An RGB-D Camera based Visual Positioning System for Assistive Navigation by a Robotic Navigation Aid

He Zhang, Lingqiu Jin, Cang Ye, *Senior Member, IEEE*

***Abstract*—There are about 253 million people with visual impairment worldwide. Many of them use a white cane and/or a guide dog as the mobility tool for daily travel. Despite decades of efforts, electronic navigation aid that can replace white cane is still research in progress. In this paper, we propose an RGB-D camera based visual positioning system (VPS) for real-time localization of a robotic navigation aid (RNA) in an architectural floor plan for assistive navigation. The core of the system is the combination of a new 6-DOF depth-enhanced visual-inertial odometry (DVIO) method and a particle filter localization (PFL) method. DVIO estimates RNA's pose by using the data from an RGB-D camera and an inertial measurement unit (IMU). It extracts the floor plane from the camera's depth data and tightly couples the floor plane, the visual features (with and without depth data), and the IMU's inertial data in a graph optimization framework to estimate the device's 6-DOF pose. Due to the use of the floor plane and depth data from the RGB-D camera, DVIO has a better pose estimation accuracy than the conventional VIO method. To reduce the accumulated pose error of DVIO for navigation in a large indoor space, we developed the PFL method to locate RNA in the floor plan. PFL leverages geometric information of the architectural CAD drawing of an indoor space to further reduce the error of the DVIO-estimated pose. Based on VPS, an assistive navigation system is developed for the RNA prototype to assist a visually impaired person in navigating a large indoor space. Experimental results demonstrate that: 1) DVIO method achieves better pose estimation accuracy than the state-of-the-art VIO method and performs real-time pose estimation (18 Hz pose update rate) on a UP Board computer; 2) PFL reduces the DVIO-accrued pose error by 82.5% on average and allows for accurate wayfinding (endpoint position error ≤ 45 cm) in large indoor spaces.**

***Index Terms*—visual positioning system, visual-inertial odometry, pose estimation, simultaneous localization and mapping, assistive navigation, robotic navigation aid.**

I. Introduction

According to Lancet Global Health [1], there are about 253 million people with visual impairment, of which 36 million are blind. Since age-related diseases (glaucoma, macular degeneration, diabetes, etc.) are the leading cause of vision loss and the world population is rapidly aging, more and more people become blind or visually impaired (BVI). Therefore, there is a crucial need for developing navigation aids to help the BVI with their daily mobility and independent lives. The problem of independent mobility of a BVI individual includes wayfinding and obstacle avoidance. Wayfinding is a global problem of planning and following a path towards the destination while obstacle avoidance is a local problem of taking steps without colliding, tripping, or falling.

To provide wayfinding and obstacle avoidance functions to a BVI traveler at the same time, the location of the traveler and the 3D map of the surroundings must be accurately acquired. The technique to address the problem is called simultaneous localization and mapping (SLAM). In the literature, several Robotic Navigation Aids (RNAs) [2], [3], [4], [5], [6], [7], [8] that use SLAM [9] for assistive navigation of the blind have been introduced. The performance of these RNAs relies on the pose estimation accuracy since the pose information is used to build a 3D map of the environment, locate the blind traveler in the environment, and guide the traveler to the destination. Stereo camera [2], [3] and RGB-D camera [4], [7] based visual SLAM methods have been developed to estimate the pose of RNA and detect surrounding obstacles from the generated 3D point cloud map. To improve the pose estimation accuracy, geometric features [5], [6], and inertial data [7], [8] have been utilized in existing SLAM methods. However, the pose error can still accrue over time and may become large enough (in case of a long trajectory) to break down RNA's navigation function. To tackle this problem, visual maps [10], [11], Bluetooth low energy beacons [12], radio-frequency identifications ([13], and near field communication tags [14] have been employed to correct accumulative pose estimation error. However, building a visual map ahead of time can be time-consuming as it requires extraction and storage of visual features point-by-point in a certain spatial interval to cover the entire navigational space, while the approach of using beacons or the alike requires re-engineering the environment and is thus not practical for assistive navigation.

To address these disadvantages, we propose, in this paper, a vision position system (VPS) that uses an RGB-D (i.e., color-depth) camera and an inertial measurement unit (IMU) to estimate the pose of an RNA for wayfinding applications. The system uses the floor plan (i.e., architectural CAD drawing) of an indoor space to reduce the accumulative pose estimation errors by a 2-step scheme. First, a new visual-inertial odometry (VIO) method is used to estimate 6-DOF RNA poses along the path. Second, a 3D point cloud map (local map) is built (by using the estimated poses) and projected onto the floor plane to

This work was supported by the NIBIB and the NEI of the National Institutes of Health under award R01EB018117. The content is solely the responsibility of the authors and does not necessarily represent the official views of the funding agencies.

H. Zhang, L. Jin and C. Ye are with the Computer Science Department, Virginia Commonwealth University, Richmond, VA 23284, USA. (email: hzhang8@vcu.edu, jinl@mymail.vcu.edu, cye@vcu.edu).

create a 2D map, which is then aligned with the floor plan by a particle filter localization (PFL) method (i.e., the 2D geometric features such as walls, doors, corners, and junctions of the two maps are aligned) to reduce RNA position and heading errors on the floor plan for wayfinding.

The RNA prototype uses a sensor suite consisting of an RGB-D camera and an IMU for localization, making the device an RGB-D-camera-based visual-inertial system (RGB-D VINS). A VINS employs a SLAM technique, known as VIO, to estimate the system's motion variables by jointly using its visual-inertial data. In [15], three state-of-the-art VIO methods, namely, OKVIS [16], VINS-Mono [17], and VIORB [18], are compared in the context of RNA pose estimation. The results show that VINS-Mono outperforms the other two. However, to enable real-time computation of VINS-Mono on the UP Board [39] computer used by the RNA, some modifications, such as using a constant inverse depth for each visual feature in the iterative optimization process, and reducing the size of the sliding-window, must be made to the algorithm/implementation. These modifications, however, trade the method's pose estimation accuracy for computational efficiency. To address the problem, we propose a so-called depth-enhanced visual-inertial odometry (DVIO) to estimate the RNA's pose for assistive navigation. DVIO is developed based on the framework of VINS-Mono and it improves VINS-Mono's pose estimation accuracy by 1) using the geometric feature (the floor plane extracted from the camera's depth data) to create additional constraints between the graph nodes to reduce the accumulative pose error; 2) using the depth data from the RGB-D camera for visual feature initialization and update to avoid iterative computation of the visual features' inverse depth in each step of the optimization process. Unlike VINS-Mono, DVIO does not need to estimate the metric scale, which is known from the depth data. As a result, it is free of pose estimation error induced by inaccurate scale estimation. Based on the DVIO-estimated egomotion, a PFL method is employed to determine the RNA's pose (3-DOF pose including position and heading) on the floor plan of the navigational space for wayfinding. The main contributions of this paper include:

- We propose a new VIO method, called DVIO, to estimate the 6-DOF pose of RNA. The method achieves better accuracy in pose estimation by using the depth data from an RGB-D camera.
- We introduce a VPS to estimate the RNA's 3-DOF pose on the floor plan for wayfinding. VPS employs PFL to estimate the pose based on the DVIO-estimated egomotion. PFL helps to improve pose estimation accuracy.
- We develop an assistive navigation system based on VPS and validate its efficacy by experiments with the RNA prototype in the real world.

II. Related Work

A. Related Work in VIO

Existing VIO methods can be classified into two categories, namely loosely-coupled [19], [20], [21] and tightly-coupled [22], [16], [17], [18]. In this section, we provide an overview of the tightly-coupled methods as the proposed DVIO falls into the same category. MSCKF [22] is an extended Kalman filter (EKF) based visual-inertial SLAM method. It utilizes IMU measurements to predict the filter state and employs visual feature measurements to update the state vector. Unlike a traditional EKF, it simultaneously updates multiple camera poses (in the state vector) by using a novel measurement model for the visual features. This model estimates a visual feature's 3D location by using its multi-view geometric constraint, computes the feature's reprojection residuals on multiple images, and use them as innovation to update the state vector. The method adopts a *delayed* state update strategy, i.e., a tracked visual feature is used to update the state vector only when it is no longer detected, to get the most from the multi-view constraint. In so doing, it uses much fewer visual features for state estimation as those features that are currently tracked are not used.

On the contrary, the smoothing-based VIO method [16], [17], [18] use all visual measurements of the related keyframes to estimate the current motion state and may achieve a more accurate result. OKVIS [16] is a smoothing-based VIO method that performs a nonlinear optimization by using a cost function consisting of the sensor measurements at several keyframes. Specifically, the cost function is formulated as the weighted sum of the residuals of the visual features' reprojections and the inertial measurements. This formulation incorporates all visual features' measurements and leads to better pose estimation accuracy than that of MSCKF [16]. OKVIS performs well on a stereo VINS. However, its performance may significantly degrade in the case of a monocular VINS. This is because it lacks a reliable approach to accurately estimating the initial values of the state variables (e.g., gyroscope bias, metric scale). Due to the non-convexity of the cost function, a poor estimate of the initial state will likely cause the optimization process to be stuck at a local minimum and result in an incorrect pose estimation. To mitigate this issue, VIORB [18] implements a sophisticated sensor fusion procedure to bootstrap a monocular VINS with a more accurate estimate for the initial state, consisting of the pose, velocity, 3D feature locations, gravity vector, metric scale, gyroscope bias, and accelerometer bias. However, VIORB requires 15 seconds of visual-inertial data to obtain an accurate result. It is not suitable for our case that requires a scale estimation right at the beginning. ORB-SLAM3 [23] improves the initialization approach by using an inertial-only maximum-a-posterior (MAP) estimation step [24] to compute the values for the scale, velocities, gravity, and IMU biases. This step takes into account the IMU's measurement uncertainty and the gravity magnitude in producing an estimate that is accurate enough for a joint visual-inertial Bundle Adjustment (BA). The output of the inertial-only MAP is used to initialize the values of the VINS' state parameters to speed up the convergence of the visual-inertial BA. The approach allows the VINS to initialize itself in less than 4 seconds.

