_y \times \lambda_1 + \frac{f_x \times \lambda_0 \times X_s \times Y_s + f_y \times \lambda_1 \times Y_s^2}{Z_s^2}), \\ (f_x \times \lambda_0 + \frac{f_x \times \lambda_0 \times X_s^2 + f_y \times \lambda_1 \times Y_s^2}{Z_s^2}), \\ \frac{\lambda_1 \times X_s \times f_y - \lambda_0 \times Y_s \times f_x}{Z_s}, \frac{\lambda_0 \times f_x}{Z_s}, \frac{\lambda_1 \times f_y}{Z_s}, \frac{f_x \times \lambda_0 \times X_s + f_y \times \lambda_1 \times Y_s}{Z_s^2} \end{bmatrix}_{1 \times 6} \\ \frac{\partial(err_e)}{\partial(\zeta)} = \begin{bmatrix} -(f_y \times \lambda_1 + \frac{f_x \times \lambda_0 \times X_e \times Y_e + f_y \times \lambda_1 \times Y_e^2}{Z_e^2}), \\ (f_x \times \lambda_0 + \frac{f_x \times \lambda_0 \times X_e^2 + f_y \times \lambda_1 \times Y_e^2}{Z_e^2}), \\ \frac{\lambda_1 \times X_e \times f_y - \lambda_0 \times Y_e \times f_x}{Z_e}, \frac{\lambda_0 \times f_x}{Z_e}, \frac{\lambda_1 \times f_y}{Z_e}, \frac{f_x \times \lambda_0 \times X_e + f_y \times \lambda_1 \times Y_e}{Z_e^2} \end{bmatrix}_{1 \times 6} \end{cases} \quad (10)$$

TABLE 1. Comparison of RMSE [cm] in the TUM RGB-D dataset.

Sequences	Ours	RGB-D SLAM	PL-SLAM	ORB-SLAM2	PL-F SLAM	PL- SLAM2	LSD-SLAM	PTAM
fr1_xyz	0.90	1.16	1.21	0.93	0.94	*	9.00	1.15
fr1_floor	1.80	6.22	7.59	2.25	1.58	3.2	38.07	×
fr1_360	9.7	*	*	24.3	17.03	*	*	*
fr1_room	4.0	*	*	5.70	5.78	*	*	*
fr1_rpy	2.48	*	*	2.21	*	*	*	*
fr1_desk	1.58	2.03	*	1.57	1.54	9.3	10.65	×
fr2_xyz	0.37	0.40	0.43	0.40	0.39	*	2.15	0.2
fr2_desk	0.86	0.93	*	0.90	0.98	4.5	4.57	×
fr2_large_loop	11.70	10.20	*	15.10	20.99	*	×	×
fr3_str_tex_near	0.93	1.03	1.25	1.54	*	1.3	×	1.04
fr3_str_tex_far	1.07	0.92	0.89	0.98	*	1.2	7.95	0.93
fr3_long_office	0.98	1.86	1.97	1.12	0.93	6.5	38.53	×
fr3_nstr_tex_n_wl	1.09	*	×	1.38	*	*	*	*
fr3_nstr_tex_far	2.64	2.74	×	8.24	*	9.0	18.31	34.74
fr3_large_cabinet	3.32	*	*	6.19	*	*	*	*

data/datasets/rgbd-dataset/tools), is utilized to evaluate the performance of visual SLAM systems. The metric index of the absolute camera trajectory error is used to evaluate our SLAM method. Root Mean Square Errors (RMSE) of the absolute translation error (ATE) is considered as the main evaluation criterion and applied to measure the difference between the ground-truth and the estimated. RMSE is an important evaluation indicator, and the RMSE of ATE reflects the accuracy and robustness of SLAM generally. For the state-of-the-art SLAM systems (e.g., ORB-SLAM2 [31], LSD-SLAM [23]), the RMSE is usually utilized to evaluate the overall performance of SLAM. Given the trajectory estimation \hat{X}_i of a frame and the ground truth X_i , the value of RMSE is calculated by using (18) as follows:

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^n \|\text{trans}(\hat{X}_i) - \text{trans}(X_i)\|^2} \quad (18)$$

n is the number of frames. The experimental results are described in detail below.

