Plane-Aided Visual-Inertial Odometry for 6-DOF Pose Estimation of a Robotic Navigation Aid

HE ZHANG AND CANG YE (Senior Member, IEEE)
Computer Science Department, Virginia Commonwealth University, Richmond, VA 23284, USA
Corresponding author: Cang Ye (e-mail: cye@vcu.edu)

This work was supported by the NIBIB and the National Eye Institute of the NIH under Award R01EB018117.

ABSTRACT The classic visual-inertial odometry (VIO) method estimates the 6-DOF pose of a moving camera by fusing the camera’s ego-motion estimated by visual odometry (VO) and the motion measured by an inertial measurement unit (IMU). The VIO attempts to updates the estimates of the IMU’s biases at each step by using the VO’s output to improve the accuracy of IMU measurement. This approach works only if an accurate VO output can be identified and used. However, there is no reliable method that can be used to perform an online evaluation of the accuracy of the VO. In this paper, a new VIO method is introduced for pose estimation of a robotic navigation aid (RNA) that uses a 3D time-of-flight camera for assistive navigation. The method, called plane-aided visual-inertial odometry (PAVIO), extracts planes from the 3D point cloud of the current camera view and track them onto the next camera view by using the IMU’s measurement. The covariance matrix of each tracked plane’s parameters is computed and used to perform a plane consistent check based on a chi-square test to evaluate the accuracy of VO’s output. PAVIO accepts a VO output only if it is accurate. The accepted VO outputs, the information of the extracted planes, and the IMU’s measurements over time are used to create a factor graph. By optimizing the graph, the method improves the accuracy in estimating the IMU bias and reduces the camera’s pose error. Experimental results with the RNA validate the effectiveness of the proposed method. PAVIO can be used to estimate the 6-DOF pose for any 3D-camera-based visual-inertial navigation system.

INDEX TERMS Visual-inertial odometry, VO/SLAM, pose estimation, robotic navigation aid.

I. INTRODUCTION Visual impairment reduces a person’s independent mobility and severely deteriorates the quality of life. According to the World Health Organization, there are about 285 million people with visual impairment, of which 39 million are blind. Because age-related diseases are the leading causes of vision loss, the visually impaired community is growing as the population continues to age. Therefore, there is a pressing need to develop new mobility tools that may help the visually impaired move around independently. In the literature, a number of Robotic Navigation Aids (RNAs) [1]–[8] have been introduced to guide the blind in indoor environments. Among these RNAs, vision-based systems are becoming more and more popular because the cameras used in these RNAs, such as monocular camera [2], [9] stereo cameras [3], [4], [10], RGB-D cameras [5], [6], [11] and 3D time-of-flight (TOF) cameras [7], [8], can provide better navigational information, including 6-DOF device pose (position and orientation) and object-level information. The pose information can be used to build a 3D map of the environment, locate a blind traveler in the environment, and guide the traveler to the destination. The problem of camera pose estimation is also known as visual simultaneous localization and mapping (SLAM). The state-of-the-art visual SLAM algorithms [12]–[15] have demonstrated their effectiveness in existing research. However, they are effective only in a feature-rich environment. Their performances may be compromised when the operating environment is feature-sparse. To address this problem, two approaches have been taken in the literature. The first approach is to use the geometric features (e.g., planes, lines) of the operating environment to limit the pose estimation error because they are ubiquitous in a man-made environment. Plane features have been used in EKF-SLAM [16] and pose-graph SLAM [17]–[19] to reduce pose error accumulated by visual odometry (VO). In [7] and [20], floor plane and wall lines were extracted from the point cloud data of a 3D TOF camera and used to
reduce the pose error of an RNA for wayfinding in an indoor environment. However, the available geometric features in some cases may be inadequate for a full correction to the 6-DOF pose. For instance, there are often only one or two planes that are observable when using the RNA [7], [20] in an indoor environment. In this case, only part of the 6-DOF pose estimation error can be corrected because at least three intersecting planes are needed for a full pose error correction. The second approach is to use an additional sensor, usually an inertial measurement unit (IMU) or a gyro, and perform sensor fusion to reduce pose estimation error. In the robotics community, the combination of a camera and an IMU has become standard equipment for SLAM. The Camera-IMU-based SLAM method is termed visual-inertial odometry (VIO). The state-of-the-art VIO [21]–[25] uses VO to periodically update the estimates of the IMU’s biases and integrate the IMU’s measurements for pose estimation when VO malfunctions. A more efficient way to incorporate the data is to fuse the IMU’s measurements with the visual observations by incremental smoothing. However, VO can fail or produce a large pose error in a feature-sparse environment. In either case, the estimates of the IMU biases can become inaccurate, resulting in a large pose estimation error. Although a VO failure may be detected, it is difficult to assess VO accuracy online. To address this issue, we propose a new method, called Plane-Aided VIO (PA VIO), for pose estimation of an RNA that uses a 3D TOF camera and an IMU for wayfinding. The method extracts plane features from the camera’s point cloud data and tracks these features over the camera’s data frames to associate plane features between the frames. It then uses the plane association information to reduce pose estimation error by minimizing the residual of a factor graph that integrates the VO-estimated pose changes (PCs), the IMU-estimated PCs, and the information of the associated planes. The main contributions of this paper include:

- We propose a plane-aided VIO method that improves pose estimation accuracy by tightly coupling the observed planes with the VO-computed pose change estimates and the IMU’s measurements in a graph optimization framework.
- We propose a plane-consistency check (PCC) method to detect and reject inaccurate pose change estimates by the VO. The PCC prevents the proposed VIO method from using inaccurate VO data and therefore its use results in more accurate pose estimation.
- We implemented the proposed plane-aided VIO method on the visual-inertial navigation system of an RNA prototype for assistive navigation of the visually impaired and carried out experiments in real world environments. The results validate its practicality for the wayfinding application.