Qin *et al.* [25] discover that the metric scale error is linearly dependent upon the accelerometer bias and simultaneously estimating the scale and the accelerometer bias requires a long duration of sensor data collection. To overcome the problem, they propose the VINS-Mono [17] method, where the initialization process is simplified by ignoring the accelerometer bias. The method uses a two-step approach to initializing the VINS' motion state. First, it builds a scale-dependent 3D structure by a visual-only structure-from-motion method. Second, it aligns the IMU integration with the visual-

only structure to recover the scale, gravity, velocity, and gyroscope bias. This initialization approach converges much faster (in ~100 ms) with negligible accuracy loss [25]. However, VINS-Mono [17] still falls short of real-time computing performance on a computer with limited computing power [26]. Using a smaller sliding-window may speed up the computation. But it may result in unwanted loss of accuracy.

Research efforts for the above state-of-the-art VIO methods have been mainly focused on monocular VINS [17], [18], [22], or stereo VINS [16], [27]. RGB-D-camera-based VIO is a less-explored area. In [28], an EKF based VIO method is introduced for pose estimation of an RGB-D VINS. The method uses the egomotion estimated by IMU preintegration to generate state prediction and treats the pose estimated by using the visual-depth data as the observation to compute the state update. Lin *et al.* present a smoothing-based VIO method [29] for RGB-D VINS. The method determines the metric scale from the depth data and obtains the VINS' initial motion state by simply aligning the visual-based pose with the IMU-preintegration-based pose. While it uses the standard VIO framework to estimate the VINS's motion state, the visual features' inverse depths are initialized by using the camera's depth data and are kept as constant values in the optimization process. Shan, *et al.* [30] proposed VINS-RGBD, a smoothing-based VIO method that exploits the depth information in the framework of VINS-Mono [17]. In the initialization process, it uses corner points [31] and employs a 3D-2D-PnP [32] method to build the visual structure. After the initialization, it estimates the VINS' motion state and the inverse depths of the tracked visual features through a nonlinear optimization process. If a feature's depth is provided by the RGB-D camera, the inverse depth value is treated as a constant. Otherwise, the depth value is estimated by triangulation, and the inverse depth value is iteratively updated in the later optimization process. The triangulated depths for far-range visual features are not accurate, and the depth estimation error can reduce the pose estimation accuracy during optimizaiton. Instead, we avoid depth estimation for the far-range visual features, and we utilize the epipolar constraint to model their measurement residuals in the optimization step for pose estimation. Also, we exploit the geometric feature (the floor plane extracted from depth data) to reduce the accumulated pose estimation error. The proposed DVIO method improves the above smoothing based VIO methods by incorporating visual features without depth and geometric feature into the graph for more accurate pose estimation. It achieves real-time computation (~18 Hz) with decent accuracy on a UP Board computer.

B. Related Work in Localization

As an incremental state estimation method, VIO accrues pose errors over time. When using VIO for navigation in a large space, a loop closure can be used to eliminate the accumulated pose error. However, if a loop closure cannot be detected or it is not detected in a timely fashion, the accrued pose error may become big enough to make the navigation system malfunction. To address the problem, the floor plans of the operating environments have been used to reduce accumulative pose errors in the robotics community. Boniardi *et al.* [33], [34] introduce a pose-graph method to track the 3-DOF pose of a robot in a floor plan (CAD drawing) by using a 2D LiDAR. A scan-to-map matching is first performed to align the LiDAR scans with the floor plan and determine the robot's pose with respect to the floor plan. Then, this relative pose measurement is used to create additional edges (between the related nodes), which serve as prior constraints in the graph structure to incorporate the floor plan into the graph. The use of the floor plan in the graph SLAM process helps to reduce the accrued pose error. The method has a 50-ms runtime on an Intel Core i7 CPU (8-core, 4.0 GHz). It is difficult to achieve real-time computation on an Up Board computer (with a 4-core 1.92GHz Intel ATOM CPU). In [35], Watanabe et al. present a method to localize a robot in indoor space by using an architectural floor plan and depth data of an RGB-D camera. The method first extracts a number of planes from the depth image at the robot's current pose and projects the 3D points belonging to the planes onto the floor to produce a 2D source point cloud. It then uses a ray-tracing algorithm to generate a simulated 2D target point cloud from the floor plan. Finally, the robot pose (with respect to the floor plan) is determined by aligning the source point cloud with the target point cloud using the GICP algorithm [36]. However, the method can malfunction when the GICP algorithm is stuck to a local minimum or the scene is not geometrically feature-rich.

The multimodel PFL [37] based method is more robust for pose tracking. Winterhalter *et al.* employ a 6-DOF PFL approach [38] to track the camera pose for a Google Tango tablet in an indoor environment by using the data from the device's RGB-D camera and IMU. The method utilizes the VIO-estimated motion to predict the pose for each particle. It computes an importance weight for each particle, which is proportional to the observation likelihood of the measurement given the particle's state. The likelihood value is estimated by comparing the actual depth data with the expected depth data (from the floor plan) given the predicted pose. A particle survives with a probability proportional to its importance weight in the re-sampling step. To reliably track the device's 6-DOF pose, 5000 particles are used. This results in a high computational cost. To achieve real-time computation, the PFL algorithm must run on a backend server. In this work, we simplify the method and employ a 3-DOF PFL method to estimate the RNA's position and orientation on a 2D floor plan for real-time assistive navigation. Our method uses only 100 particles for pose tracking, resulting in real-time computation (~50-ms runtime) on an Up Board computer. The proposed method creates a local submap by registering several frames of depth data (instead of using just one frame of depth data [38]) and aligns this map with the floor plan to determine the device pose with respect to the floor plan. The multi-frame local submap is less likely to be geometrically featureless, makes our method more robust to depth data noise.

III. RNA Prototype and Notations

As depicted in Fig. 1, the RNA prototype uses an Intel Realsense D435 (RGB-D) Camera and an IMU (VN100 of VectorNav Technologies, LLC) for motion estimation. The D435 consists of a color camera that produces a color image of the scene and an IR stereo camera that generates the corresponding depth data. Their resolutions are set to 424×240 to produce a 20 fps data stream to the UP Board computer [39]. The D435 is mounted on the cane with a 25° tilt-up angle to

keep the cane's body out of the camera's field-of-view. The VN100 is set to output the inertial data at 100 Hz. The prototype uses a mechanism called active rolling tip (ART) [40] to steer the cane to the desired direction of travel to guide the user. The ART consists of a rolling tip, a gear motor (with a built-in encoder), a motor drive, and a clutch. A custom control board is used to engage and disengage the clutch. When the clutch is engaged, RNA enters the robot-cane mode and the motor drives the rolling tip and steers the cane into the desired direction. The slippage at the rolling tip is detected by comparing the encoder and gyro data. If the slippage is above a threshold, RNA switches itself into the white-cane mode temporarily until the slippage drops below the threshold. Details on this human-intent detection scheme for automatic mode switching is referred to [5]. When the ART is disengaged, the rolling tip is disconnected from the gear motor, turning RNA into the white-cane mode, and the user can swing the RNA just like using a white cane. In this case, a coin vibrator (on the grip) vibrates to indicate the desired direction. The user can switch between the two modes by pressing a push-button on the grip. The clutch

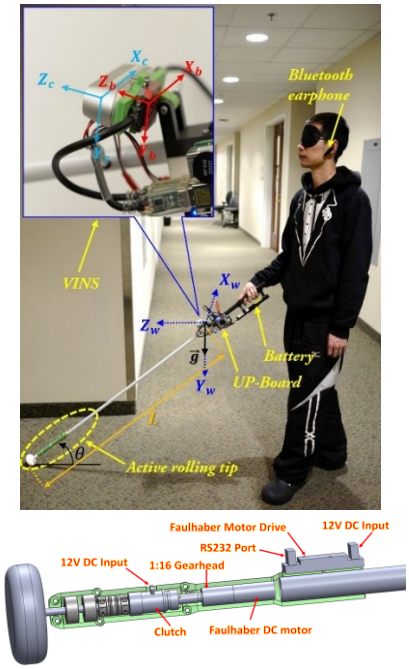


Fig.1 Top: RNA prototype. The body, camera, and world coordinate systems are denoted by $\{B\}$ (or $X_bY_bZ_b$), $\{C\}$ (or $X_cY_cZ_c$), $\{W\}$ (or $X_wY_wZ_w$), respectively. The initial $\{B\}$ is taken as the world coordinate system $\{W\}$ after performing a rotation to make the Z-axis level and align the Y-axis with the gravity vector $\vec{g}$. In this paper, the superscripts b and c describe a variable in $\{B\}$ and $\{C\}$, respectively. Bottom: Solidworks drawing of the ART.

controller and the motor drive are controlled by the Up Board via its general IO port and RS-232 port, respectively.

The body (i.e., IMU) and camera coordinate systems are denoted by $\{B\}$ $(X_bY_bZ_b)$ and $\{C\}$ $(X_cY_cZ_c)$, respectively. The initial $\{B\}$ at the beginning of the navigation task is taken as the world coordinate system, denoted $\{W\}$ ($X_wY_wZ_w$), after performing a rotation to make the Z-axis level and align the Y-axis with the gravity vector $\vec{g}$. the floor plane extracted from the k^{th} frame is described in $\{B\}$ by $\boldsymbol{l}_k^b = \left[\boldsymbol{n}_k^b, d_k^b\right]^T$ or in $\{W\}$ by $\boldsymbol{l}_k^w = [\boldsymbol{n}_k^w, d_k^w]^T$. Here, $\boldsymbol{n}_k^b/\boldsymbol{n}_k^w$ represents the plane's normal vector and d_k^b/d_k^w the distances from the origin to the plane. The transformation from $\{C\}$ to $\{B\}$ is pre-calibrated and it is denoted by $\mathbf{T}_c^b = [\mathbf{R}_c^b\ \mathbf{t}_c^b]$, i.e., $\boldsymbol{\xi}_c^b = \{\mathbf{t}_c^b, \mathbf{q}_c^b\}$ is known a priori. The color and depth cameras' intrinsic parameters have been calibrated and their data have been properly associated. The 3D point cloud of the k^{th} frame is denoted by $\mathbf{P}_k^b$ or $\mathbf{P}_k^w$ in the body or world coordinate system, respectively.