A. TUM DATASETS

The TUM datasets is a public benchmark dataset and usually used to evaluate the performance of visual SLAM system [17]. The datasets contains kinds of indoor scene sequences, which are recorded using an RGB-D camera. These image sequences are recorded in real indoor scenes at a frame rate of 30Hz with a 640×480 size and provide the ground-truth trajectories traced by a motion-capture system with higher positioning accuracy. The TUM RGB-D datasets is also a challenging benchmark dataset containing some

blurred, rotation, and low texture sequences, which makes it difficult to estimate camera pose accurately. Three typical examples of image sequences are shown in Fig.8. As a result, many outstanding visual SLAM systems adopted the TUM RGB-D datasets as their benchmark datasets to evaluate the accuracy and robustness of SLAM, for example, the ORB-SLAM2 [31] and RGB-D SLAM [44].

In our practical experiments, the number of extraction point and line features are set as 1000 and 300, respectively. The experimental results of using TUM datasets are shown in Table 1. We compared the performance of our algorithm with RGB-D SLAM [44], PL-SLAM2 [50], PL-F-SLAM [45], and PL-SLAM [14] which are based on the 2D point-line features. We also compared our RGB-D SLAM method with the classical ORB-SLAM2 [31], PTAM [22], and LSD-SLAM [23]. The RMSE of experimental results are presented in Table 1. All statistics are from [14], [44], [45] and our real experiments. “*” indicates that the algorithm does not provide the relevant experimental results. “×” means that tracking lost at some point or a significant portion of the sequences is not processed by the visual SLAM system. The best value is bold and denotes the most accurate result for pose estimation. Some intuitive results and the corresponding ATE of trajectories estimated by ORB-SLAM2 [31] and our RGB-D SLAM are shown in Fig.9. In Fig.9, the black lines represent the ground-truth of trajectories. The blue lines denote the estimated trajectories by our proposed SLAM method and ORB-SLAM2 [31]. The red lines represent the error metric between the ground-truth and the estimated trajectories.

TABLE 2. Comparison of RMSE [cm] in the ICL_NUIM datasets.

Sequences	Living 0	Living 1	Living 2	Living 3	Office 0	Office 1	Office 2	Office 3
Ours	0.60	0.91	1.5	1.21	1.32	1.38	0.96	1.00
ORB-SLAM2	0.66	13.8	1.42	1.33	2.32	11.01	1.17	1.77

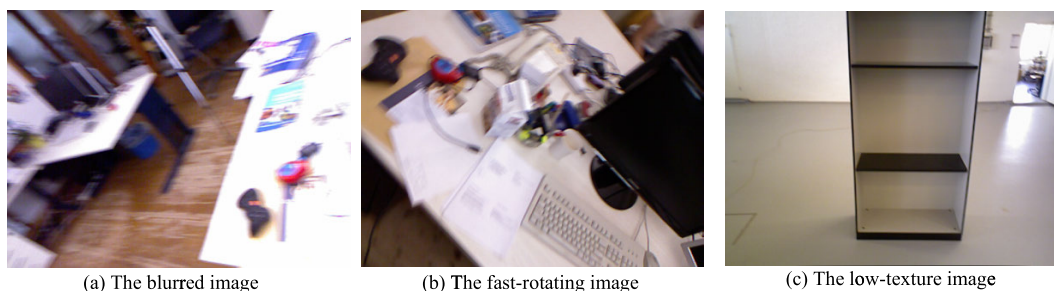


FIGURE 8. Different image sequences of TUM RGB-D datasets.