II. OVERVIEW OF THE RNA
The RNA prototype that we have fabricated is depicted in Fig. 1. It is a computer-vision-enhanced white cane that allows a blind traveler to “see” the surrounding environment. It provides the user with the desired travel direction in an intuitive way and offers a friendly human-device interface. The computer vision system uses a 3D TOF camera (SwissRanger SR4000 of Mesa Imaging) for 3D perception, an IMU (VN-100 of VectorNav Technologies, LLC) for motion measurement and an Up Board computer for computation. The SR4000 illuminates the environment with modulated infrared light and measures range up to 5 meters (accuracy: ∼1 cm, resolution: 176 × 144) by using phase shift measurement. The camera produces range, intensity and confidence data at a rate up to 54 frames per second. The SR4000 has a much better range measurement accuracy for a distant objects and has better data completeness than a stereo camera. This may result in better pose estimation performances. The IMU measures the angular velocity and acceleration at 200 Hz. All software codes are implemented with the Up Board. The system provides both pose estimation and obstacle detection functions. The results are used to guide the blind traveler for wayfinding and obstacle avoidance. The RNA has an active rolling tip (ART) that can steer itself into the desired travel direction. A speech interface using a Bluetooth headset is used for human-device interaction. The ART (the bottom image in Fig. 1) is an assembly of a servo motor, an encoder, a gear head, and an electromagnetic clutch that is connected to a ball-bearing-supported rolling tip by a flexible coupling. A servo controller and custom microcontroller boards are used to control the motor and the clutch, respectively. Both controller boards are connected to the UP Board’s USB ports.

III. NAVIGATION SYSTEM AND NOTATION

In this paper, a new VIO method is developed to estimate the IMU’s pose change by incorporating the IMU’s measurements, pose changes estimated by the SR4000-based VO [26], and geometry features (i.e., planes) extracted from the camera’s point cloud data. The VO estimates the pose...
change between two keyframes. The first data frame of the camera is taken as the first keyframe. A subsequent keyframe is defined as one with a substantial translation (>0.1 meters) or rotation (>2°) from the previous keyframe. The ith keyframe contains intensity image i (called image i for simplicity) and depth image i. There are multiple IMU measurements between two keyframes. These inertial measurements are integrated to produce another pose change estimate (PCE). The two PCEs together with the plane features extracted from the point cloud data are used to create a factor graph [27] for estimating the IMU’s pose in the world coordinate system. The use of the plane features helps to reduce pose estimation error. The proposed method is detailed in sections III and IV.

The coordinate systems of the IMU and the camera denoted \(X_iY_iZ_i\) and \(X_jY_jZ_j\), respectively, are defined in Fig. 1. The world coordinate system \(X_wY_wZ_w\) is defined as the IMU’s coordinate system at the first keyframe. Throughout this paper, a matrix is represented by an upper case letter and a vector a bold lowercase letter. The right superscript of a letter represents the coordinate system, in which the variable is expressed, whereas a left superscript indicates the type of the variable. For example, the camera’s transformation matrix from keyframe i to keyframe j is denoted by \(T_{ij}^w\). Similarly, the transformation matrix from \(X_wY_wZ_w\) to \(X_jY_jZ_j\) at keyframe i is denoted \(T_{iw}^w\). We use Riemannian geometry notation to describe the pose of a rigid body. Let the IMU’s pose at keyframe i be denoted \(\xi_i^w = [(\omega_i^w) \cdot (t_i^w)]^T\), where \(\omega_i^w \in \mathbb{R}^3\) and \(t_i^w \in \mathbb{R}^3\) are the IMU’s orientation and position in \(X_wY_wZ_w\), respectively. Similarly, we use \(\xi_{ij}^w = [(\omega_{ij}^w) \cdot (t_{ij}^w)]^T\) to denote the IMU’s pose change from keyframe i to keyframe j and \(T_{ij}^w\) to denote the corresponding transformation matrix. The relation between \(\omega_i^w\) and the rotation matrix \(R_{iw}^w \in SO(3)\) is determined by the exponential/logarithm map [25]. Specifically, \(R_{iw}^w = \exp((\omega_i^w)^\land)\) and \((\omega_i^w)^\land = (\log(R_{iw}^w))^\lor\), where \((\omega_i^w)^\land \in \mathfrak{so}(3)\) is the corresponding Lie algebra element of \(\omega_i^w\) the hat operator \(\land\) maps a vector to a skew-symmetric matrix in \(\mathfrak{so}(3)\), and the vee operator \(\lor\) indicates the inverse operation. The transformation matrix from \(X_iY_iZ_i\) to \(X_wY_wZ_w\) at keyframe i is computed from \(\xi_i^w\) by

\[
T_{iw}^w = \begin{bmatrix} R_{iw}^w & t_i^w \end{bmatrix} = \begin{bmatrix} \exp((\omega_i^w)^\land) & t_i^w \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}
\]

The IMU’s pose \(\xi_{ij}^w\) at keyframe j can be computed from \(T_{iw}^w\) by

\[
\xi_{ij}^w = \begin{bmatrix} \omega_{ij}^w \\ t_{ij}^w \end{bmatrix} = \begin{bmatrix} (\log(\exp((\omega_i^w)^\land))) \exp((\omega_{ij}^w)^\land) \end{bmatrix} \begin{bmatrix} R_{iw}^w & t_i^w \end{bmatrix}^{-1} t_i^w \]

Let \(x_i^w = [(\xi_i^w) \cdot (v_i^w)]^T\) denote the IMU’s state and \(b_i^w = [(c_b^i) \cdot (c_v^i)]^T\) its calibration parameters, where \(v_i^w \in \mathbb{R}^3\) is the IMU’s linear velocity, and \(c_b^i \in \mathbb{R}^3\) and \(c_v^i \in \mathbb{R}^3\) are the bias of the gyroscope and accelerometer, respectively.