IV. Depth-Enhanced Visual-Inertial Odometry

The motion state of RNA is estimated by the proposed DVIO method which consists of three parts: feature tracker, floor detector, and state estimator. The feature tracker extracts visual features from a color image and tracks them to the next image. It also selects keyframes based on the average parallax difference. If the average parallax of the tracked features between the current frame and the latest keyframe is larger than a threshold (10 pixels), this frame is treated as a keyframe. The tracked features in the keyframes are passed to the optimization process to estimate the VINS' motion state. The features extracted in the non-keyframes are only used for tracking. The floor detector extracts the floor plane from the D435's depth data. The state estimator estimates the state of the IMU by using the visual features, the floor plane, the depth data, and the IMU measurements. The details of each part are described below.

A. Feature Tracker

The feature tracker detects Harris corner features [41] at each image frame. To obtain a higher processing speed without compromising pose estimation accuracy, the image is evenly divided into 8×8 patches, within which at most 4 features are extracted and tracked. These features (at most 256) are tracked across image frames by the KLT tracker [42]. A RANSAC process based on the fundamental matrix is devised to remove outliers that do not satisfy epipolar constraint. Inliers are passed to the state estimator for pose estimation.

B. Floor Detector

Given the pose estimate $\hat{\boldsymbol{\xi}}_{b_{k-1}}^w$ for the $(k-1)^{th}$ and the IMU-measured pose change $\hat{\boldsymbol{\xi}}_{b_k}^{b_{k-1}}$, the RNA's pose for the k^{th} frame can be predicted by $\hat{\boldsymbol{\xi}}_{b_k}^w = \hat{\boldsymbol{\xi}}_{b_{k-1}}^w \circ \hat{\boldsymbol{\xi}}_{b_k}^{b_{k-1}}$, where $\circ$ is pose composition operator. The k^{th} frame point cloud data $\mathbf{P}_k^{b_k}$ of the D435 can be described in $\{W\}$ by $\mathbf{P}_k^w = \hat{\boldsymbol{\xi}}_{b_k}^w \mathbf{P}_k^{b_k}$. The floor plane can be extracted by RANSAC from the points having a Y-coordinate in $[d_{\mathrm{k}} - 0.15,\ d_{\mathrm{k}} + 0.15]$, where d_{k} is the estimated floor plane height (along Y_w-axis) and it can be computed from $\hat{\boldsymbol{\xi}}_{b_k}^w$ and L (see Fig. 1). The extracted plane is accepted as the floor plane measurement if it contains more than 3000 points and the angle between its normal and Y_w is between 175° and 185°. The detected floor plane in $\{W\}$ is then expressed by $\boldsymbol{l}_k^{\mathrm{w}} = [\boldsymbol{n}_k^{\mathrm{w}}, d_k^{\mathrm{w}}]^T$, where $\boldsymbol{n}_k^{\mathrm{w}}$ and d_k^{w} are the normal vector and the distance from the origin to the floor plane. It can be described in $\{B\}$ by

$$\boldsymbol{l}_k^b = \left[\boldsymbol{n}_k^b, d_k^b\right]^T = \left[\hat{\mathbf{R}}_w^{b_k}\boldsymbol{n}_k^w, \hat{\mathbf{t}}_w^{b_k}\boldsymbol{n}_k^w + d_k^w\right]^T \quad (1)$$

C. State Estimator

A sliding window-based nonlinear optimization process is employed for state estimation. The full state vector in the sliding

window is defined as $\boldsymbol{\chi} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, where $\mathbf{x}_i = \{\mathbf{t}_{b_i}^w, \mathbf{v}_{b_i}^w, \mathbf{q}_{b_i}^w, \mathbf{b}_a, \mathbf{b}_g\}$ ($i \in [1, n]$) is the IMU's motion state (translation, velocity, rotation, accelerometer bias, and gyroscope bias) at the time when the i^{th} keyframe is captured. n is the number of keyframes in the window ($n = 4$ in this work). Visual features with a known depth are used to build a perspective reprojection model to constrain pose estimation in the optimization process. Visual features with an unknown depth are also used in the process as they constrain the estimation of the rotation. Moreover, the floor plane (extract from the depth

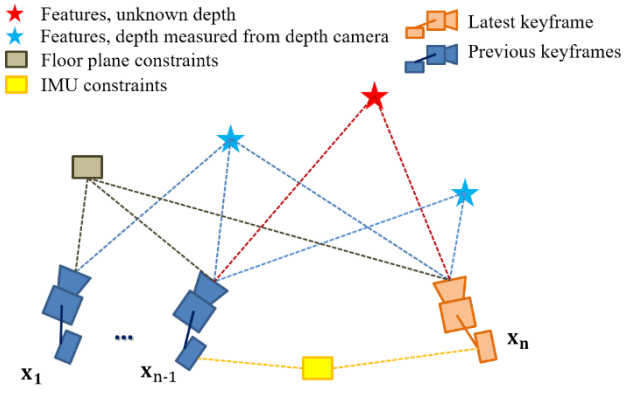


Fig. 2 Graph structure for DVIO

data) is incorporated into the graph to reduce the pose estimation error. Fig. 2 shows one example graph of the DVIO method.

We estimate the state vector $\boldsymbol{\chi}$ by minimizing the sum of the squared Mahalanobis distances of the prior and measurement residuals, given by:

$$\boldsymbol{\chi}_k^* = \underset{\boldsymbol{\chi}_k}{argmin}\Big(\|\boldsymbol{r}_o\|_{\boldsymbol{\Sigma}_o}^2 + \sum_{(k-1,k)} \left\|\boldsymbol{r}_{\boldsymbol{k-1,k}}^{\boldsymbol{b}}\right\|_{\boldsymbol{\Sigma}_b}^2 + \sum_{(f,k)\in C_p} \left\|\boldsymbol{r}_{\boldsymbol{f,k}}^{\boldsymbol{p}}\right\|_{\boldsymbol{\Sigma}_f}^2 + \sum_{(i,k)\in C_{v1}} \left\|\boldsymbol{r}_{\boldsymbol{i,k}}^{\boldsymbol{v}}\right\|_{\boldsymbol{\Sigma}_v}^2 + \sum_{(i,k)\in C_{v2}} \left\|\boldsymbol{r}_{\boldsymbol{i,k}}^{\boldsymbol{v}}\right\|_{\boldsymbol{\Sigma}_v}^2\Big) \quad (2)$$

where $\|\boldsymbol{r}\|_{\boldsymbol{\Sigma}}^2 = \boldsymbol{r}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{r}$ represents the squared Mahalanobis distance for residual $\boldsymbol{r}$; $\boldsymbol{r}_o$, $\boldsymbol{r}_{\boldsymbol{k-1,k}}^{\boldsymbol{b}}$, $\boldsymbol{r}_{\boldsymbol{f,k}}^{\boldsymbol{p}}$ and $\boldsymbol{r}_{\boldsymbol{i,k}}^{\boldsymbol{v}}$ are the residuals related to the prior information from marginalization, IMU preintegration between keyframes $k-1$ and k, floor plane, and visual feature measurements, respectively; $\boldsymbol{\Sigma}_o$, $\boldsymbol{\Sigma}_b$, $\boldsymbol{\Sigma}_f$, and $\boldsymbol{\Sigma}_v$ are the covariance matrices used to compute the squared Mahalanobis distances; C_p, C_{v1}, and C_{v2} represent the set of measurements for the floor plane, visual features with depth and visual features without depth, respectively. In this work, $\boldsymbol{r}_o$ and $\boldsymbol{r}_{\boldsymbol{k-1,k}}^{\boldsymbol{b}}$ and their covariance matrices ($\boldsymbol{\Sigma}_o$ and $\boldsymbol{\Sigma}_b$) are computed by using Qin's method [21]. The computation of $\boldsymbol{r}_{\boldsymbol{f,k}}^{\boldsymbol{p}}$ and $\boldsymbol{r}_{\boldsymbol{i,k}}^{\boldsymbol{v}}$ and the related Jacobians are given later in this section. We employ Ceres solver to solve the optimization problem in Equation (1). As the D435 uses an IR stereo camera to measure depth, the measurement error increases quadratically with the true depth. To attain a good pose estimation accuracy, DVIO should only use the depth data of near-range visual features. To determine the depth threshold, we experimented (as shown in Fig. 3) to characterize the D435 camera. The method in [43] was used for the characterization study. It can be seen that the measurement is of high accuracy (error < 2.2 cm) if the depth is no greater than 2.2 m. Therefore, a near-range (≤ 2.2 m) visual feature is assigned the depth measurement from the

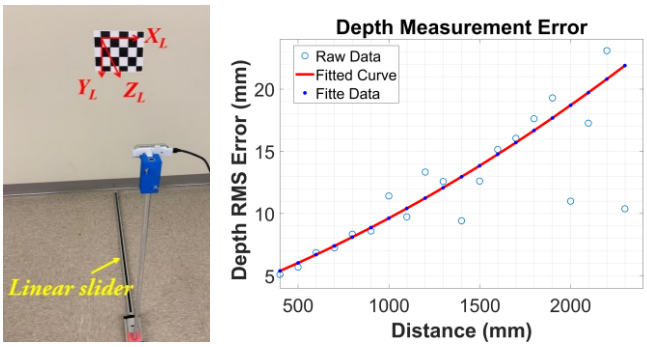


Fig. 3. Characterization of the D435 camera: the linear motion table moves the camera from 400 mm to 2400 mm with a step-size of 100 mm. At each position, 300 frames of depth data were captured and used to compute the mean and RMS of the measurement errors. The method in [43] was employed to estimate the ground truth depth, which is then refined by using the known camera movement (100 mm) to obtain the ground truth depth. Given a camera pose, the wall plane is projected to the camera frame as the ground truth plane.

camera and a far-range visual feature is assigned an unknown depth. Features with an unknown depth are also used for state estimation as they contain the information about the RNA's rotation and the direction of its translational movement.