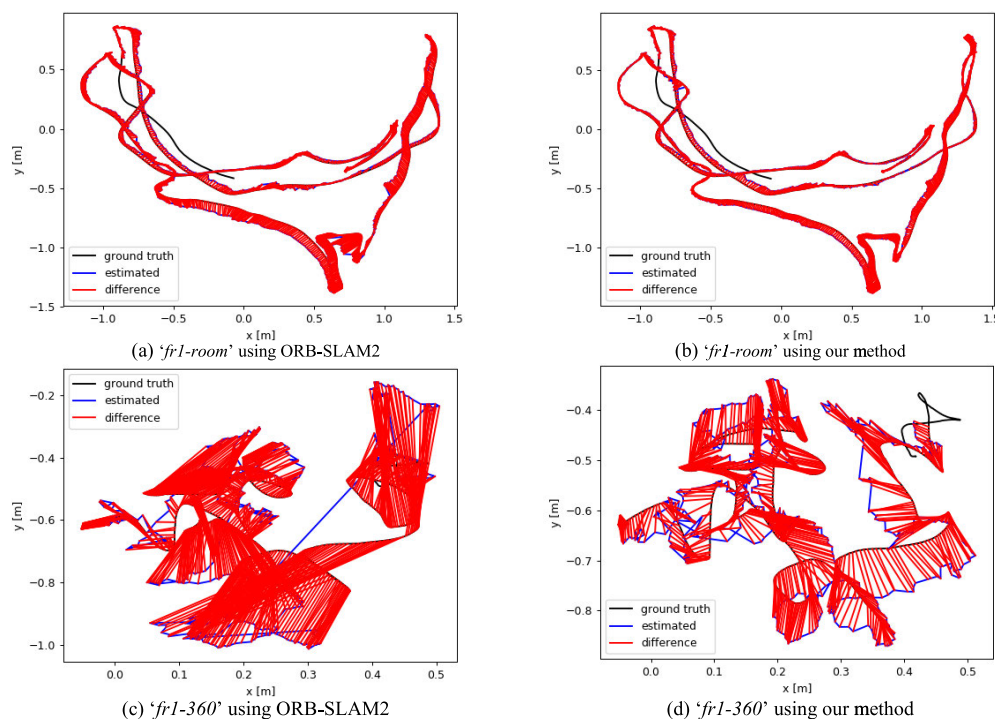


FIGURE 9. The trajectories are estimated using ORB-SLAM2 and our proposed method on TUM datasets.

In Table 1, although ORB-SLAM2 achieved the best accuracy in one sequence, ORB-SLAM2 performed all sequences and showed higher reliability. PL-SLAM yielded the best value in a sequence, and the system was not robust because tracking loss occurred in 2 image sequences. RGB-D SLAM, PL-F SLAM, and PL-SLAM2, which all were classical RGB-D SLAM based on point and line features, achieved four best results in all sequences. However, they also performed a comparable accuracy and the tracking loss did not occur in the experiments. It was noted that our RGB-D SLAM

performed the best accuracy in 8 out of 15 sequences. Even if our method did not perform the best value in 7 of all sequences, our proposal achieved considerable performance compared to the RGB-D SLAM systems above. Moreover, our SLAM method achieved some moderate improvements in these common scene sequences (*fr1-xyz, fr2-xyz*), it was clear that our method achieved significant improvements in accuracy for these fast-rotation and low-texture sequences (*fr1-360, fr3*). Furthermore, from the visible comparisons in Fig. 9, our proposal had a robust performance than ORB-SLAM2.

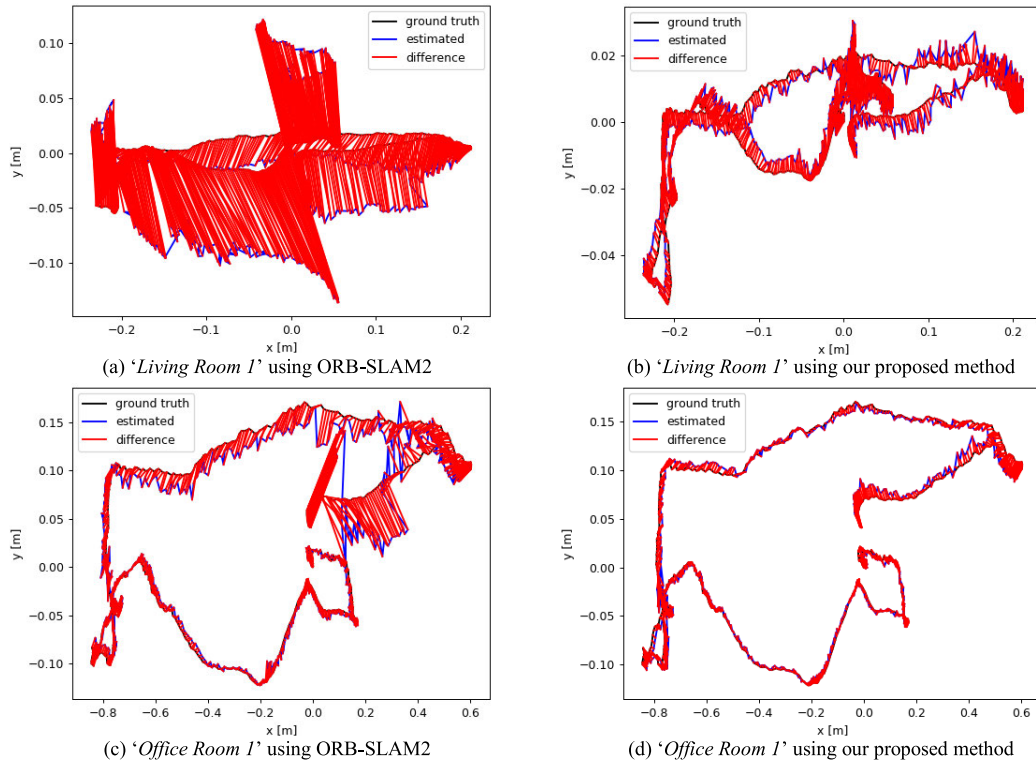


FIGURE 10. The trajectories are estimated using ORB-SLAM2 and our proposed method on ICL-NUIM RGB-D datasets.



