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

Structure Reconstruction Using Ray-Point-Ray Features: Representation and Camera Pose Estimation

Yijia He*, Xiangyue Liu[1*], Xiao Liu[2], Ji Zhao†

Abstract— **Straight line features have been increasingly utilized in visual SLAM and 3D reconstruction systems. The straight lines' parameterization, parallel constraint, and coplanar constraint are studied in many recent works. In this paper, we explore the novel intersection constraint of straight lines for structure reconstruction. First, a minimum parameterized representation of ray-point-ray (RPR) structures is proposed to represent the intersection of two straight lines in the 3D space. Second, an efficient solver is designed for the camera pose estimation, which leverages the perpendicularity and intersection of straight lines. Third, we build a stereo visual odometry based on RPR features and evaluate it on the simulation and real datasets. The experimental results verify that the intersection constraints from RPR can effectively improve the accuracy and efficiency of line-based SLAM and reconstruction system.**

I. INTRODUCTION

Considering that straight lines are ubiquitous in man-made environments, line features are incorporated into the structure reconstruction and SLAM systems to improve the accuracy and robustness of the system, especially in weak texture areas [1]–[3]. In addition, compared to point features, straight lines provide more structural and semantic information, such as line-based building wireframe estimation [4]. The parameterized representation of straight lines in 3D space and the further camera pose estimation are two important issues.

For the parameterized representation of straight lines, Zhang et al. [5] used the Plücker line coordinates with six parameters to initialize and transform spatial straight lines in their SLAM system. However, straight line has only four degrees of freedom (DoF). To avoid over-parameter problems during optimization, the orthogonal representation [6] and the closest point representation [7], which contain only four parameters, are used in bundle adjustment optimization. In many structured environments such as Manhattan World and Atlanta World, the parameters of straight lines can be further reduced due to the parallelism between straight lines. Zou et al. [8] realized the StructVIO system that only used the two parameters, inverse depth and angle, to represent straight lines in a specific direction. In addition, the constraints of the coplanarity between straight lines can be introduced into the bundle adjustment [9], [10]. Although the previous works have explored many geometric constraints, the constraint from line intersection is rarely mentioned.

The absolute camera pose can be solved from the 3D straight lines. The typical Perspective-3-Line (P3L) problem is to obtain the camera pose given three 3D straight lines and their corresponding 2D image projections. The order of the polynomial equation to solve the general P3L problem is eight [11]. Using constraints such as parallelism and perpendicularity can reduce the order of the polynomial and make the solution more efficient [12], [13]. However, these algorithms rely on the assumption that two lines are parallel and the third line is perpendicular to these two lines. Kuang et al. [14] proposed a three-variable polynomial system based on points and direction vectors to solve the rotation of the camera, in which the intersection of two straight lines is transformed into a point and two direction constraints. But this method needs to be solved by Gröbner basis [15], and its computational efficiency is limited. Xu et al. [16] used the structure that three straight lines intersect at one point to construct a virtual camera coordinate system to obtain the rotation matrix, by efficiently solving a quartic equation.

In our previous work [17], [18], we proposed a detection and matching algorithm based on ray-point-ray (RPR) features, and used the 2D-2D RPR matching pairs in image plane to solve the *relative pose between two cameras*. The representation of RPR in 3D space and the usage of RPR to solve the absolute pose are not involved. In this paper, we further consider these two questions: 1) how to form the minimum parameterized representation of RPR for 3D reconstruction? 2) how to leverage the perpendicular intersected lines, which are common in the artificial environment, to realize the *absolute camera pose estimation*?

The contributions of this paper are as follows:

- We propose a minimum parameterized representation of RPR structures in the 3D space, based on which the corresponding geometric transformation, initialization, and measurement model are derived.
- Using the RPR90 structure (i.e. the two intersected lines are perpendicular) in the 3D space, a solver for *absolute camera pose estimation* is proposed, which has high precision and good efficiency.
- A stereo odometry system is built based on the above methods. Using RPR measurement model in bundle adjustment optimization, and using RPR90 structure in camera absolute pose estimation, the system accuracy can be effectively improved.
- The experimental results on the simulation data and real data show that our method outperforms the state-of-the-

* Equal contribution, † Corresponding author
Yijia He. E-mail: heyijia2016@gmail.com
[1]Xiangyue Liu is with the School of Software, Beihang University, Beijing, China. E-mail: liuxiangyue@buaa.edu.cn
[2]Xiao Liu is with the Megvii (Face++) Technology Inc., Beijing, China. E-mail: liuxiao@megvii.com
Ji Zhao. E-mail: zhaoji84@gmail.com

art methods in terms of efficiency and accuracy.