At keyframe i, K planes are detected and described in \(X_iY_iZ_i\) as \(l_{ik} = [n_{ik}^T, d_{ik}^T]^T\) \((k = 1, \ldots, K)\), where \(n_{ik}^T\) and \(d_{ik}\) are the plane’s normal vector and the distance from the origin of \(X_iY_iZ_i\) to the plane, respectively. The representation of the plane in \(X_wY_wZ_w\) is given by

\[
l_{ik}^* = \hat{g}(T_{iw}^w, l_{ik}) = \begin{bmatrix} n_{ik}^w \\ d_{ik}^w \end{bmatrix} = \begin{bmatrix} (R_{iw}^w)^T n_{ik}^w + d_{ik} \end{bmatrix}
\]

IV. FACTOR GRAPH FORMULATION

In this paper, we use a factor graph to model the SLAM problem. A factor graph [27], [28] is a bipartite graph consisting of nodes and edges. There are two types of nodes: variable nodes and factor nodes. A variable node represents the random variables to be estimated and a factor node encodes a measurement model defined by a probabilistic distribution function (PDF) of these variables. We denote the set of variables up to \(M\) keyframes by \(\Theta = \{X, B, C\}\), where \(X = \{x_i^w\}, B = \{b_i^w\}\) and \(C = \{l_{ik}^*\}\) for \(i = 1, \ldots, M\), \(k = 1, \ldots, K\). The graph is denoted by \(G = (F, \Theta, E)\), where variable node \(\theta_i \in \Theta\) represents the variables to be estimated; factor node \(f_i \in F\) represents the measurements; and edges \(e_{ij} \in E\) represents the connection/relation between a factor node and a variable node. Graph \(G\) defines a function given by

\[
F(\Theta) = \prod_i f_i(\theta_i)
\]

where each factor \(f_i\) is a function of \(\theta_i\) and \(\Theta = \{\theta_i\}\) is the set of variables involved in factor \(f_i\). The dependence relationships between the nodes are encoded by edges \(e_{ij}\). By using a probabilistic representation for \(f_i\) and assuming Gaussian transformation models,

\[
f_i(\theta_i) \propto \exp(-\frac{1}{2} r_i^2)
\]

Here, \(r_i^2 = e_i^T \Sigma^{-1} e_i\) is the squared Mahalanobis distance. \(e_i = h_i(\theta_i) - z_i\) is the residual, where \(h_i(\theta_i)\) is a measurement prediction function and \(z_i\) is the actual measurement. \(\Sigma_i\) is the covariance matrix for \(e_i\). The optimal value \(\Theta^*\) that maximizes \(F(\Theta)\) is given by:

\[
\Theta^* = \arg\max_{\Theta} \left( -\sum_i \log F(\Theta) \right) = \arg\min_{\Theta} \left( \sum_i r_i^2 \right)
\]

The nonlinear LS problem cannot be solved directly but requires a numerical solution that starts with an initial estimate \(\hat{\Theta} = \{\theta_i\}_{i=1}^{M}\) and successively approximate the optimal solution. The Levenberg-Marquardt (LM) algorithm [29] is employed to solve the problem. The LM method iterates the following steps until the optimal \(\Theta^*\) is found: 1) approximates residual \(e_i\) around \(\hat{\Theta}\) by \(e_i(\hat{\Theta} + \delta\Theta) \approx e_i(\hat{\Theta}) + J_i^T \delta\Theta\), where \(J_i\) is the Jacobian of \(e_i(\hat{\Theta})\) computed at \(\hat{\Theta}\); 2) linearizes the \(F(\Theta)\) by using the approximated \(e_i\) and find \(\delta\Theta^*\) that minimizes \(F(\Theta^*); 3) updates \(\Theta\) by \(\Theta = \Theta + \delta\Theta^*\). The iterative process terminates when \(\delta\Theta^*\) approaches 0. And the resulted \(\hat{\Theta}\) is taken as the optimal solution.
There are four types of variable nodes and four types of factor nodes. The four variable nodes are related to the variables for the pose, velocity, IMU bias, and planes. For keyframe i with K planes observed, these variable nodes are denoted by $\xi^{w}_{i1}$, $v^{w}_{i1}$, $b^{w}_{i1}$, and $i^{w}_{ik}$ ($k = 1, \ldots, K$), respectively. Our goal is to estimate the values of these variables that minimize the cost function of the graph. The prior factor node reflects the prior knowledge on the device’s initial pose and it fixes the first pose node in the graph. Each subsequent pose node is added by using the IMU preintegration as the PCE. The other three factor nodes represent the measurements obtained from IMU preintegration, VO, and plane-fitting. An IMU preintegration factor constrains the estimates for the pose change, velocity change and IMU bias drift. Therefore, it connects the two neighboring pose nodes, velocity nodes, and IMU bias nodes. A VO factor constrains the PCE between the two pose nodes and therefore it connects them. A factor graph example that consists of four keyframes ($i = 1, \ldots, 4$) and two planes ($I^{w}_{21}$ and $I^{w}_{22}$) observed at keyframe 2 is shown in Fig. 2. In this paper, we propose to evaluate the accuracy of the VO-computed PCE by checking the consistency of plane observation (see section V). If a PCE fails the consistency check, no VO factor node will be created. Instead, a plane factor is added to the graph to constrain the PCE. A plane factor involves a plane node and a pose node at which the plane is observed. By connecting each variable node to one or more factor nodes, the graph structure encodes a joint probability density function $F(\Theta^{*})$ of these variables. The LS problem (6) is given by