FIGURE 11. Example of images captured by Kinect1 in the real scenario. The first row and second row: examples of frames and the basic layout of the corridor.

It was distinct to conclude that our proposal improved the camera poses accuracy effectively.

B. ICL NUIM DATASETS

The ICL-NUIM (Imperial College London and National University of Ireland Maynooth) data- sets [18] comprises the images from a hand-held RGB-D camera in generated environments and the corresponding ground truth of poses are also provided. The image sequences are captured in a living and office room with the ground-truth to measure the accuracy of visual odometry or SLAM. We compared the performance of our proposed SLAM with the ORB-SLAM2 [31] too. The statistics are from our real experiments, and summarized in Table 2. The best values are still bold and represent the most accurate result of camera pose estimation in Table 2. For visible comparison, some experimental results

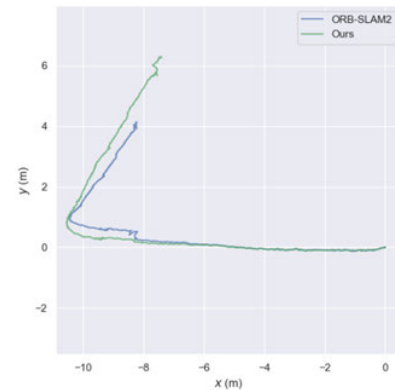


FIGURE 12. The trajectories are estimated using ORB-SLAM2 and our proposed method on the real scene. The trajectory marked by the blue solid line is generated by ORB-SLAM2, while the trajectory marked using the green solid line is from our proposed method.

and the corresponding ATE of trajectories estimated by the ORB-SLAM2 [31] and our proposed method are shown in Fig.10. In Fig.10, the black lines represent the ground-truth trajectories. The blue lines denote the estimated trajectories of our method and ORB-SLAM2 [31]. The red lines represent the difference between the truth and the estimated trajectories.

C. REAL SCENARIO

In this section, we performed our experiments in the real scenario by using ORB-SLAM2 and our method. The image frames were captured by an RGB-D camera of Microsoft

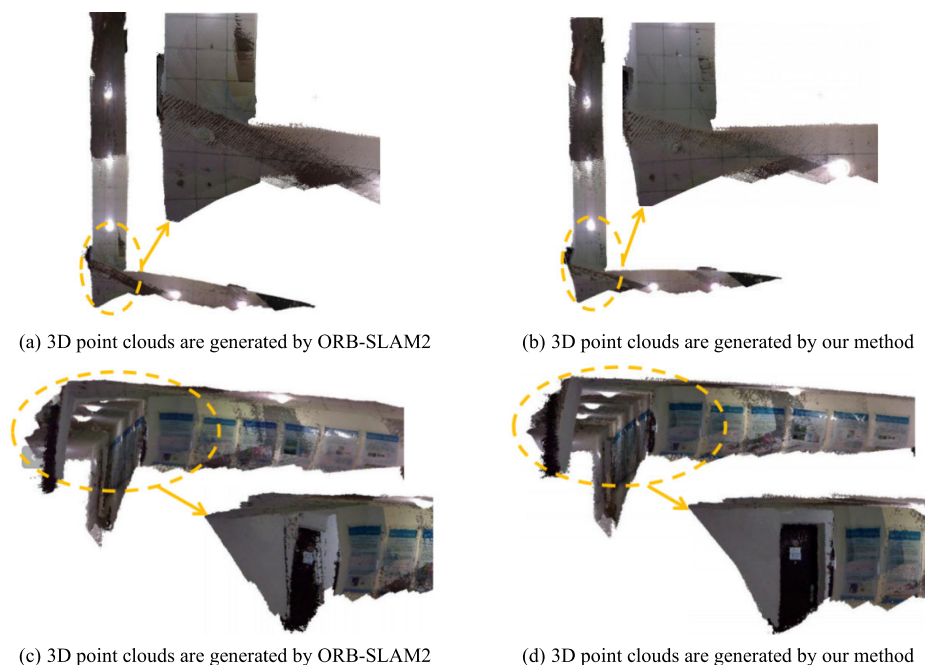


FIGURE 13. Comparison of 3D point clouds from our method and ORB-SLAM2.