II. 3D RPR Representation

A. Preliminaries

In this paper, the k-th 3D point in the i-th camera frame is represented by bold capital letters, such as $\mathbf{P}_k^{c_i}$. The corresponding bold lowercase letters, $\mathbf{p}_k^{c_i}$, are used to represent the projection coordinates of the 3D points in image plane. In order to ignore the instrinsic parameters of camera, we only consider normalized image plane (focal length is 1) in this paper. For the relative pose from the world frame w to the camera frame c, we use $\mathbf{R}_{cw} \in SO(3)$ and $\mathbf{t}_{cw} \in \mathbb{R}^3$ to represent the rotation and translation, respectively. $SO(3)$ is the Lie group of rotation matrix, and the Lie algebra $so(3)$ is the minimal representation of the rotation, which are used in camera pose optimization.

B. RPR Representation

In the 3D space, a straight line can be represented by a 3D point $\mathbf{P}$ and a direction vector $\mathbf{d}$. However, this is an over-parametrized representation. Orthonormal Representation [6] or Closest Point Representation [7] provides the 4-DoF representation of infinite straight line. For RPR structure, presentation with two independent straight lines is over-parametrized, because the intersection constraint is not considered.

An intuitive parameter representation of k-th RPR in world frame is a triplet $\mathcal{T}_k^w = \left(\mathbf{P}_k^w, \mathbf{d}_{s_k}^w, \mathbf{d}_{e_k}^w\right)$, where $\mathbf{P}_k^w \in \mathbb{R}^3$ is the junction point in world frame, and $\mathbf{d}_{s_k}^w, \mathbf{d}_{e_k}^w$ are unit direction vectors indicating the two intersected straight lines. The 3D unit direction vector is parameterized on unit sphere S^2, so the triplet representation of RPR has 7 DoF. Specially, in the RPR90 case when the two lines are perpendicular, $\mathbf{d}_s^\top \mathbf{d}_e = 0$, the RPR90 representation only has 6 DoF. However, the perpendicular constraint on $\mathbf{d}$ can not be directly used in the least square optimization.

In order to give a minimal representation of general RPR structure, we build a frame r on the RPR structure. As shown in Fig.1, in the RPR frame r, the origin is the junction point, the x axis is along the direction $\mathbf{d}_s$, and the two direction vector decide the xy plane. Thus, the proposed parameterized representation of RPR is given by $\mathcal{F}_k^w = (\mathbf{R}_{wr_k}, \mathbf{P}_k^w, \phi) \in SO(3) \times \mathbb{R}^3 \times \mathbb{R}^1$. Where $\phi = \arccos\left(\mathbf{d}_{s_k}^{w\top} \mathbf{d}_{e_k}^w\right)$ is the angle between the two directions, $\mathbf{R}_{wr_k}$ is the rotation matrix constructed by the two unit directions,

$$\mathbf{R}_{wr_k} = \begin{bmatrix} \mathbf{d}_{s_k}^w & \left(\mathbf{d}_{s_k}^w \times \mathbf{d}_{e_k}^w\right) \times \mathbf{d}_{s_k}^w & \mathbf{d}_{s_k}^w \times \mathbf{d}_{e_k}^w \end{bmatrix} \tag{1}$$

For RPR90, ϕ is constant value $\frac{\pi}{2}$.

Transforming the minimum representation $\mathcal{F}_k$ to the standard representation $\mathcal{T}_k$ is given by:

$$\mathbf{d}_s^w = \mathbf{R}_{wr} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{d}_e^w = \mathbf{R}_{wr} \begin{bmatrix} \cos\phi \\ \sin\phi \\ 0 \end{bmatrix} \tag{2}$$

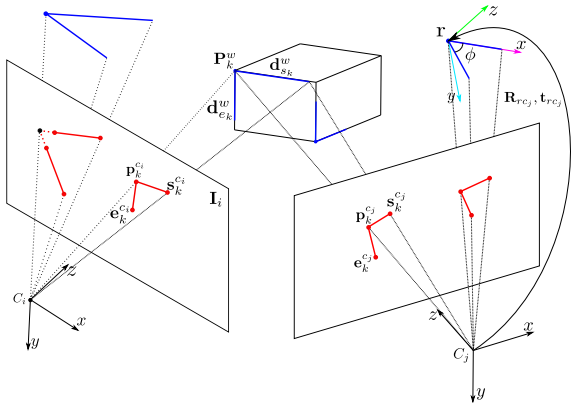


Fig. 1. Illustration of the geometric relation between the RPR structures (blue lines) and the cameras. The projection of RPR structure is shown in red. Three axes of an RPR frame are shown in magenta, cyan and green respectively.

C. RPR Triangulation

The observation of the RPR structure on each image plane is $\mathbf{z} = (\mathbf{p}, \mathbf{s}, \mathbf{e})$, where $\mathbf{p} = [u_p \ v_p \ 1]^\top$ is the detected junction point, $\mathbf{s} = [u_s \ v_s \ 1]^\top$ and $\mathbf{e} = [u_e \ v_e \ 1]^\top$ are the endpoints on the two line segments respectively.