$$\Theta^{*} = argmin_{\Theta} \left( \alpha_{ij}^2 + \sum_{ij} \epsilon r^2_{ij} + \sum_{i} \epsilon r^2_{i,i+1} + \sum_{ik} \epsilon r^2_{ik} \right) \tag{7}$$

where $\alpha r^2_{ij}, \epsilon r^2_{ij}, \epsilon r^2_{i,i+1},$ and $\epsilon r^2_{ik}$ are the squared Mahalanobis distance related to the factors of prior, VO, IMU and plane measurement, respectively. The computation of $\alpha r^2_{ij}$ for the prior factor and $\epsilon r^2_{i,i+1}$ for the IMU factors $F^{I}$ follows Forster’s VIO method [25]. The error functions and covariance matrices related to the VO factor $F^{O}$ and $F^{L}$ are detailed as follows.

A. VISUAL ODOMETRY FACTOR $F^{O}$

The VO estimates the PC between keyframes $i$ and $j$ by extracting visual features from image $i$ and tracking them onto image $j$. SIFT features [30] are used in this work. Note that we use the keypoints only to avoid the computation of the descriptors and keep the computational cost low. Given the IMU-measured PC, $I^{w}_{ij}$, a space correspondence search is employed to find the matched features between the two images and the RANSAC method [26] is used to identify the inliers, from which PCE is computed. Finally, we follow the method in [31] and employ Bundle Adjustment (BA) to refine the PCE $T^{w}_{ij}$ and computes its covariance $\Sigma^{w}_{ij}$. From our experiments, we observed that the PCE produced by BA could be inaccurate even when it was computed from a good number of inliers. Fig. 3 shows a case with 22 inliers. The point cloud data of the two keyframes are registered by using the estimated PC. The misalignment of the point cloud surfaces (as well as the ground surfaces) indicates an inaccurate PCE. In this work, we propose to use the associated planes between the two keyframes to evaluate the accuracy of the BA-estimated PC and determine the acceptance/rejection of the VO-computed PCE for graph construction. The proposed VO method is illustrated in Algorithm 1. The method computes $[T^{w}_{ij}, \Sigma^{w}_{ij}]$ (the PCE and the covariance matrix) and $L$ (the associated plane-pairs) between keyframes $i$ and $j$. If it returns 0 (i.e., the method successfully results in an accurate PCE), then $[T^{w}_{ij}, \Sigma^{w}_{ij}]$ is used to construct the VO factor in the graph and $L$ is not used to create plane node factors. Otherwise, $L$ is used to add plane nodes and construct plane factors in the graph but no VO factor is created. The PlaneDetection function extracts planes from keyframe $j$ (after subtracting the point cloud data of the tracked planes) by using the clustering approach in [32]. It is noted that the algorithm returns 1 (failure) if the inlier number $N$ is smaller than 12. This is because that the SR4000-based VO requires at least 12 matched points to compute an accurate PCE [16]. The rest of the algorithm is self-explanatory.
Based on the VO output \( T^c_{ij} \), the IMU’s PC measurement can be computed by \( T^s_{ij} = (T^{cs})^{-1} T^c_{ij} T^{cs} \), where \( T^{cs} \) (the transformation matrix from \( X_s, Y_s, Z_s \) to \( X_i, Y_i, Z_i \)) is known a priori by calibration. The predicted IMU’s PC is given by \( \hat{T}^s_{ij} = (\hat{T}^{ws})^{-1} \hat{T}^s_{ij} \), where \( \hat{T}^{ws} \) and \( \hat{T}^s_{ij} \) are the transformation matrices corresponding to the estimated IMU poses at keyframes \( i \) and \( j \). The residual vector for factor \( x_O \) is then defined by \( e_{ij} = \left( \nabla o_{ij}, \nabla t_{ij} \right)^T \) with \( \nabla o_{ij} = \left( \log([R^i]^{-1} R^j \left| \Sigma^o_{ij} \right|^{-1} R^j R^i \right)^\top \). \( R \) and \( t \) represent the transformation matrix and translation vector, respectively. The squared Mahalanobis distance for the visual odometry factor can be computed as \( c_{r^2} = e_{ij}^T \left( \Sigma^o_{ij} \right)^{-1} e_{ij} \), where \( \Sigma^o_{ij} = (R^s)^T \Sigma^s_{ij} R^c \).