Kinect-1, and the examples of the sequences were shown in Fig.11. The dataset is recorded in an L-shaped corridor and owns various situations, for instance, rolling, low-texture scene, and dim light, which is also a challenging image sequence for visual SLAM. Our proposed RGB-D SLAM and ORB-SLAM2 performed the experiments. We also displayed the 3D point clouds generated from our method and ORB-SLAM2 in Fig.13 visually.

Fig.12 shows the trajectories estimated by ORB-SLAM2 and our proposed method, while Fig.13 provides a comparison of 3D point clouds generated from ORB-SLAM2 and our proposal from different views. Visually, compared to ours, ORB-SLAM2 yielded an obvious drifting shown in Fig.13 (a). Moreover, in the yellow ellipse of Fig.13 (d), our proposed method outputted more stable 3D point clouds than ORB-SLAM2, such as the ceiling and door in the corner. As the real scenario dataset includes amounts of frames with the low-texture and structure information in the corner of the corridor, it is manifested to find out that the ORB-SLAM2 generated big drafting in Fig.12 and the warping point clouds visibly in Fig.13. Consequently, it can be summarized that our proposed method is more robust and feasible than ORB-SLAM2 in the real scenario.

D. RUNTIME ANALYSIS

In addition to the accuracy, efficiency is also an important evaluation indicator for SLAM, and we investigate the efficiency performance of our SLAM. We take the “*fr2_large_loop*” as an example and employ the evaluation indicator of average processing frames per second (FPS) to evaluate the efficiency performance. There are 5182 frames in the “*fr2_large_loop*” sequence with a distance of 39.11 meters.

ORB-SLAM2 operates one frame in 0.032s at a higher FPS of 31.25. Since our proposed RGB-D SLAM needs to deal with the 2D and 3D information within points and lines, our proposed method consumes a little time more than ORB-SLAM2. Although our proposal processes one frame at a lower FPS of 16.67, our method merges the 2D and 3D of points and lines to estimate the camera pose with higher accuracy.

IV. DISCUSSION

In the experimental results of Table 1 and Fig.9, PL-F-SLAM [45] and our method outperform other SLAM solutions. Comparing to the SLAM methods which are based on the 2D re-projection error of point and line features [14], [44], [45], our proposed method is established based on the 2D and 3D points and lines and achieves obvious improvement in location accuracy for the indoor scene sequences. As RGB-D camera can provide depth information, our method makes use of 2D and 3D information within points and lines, which is the distinct difference from the RGB-D SLAM with points and lines [44], [45]. In contrast to the PL-SLAM with a monocular camera [14], it is inessential to perform the initialization for acquiring 3D information. Our method reduces amount of computation simultaneously, which is another advantage of our method. In contrast to the ORB-SLAM2, our proposal acquires obvious improvements in low-texture sequences (*fr3*) and the rotation sequence (*fr1_360*). We further plot the ATE of estimation trajectories in Fig.9. As shown in Fig.9, our method achieves a better performance than ORB-SLAM2. However, for the fast-rotation and weak-matching scene image

sequences (*fr1_rpy*, *fr1_room*), our method just obtains a mildly poor performance.

In the experimental results of Table 2 and Fig.10, both ORB-SLAM2 and our method run 8 image sequences successfully. In *Living 1* and *Office 1*, our method performs more noticeable improvements in location accuracy, while ORB-SLAM2 only achieves the best performance in *Living 2*. The following reasons are accounting for it: As the ORB-SLAM2 is run by utilizing points only, and the low-texture scene sequences contribute that the extraction point features are too concentrated to distribute in the whole image region uniformly, therefore the accuracy and robustness of camera pose is poor. Moreover, the smaller numbers of extraction points in the frames recorded in the low-texture scene is another reason for the poor performance of ORB-SLAM2. For further visual comparison, we also plot the ATE of estimation trajectories in Fig.10. As shown in Fig.10, our method also achieves better performance than ORB-SLAM2.