1) Floor plane measurement $\boldsymbol{r}_{\boldsymbol{f,k}}^{\boldsymbol{p}}$

Given the IMU's pose $\boldsymbol{\xi}_{b_k}^w = \{\mathbf{t}_{b_k}^w, \mathbf{q}_{b_k}^w\}$, the expected measurement of the floor plane in $\{B\}$ is $\hat{\boldsymbol{l}}_k^b = [\hat{\boldsymbol{n}}_k^b, \hat{d}_k^b]^T$ with $\hat{\boldsymbol{n}}_{\boldsymbol{k}}^{\boldsymbol{b}} = \mathbf{R}_w^{b_k}\boldsymbol{n}_0^w$ and $\hat{d}_k^b = \mathbf{t}_w^{b_k}\boldsymbol{n}_0^w + d_0^w$, where $\boldsymbol{n}_0^w$ and d_0^w are the parameters of the plane detected at the beginning of the navigation task (i.e., frame 0). To avoid over parameterization, the normal vector is described as a 2D vector $\boldsymbol{\rho} \in \mathbb{R}^2$ in the tangent space $T_{\boldsymbol{n}}(S^2)$ with basis $\mathbf{B}_{\mathbf{n}}$, where $S^2 = \{\boldsymbol{n} \in \mathbb{R}^3 | \, \|\boldsymbol{n}\| = 1\}$ and $T_{\boldsymbol{n}}(S^2) \triangleq \{\boldsymbol{\varphi} \in \mathbb{R}^3 | \, \boldsymbol{n}^T\boldsymbol{\varphi} = 0\}$. $\boldsymbol{\pi} = \mathbf{B}_{\mathbf{n}}\boldsymbol{\rho}$, where $\boldsymbol{\rho} \in \mathbb{R}^2$ is the 2D coordinate of $\boldsymbol{\varphi}$ in the tangent plane with the basis $\mathbf{B}_{\mathbf{n}}$. Given the actual measurement $\boldsymbol{l}_k^b$ and the predicted measurement $\hat{\boldsymbol{l}}_k^b$, the residual $\boldsymbol{r}_{f,k}^p$ is calculated by

$$\boldsymbol{r}_{f,k}^p = [(\mathbf{B}_{\boldsymbol{n}_k^b})^{\mathrm{T}}\hat{\boldsymbol{n}}_k^b, d_k^b - \hat{d}_k^b]^{\boldsymbol{T}}. \quad (3)$$

The Jacobian matrix is

$$\boldsymbol{J}_{fk} = \frac{\partial \boldsymbol{r}_{f,k}^p}{\partial \boldsymbol{\xi}_{b_k}^w} = \begin{bmatrix} \mathbf{0}_{2\times 3} & (\mathbf{B}_{\boldsymbol{n}_k^b})^T[(\mathbf{R}_{b_k}^w)^T\boldsymbol{n}_f^w]_\times \\ (-\mathbf{t}_{b_k}^w)^T & \mathbf{0}_{1\times 3} \end{bmatrix} \quad (4)$$

where $[\boldsymbol{n}]_\times$ is the skew matrix of $\boldsymbol{n}$. The basis $\mathbf{B}_{\mathbf{n}} = [\boldsymbol{b}_1|\boldsymbol{b}_2]$ for $T_{\boldsymbol{n}}(S^2)$ is computed by $\boldsymbol{b}_1 = \boldsymbol{b}'/||\boldsymbol{b}'||$ and $\boldsymbol{b}_2 = \boldsymbol{n} \times \boldsymbol{b}_1$, where $\boldsymbol{b}' = \boldsymbol{n} \times \boldsymbol{a}$. To ensure that $\boldsymbol{a}$ is not parallel to $\boldsymbol{n}$, we set $\boldsymbol{a}$ to $[1,0,0]^T$, $[0,1,0]^T$ and $[0,0,1]^T$ if n_x, n_y or n_z dominates the other two elements, respectively. $\boldsymbol{\Sigma}_f$ is computed by using a linear regression model and the details can be found in [44].

2) Visual Feature Measurement with known depth

For the i^{th} visual feature that is anchored on the j^{th} image, the residual for the observation on the k^{th} image is defined as

$$\boldsymbol{r}_{i,k}^v = [u_i^k, v_i^k, 1]^T - \frac{\hat{\boldsymbol{p}}_i^{c_k}}{\hat{p}_z} \quad (5)$$

where $\hat{\boldsymbol{p}}_i^{c_k} = [\hat{p}_x, \hat{p}_y, \hat{p}_z]^T = \boldsymbol{\xi}_{c_u}^{c_k} \circ (\rho_i[u_i^j, v_i^j, 1]^T)$; u_i^k, v_i^k and u_i^j, v_i^j are the normalized coordinates of the i^{th} visual feature at the k^{th} and j^{th} images, respectively; and ρ_i is the depth

estimate for visual feature i at keyframe j. From this equation, the Jacobian matrix

$$\boldsymbol{J}_{ik} = \frac{\partial \boldsymbol{r}_{i,k}^{v}}{\partial \boldsymbol{\xi}_{b_k}^{w}} = \left[-\boldsymbol{J}_1 \mathbf{R}_{w}^{c_k} \quad \boldsymbol{J}_1 \mathbf{R}_{b}^{c} \left[\hat{\boldsymbol{p}}_{i}^{b_k} \right]_{\times} \right] \tag{6}$$

where $\boldsymbol{J}_1 = \left. \frac{\partial \boldsymbol{r}_{i,k}^{v}}{\partial \hat{\boldsymbol{p}}_{i}^{c_k}} \right|_{\hat{\boldsymbol{p}}_{i}^{c_k} = [\hat{p}_x, \hat{p}_y, \hat{p}_z]^T} = \begin{bmatrix} 1/\hat{p}_z & 0 & \hat{p}_x/(\hat{p}_z)^2 \\ 0 & 1/\hat{p}_z & \hat{p}_y/(\hat{p}_z)^2 \end{bmatrix}$.

The measurement covariance is defined by $\boldsymbol{\Sigma}_{\mathrm{v}} = diag(\frac{\sigma_\tau^2}{f^2}, \frac{\sigma_\tau^2}{f^2})$, where σ_τ is the image noise (σ_τ=1.5 pixels in this work) and f is the camera's focal length. Since the accuracy of inverse depth is critical to attaining an accurate pose estimation, the i^{th} visual feature on the j^{th} keyframe is assigned the depth measurement from the RGB-D camera only if the measured value is no greater than 2.2 meters. Once assigned, the value ρ_i on keyframe j is kept constant during the iterations of the optimization process. At the time keyframe j is marginalized, ρ_i is then handed to keyframe j+1 if the visual feature is also observed on keyframe j +1. This is advantageous over a monocular VIO method that needs to update the depth value throughout the pose estimation process. Unlike VINS-Mono that uses the depth estimate of a feature point at its first observation, DVIO uses the smallest depth of the frames within the sliding window for ρ_i. In addition, if the feature is tracked onto the next keyframe with a smaller depth, then ρ_i is updated with that depth value. We also make the next keyframe as the anchoring keyframe for visual feature i. These treatments aim to minimize the measurement error for ρ_i.

3) Visual Feature Measurement with an unknown depth

Assuming that the i^{th} visual feature ($\boldsymbol{X}_i^{c_j} = [u_i^{c_j}, v_i^{c_j}, 1]^T$) is observed on the j^{th} image and tracked onto the k^{th} image as $\boldsymbol{X}_i^{c_k} = [u_i^{c_k}, v_i^{c_k}, 1]^T$, the epipolar error is defined as the residual given by

$$\boldsymbol{r}_{i,k}^{v} = \left(\mathbf{R}_{c_k}^{c_j} \boldsymbol{X}_i^{c_k} \right)^T \left(\left[\mathbf{t}_{c_k}^{c_j} \right]_{\times} \boldsymbol{X}_i^{c_j} \right) \tag{7}$$

The Jacobian matrix of $\boldsymbol{r}_{i,k}^{v}$ with regard to $\boldsymbol{\xi}_{b_k}^{w}$ is given by

$$\boldsymbol{J}_{ik} = \frac{\partial \boldsymbol{r}_{i,k}^{v}}{\partial \boldsymbol{\xi}_{b_k}^{w}} = \frac{\partial \boldsymbol{r}_{i,k}^{v}}{\partial \boldsymbol{\xi}_{c_k}^{c_j}} \frac{\partial \boldsymbol{\xi}_{c_k}^{c_j}}{\partial \boldsymbol{\xi}_{b_k}^{w}} \tag{8}$$

where $\frac{\partial \boldsymbol{r}_{i,k}^{v}}{\partial \boldsymbol{\xi}_{c_k}^{c_j}} = \left[\left(\left[\mathbf{R}_{c_k}^{c_j} \boldsymbol{X}_i^{c_k} \right]_{\times} \boldsymbol{X}_i^{c_j} \right)^T \quad -\left(\left[\mathbf{t}_{c_k}^{c_j} \right]_{\times} \boldsymbol{X}_i^{c_j} \right)^T \mathbf{R}_{c_k}^{c_j} \left[\boldsymbol{X}_i^{c_k} \right]_{\times} \right]$

and $\frac{\partial \boldsymbol{\xi}_{c_k}^{c_j}}{\partial \boldsymbol{\xi}_{b_k}^{w}} = \begin{bmatrix} \mathbf{R}_{w}^{c_j} & -\mathbf{R}_{b_k}^{c_j} [\mathbf{t}_c^b]_{\times} \\ \mathbf{0}_{3\times 3} & (\mathbf{R}_c^b)^{-1} \end{bmatrix}$.