**B. PLANE FACTOR \( x_L \)**

For each keyframe, the clustering approach in [33] was used to extract planes from the SR4000’s depth data. The \( k \)th plane detected at keyframe \( i \) is denoted by \( T^c_{ik} = [n^c_{ik}, d^c_{ik}]^T \). Its covariance matrix is denoted \( \Sigma^c_{ik} \) is computed by forward-propagating the covariances of the points on the plane [20]. This method requires the numerical computation of the Jacobians for the plane’s parameters \( (n^c_{ik}, d^c_{ik}) \) with respect to each point of the plane and is thus time-consuming or even infeasible in case of a large number of points. To tackle this issue, we generate a linear plane function based on the inverse depth parameterization [34] and then analytically compute \( \Sigma^c_{ik} \) from the function by using the theory of linear regression [35]. The plane equation

\[
 n_x x + n_y y + n_z z = d
\]

is nonlinear in terms of the plane parameter \( [n, d]^T = [n_x, n_y, n_z, d]^T \), where \( n_z = \sqrt{1 - n_x^2 - n_y^2} \). Equation (8) can be written as

\[
 \frac{n_x x}{d} + \frac{n_y y}{d} + \frac{n_z}{d} = \frac{1}{z}
\]

By substituting \( u = \frac{x}{z}, v = \frac{y}{z}, a = \frac{n_x}{d}, b = \frac{n_y}{d}, c = \frac{n_z}{d} \), and \( q = \frac{1}{z} \) into equation (9), we have

\[
 au + bv + c = q
\]

Equation (10) is a linear linear function in terms of parameters \( \beta = [(a, b, c)]^T \). The implicit constraints are \( d \neq 0 \) and \( z \neq 0 \). In our application, these conditions are always met because a plane extracted from the SR4000’s data does not pass through the optical center \( d \neq 0 \), and all data points have a positive \( z \) value.

The plane fitting problem is to minimize the squared error \( E(z, \beta) = (\beta - Q)^T (\beta - Q) \), where the input data with \( M \) observations are \( z = [(u_1, v_1, q_1), (u_2, v_2, q_2), \ldots, (u_M, v_M, q_M)] \). \( A = [(u_1, v_1, 1)^T, (u_2, v_2, 1)^T, \ldots, (u_M, v_M, 1)^T] \), \( \beta = [(a, b, c)]^T \). \( Q = [q_1, q_2, \ldots, q_M]^T \).

According to the linear regression theory [35], the optimal estimate for \( \hat{\beta} \) and the covariance matrix \( \hat{C}(\hat{\beta}) \) is given by

\[
 \hat{\beta} = (A^T A)^{-1} A^T Q
\]

and

\[
 \hat{C}(\hat{\beta}) = (A^T A)^{-1} \sigma^2
\]

where \( \sigma^2 \) is the variance of the plane-fitting error. An unbiased estimate of \( \sigma^2 \) is:

\[
 \sigma^2 = \frac{E(z, \hat{\beta})}{M - 3}
\]

is the dimensionality of the plane. The plane’s parameters can be computed by \( d = 1/\beta \) and \( n = \hat{\beta}/\beta \). The plane’s covariance matrix can be computed by forward propagation and is given by

\[
 C(I) = \begin{bmatrix} I_3 & -\hat{\beta} d^3 \\ -\hat{\beta} d^3 & -\hat{\beta} d^3 \end{bmatrix} C(\hat{\beta}) \begin{bmatrix} I_3 & -\hat{\beta} d^3 \\ -\hat{\beta} d^3 & -\hat{\beta} d^3 \end{bmatrix}^T
\]

Therefore, the covariance matrix of the \( k \)th plane at keyframe \( i \) is given by \( \Sigma_{ik} = C(I_{ik}) \). Since the factor graph optimization is performed in the variables’ tangent spaces, we transform \( \Sigma_{ik} \) to its tangent space by

\[
 S_{ik} = \begin{bmatrix} B_{ik}^T & 0 \\ 0 & 1 \end{bmatrix} \Sigma_{ik} \begin{bmatrix} B_{ik}^T & 0 \\ 0 & 1 \end{bmatrix}^T
\]
The computation of $B_{ij}^{c}$ is given in Appendix A. The actual measurement of the plane’s parameters and its covariance matrix in $X_i, Y_i, Z_i$ can be computed by $T_i^{c} = \tilde{g}(T_{i}^{w}, I_i^{c})$ and $S_{ik}^{c} = J_{sc}S_{ik}^{c}J_{sc}^{T}$, where $J_{sc}$ is given by

$$J_{sc} = \begin{bmatrix} B_{nk}^{cT}(R_{c}^T)^T B_{nk}^{cT} & B_{nk}^{cT}t_{c}^T & 1 \end{bmatrix}$$

The predicted measurement of the $k^{th}$ plane at keyframe $i$ in $X_i, Y_i, Z_i$ is given by $I_{ik}^{c} = \tilde{g}(T_{i}^{w}, I_i^{c})$, where $T_{i}^{w}$ is the transformation matrix from $X_i, Y_i, Z_i$ to $X_i, Y_i, Z_i$ and $I_{ik}^{c}$ is the estimated $k^{th}$ plane in $X_i, Y_i, Z_i$. In fact, $I_{ik}^{c}$ inherits the plane parameters from the previous step (i.e., $I_{ik}^{c} = T_{i}^{wI_{ik}^{c}}$). Using the Geodesic distance [36], the residual vector of the plane factor is defined by

$$e_{ik} = [\rho_{ik}^{T}, \delta_{ik}^{T}]^{T}$$

where $\rho_{ik} = B_{nk}^{cT}(R_{c}^T)^T(n_{ik}^{c} - \hat{n}_{ik}^{c}\cos(\theta))$, with $\theta = \cos^{-1}((\hat{n}_{ik}^{c})^{T}n_{ik}^{c})$. Then, the Mahalanobis distance for the plane factor can be computed as $(r_{ik}^{c})^{2} = e_{ik}^{T}(S_{ik}^{c})^{-1}e_{ik}$. 