It can be concluded from the experimental results as follows: our proposed method makes full use of the 2D and 3D geometric information within the points and lines to establish constraint, model and it is different from the visual SLAM methods based on 2D re-projection error of point and line features. As our proposal integrates the 2D and 3D information within points and lines, our SLAM method achieves an improvement in accuracy performance than PL-SLAM using 2D point and line features only. Besides, compared to ORB-SLAM2, PTAM [22], and LSD-SLAM [23], although our method performs some slight improvements in the pose estimation in common scenarios, it can achieve comparable accuracy in those low-texture scenes (*Living 1*, *Office 1*, *fr3*). Finally, in the light of the experimental results of the real scenario, our proposed method obtains higher accuracy with strong robustness and can be applied in the pose estimation model of an RGB-D camera for real application.

V. CONCLUSION AND FUTURE WORK

In this paper, we proposed a new geometric constraint model of RGB-D SLAM with points and lines. The main innovations of our method were summarized as follows: (1) for the point, in addition to the 2D re-projection error of points, as an RGB-D camera can provide the per-depth information, the per-depth was utilized fully and the constraint error of per-depth was added in our method. For the line, we made the most of the 2D and 3D geometric information within the lines to construct our proposed method, and our method was different from the RGB-D SLAM systems based on the 2D re-projection error of point and line features merely. (2) we established our geometric constraint model using the 2D and 3D information of points and lines and extended it to the BA model.

In the experimental section, the qualitative experiment results proved our method improved the accuracy and robustness of ORB-SLAM2 based on point features. We also presented the experiments performed by using the TUM and ICI_NUIM RGB-D datasets. The experimental results

showed that our SLAM method achieved a moderate improvement in regular scenes, and achieved obvious robustness and improvement in the low-texture and blurred image sequences. Moreover, we also performed our experiment in a real scene, and our proposal outperformed the ORB-SLAM2 on the basis of the comparisons of point cloud visually. Furthermore, we compared these RGB-D SLAM methods [44], [45] which are all state-of-the-art RGB-D SLAM systems based on point and line features. These RGB-D SLAM methods utilized 2D information with points and lines to estimate pose. By contrast, our proposed method was established by the 2D and 3D information of points and lines and as such, achieved better performance generally.

In the future, our work will focus on two aspects listed as follows: (1) we will extend our approach into RGB-D SLAM based on multiple features (point-line-plane), which provide more robust pose estimation for some special scenes containing low-texture and structure information. (2) as our proposed SLAM method is based on the 2D and 3D information within point and line, which requires more time than the ORB-SLAM2, the operation efficiency of our SLAM method will be planned to improve in our future work.

ACKNOWLEDGMENT

The author would like to thank the Computer Vision Group from Technical University of Munich and Imperial College London and National University for making the RGB-D SLAM dataset publically available. He also thank Qiang Zhao from Huawei Technologies Co. Ltd., Yueqian Shen from Hohai university for their help on this research.

REFERENCES

- [1] R. Smith, M. Self, and P. Cheeseman, "Estimating uncertain spatial relationships in robotics," in *Proc. IEEE Int. Conf. Robot. Autom.*, Raleigh, NC, USA, Mar./Apr. 1987, p. 850.
- [2] T. Mouats, N. Aouf, L. Chermak, and M. A. Richardson, "Thermal stereo odometry for UAVs," *IEEE Sensors J.*, vol. 15, no. 11, pp. 6335–6347, Nov. 2015.
- [3] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: Part I," *IEEE Robot. Autom. Mag.*, vol. 13, no. 2, pp. 99–110, Jun. 2006.
- [4] L. Shaofang and L. Dawei, "A survey of research situation on navigation by autonomous mobile robot and its related techniques," *Trans. Chin. Soc. Agricult. Machinery*, vol. 33, no. 2, pp. 112–116, 2002.
- [5] A. Hornung, M. Phillips, E. Gil Jones, M. Bennewitz, and S. Chitta, "Navigation in three-dimensional cluttered environments for mobile manipulation," in *Proc. IEEE Int. Conf. Robot. Automat.*, Saint Paul, MN, USA, May 2012, pp. 423–429.
- [6] M. Liu and R. Siegwart, "Navigation on point-cloud—A Riemannian metric approach," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, Hong Kong, Jun. 2014, pp. 4088–4093.
- [7] J. K. Makhubela, T. Zuva, and O. Y. Agunbiade, "A review on vision simultaneous localization and mapping (VSLAM)," in *Proc. Int. Conf. Intell. Innov. Comput. Appl. (ICONIC)*, Plaine, Magnien, Dec. 2018, pp. 1–5.
- [8] S. A. Sadat, K. Chutskoff, D. Jungic, J. Wawerla, and R. Vaughan, "Feature-rich path planning for robust navigation of MAVs with mono-SLAM," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Hong Kong, May 2014, pp. 3870–3875.
- [9] K. Di, F. Xu, J. Wang, S. Agarwal, E. Brodyagina, R. Li, and L. Matthies, "Photogrammetric processing of rover imagery of the 2003 mars exploration rover mission," *ISPRS J. Photogramm. Remote Sens.*, vol. 63, no. 2, pp. 181–201, Mar. 2008.
- [10] B. Wang, J. Zhou, and G. S. Tang, "Research on visual localization method of lunar rover," *Sci. China Inf. Sci.*, vol. 44, no. 4, pp. 260–452, 2014.