Given more than two frames of camera observations $\mathbf{z}_i$ and the corresponding poses $\mathbf{R}_{wc_i}$, the RPR can be reconstructed. The triangulation of junction point $\mathbf{P}$ is calculated in the same way with the triagulation of point feature. For the triangulation of the direction vector $\mathbf{d}$, we adopt the constraint that the direction vector falls in the plane formed by junction point, endpoint and the origin point of camera coordinate. For endpoint $\mathbf{s}$, we have

$$\mathbf{n}^{c\top} \mathbf{R}_{cw} \mathbf{d}_s^w = \mathbf{0}, \ \mathbf{n}^c = \frac{[\mathbf{p}]_\times \mathbf{s}}{\left\| [\mathbf{p}]_\times \mathbf{s} \right\|} \tag{3}$$

where $\mathbf{n}^c$ is the plane normal vector, $[\cdot]_\times$ is an operator transforming a 3D vector to a skew symmetric matrix. When multiple camera poses and observations are given, a set of linear equations can be built:

$$\mathbf{A}\mathbf{d}_s^w = \begin{bmatrix} \left(\mathbf{R}_{wc_1} \mathbf{n}_s^{c_1}\right)^\top \\ \cdots \\ \left(\mathbf{R}_{wc_m} \mathbf{n}_s^{c_m}\right)^\top \end{bmatrix} \mathbf{d}_s^w = \mathbf{0} \tag{4}$$

The 3D direction $\mathbf{d}_s^w$ is given by the eigenvector corresponding to the smallest eigenvalue of $\mathbf{A}^\top \mathbf{A}$.

D. RPR Measurement Model

Two types of constraints can be constructed between the 3D RPR structure and the image observation $\mathbf{z}$. Before building the measurement model, the RPR $\mathcal{T}$ must be transformed from world frame to camera frame:

$$\begin{aligned} \mathbf{P}^c &= \mathbf{R}_{wc}^\top \left(\mathbf{P}^w - \mathbf{t}_{wc}\right) = \begin{bmatrix} p_x^c & p_y^c & p_z^c \end{bmatrix}^\top \\ \mathbf{d}^c &= \mathbf{R}_{wc}^\top \mathbf{d}^w \end{aligned} \tag{5}$$

1) Junction Point Measurement Model: The junction point is projected to image plane, and the re-projection measurement model is calculated with the detected junction $\mathbf{p}$:

$$\mathbf{r}_p = \begin{bmatrix} \frac{p_x^c}{p_z^c} - u_p & \frac{p_y^c}{p_z^c} - v_p \end{bmatrix}^\top \tag{6}$$

2) Direction Measurement Model: Firstly, two straight lines of an RPR are projected to the normalized image plane to get the projection line segments. To obtain the projection line equation $\mathbf{l}$, we take two 3D points on a straight line of RPR, such as $\mathbf{P}^c$ and $(\mathbf{P}^c + \mathbf{d}^c)$, and project them to the normalized image plane. $\mathbf{l}$ can be calculated by the cross product of these two projection points :

$$\mathbf{l} = \frac{1}{p_z^c} \frac{1}{p_z^c + d_z^c} [\mathbf{P}^c]_\times (\mathbf{P}^c + \mathbf{d}^c) \tag{7}$$

Then, the direction measurement model is the distance from the detected endpoints $\mathbf{s}$ and $\mathbf{e}$ to the predicted line segments $\mathbf{l}_s$ and $\mathbf{l}_e$, respectively:

$$\mathbf{r}_d = \begin{bmatrix} \frac{\mathbf{s}^\top \mathbf{l}_s}{\sqrt{l_{sx}^2 + l_{sy}^2}} & \frac{\mathbf{e}^\top \mathbf{l}_e}{\sqrt{l_{ex}^2 + l_{ey}^2}} \end{bmatrix}^\top \tag{8}$$

III. Absolute Camera Pose Estimation

In this section, the absolute camera pose estimation problem based on minimal 3D-2D constraints is firstly defined. Then, an efficient and stable solver based on the RPR90 structure is proposed to solve the camera pose $\mathbf{R}_{wc}, \mathbf{t}_{wc}$.

A. Absolute Camera Pose Estimation with Line Segments

A 3D straight line and its corresponding 2D line segment observation can not only provide direction constraints in Eq. (3), but also provide a point on plane constraint:

$$\mathbf{n}^{c\top} (\mathbf{R}_{cw} \mathbf{P}^w + \mathbf{t}_{cw}) = 0 \tag{9}$$

where $\mathbf{P}^w$ is a point on 3D straight line, and $\mathbf{n}^c$ is the normal vector of the plane where the camera center and the observed line segment lie on. The P3L problem is to use 6 constraints provided by 3 straight lines to solve a camera pose with 6 DoF [16]. The solution is divided into two steps. First, the rotation matrix is solved by the direction constraint (3), then the translation vector is solved by the point constraint (9). There are two types of methods to solve the rotation matrix. One method is to represent the rotation matrix by the quaternion or Cayley representation, then use direction constraints to construct a polynomial system, finally use Gröbner basis to solve it [14]. Another method is to use the intermediate coordinate system to simplify the representation of the rotation matrix, thereby obtaining an efficient closed-form solution [16].