### V. PLANE ASSOCIATION AND CONSISTENCY CHECK

#### A. PLANE ASSOCIATION

Assuming $T_{ij}^{c}$ (the camera’s PC measured by the IMU) is accurate, $I_{ik}^{c}$ (the $k^{th}$ plane observed at keyframe $i$) can be tracked onto keyframe $j$ as $\hat{I}_{jk}^{c} = [\hat{n}_{jk}^{c}, \hat{d}_{jk}^{c}]^{T} = \tilde{g}(T_{ij}^{c}, I_{ik}^{c})$ to speed up the search for its associated plane $I_{jk}^{c}$ at keyframe $j$. The distances between each data point of keyframe $j$ and $I_{jk}^{c}$ is computed. Data points with a distance below $2\sigma_{jk}$ are treated as candidate data points of the plane $I_{jk}^{c}$. If the plane $I_{jk}^{c}$ is the plane-fitting error for $I_{jk}^{c}$ because plane $I_{jk}^{c}$ is in the neighborhood of $I_{jk}^{c}$. In this work, a RANSAC plane-fitting process is used to extract the plane $I_{jk}^{c}$ from the candidate data points. After the RANSAC process, $I_{jk}^{c}$ is extended by adding data points satisfying the distance criteria. Finally, the plane’s parameters are recomputed by following the process in section III.B. If the number of data points of $I_{jk}^{c}$ is above a threshold, $\{I_{jk}^{c}, I_{jk}^{c}\}$ is taken as a pair of associated/tracked planes.

#### B. PLANE CONSISTENCY CHECK (PCC)

Plane consistency check (PCC) is to evaluate, based on the VO-computed PCE, how good the match between $I_{ik}^{c}$ and $I_{jk}^{c}$ is. The evaluation result determines if the VO-computed pose is accurate for graph construction. Given the VO-computed PCE $T_{ij}^{c}, \Sigma_{ij}^{c}$, the predicted plane for $I_{ik}^{c}$ in keyframe $j$ is $\hat{I}_{jk}^{c} = [\hat{n}_{jk}^{c}, \hat{d}_{jk}^{c}]^{T} = \tilde{g}(T_{ij}^{c}, I_{ik}^{c})$. The goodness of the match between $I_{ik}^{c}$ and $I_{jk}^{c}$ is evaluated by using the angle between $\hat{n}_{ik}^{c}$ and $\hat{n}_{jk}^{c}$ and $\delta_{i} = \hat{d}_{jk}^{c} - \hat{d}_{ik}^{c}$. If the angle is larger than $5^\circ$ or $\delta_{i} > 0.15 m$, the PCE is inaccurate and thus rejected. If the angle is less than $0.5^\circ$ and $\delta_{i} < 0.01 m$, the PCE is accurate and accepted.

For the rest, we compute the covariance of $\hat{n}_{jk}^{c}$ by

$$\hat{S}_{jk}^{c} = J_{lk}^{c}\Sigma_{lk}^{c}J_{lk}^{cT} + J_{lk}^{c}S_{ik}^{c}J_{lk}^{cT}$$

where $J_{lk}^{c}$ and $J_{lk}^{c}$ are the Jacobians of $\hat{n}_{jk}^{c}$ with respect to $\xi_{ij}^{c}$ and $n_{ik}^{c}$, respectively. Computations of $J_{lk}^{c}$ and $J_{lk}^{c}$ are given in Appendix B. The consistency between the predicted plane $\hat{I}_{jk}^{c}$ and the associated plane $I_{jk}^{c}$ is evaluated based on the error vector defined in equation (17). When $\theta$ (the angle between normal vectors $\hat{n}_{jk}^{c}$ and $n_{ik}^{c}$) is small, the difference between them can be approximated by $\eta_{k} = B_{n_{jk}^{c}, n_{ik}^{c}}$ and the vector of distance error for the plane pair $\{I_{jk}^{c}, I_{jk}^{c}\}$ is computed by

$$e_{k} = [\eta_{k}, \delta_{j}] \approx [B_{n_{jk}^{c}, n_{ik}^{c}}]$$

The covariance matrix of $e_{k}$ is computed by

$$\Sigma_{ek} = J_{el}^{c}\hat{S}_{jk}^{c}J_{el}^{cT} + J_{el}^{c}S_{ik}^{c}J_{el}^{cT}$$

where $J_{el}^{c}$ and $J_{el}^{c}$ are the Jacobians of $e_{k}$ with respect to the parameters of $\hat{I}_{jk}^{c}$ and $I_{jk}^{c}$, respectively. The computations of $J_{el}^{c}$ and $J_{el}^{c}$ are given in Appendix B. Then, for the $k^{th}$ tracked plane-pair, the Mahalanobis distance is computed by $r_{k}^{2} = e_{k}^{T}\Sigma_{ek}^{-1}e_{k}$. If $K$ tracked plane-pairs are found between keyframes $i$ and $j$, the PCE is evaluated by using the maximum value of $r_{k}^{2}$, i.e., $E_{r} = \max(r_{k}^{2})$. If $E_{r} < \chi_{k,0.9}^{2} = 6.25$, the PCE of VO is accepted. Otherwise, it is rejected.
FIGURE 4. Experiments with the RNA in various indoor environments: The trajectories produced by VINS-Fusion, VIO (N = 12), VIO (N = 20), VIO (N = 30), PAVIO w/o PCC, and PAVIO are plotted in black, red, blue, green, purple, and yellow, respectively, over the 3D point cloud map built by using the PAVIO-computed poses. The yellow dot shows the location where the snapshot was taken.