- [11] Z. Liu, K. Di, J. Li, J. Xie, X. Cui, L. Xi, W. Wan, and M. Peng, "Landing site topographic mapping and rover localization for Chang'e-4 mission," *Sci. China Inf. Sci.*, vol. 63, no. 4, 2020, Art. no. 140901.
- [12] C. Zhang, T. Huang, and Q. Zhao, "A new model of RGB-D camera calibration based on 3D control field," *Sensors*, vol. 19, no. 11, pp. 5082–5101, 2019.
- [13] R. Guo, K. Peng, W. Fan, Y. Zhai, and Y. Liu, "RGB-D SLAM using point-plane constraints for indoor environments," *Sensors*, vol. 19, no. 12, p. 2721, 2019.
- [14] A. Pumarola, A. Vakhitov, A. Agudo, A. Sanfeliu, and F. Moreno-Noguer, "PL-SLAM: Real-time monocular visual SLAM with points and lines," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2017, pp. 4503–4508.
- [15] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *Proc. Int. Conf. Comput. Vis.*, Barcelona, Spain, Nov. 2011, pp. 2564–2571.
- [16] R. G. von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall, "LSD: A fast line segment detector with a false detection control," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 4, pp. 722–732, Apr. 2010.
- [17] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Vilamoura, Portugal, Oct. 2012, pp. 573–580.
- [18] A. Handa, T. Whelan, J. McDonald, and A. J. Davison, "A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Hong Kong, May 2014, pp. 1524–1531.
- [19] R. Wang, K. Di, W. Wan, and Y. Wang, "Improved point-line feature based visual SLAM method for indoor scenes," *Sensors*, vol. 18, no. 10, p. 3559, Oct. 2018.
- [20] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proc. 7th IEEE Int. Conf. Comput. Vis.*, Kerkyra, Greece, Sep. 1999, pp. 1150–1157.
- [21] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features," *Comput. Vis. Image Und.*, vol. 110, no. 3, pp. 346–359, 2008.
- [22] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proc. 6th IEEE ACM Int. Symp. Mixed Augmented Reality*, Nara, Japan, Nov. 2007, pp. 225–234.
- [23] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Zurich, Switzerland, Sep. 2014, pp. 834–849.
- [24] R. Kummerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g2o: A general framework for graph optimization," in *Proc. IEEE Int. Conf. Robot. Autom.*, Shanghai, China, May 2011, pp. 3607–3613.
- [25] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, "RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments," *Int. J. Robot. Res.*, vol. 31, no. 5, pp. 647–663, Apr. 2012.
- [26] A. S. Huang, A. Bachrach, P. Henry, and M. Krainin, "Visual odometry and mapping for autonomous flight using an RGB-D camera," in *Proc. 15th Int. Symp. Robot. Res. (ISRR)*, Flagstaff, AZ, USA, Sep. 2011, pp. 235–252.
- [27] F. Endres, J. Hess, J. Sturm, D. Cremers, and W. Burgard, "3-D mapping with an RGB-D camera," *IEEE Trans. Robot.*, vol. 30, no. 1, pp. 177–187, Feb. 2014.
- [28] G. Hu, S. Huang, L. Zhao, A. Alempijevic, and G. Dissanayake, "A robust RGB-D SLAM algorithm," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Vilamoura, Portugal, Oct. 2012, pp. 1714–1719.
- [29] C. Kerl, J. Sturm, and D. Cremers, "Dense visual SLAM for RGB-D cameras," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Tokyo, Japan, Nov. 2013, pp. 2100–2106.
- [30] K. Di, Q. Zhao, W. Wan, Y. Wang, and Y. Gao, "RGB-D SLAM based on extended bundle adjustment with 2D and 3D information," *Sensors*, vol. 16, no. 8, p. 1285, Aug. 2016.
- [31] R. Mur-Artal and J. D. Tardos, "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, Oct. 2017.
- [32] D. Duan, M. Xie, Q. Mo, Z. Han, and Y. Wan, "An improved Hough transform for line detection," in *Proc. Int. Conf. Comput. Appl. Syst. Model. (ICCASM)*, Taiyuan, China, Oct. 2010, pp. 354–357.