Considering an RPR structure provides 2 line segments with a junction, we can use 1.5 RPRs to determine the absolute pose: one complete RPR90 and a junction point of another RPR. Then we have 6 constraints to estimate the absolute camera pose. Being different from ordinary straight line, the perpendicular intersection constraint of RPR90 can improve the stability and efficiency of the pose estimation.

B. Efficient Solver with One RPR90 and One Point

Similar to Xu's method [16], we first construct a virtual camera coordinate system to reduce the order of the polynomial. Different from the way that they use two variables to represent the rotation matrix, we take advantage of the perpendicular intersection of the RPR90 structure to express

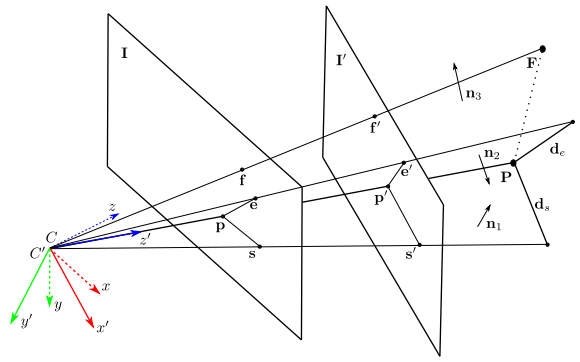


Fig. 2. The geometry of a virtual camera frame c', the camera c, and an RPR feature.

the rotation matrix with only one variable, which further improves the efficiency of the rotation matrix solving.

From the calculation of the normal vector in Eq. (3), it can be seen that if the line segment observation intersects with the camera's z-axis, the corresponding normal vector is $\mathbf{n} = \frac{[\mathbf{z}]_\times \mathbf{s}}{\|[\mathbf{z}]_\times \mathbf{s}\|} = [n_x \ n_y \ 0]^\top$, where $z = [0 \ 0 \ 1]^\top$. The $n_z = 0$ will simplify the polynomial equation (3) by eliminating the third row of the rotation matrix. Inspired by this, we build a virtual camera c' that the coordinate origin is the same with that of the camera coordinate c and the z-axis of the virtual camera is a unit vector point from camera center to the junction point $\mathbf{P}$, as shown in Fig. 2. The x-axis of the virtual camera is parallel to the observation vector pointing from $\mathbf{p}'$ to $\mathbf{s}'$ in virtual image plane. The parameter of the virtual image plane $\mathbf{I}'$ expressed in the camera frame c is $\boldsymbol{\pi}'^c = \begin{bmatrix} \mathbf{v}_z^\top & -1 \end{bmatrix}^\top$, where $\mathbf{v}_z = \frac{\overrightarrow{\mathbf{Cp}}}{\|\overrightarrow{\mathbf{Cp}}\|}$. The coordinate of the virtual observation $\mathbf{p}'^c$ in the camera frame c can be obtained by the intersection of the ray $\overrightarrow{\mathbf{Cp}}$ and the virtual camera plane $\boldsymbol{\pi}'^c$. The rotation matrix $\mathbf{R}_{cc'}$ from virtual camera to actual camera can be obtained directly.

$$\mathbf{R}_{cc'} = \begin{bmatrix} \mathbf{v}_x & [\mathbf{v}_z]_\times \mathbf{v}_x & \mathbf{v}_z \end{bmatrix}, \quad \mathbf{v}_x = \frac{\overrightarrow{\mathbf{p}'^c \mathbf{s}'^c}}{\|\overrightarrow{\mathbf{p}'^c \mathbf{s}'^c}\|} \tag{10}$$

Since the 3D coordinates of RPR have been given, i.e. the rotation matrix $\mathbf{R}_{rw}$ between the coordinate system r constructed by RPR and the world coordinate system w is known, the camera pose estimation problem $\mathbf{R}_{cw} = \mathbf{R}_{cc'} \mathbf{R}_{c'r} \mathbf{R}_{rw}$ is transferred to solving $\mathbf{R}_{c'r}$.

The direction vector $\mathbf{d}_s$, the x-axis of the RPR90, is in the xz plane of the virtual camera coordinate. Assume the angle between the virtual x-axis and the direction vector is $\alpha \in \left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$, then the vector $\mathbf{d}_s$ in virtual camera frame c' is:

$$\mathbf{d}'_s = \mathbf{R}_y(\alpha) [1 \ 0 \ 0]^\top = [\cos(\alpha) \ \ 0 \ \ \sin(\alpha)]^\top \tag{11}$$

where $\mathbf{R}_y(\cdot)$ is a rotation matrix around the y-axis. Since $\mathbf{d}_e'^\top \mathbf{d}'_s = 0$ and $\mathbf{d}_e'^\top \mathbf{n}'_2 = 0$, where $\mathbf{n}'_2$ is the normal vector of the plane decided by the points $\mathbf{p}'$, $\mathbf{e}'$ and C', the direction vector $\mathbf{d}_e$ in virtual camera frame $\mathbf{d}'_e$ can be calculated by:

$$\mathbf{d}'_e = \frac{\mathbf{v}'_e}{\|\mathbf{v}'_e\|}, \quad \mathbf{v}'_e = \begin{cases} [\mathbf{n}'_2]_\times \mathbf{d}'_s, & \alpha > 0. \\ -[\mathbf{n}'_2]_\times \mathbf{d}'_s, & \alpha < 0. \end{cases} \tag{12}$$

where $\alpha = 0$ is a degeneration case that $\mathbf{d}'_e$ can not be solved correctly, we will discuss this in Sec. III-C. Thus, the rotation matrix $\mathbf{R}_{c'r}$ from RPR frame to virtual camera can be represented with only one parameter α:

$$\mathbf{R}_{c'r} = \begin{bmatrix} \mathbf{d}'_s & \mathbf{d}'_e & \mathbf{d}'_s \times \mathbf{d}'_e \end{bmatrix} \tag{13}$$