(g) Dataset 7: Hallway and Stairway. The subject walked on the 2nd floor, then took stairways 1,2 and went upstairs to the 3rd floor. He walked in the hallway of the 3rd floor and took stairways 3,4 to return to the starting point.
function on the same datasets. The use of the percentage EPEN allows us to compute the statistical results (the mean and standard deviation of EPEN over the seven datasets) of each method and compare their overall performances (the last row of Table 1).

Since VINS-Fusion diverged for datasets 3-7, only the results on datasets 1 and 2 are included. From Table 1, it can be observed that: (1) PAVIO outperforms the other methods for each dataset; (2) PAVIO has an overall much better performance (mean EPEN: 2.47%) in pose estimation than the VIO method (mean EPEN: 3.75%) and VINS-Fusion (mean EPEN: 8.1%); (3) The use of the PCC greatly improves the pose estimation accuracy (reduce the EPEN more than 24.9% on average); (4) PAVIO has better repeatability in pose estimation (standard deviation of EPEN: 1.03%) than VIO (standard deviation: 1.74%), meaning that it is more robust to the variables of operating environments. As shown in Table 1, the VIO’s performance may be improved by adjusting the value of \(N\). However, PAVIO still exhibits a better overall performance in terms of pose estimation accuracy and robustness) than VIO. The results in Table 1 show that the VIO’s performance can be improved by decreasing the inlier threshold (\(N\) value). A smaller \(N\) increases the total number of VO outputs and thus increases the number of VO factor nodes. This may result in a better VIO-estimated pose at the beginning. However, it decreases the VO-computed PCE accuracy overall and may deteriorate the VIO-computed pose at some points. It is not possible to find an appropriate \(N\) a priori because the VO’s accuracy can be affected not only by \(N\) but also by the accuracy of the visual features’ depth data (the larger the depth the lower the accuracy). The PAVIO method addresses this problem by using a relatively smaller threshold (\(N = 12\)) and employing the PCC to evaluate the VO-computed PCE online and reject those inaccurate PCE values. VINS-Fusion has the worst pose estimation accuracy because its visual feature tracking method and outlier removal method are least accurate. Fig. 4 shows the trajectories generated by these methods over the point cloud map built by using the PAVIO-estimated poses for each experiment. It can be seen that the trajectories estimated by PAVIO are more accurate.

VII. CONCLUSION

We have presented a new visual-inertial odometry (VIO) method for pose estimation of a robotic navigation aid (RNA) that uses a 3D time-of-flight camera and an IMU for assistive navigation. The method, called plane-aided visual-inertial odometry (PAVIO), extracts planes from the 3D point cloud of the camera’s current frame and track them onto the next camera frame by using the IMU’s inertial measurement. The tracking result is used to evaluate the pose change estimate (PCE) of the visual odometry (VO) and accept the PCE only if it is accurate. The accepted VO outputs, the information of the extracted planes, and the IMU measurements over time are used to create a factor graph. By optimizing the graph, the method improves the accuracy in estimating the IMU bias and reduces the pose estimation error for the RNA. Experimental results with the RNA demonstrate that the proposed method outperforms the state-of-the-art VIO method in pose estimation in terms of both accuracy and repeatability. The proposed method can be used to produce a more accurate 3D map of the operating environment for object/obstacle detection and to locate a visually impaired traveler in the environment for assistive navigation. It can also be used for the 6-DOF pose estimation of any 3D-camera-based visual-inertial navigation system.

APPENDIX A

COMPUTATION OF \(B_n\)

A set of normal vectors form a smooth manifold—a 2-dimensional sphere \(S^2\). A normal vector \(n\) represents a point on the surface of \(S^2 \subseteq \mathbb{R}^3\), where \(n = [n_x, n_y, n_z]^T\) is a unit vector. Each point in the space of \(S^2\) corresponds to a normal vector of a plane. The tangent space \(T_n(S^2)\) at a point (corresponding to normal vector \(n\)) is composed of 3D vectors \(\phi : T_n(S^2) \ni \{\phi \in \mathbb{R}^3 | n^T \phi = 0\}\). It is a hyperplane in \(\mathbb{R}^3\) and it is in the null space of the normal vector \(n\). A basis than spans the tangent space \(T_n(S^2)\) is denoted as \((b_1, b_2)\).

It is computed as follows: \(b_1 = b_1' / \|b_1'\|\) and \(b_2 = b_2' / \|b_2'\|\), where \(b_1' = n \times a\), \(b_2' = n \times b_1\) and vector \(a\) is set to \([1, 0, 0]^T, [0, 1, 0]^T\) or \([0, 0, 1]^T\) if \(n_x\), \(n_y\) or \(n_z\) of \(n\) dominates the other two elements. The matrix \(B_n = [b_1 | b_2]\) is, therefore, the basis matrix for a normal vector \(n\).
APPENDIX B
COMPUTATIONS OF $J_{el}$ AND $J_{kl}$