- [33] C. Akinlar and C. Topal, "EDLines: A real-time line segment detector with a false detection control," *Pattern Recognit. Lett.*, vol. 32, no. 13, pp. 1633–1642, Oct. 2011.
- [34] R. Gomez-Ojeda and J. Gonzalez-Jimenez, "Robust stereo visual odometry through a probabilistic combination of points and line segments," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Stockholm, Sweden, May 2016, pp. 2521–2526.
- [35] R. Gomez-Ojeda, F.-A. Moreno, D. Zuniga-Noel, D. Scaramuzza, and J. Gonzalez-Jimenez, "PL-SLAM: A stereo SLAM system through the combination of points and line segments," *IEEE Trans. Robot.*, vol. 35, no. 3, pp. 734–746, Jun. 2019.
- [36] Y. Zhao and P. A. Vela, "Good line cutting: Towards accurate pose tracking of line-assisted VO/VSLAM," in *Proc. 15th Eur. Conf.*, Munich, Germany, Sep. 2018, pp. 527–543.
- [37] G. Zhang, J. H. Lee, J. Lim, and I. H. Suh, "Building a 3-D line-based map using stereo SLAM," *IEEE Trans. Robot.*, vol. 31, no. 6, pp. 1364–1377, Dec. 2015.
- [38] X. Zuo, X. Xie, Y. Liu, and G. Huang, "Robust visual SLAM with point and line features," 2017, *arXiv:1711.08654*. [Online]. Available: <http://arxiv.org/abs/1711.08654>
- [39] 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. 12816–12833, Jun. 2015.
- [40] Y. Yang, P. Geneva, and K. Eickenhoff, "Visual-inertial odometry with point and line features," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Macau, China, Nov. 2019, pp. 2447–2454.
- [41] 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.
- [42] Y. Lu and D. Song, "Robust RGB-D odometry using point and line features," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Santiago, Chile, Dec. 2015, pp. 3934–3942.
- [43] M. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. Assoc. Comp.*, vol. 24, no. 6, pp. 381–395, 1981.
- [44] Q. Fu, H. Yu, L. Lai, J. Wang, X. Peng, W. Sun, and M. Sun, "A robust RGB-D SLAM system with points and lines for low texture indoor environments," *IEEE Sensors J.*, vol. 19, no. 21, pp. 9908–9920, Nov. 2019.
- [45] B. Fang and Z. Zhan, "A visual SLAM method based on point-line fusion in weak-matching scene," *Int. J. Adv. Robotic Syst.*, vol. 17, no. 2, pp. 1–10, 2020.
- [46] L. Lu and S. Liu, "Combined point and line features PL-SLAM algorithm research," in *Proc. 2nd Int. Conf. Comput. Sci. Softw. Eng. (CSSE)*, Xi'an, China, 2019, pp. 62–66.
- [47] Y. Zou, A. Eldemiry, Y. Li, and W. Chen, "Robust RGB-D SLAM using point and line features for low textured scene," *Sensors*, vol. 20, no. 17, pp. 4984–5005, 2020.
- [48] H. Yu, Q. Fu, Z. Yang, L. Tan, W. Sun, and M. Sun, "Robust robot pose estimation for challenging scenes with an RGB-D camera," *IEEE Sensors J.*, vol. 19, no. 6, pp. 2217–2229, Mar. 2019.
- [49] Z. Zhang, "Flexible camera calibration by viewing a plane from unknown orientations," in *Proc. 7th IEEE Int. Conf. Comput. Vis.*, Kerkyra, Greece, Sep. 1999, pp. 666–673.
- [50] L. Zhang and R. Koch, "An efficient and robust line segment matching approach based on LBD descriptor and pairwise geometric consistency," *J. Vis. Commun. Image Represent.*, vol. 24, no. 7, pp. 794–805, Oct. 2013.



CHENYANG ZHANG was born in 1991. He received the M.S. degree from the School of Surveying and Mapping, Shandong University of Science and Technology, Qingdao, China, in 2017. He is currently pursuing the Ph.D. degree with the School of Earth Science and Engineering, Hohai University. His current research interests include visual SLAM, photogrammetry, and computer vision.