Given one more direction constraint, the rotation matrix can be solved. Choose an observed 3D point $\mathbf{F}$, then the connection between the point $\mathbf{F}$ and the junction point $\mathbf{P}$ can determine a new direction vector $\mathbf{v}_3 = \frac{\overrightarrow{\mathbf{PF}}}{\|\overrightarrow{\mathbf{PF}}\|}$ in RPR frame, and the corresponding plane normal vector in virtual camera frame can be calculated by Eq. (3), $\mathbf{n}'_3 = \begin{bmatrix} n'_{3x} & n'_{3y} & 0 \end{bmatrix}^\top$. A quartic equation of $\sin(\alpha)$ can be build from the constraint $\mathbf{n}_3'^\top \mathbf{R}_{c'r}\mathbf{v_3} = 0$. Let x denote $\sin(\alpha)$, we have

$$\begin{aligned} \sum_{i=0}^{4} \delta_i x^i &= 0 \\ \delta_4 &= a^2 + d \\ \delta_3 &= 2ab \\ \delta_2 &= b^2 + 2ac + e - d \\ \delta_1 &= 2bc \\ \delta_0 &= c^2 - e \end{aligned} \tag{14}$$

where $a = -n'_{3x}n'_{2x}v_{3z}$, $b = -(n'_{3y}n'_{2x} - n'_{3x}n'_{2y})v_{3y}$, $c = -n'_{3y}n'_{2y}v_{3z}$, $d = n'^2_{2x}n'^2_{3x}v^2_{3x}$, and $e = n'^2_{2y}n'^2_{3x}v^2_{3x}$. In addition, the equation system (14) holds no matter whether α is greater or less than 0. After the quartic equation is solved, we compute $\arcsin(\cdot)$ to get α and recover the rotation matrix by Eq. (13). Xu's method uses two variables to parameterize the rotation matrix and will have up to 8 solutions. In comparison, our method has at most four solutions.

After rotation matrix is solved, we use the point observation to solve translation. Based on the point constraint, we have:

$$\begin{bmatrix} \begin{bmatrix}\mathbf{p}^1\end{bmatrix}_\times \\ \begin{bmatrix}\mathbf{p}^2\end{bmatrix}_\times \end{bmatrix} \mathbf{t}_{cw} = \begin{bmatrix} -\begin{bmatrix}\mathbf{p}^1\end{bmatrix}_\times \mathbf{R}_{cw}\mathbf{P}^1 \\ -\begin{bmatrix}\mathbf{p}^2\end{bmatrix}_\times \mathbf{R}_{cw}\mathbf{P}^2 \end{bmatrix} \Rightarrow \mathbf{H}\mathbf{t}_{cw} = \mathbf{b} \tag{15}$$

where $rank(\mathbf{H}) = 3$. The translation vector can be recovered by $\mathbf{t}_{cw} = \left(\mathbf{H}^\top\mathbf{H}\right)^{-1}\mathbf{H}^\top\mathbf{b}$.

C. Degeneration Cases

In the case that $\alpha = 0$, the x-axis of RPR frame is strictly parallel to the x-axis of the virtual frame. However, such cases rarely occur. Another degenerate situation is that the selected 3D point is localized on one of the straight lines corresponding to the RPR. Fortunately, we can avoid this situation when selecting points.

IV. Stereo Visual Odometry with RPR

In this section, we build a stereo visual odometry (VO) system that only uses RPR features. The system consists of a front-end tracking module and a back-end local bundle adjustment module, which are described as follows.

A. Front-end Tracking Module

The front-end module provides feature extraction and initial relative pose estimation between frames. For each stereo frame, we use the method [17] to detect the RPR features in left image and match them with those in the latest keyframe to complete data association. Then, we randomly select an RPR90 feature and a junction point from the other RPR features, to estimate the relative pose from the new frame to latest keyframe with our pose solver in Sec. III. The RANSAC algorithm is also used to select the best pose estimation result. If the translation or rotation angle of the relative pose exceeds a threshold, we will select a new keyframe. For each keyframe, we will extract new RPR features in stereo frames and triangulate the RPR with the observations from left and right cameras. The left camera coordinate system of the first frame is set as the world coordinate.