V. Visual Positioning System for Assistive Navigation

The DVIO-estimated pose is used to 1) generate a 3D point cloud map for obstacle avoidance, and 2) obtain a refined 2D pose by PFL on a floor plan map for wayfinding. DVIO and PFL form a visual positioning system, based on which an assistive navigation system is created as shown in Fig. 4. The system was developed based on the robot operating system (ROS) framework. Each ROS node is an independent functional module and it communicates with the others through a messaging mechanism. The Data Acquisition node acquires and publishes the camera's and the IMU's data, which are subscribed by the DVIO node for pose estimation. The Terrain Mapping node registers the depth data captured with different camera poses to form a 3D point cloud map, which is then

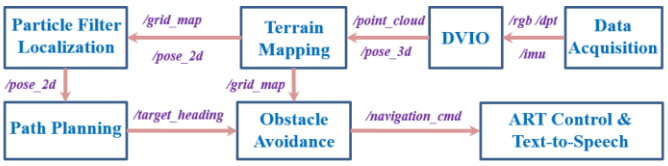


Fig. 4. Software pipeline for the assistive navigation software

reprojected onto the floor plane to create a 2D local grid map for obstacle avoidance and localization of RNA in the 2D floor plan. Based on the RNA's location in the floor plan, the Path Planning module [45] determines the desired heading to direct RNA towards the next Point Of Interest (POI). This information is passed to the Obstacle Avoidance module [46] to compute the Desired Direction of Travel (DDT) that will move RNA towards the POI without colliding with the surrounding obstacle(s). Based on the DDT, the ART Controller steers RNA into the DDT, and the speech interface sends audio navigation messages to the blind traveller via the Bluetooth headset. Both the tactile and audio information will guide the blind traveller to move along the planned path. The details of the major modules such as PFL, Path Planning, Obstacle Avoidance, and ART Control are described below.

A. Particle Filter based Localization

DVIO accrues a pose error over time and the accumulated error may cause the navigation system to fail. To reduce pose error, we adopt Winterhalter's idea [38] and employ a Particle Filter (PF) to localize RNA in a 2D floor plan. Winterhalter's method computes a mobile device's 6-DOF pose in an architectural floor plan by generating a prior 3D map from the architectural drawing and aligning the captured 3D depth data with this map. The method is slow because a large number (5000) of particles are required to represent the distribution of a 6-DOF pose variable. To overcome this problem, we simplify the problem by estimating the RNA's 3-DOF pose in a 2D floor plan and reduce the required number of particles from 5000 to 100. This makes real-time computation on the UP Board computer possible. Our method generates range measurements both from the local grid map [47] and the floor plan, and align these two sets of range measurements to localize RNA in the floor plan. A 2D laser scanner simulator [48] is devised to produce range measurements from -45° to 225° (with 1° interval) at the current RNA pose. 3D points that are above the floor plane (>0.1 meters) are projected onto the floor plane (X_w-Z_w), which is divide into 0.1×0.1 m^2 grid cells. A cell is labeled as an occupied one if it contains one or more projected points, or as a free cell otherwise. An occupied cell at (x, z) produces a measurement $\mathrm{z}_{t,\beta} = \sqrt{x^2 + z^2}$ with a bearing angle $\beta = int(\tan^{-1}(z/x))$ if $\beta \in [-45°, 225°]$, while a free cell produces an infinite measurement value.

PFL consists of three steps, motion prediction, weight update, and resampling. First, the RNA's egomotion is computed from the DVIO-estimated poses at time steps $t-1$ and t and it is used to predict the RNA's pose. At time step t, the predicted pose for particle i is given by $\boldsymbol{x}_t^i = \boldsymbol{x}_{t-1}^i + \Delta\boldsymbol{\xi}_t^{w'} + \mathbf{n}_{\mathrm{o}}$, where $\Delta\boldsymbol{\xi}_t^{w'}$ is the projected RNA egomotion on X_w–Z_w

plane and $\mathbf{n}_o \sim \mathcal{N}(0, \mathbf{\Lambda}_o)$ is the pose noise with $\mathbf{\Lambda}_o = diag(\sigma_x^2, \sigma_y^2, \sigma_\psi^2)$. In our implementation, $\sigma_x = \sigma_z = 0.03$ and $\sigma_\psi = 3°$. Second, given the pose $\boldsymbol{x}_t^i$ and the floor plan map $\mathfrak{m}$, the likelihood of making the measurement $\mathbf{z}_t$ is computed by the sensor model $p(\mathbf{z}_t | \boldsymbol{x}_t^i, \mathfrak{m}) \propto \prod_{\beta=-45}^{225} p(z_{t,\beta} | \boldsymbol{x}_t^i, \mathfrak{m})$, where $p(z_{t,\beta} | \boldsymbol{x}_t^i, \mathfrak{m})$ with $z_{t,\beta} \sim \mathcal{N}(\hat{z}_{t,\beta}, \sigma_d^2)$ is the measurement model for the laser scanner. In this work, $\sigma_d = 0.2$ meters. The expected measurement $\hat{z}_{t,\beta}$ is obtained by running the laser scanner simulator on $\mathfrak{m}$ with pose $\boldsymbol{x}_t^i$. The importance weight of

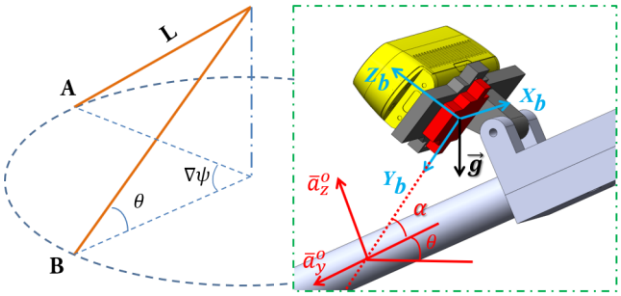


Fig. 5. Left: RNA swings from A to B, Right: Computation of θ from the accelerometer data.

the i^{th} particle is then updated by $w_t^i = \eta w_{t-1}^i \, p(\mathbf{z}_t | \boldsymbol{x}_t^i, \mathfrak{m})$, where η is a normalizer. Third, the adaptive strategy [49] is employed for resampling. The effective number of samples $N_{\text{eff}} = 1/\sum_{i=1}^{N} (w_t^i)^2$ is used to evaluate how well the N-particle set represents the target posterior. If $N_{\text{eff}} < 0.8$, a resampling operation is performed. In this work, N=100. At time step t, the output of the PF is given by $\boldsymbol{x_t} = \sum_{i=1}^{N} w_t^i \boldsymbol{x}_t^i$.

B. Path Planning

We use our earlier Point of Interest (POI) graph method [45] for path planning. The graph's nodes are the POIs (hallway junctions, elevators, etc.) and each edge between two nodes has a weight equal to the distance between them. The A* algorithm is used to find the shortest path from the starting point to the destination. At each POI along the path, a navigational message is generated based on the next POI. This message is conveyed to the user by the speech interface. In addition, at each junction POI where a turn is required, the needed heading angle change is computed as the difference between the current heading angle and the angle required to move towards the next POI.

C. Obstacle Avoidance

In this work, we employ the Traversability Field Histogram (TFH) [46] method to determine an obstacle-free direction for RNA. First, a local terrain map surrounding RNA is converted into a Traversability Map (TM). Then, a Polar Traversability Index (PTI) is computed for each 5° sector of the TM. The smaller the PTI, the more traversable the direction. The PTIs are structured in the form of a histogram. Consecutive sectors with a low PTI form a histogram valley, indicating a walkable direction to RNA. The valley closest to the RNA's target direction is selected and the DDT for RNA is thus determined. The steering angle for RNA is calculated based on the DDT and the current RNA heading. The steering angle is then used to control the ART. In addition, a navigational message is generated based on the next POI. This message is conveyed to the user via the speech interface.

D. ART Control

To steer the rolling tip of RNA from position A to B and make a heading angle change $\Delta\psi$ (see Fig. 5), the required rotation of the motor is computed by $\Delta\mu = CL\Delta\psi\cos(\theta)/r$, where C, L, θ, and r are the gearhead reduction ratio, the cane's length, the cane's tilt angle, and the rolling tip's radius, respectively. This means that the cane's turning angle can be accurately controlled by the motor. In other words, RNA may use its motor control system to steer itself into the desired direction for the user to follow. In our case, $C = 16$, $L = 1.47$ m, and $r = 0.04$ m. The tilt angle θ (see Fig. 1) is mainly determined by the user's height. It may undergo a small change when the user is walking. We estimate θ at the beginning of each navigation task (when the user holds the cane steadily) based on the accelerometer reading. The averaged value of the first 100 IMU readings, denoted $\bar{\boldsymbol{a}}^{\boldsymbol{b}} = \{\bar{a}_x^b, \bar{a}_y^b, \bar{a}_z^b\}$, is used to estimate the tilt angle by $\theta = |\arctan(\bar{a}_z^o / \bar{a}_y^o)|$, where $\bar{a}_y^o = \bar{a}_y^b \cos\alpha + \bar{a}_z^b \sin\alpha$ and $\bar{a}_z^o = -\bar{a}_y^b \sin\alpha + \bar{a}_z^b \cos\alpha$. α is the angle between Y_b and the cane body and it is known a priori.