Since the $k^{th}$ plane’s parameter from keyframe $i$ is transformed to keyframe $j$ using $\hat{T}_{jk} = \hat{g}(T_{ij}, T_{ik})$, we can derive the Jacobians $J_{\hat{n}}$ and $J_{\hat{b}}$ as follows:

$$J_{\hat{n}} = \frac{\partial \hat{T}_{jk}}{\partial \hat{n}_{jk}} = \begin{bmatrix} -B_{n,k,j}^{T} R_{c,j}^{T} \left[ n_{jk} \times \right] & 0 \\ 0 & \left[ n_{jk} \right]^{T} \end{bmatrix} \quad (21)$$

$$J_{\hat{b}} = \frac{\partial \hat{T}_{jk}}{\partial \hat{b}_{jk}} = \begin{bmatrix} B_{\hat{n},n,k}^{T} R_{c,j}^{T} B_{\hat{n},c,j}^{T} & B_{\hat{n},t,j}^{T} t_{j}^{T} \\ 0 & 1 \end{bmatrix} \quad (22)$$

where $[n]_{\times}$ is the skew matrix of $n$. Similarly, since the error vector is estimated in equation 19, the Jacobians $J_{el}$ and $J_{el}$ can be derived using the chain rule:

$$J_{el} = \frac{\partial e_{k}}{\partial \hat{n}_{jk}} = \begin{bmatrix} \left[ n_{jk} \right]^{T} (\frac{\partial \hat{b}_{1}}{\partial \hat{n}_{jk}}) & 0 \\ 0 & 0 \end{bmatrix} \quad (23)$$

$$J_{el} = \frac{\partial e_{k}}{\partial \hat{b}_{jk}} = \begin{bmatrix} \left[ n_{jk} \right]^{T} (\frac{\partial \hat{b}_{2}}{\partial \hat{b}_{jk}}) & 0 \\ 0 & 1 \end{bmatrix} \quad (24)$$

where $\frac{\partial \hat{b}_{1}}{\partial \hat{n}_{jk}} = J_{\hat{n},1,1} [\vec{n}]_{\times} B_{\hat{n},k}$. According to Appendix A, $B_{\hat{n},k} = [\hat{b}_{1} | \hat{b}_{2}]$ and $J_{\hat{n},1,1}$ is derived as:

$$J_{\hat{n},1,1} = \frac{\partial \hat{b}_{1}}{\partial \hat{n}_{jk}} = \frac{1}{|\hat{b}_{1}|^{1/2}} \begin{bmatrix} b_{x}^{0} + b_{x}^{2} & 0 & -b_{y}^{0} - b_{y}^{2} \\ b_{y}^{0} + b_{y}^{2} & 0 & -b_{x}^{0} - b_{x}^{2} \\ -b_{x}^{0} - b_{x}^{2} & b_{y}^{0} + b_{y}^{2} & -b_{z}^{0} - b_{z}^{2} \\ b_{x}^{0} + b_{x}^{2} & -b_{y}^{0} - b_{y}^{2} & b_{z}^{0} + b_{z}^{2} \end{bmatrix} \quad (25)$$

Likewise, the Jacobian $\frac{\partial \hat{b}_{2}}{\partial \hat{n}_{jk}}$ can be derived as $\frac{\partial \hat{b}_{2}}{\partial \hat{n}_{jk}} = J_{\hat{n},2,2} [\vec{n}]_{\times}$.
[29] J. J. Moré, “The Levenberg-Marquardt algorithm: Implementation and theory,” in *Numerical Analysis*. Berlin, Germany: Springer, 1978, pp. 105–116.

[30] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, Nov. 2004.

[31] S. Lynen, M. W. Achtelik, S. Weiss, M. Chili, and R. Siegwart, “A robust and modular multi-sensor fusion approach applied to MAV navigation,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Nov. 2013, pp. 3923–3929.

[32] J. W. Weingarten, G. Gruener, and R. Siegwart, “Probabilistic plane fitting in 3D and an application to robotic mapping,” in *Proc. IEEE Int. Conf. Robot. Autom.* (ICRA), vol. 1, Apr. 2004, pp. 927–932.

[33] C. Feng, Y. Taguchi, and V. R. Kamat, “Fast plane extraction in organized point clouds using agglomerative hierarchical clustering,” in *Proc. IEEE Int. Conf. Robot. Autom.* (ICRA), May 2014, pp. 6218–6225.

[34] T. J. J. Tang, W. L. D. Lui, and W. H. Li, “A lightweight approach to 6-DOF plane-based egomotion estimation using inverse depth,” in *Proc. Australas. Conf. Robot. Autom.*, 2011, pp. 1–10.

[35] S. M. Kay, *Fundamentals of Statistical Signal Processing*. Upper Saddle River, NJ, USA: Prentice-Hall, 1993.

[36] (Jun. 21, 2012). *Basic Differential Geometry*, [Online]. Available: http://ani.stat.fsu.edu/~anuj/CVPR_Tutorial/Part2.pdf

[37] T. Qin, J. Pan, S. Cao, and S. Shen, “A general optimization-based framework for local odometry estimation with multiple sensors,” 2019, arXiv:1901.03638. [Online]. Available: http://arxiv.org/abs/1901.03638

---

**HE ZHANG** received the B.E. degree from the China University of Mining and Technology (Beijing), Beijing, in 2009, and the Ph.D. degree from the University of Arkansas at Little Rock, Little Rock, in 2018. He is currently a Postdoctoral Researcher with the Department of Computer Science, Virginia Commonwealth University. His research interests include simultaneous localization and mapping, visual-inertial odometry, rehabilitation robotics, and 2D/3D computer vision.

**CANG YE** (Senior Member, IEEE) received the B.E. and M.E. degrees from the University of Science and Technology of China, Hefei, in 1988 and 1991, respectively, and the Ph.D. degree from the University of Hong Kong, Hong Kong, in 1999. He is currently a Professor with the Department of Computer Science, Virginia Commonwealth University. His research interests are in mobile robotics, computer vision, assistive technology, and intelligent systems.