B. Back-end Bundle adjustment Module

The back-end module uses local bundle adjustment method to jointly optimize the keyframe pose and RPR parameters. A sliding window with n active keyframes and m RPRs observed by these active keyframes are jointly optimized. If these RPRs are observed by anchor keyframes outside of the window, the observation of anchor frames will also be added in optimization, but these pose will be fixed during the optimization. Denote $\mathcal{C}_a = \{\mathbf{R}_{wc_i}, \mathbf{t}_{wc_i}\}$ as the set of active keyframes, $\mathcal{C}_b$ as the set of anchor keyframes, and $\mathcal{F} = \left\{\mathcal{F}_j^w\right\}$ as the observed RPRs. The optimal camera poses and RPR parameters are obtained by iteratively optimizing the following cost function

$$\arg\min_{\mathcal{C}_a, \mathcal{F}} \sum_{i \in \mathcal{C}_a, \mathcal{C}_b} \left(\sum_{k \in \mathcal{B}} \rho\left(\left\| \mathbf{r}_{z_k}^{c_i} \right\|^2 \right) \right) \tag{16}$$

where the set $\mathcal{B}$ includes all RPR measurements of $\mathcal{F}$, and ρ is the robust Cauchy cost function. $\mathbf{r}_{z_k}^{c_i}$ is the reprojection residual vector containing the junction point measurement model and direction measurement model by projecting the k-th RPR into i-th keyframe. After the optimization, RPRs with reprojection residuals larger than 5 pixels will be discarded. If the angle ϕ of an RPR is close to 90 degrees, we will regard it as an RPR90 for the subsequent camera pose estimation.

V. EXPERIMENTS

In this section, we used two Monte Carlo simulation experiments and a real dataset experiment to verify the effectiveness of the RPR representation and pose estimation.

A. PRP Representation

Using line segments for reconstruction, the parameterization representation of lines has an significant influence on the accuracy and efficiency of reconstruction. In order to verify the advantages of RPR parameters over the line parameterization in bundle adjustment, a Monte Carlo simulation experiment is carried out. Eight house models constructed by straight lines in outdoors is generated. Each house is

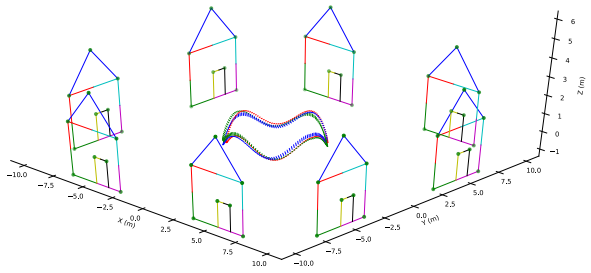


Fig. 3. Simulation setup for RPR representation. Different RPRs are marked on the stick figure house with different colors. All junction points are marked with green dots.

composed of 7 RPRs and 14 lines in total, as shown in Fig. 3. The observation of the line segment is captured by a monocular camera, which moves along the sinusoidal motion trajectory in a region with the radius of 3 meters. 150 camera poses were generated in total. Gaussian White noise is used to mimic the noise in practical application. Each line segment observation consists of two endpoints, and each endpoint measurement has the Gaussian white noise with a standard deviation of 1 pixel. Each camera pose is also perturbed by Gaussian white noise, in which the standard deviation of noise is 0.1m in translation, and 1 degree in rotation angle.

In each simulation experiment, the camera pose and RPR or line are optimized by bundle adjustment. During optimization, the proposed two parameterization methods are tested for RPR. The triplet parameterization, RPR is expressed as a point and two direction vectors, is denoted as RPR-t. The coordinate system parameterization is denoted as RPR-c. For line parameterization, we use the orthogonal representation [6]. In addition, the intersection points as point features is also added to the comparison experiment. The absolute pose error (APE) of each camera pose is calculated and the root mean square error (RMSE) is used for evaluation [19]. The optimization method adopts Levenberg-Marquardt algorithm, and the number of iterations to converge is also regarded as an evaluation metric. The average error and number of iterations after 100 simulation experiments is recorded in the Tab. I.

The experimental results show that the accuracy of point features is the lowest because there are few points in the scene and only the constraints of points are used while ignoring the direction constraints of straight lines. The two different RPR representation methods have the same accuracy, showing that the introduction of line intersection constraint can improve the accuracy. In addition, the RPR representation based method needs the least iteration number to achieve convergence due to its compact representation.

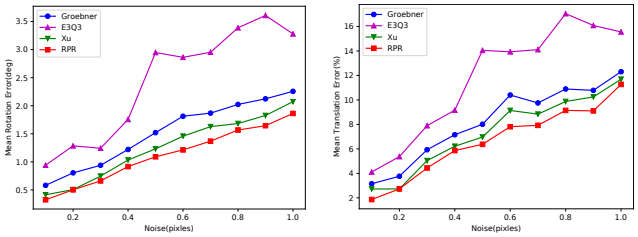
B. Camera Pose Estimation

In this experiment, our RPR based pose solver is compared with the state-of-the-art line based methods: The polynomial based solution proposed by Kuang et al. [14] and Xu's solution [16] based on the intersection of three straight lines. To solve the polynomial systems derived from [14], we use the state-of-the-art automatic solver for polynomial equation systems provided by Larsson et al. [15]. Since this method is

TABLE I

THE AVERAGE RMSE AND THE NUMBER OF ITERATIONS FOR POINT, LINE, RPR-T, RPR-C ARE LIST AS FOLLOWS. THE BEST RESULTS ARE MARKED IN **BOLD** .