TABLE I: COMPARISON OF EPENS (METERS) OF VINS-MONO, VINS-RGBD AND DVIO

Dataset	VINS-Mono	VINS-RGBD	DVIO Variants		
			D	DF	DFV
D1	1.34	1.03	0.87	0.82	0.83
D2	1.16	1.14	0.44	0.33	0.25
D3	0.76	0.49	0.57	0.45	0.38
D4	X	0.63	0.82	0.58	0.46
D5	1.34	1.01	1.74	1.25	1.09
D6	0.63	1.00	1.01	0.72	0.62
D7	1.21	0.81	1.09	0.94	0.82
Mean	1.07	0.87	0.93	0.73	0.64

X indicates that the method diverged.

VI. EXPERIMENTS

A. DVIO Accuracy: D435 + VN100

The performance of DVIO was compared with that of VINS-Mono [17] and VINS-RGBD [30] by experiments. Eight datasets were collected by holding RNA and walking at a speed of ~0.7 m/s. During each data collection session, the user swung RNA just like using a white cane. The ground truth positions of the start point and endpoint are [0, 0, 0] and [0, 0, 20 m], respectively. We use the endpoint position error norm (EPEN) as the metric for pose estimation accuracy. DVIO's pose estimation accuracy and computational cost can be tuned by adjusting the size of the sliding window. For the sake of real-time computation, we used a small window consisting of 4 pose-nodes for DVIO. For the fairness of comparison, VINS-Mono and VINS-RGBD also used a 4-node sliding window and their loop closure functions were disabled. To demonstrate that the use of the floor plane and the visual features with unknown depth improves pose estimation accuracy, we ran DVIO with three different conditions, denoted DVIO-DFV, DVIO-DF, and DVIO-D, representing the full DVIO implementation, DVIO that does not use visual features without depth, and DVIO that does not use visual features without depth and the floor plane, respectively. Their pose estimation accuracies are compared with that of VINS-Mono and VINS-RGBD in Table I. It can be

seen that: 1) using the floor plane reduced the EPEN of DVIO-D by 21.5%; 2) using visual features without depth reduced the EPEN of DVIO-DF by 12.3%. Therefore, the full DVIO has the best accuracy. On average, it reduced the EPEN by 40.2% and 26.4% when compared with VINS-Mono and VINS-RGBD, respectively.

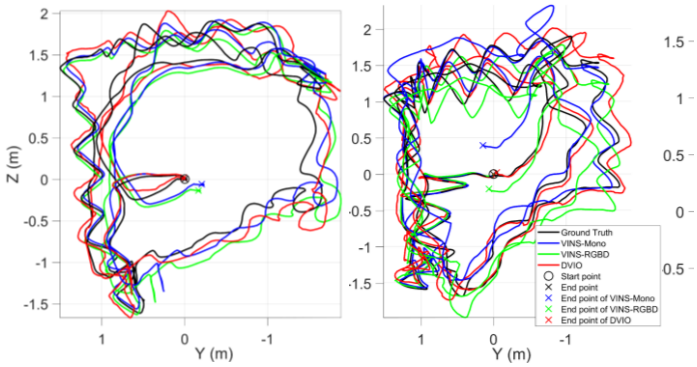


Fig. 7 From left to right: trajectory comparison for datasets S1, S2, S5, and S8 DVIO are plotted in black, blue, green, and red, respectively. o indicates the s

B. DVIO Accuracy: Structure Core

We collected eight more datasets from the most updated RGB-D camera with an integrated IMU—Occipital Structure Core (SC)—that can provide synchronized image, depth (0.7-7 meters), and inertial data and compared DVIO's pose estimation performance with that of VINS-Mono and VINS-RGBD by using these datasets. We characterized the SC by using the method in [43] and found that the depth measurement is of high accuracy (error < 2 cm) if the depth is no greater than 4.0 m (Fig. 6b).

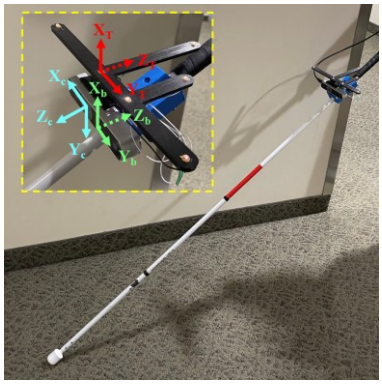
We installed the SC on a white cane in a way similar to D435 (see Fig. 6a) and collected eight datasets by swinging the cane and walking (~0.7 m/s) in our laboratory. Based on the ground truth poses provided by the OptiTrack motion capture (MoCap) system, we calculated the absolute pose error for each point on the trajectories generated by DVIO, VINS-Mono, and VINS-RGBD. Table II summarizes the results. It can be observed that DVIO has the smallest RMSE in seven of the eight experiments. Its RMSE is only slightly larger than that of VINS-RGBD in one experiment. This demonstrates that DVIO has a much more accurate pose estimation than the other methods. On average, it reduced the RMSE by 57.1% and 23.7% when compared with VINS-Mono and VINS-RGBD, respectively. The trajectories generated by the three methods for four of the experiments are compared in Fig. 7, which show that the trajectories generated by DVIO are more accurate than that of VINS-Mono/VINS-RGBD.

TABLE II: RESULTS ON THE LAB DATASETS: RMSE OF THE ESTIMATED TRAJECTORY OF EACH VIO METHOD. TL - TRAJECTORY LENGTH.

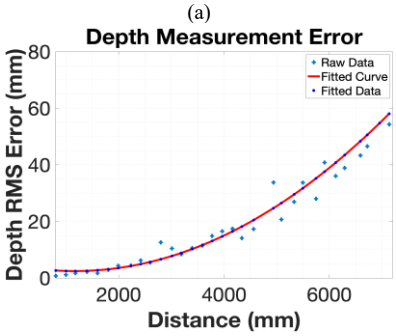
Dataset	TL(m)	VINS-Mono	VINS-RGBD	DVIO
S1	30.4	0.129	0.124	0.105
S2	43.0	0.209	0.151	0.085
S3	56.3	0.352	0.248	0.233
S4	23.5	0.23	0.107	0.065
S5	22.4	0.121	0.065	0.054
S6	26.0	0.126	0.078	0.067
S7	16.8	0.28	0.069	0.077
S8	22.2	0.282	0.158	0.075
Mean		0.21	0.118	0.09

C. Runtimes of DVIO and Other Modules

Table III shows the runtimes of the major modules of the assistive navigation software system depicted in Fig. 4. The average runtimes of DVIO, Terrain Mapping, PFL, and Obstacle



(a)



(b)

Fig. 6 (a) Structure Core sensor on a white cane for data collection. The coordinate systems of the body (IMU), color camera, and the LED-target are denoted by $X_bY_bZ_b$, $X_cY_cZ_c$, and $X_TY_TZ_T$, respectively. The LED-target will be tracked by the MoCap system to produce the ground truth poses. (b) Measurement error vs distance of the Structure Core sensor.

Avoidance are 55.6 ms, 19.9 ms, 17.5 ms, and 0.5 ms, respectively. Since each module runs as an independent thread on a different core of the CPU, the software system achieves real-time computation on the UP Board computer (~18 fps).

D. PFL Performance Evaluation

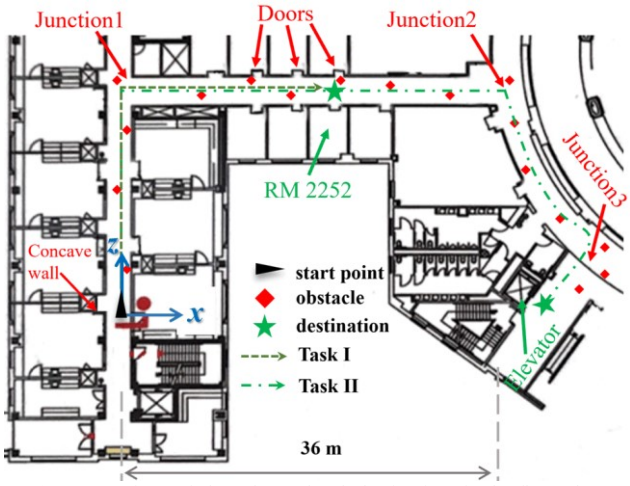
To evaluate RNA's localization performance, we carried out experiments by holding RNA and walking along several different paths on the second floor of the Engineering East Hall

TABLE III RUNTIME FOR MODULES OF RNA

Modules	DVIO	Terrain Mapping	PFL	Obstacle avoidance
Runtime (ms)	55.6 ± 7.8	19.9 ± 13.8	17.5 ± 51.6	0.5 ± 0.6

of Virginia Commonwealth University. We created the floor plan map (as shown in Fig. 8a) from the architectural floor plan drawing after performing necessary editing to the doors (to show the geometric shapes of the closed doors along the paths). The distinctive geometric shapes of the areas around the doors, junctions, and corners will be used by PFL for RNA localization in the floor plan. For each experiment, the target and actual endpoints of RNA were recorded and their difference is calculated as the EPEN for performance evaluation. Table IV

summarizes the EPENs of the experiments. The trajectories



(a) Locations of obstacles and task destinations in the floor plan

(b) Snapshots of the scenes at the start point and Junctions 1, 2

Fig. 8 Experimental settings for localization/wayfinding experiments

estimated by PFL (i.e., DVIO + PF) and that by DVIO only are compared in Fig. 9 to demonstrate the improved localization accuracy. It can be seen that PFL has a smaller EPEN for each experiment. Its mean EPEN over all experiments is 0.58%, i.e., 82.5% smaller than that of DVIO, meaning that the particle filter reduces the DVIO-accrued pose error by 82.5% on average. It is noted that the use of EPEN in the percentage of path-length allows us to compute the mean value over experiments with different paths for overall performance comparison. In principle, PFL eliminates DVIO-accrued pose error whenever RNA 'sees' a geometrically featured region. When RNA moves in a corridor (between two featured regions), PFL can eliminate the lateral but not the longitudinal position error. As a result, PFL's pose error is the PF alignment error plus the uncorrected DVIO pose error since the last alignment (occurred at the last geometrically featured region). This means that the path-length does not affect the EPEN of the PFL method. One can see from Table IV that the EPEN of data sequence DS6/DS7 is much smaller than that of DS4 even if its path-length is much longer. This is because the endpoint of DS6 locates at junction 1 and the last concave wall of DS7 that RNA "saw" is very close to the endpoint while the elevator (endpoint for DS4) is much farther from junction 3 (the last-seen feature).