	Point	Line	RPR-t	RPR-c
Trans. (cm)	2.24	1.86	**1.74**	**1.74**
Rot. (deg)	0.109	0.092	**0.085**	**0.085**
Iter.	**5**	11	27	**5**



(a) rotation error (b) translation error

Fig. 4. Accuracy comparison under different noise levels.

based on Gröbner basis, it will be referred to as Gröbner in the following. In addition, an efficient E3Q3 solver for three quadrics in three unknowns [20] is also used for comparison to solve the polynomial systems in [14].

To ensure that the measurements given to different solvers are the same, the three direction vectors used to solve the rotation matrix are derived from two direction vectors provided by RPR and one direction vector provided by RPR intersection point and another point. After the rotation matrix is solved, Eq. (15) is used to solve the translation. The simulation camera is generated at the origin of the world coordinate system with 640×480 pixels and focal length 300 pixels. The intersection points of RPR are randomly distributed in the x-, y- and z-range of $[-3,-3]\times[-3,-3]\times[2,8]$. The two direction vectors of RPR are generated by the first and second columns of a random rotation matrix.

The Gaussian white noises are added to the endpoints of the observed line segment, whose standard deviation δ is increased gradually from 0.1 pixel to 1 pixel. With each standard deviation, 1000 random samples were taken. The total runtime of running 10000 times for the different algoritms are shown in Tab. II. All the solvers are implemented in C++ and run on a computer with Intel Core i7 3.20GHz and 16 GB memory.

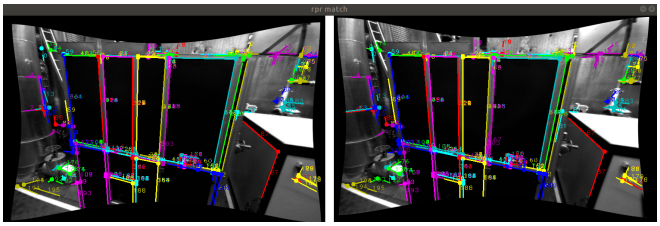
In terms of computational efficiency, our method is the most efficient, because our method only needs to solve a univariate quartic equation. The Gröbner basis method has the lowest efficiency because it generates a 20×20 action matrix and the solution is obtained by eigenvalue decomposition. E3Q3 transforms the problem into an 8-order polynomial equation, and the solution of coefficients requires thousands of multiplications and additions. Xu's method is also a univariate quartic equation, but the introduction of two variables in the intermediate results in more complex coefficient calculation.

In addition, we also compared the algorithms' robustness

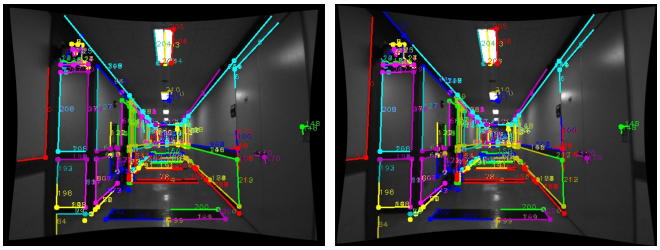
TABLE II

TOTAL TIME-CONSUMING OF 10,000 TESTS FOR DIFFERENT METHODS

Method	Gröbner [15]	E3Q3 [20]	Xu [16]	RPR
Time (ms)	192.44	40.34	15.35	9.24



(a) EuRoc



(b) it3f

Fig. 5. Examples of detected and matched RPRs. The same RPRs in the left and right views are marked by the same color and the same number.

to noise. The evaluation criteria of rotation and translation error are described in [17]. The evaluation results are shown in Fig. 4. Our method has the highest accuracy due to the usage of the constraint of perpendicularity and intersection between lines of an RPR.

C. Visual Odometry

The RPR based method is evaluated in the real world data, also compared to point based odometry and line based odometry. All the odometries are based on the system proposed in Sec. IV. The odometer based on point feature only uses the intersection constraint of RPR. The odometer based on line features uses orthogonal representation [6] to represent the spatial line. Considering that the coordinate based representation RPR-c is more efficient than the triple representation, we use RPR-c in the RPR based system. The parameters of the VO system is the same for all methods.

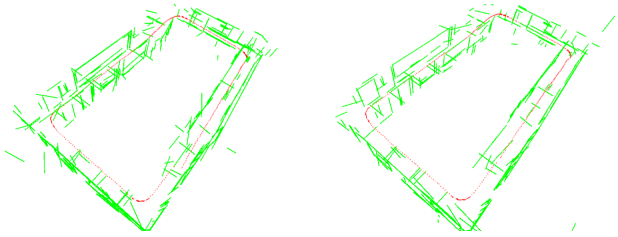
The EuRoc dataset [21] and it3f dataset [5] are used for quantitative and qualitative analysis, respectively. The EuRoc dataset is collected by a stereo camera in two indoor scenes. The intrinsic and external parameters of stereo camera and the ground truth trajectory of camera are provided. The it3f dataset is collected with a stereo camera in a rectangular corridor scene with less texture. The start and end poses in the dataset are near the same location. Detection and matching samples of RPRs are shown in Fig. 5.

For quantitative analysis, we use relative pose error (RPE) [19] to evaluate the accuracy of the VO system. The statistical results on the two sequences of EuRoc are shown in the Tab. III. It can be seen that the accuracy of point

TABLE III

THE TRANSLATION (M) AND ROTATION (DEG) ERROR OF RMSE FOR POINT, LINE, RPR ARE LIST AS FOLLOWS. IN **BOLD** THE BEST RESULT.

Sequence	Point		Line		RPR	
	trans.	rot.	trans.	rot.	trans.	rot.
MH_01	0.091	**2.217**	0.094	2.429	**0.086**	2.237
MH_03	0.107	**1.789**	0.107	1.807	**0.104**	**1.789**



(a) Line based system (b) RPR based system

Fig. 6. The reconstruction results from line based and RPR based odometry system, respectively. The green line segments are the reconstructed line map, and the red dots are the estimated camera trajectory.

based VO in richly-textured environment is higher than that of line, which shows that the direction constraints of the line feature are not tighter than the point constraints. In contrast, the RPR feature makes full use of the intersection constraint and direction constraint of the straight lines, and obtains the highest overall accuracy.

Since the *it3f* dataset does not provide ground truth trajectory, we used this dataset to qualitatively analyze the reconstruction results of line and RPR. The reconstruction results of straight line map and camera trajectory are shown in the Fig. 6. It can be seen that the gap between the start and end of the trajectory estimated by the line based system is larger than that of the RPR based system, which means that the trajectory error is larger.

VI. CONCLUSIONS

In this paper, we introduce the novel geometric constraint of intersected lines, to improve the SLAM and 3D reconstruction systems. We firstly propose a minimum parameterization representation of RPR structure, which is more compact than the ordinary representation of two independent lines. We also derive the initialization and measurement model for RPR structure. Leveraging the perpendicular and intersecting relationship of two straight lines, an efficient absolute camera pose estimation algorithm is designed. A stereo odometer system that only involves RPR features is developed. A series of experiments are conducted on simulation and real datasets. The proposed method demonstrated the high efficiency and the accuracy improvement.

ACKNOWLEDGEMENT

We would like to thank Dr. Fangbo Qin for his suggestions on scientific writing. We also thank Xin Li for his help in the simulation code.

REFERENCES

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

[2] R. Gomez-Ojeda, F. Moreno, D. Zuiga-Nol, 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.

[3] 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.

[4] Y. Zhou, H. Qi, and Y. Ma, "End-to-end wireframe parsing," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[5] 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.

[6] 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.

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

[8] 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.

[9] Y. Lu and D. Song, "Visual navigation using heterogeneous landmarks and unsupervised geometric constraints," *IEEE Transactions on Robotics*, vol. 31, no. 3, pp. 736–749, 2015.

[10] X. Li, Y. He, J. Lin, and X. Liu, "Leveraging planar regularities for point line visual-inertial odometry," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[11] H. H. Chen, "Pose determination from line-to-plane correspondences: existence condition and closed-form solutions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 6, pp. 530–541, 1991.

[12] A. Elqursh and A. Elgammal, "Line-based relative pose estimation," in *IEEE Conference on Computer Vision Pattern Recognition*, 2011.

[13] H. Li, J. Zhao, J. Bazin, W. Chen, K. Chen, and Y. Liu, "Line-based absolute and relative camera pose estimation in structured environments," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 6914–6920.

[14] Y. Kuang and K. strm, "Pose estimation with unknown focal length using points, directions and lines," in *2013 IEEE International Conference on Computer Vision*, 2013, pp. 529–536.

[15] V. Larsson, M. Oskarsson, K. Astrom, A. Wallis, Z. Kukelova, and T. Pajdla, "Beyond grobner bases: Basis selection for minimal solvers," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3945–3954.

[16] C. Xu, L. Zhang, L. Cheng, and R. Koch, "Pose estimation from line correspondences: A complete analysis and a series of solutions," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 6, pp. 1209–1222, 2016.

[17] J. Zhao, L. Kneip, Y. He, and J. Ma, "Minimal case relative pose computation using ray-point-ray features," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 5, pp. 1176–1190, 2020.

[18] J. Ma, X. Wang, Y. He, X. Mei, and J. Zhao, "Line-based stereo SLAM by junction matching and vanishing point alignment," *IEEE Access*, vol. 7, p. 181800, 2019.

[19] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, 2012, pp. 573–580.

[20] Z. Kukelova, J. Heller, and A. Fitzgibbon, "Efficient intersection of three quadrics and applications in computer vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1799–1808.

[21] 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.