From the trajectory plots (Fig. 9), it can be seen that the trajectories estimated by DVIO (blue lines) intersect with the walls or doors as the result of the accrued pose error. But the PFL method eliminated the pose errors from time to time and resulted in much more accurate trajectories (red lines).

TABLE IV: COMPARISON OF EPENS: METERS (% OF PATH-LENGTH)

Data Sequence	Trajectory Length	DVIO	DVIO + PFL
DS1	80 m	3.42 (4.28%)	0.45 (0.56%)
DS2	80 m	2.78 (3.48%)	0.85 (1.06%)
DS3	80 m	1.71 (2.14%)	0.78 (0.98%)
DS4	80 m	3.99 (4.99%)	0.50 (0.63%)
DS5	120 m	3.72 (3.10%)	0.58 (0.48%)
DS6	110 m	1.58 (1.44%)	0.17 (0.15%)
DS7	190 m	7.20 (3.79%)	0.32 (0.17%)
Mean		3.32%	0.58%

E. Wayfinding Experiments

We tested the practicality of the visual positioning system by performing two navigation tasks in the Engineering East Hall. Task I is from RM 2264 to RM 2252 (path-length: ~35 meters) and task II is from RM 2264 to the elevator (path-length: ~80 meters). Two sighted persons (blind-folded) performed these tasks. Each person conducted two experiments for each task and he/she stopped at the point when RNA indicated that the destination had been reached. The EPENs (in meters) for the experiments are tabulated in Table V. As the path-length does not affect a PFL-estimated trajectory, we use the absolute EPEN as the performance metric. The average EPEN for tasks I and II are 0.20 m and 0.45 m, respectively. Due to the small error, RNA successfully guided the users to get to the destinations in all experiments. In Table V, we also show the mean EPENs over persons and that over experiments for each task. Their values are close to the overall averaged value (0.20 m or 0.45 m), indicating a consistent localization performance.

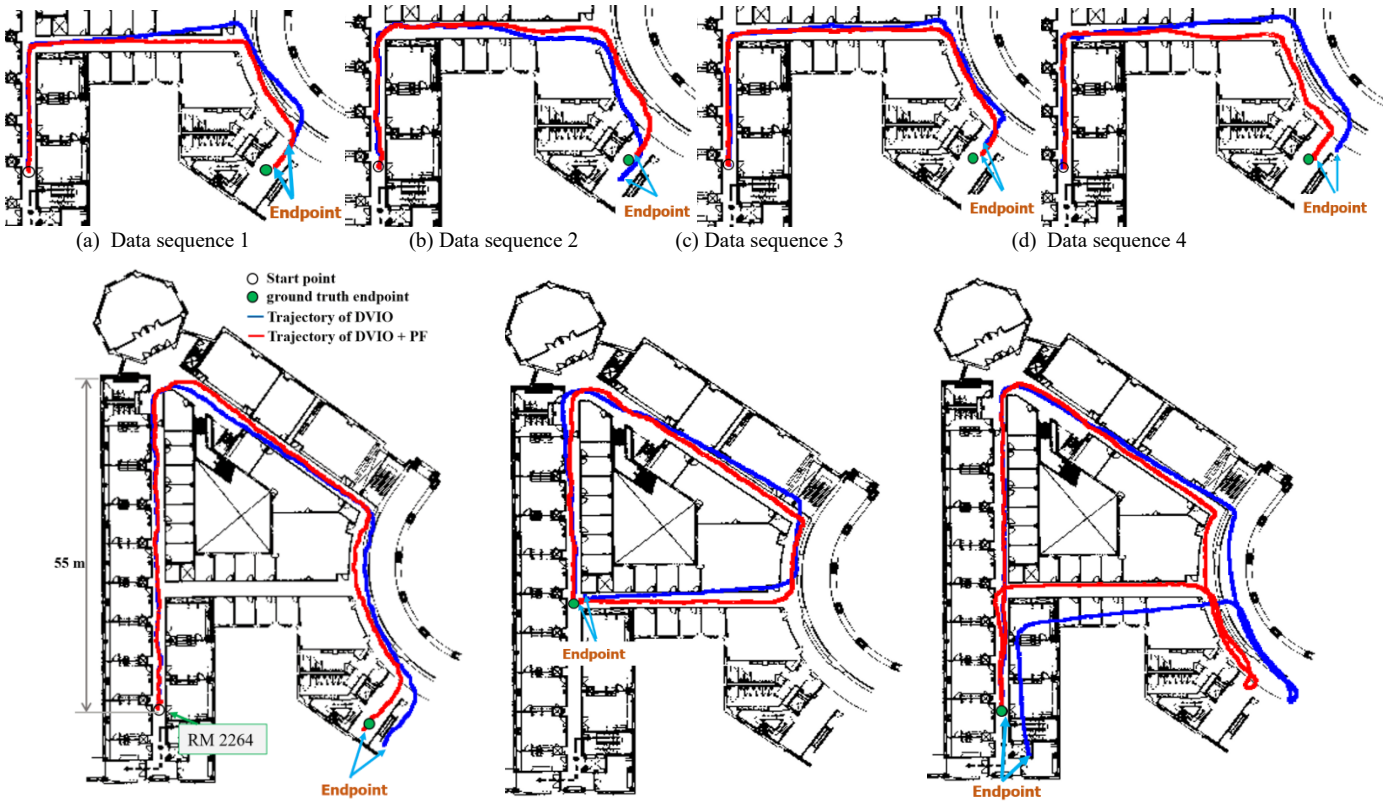
TABLE V: EPENS OF WAYFINDING EXPERIMENTS

Task \ Person	A	B	Mean
I	0.20 m	0.30 m	0.25 m
	0.10 m	0.20 m	0.15 m
Mean	0.15 m	0.25 m	0.20 m
II	0.70 m	0.50 m	0.60 m
	0.40 m	0.20 m	0.30 m
Mean	0.55 m	0.35 m	0.45 m

In these wayfinding experiments, we placed numerous obstacles along the paths to test the assistive navigation system's obstacle avoidance function. The results show that the obstacle avoidance module functioned well and the ART successfully steered RNA into an obstacle-free direction toward the destination. As this is beyond the focus of this paper, we omit the details for simplicity. Successful obstacle avoidance reflects accurate pose estimation of PFL from a different aspect.

VII. CONCLUSION AND FUTURE WORK

This paper presents a new VIO method, called DVIO, for 6-DOF pose estimation of an RGB-D-camera-based VINS. The method achieves better accuracy by using the geometric feature (the floor plane extracted from the camera's depth data) to add constraints between the graph nodes to reduce the accumulative pose error. Specifically, it tightly couples the floor plane, the visual features, and the IMU's inertial data in a graph optimization framework for pose estimation. Based on the characterization of the camera's depth measurements, visual features are classified into ones with a near-range depth and



(a) Data sequence 1 (b) Data sequence 2 (c) Data sequence 3 (d) Data sequence 4

(e) Data sequence 5 (f) Data sequence 6 (g) Data sequence 7

Fig. 9 Trajectories estimated by DVIO and PFL (DVIO+PF) on the floor plan of the Engineering East Hall. The start and endpoint for DS6/DS7 are the same.

ones with a far-range depth. For near-range visual features, the depth values are initialized and updated by directly using the camera's depth measurements because these measurements are accurate. For far-range visual features, the depths are regarded as unknown values because the camera's depth measurements are less accurate and therefore, the epipolar plane model is used to create constraints between the related nodes in the graph. The use of the floor plane and the inclusion of both visual features with and without a depth value improved the pose estimation accuracy. To support wayfinding application in a large indoor space, a PFL method is devised to limit the accumulative pose error of DVIO by using the information of the operating environment's floor plan. The PFL method builds a 2D local grid map by using the DVIO-estimated egomotion and aligns this map with the floor plan map to minimize the pose error. PFL and DVIO form a VPS for accurate device localization on the 2D floor plan map.

We validated the VPS' localization function in the context of assistive navigation RNA in a large indoor space. To extend VPS into a full navigation system, we developed other essential software modules, including Data Acquisition, Path Planning, Obstacle Avoidance, and ART Control. The ART mechanism can steer RNA into the desired direction of travel to guide the visually impaired user to avoid obstacles and move towards the destination. Experimental results validate that: 1) DVIO has better pose estimation accuracy than state-of-the-art VIO and it achieves real-time computation on a UP Board computer; 2) PFL can substantially reduce DVIO's accumulative pose error for localization in a floor plan; and 3) VPS can be effectively used for assistive navigation in a large indoor space for both wayfinding and obstacle avoidance.

In terms of future work, we will recruit visually impaired human subjects to conduct experiments in various indoor environments to validate the assistive navigation function of the RNA prototype.

REFERENC

[1] R. R A Bourne, S. R. Flaxman, T. Braithwaite, M.V. Cicinelli, *et al.*, "Magnitude, temporal trends, and projections of the global prevalence of blindness and distance and near vision impairment: a systematic review and meta-analysis," *Lancet Glob Healthm*, vol. 5, no. 9, pp. 888-897, 2017.

[2] J. M. Saez, F. Escolano, and A. Penalver, "First steps towards stereo-based 6-DOF SLAM for the visually impaired," in *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2005, pp. 23-23.

[3] V. Pradeep, G. Medioni, and J. Weiland, "Robot vision for the visually impaired," in *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2010, pp. 15-22.

[4] Y. H. Lee and G. Medioni, "RGB-D camera based navigation for the visually impaired," in *Proc. RSS Workshop on RGB-D: Advanced Reasoning With Depth Cameras*, 2011, pp. 1-6.

[5] C. Ye, S. Hong, X. Qian, and W. Wu, "Co-robotic cane: A new robotic navigation aid for the visually impaired," *IEEE Systems, Man, and Cybernetics Magazine*, no. *2*, vol. 2, 33-42, 2016.

[6] H. Zhang and C.Ye, "An Indoor navigation aid for the visualy impaired," in *Proc. IEEE Int. Conf. Robotics and Biomimetics*, 2016, pp. 467-472.

[7] B. Li, J.P. Munoz, X. Rong, Q. Chen, *et al.*, "Vision-based mobile indoor assistive navigation aid for blind people," *IEEE Transactions on Mobile Computing,* vol. 18, no. 3, pp. 702-714, 2018.

[8] H. Zhang, L. Jin, and C. Ye, "A depth-enhanced visual inertial odometry for a robotic navigation aid for blind people," in *Proc. Visual-Inertial*

Navigation: Challenges and Applications Workshop at *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems*.

[9] C. Ye, S. Hong, and A. Tamjidi, "6-DOF Pose Estimation of a Robotic Navigation Aid by Tracking Visual and Geometric Features," *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 4, pp. 1169-1180, 2015.

[10] S. Treuillet, E. Royer, T. Chateau, M. Dhome, *et al*., "Body Mounted Vision System for Visually Impaired Outdoor and Indoor Wayfinding Assistance," in *Proceedings of Conference on Assistive Technologies for People with Vision and Hearing Impairments*, 2007.

[11] K. Wang, W. Wang, and Y. Zhuang, "A Map Approach for Vision-based Self-localization of Mobile robot," *Acta Automatica Sinica,* vol. 34, no. 2 pp. 159-166, 2008.

[12] D. Ahmetovic, C. Gleason, C. Ruan, K. Kitani, *et al.,* "NavCog: a navigational cognitive assistant for the blind," in *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services*, 2016.

[13] A. Ganz, J. M. Schafer, S. Gandhi, E. Puleo, *et al*., "PERCEPT indoor navigation system for the blind and visually impaired: architecture and experimentation," *International Journal of Telemedicine and Applications,* 2012. DOI: 10.1155/2012/ 894869.

[14] A. Ganz, J. M. Schafer, Y. Tao, C. Wilson *et al*., "PERCEPT-II: Smartphone based indoor navigation system for the blind," in *Proceedings of the 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 2014, pp. 3662-3665.

[15] H. Zhang, L. Jin, H. Zhang, and C. Ye, "A Comparative Analysis of Visual-Inertial SLAM for Assisted Wayfinding of the Visually Impaired," in *Proc. IEEE Winter Conference on Applications of Computer Vision*, 2019, pp. 210-217.

[16] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, *et al*., "Keyframe-Based Visual-Inertial SLAM using Nonlinear Optimization," *International Journal of Robotics Research*, vol. 34, no. 3, pp. 314-334, 2015.

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

[18] R. Mur-Artal and J. D. Tardós, "Visual-inertial monocular SLAM with map reuse," *IEEE Robotics and Automation Letters* 2.2 (2017): 796-803.

[19] S. Weiss, M. W. Achtelik, S. Lynen, M. Chli, *et al*., "Real-time onboard visual-inertial state estimation and self-calibration of MAVs in unknown environments," in *Proc. IEEE International Conference on Robotics and Automation*, 2012, pp. 957-964.

[20] V. Indelman, S. Williams, M. Kaess, and F. Dellaert, "Information fusion in navigation systems via factor graph based incremental smoothing" *Robotics and Autonomous Systems*, vol. 61, no. 8, pp. 721–738, 2013.

[21] W. Zheng, F. Zhou, and Z. Wang, "Robust and accurate monocular visual navigation combining IMU for a quadrotor," *IEEE/CAA Journal of Automatica Sinica* 2.1 (2015): 33-44.

[22] A. I. Mourikis and S. I. Roumeliotis, "A Multi-State Constraint Kalman Filter for Vision-aided Inertial Navigation," *Proceedings IEEE International Conference on Robotics and Automation*, 2007.

[23] C. Campos, R. Elvira, J. J. G. Rodriguez, J. M. Montiel, *et al*., "ORB-SLAM3: An accurate open-source library for visual, visual-inertial and multi-map SLAM," *arXiv preprint arXiv:2007.11898*, 2020.

[24] C. Campos, J. M. Montiel, and J. D. Tardós, "Inertial-Only Optimization for Visual-Inertial Initialization," *arXiv preprint arXiv:2003.05766*, 2020.

[25] T. Qin and S. Shen, "Robust initialization of monocular visual-inertial estimation on aerial robots," in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017.

[26] J. Delmerico and D. Scaramuzza, "A benchmark comparison of monocular visual-inertial odometry algorithms for flying robots," in *Proc. of IEEE International Conference on Robotics and Automation*, 2018.

[27] K. Sun, K. Mohta, B. Pfrommer, M. Watterson, *et al*., "Robust stereo visual inertial odometry for fast autonomous flight," *IEEE Robotics and Automation Letters,* 3.2 (2018): 965-972.

[28] N. Brunetto, S. Salti, N. Fioraio, T. Cavallari, *et al.*, "Fusion of inertial and visual measurements for RGB-D SLAM on mobile devices," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2015, pp. 1–9

[29] Y. Ling, H. Liu, X. Zhu, J. Jiang, *et al*., "RGB-D inertial odometry for indoor robot via keyframe-based nonlinear optimization," in *Proc. of IEEE Int. Conf. on Mechatronics and Automation*, 2018, pp. 973–979.

[30] Z. Shan, R. Li, and S. Schwertfeger, "Rgbd-inertial trajectory estimation and mapping for ground robots," *Sensors*, 19(10):2251, 2019

[31] J. Shi, "Good features to track," in *Proc. of IEEE conference on computer vision and pattern recognition*, 1994, pp. 593–600.

[32] V. Lepetit, F. Moreno-Noguer, and P. Fua, "Epnp: An accurate o (n) solution to the pnp problem," *International journal of computer vision*, 81(2):155, 2009.

[33] F. Boniardi, T. Caselitz, R. Kummerle, and W. Burgard, "Robust LiDAR-based localization in architectural floor plans," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017, pp. 3318-3324.

[34] F. Boniardi, T. Caselitz, R. Kummerle, and W. Burgard, "A pose graph-based localization system for long-term navigation in CAD floor plans," *Robotics and Autonomous Systems*, 2019, 112: 84-97.

[35] Y. Watanabe, K. R. Amaro, B. llhan, T. Kinoshita, *et al*., "Robust Localization with Architectural Floor Plans and Depth Camera," in *IEEE/SICE International Symposium on System Integration*, 2020.

[36] A. Segal, D. Hhnel, and S. Thrun, "Generalized-ICP," in *Proc. Robotics: Science and Systems*, 2009

[37] S. Seifzadeh, B. Khaleghi, and F. Karray, "Distributed soft-data-constrained multi-model particle filter," *IEEE Transactions on Cybernetics,* 2014, 45.3: 384-394.

[38] W. Winterhalter, F. Fleckenstein, B. Steder, L. Spinello, *et al*., "Accurate indoor localization for RGB-D smartphones and tablets given 2D floor plans," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015, pp. 3138-3143.

[39] http://www.up-board.org/up

[40] H. Zhang and C. Ye, "Human-Robot Interaction for Assisted Wayfinding of a Robotic Navigation Aid for the Blind," in *Proc. IEEE International Conf. on Human System Interaction*, 2019, pp. 137-142.

[41] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, New York, 2004.

[42] B. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proc. Imaging Understanding Workshop*, 1981, pp. 121-130.

[43] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 22, no. 11, pp. 1330–1334, 2000.

[44] H. Zhang and C. Ye, "Plane-Aided Visual-Inertial Odometry for 6-DOF Pose Estimation of a Robotic Navigation Aid," *IEEE Access* 8, 2020, pp. 90042-90051.

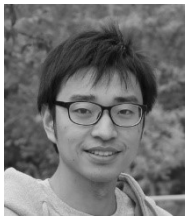
[45] H. Zhang and C. Ye, "An indoor wayfinding system based on geometric features aided graph SLAM for the visually impaired," *IEEE Transactions on Neural Systems and Rehabilitation Engineering,* vol. 25, no. 9, 1592-1604, 2017.

[46] C. Ye, "T-transformation: a new traversability analysis method for terrain navigation," in *Proc. SPIE Defense and Security Symposium,* 2004.

[47] C. Ye, "A Method for Mobile Robot Obstacle Negotiation," *International Journal of Intelligent Control and Systems*, vol. 10, no. 3, pp. 188-200, 2005.

[48] O. Wulf, K. O. Arras, H. I. Christensen, and B. Wagner, "2D mapping of cluttered indoor environments by means of 3D perception," in *Proc. IEEE International Conference on Robotics and Automation, 2004*.

[49] G. Giorgio, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," *IEEE Transactions on Robotics,* vol. 23, no. 1, pp. 34-46, 2007.



He Zhang He Zhang received the B.E. degree from 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 Postdoc with the Department of Computer Science, in Virginia Commonwealth University. His research interests include simultaneous localization and mapping, visual-inertial odometry, rehabilitation robotics and 2D/3D computer vision.

Lingqiu Jin received BS degrees in Electrical Engineering & Mathematics from the Pennsylvania State University, in 2015 and M.S. in Electrical Engineering, Columbia University in the City of New York, in 2016. He is currently a Ph.D student with the Department of Computer Science, Virginia Commonwealth University. His research interests include simultaneous localization and mapping, wearable robotics, 2D/3D computer vision on mobile devices.

Cang Ye (S'97–M'00–SM'05) received the B. E. and M. E. degrees from the University of Science and Technology of China, Hefei, Anhui, 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 system.