A Depth-Enhanced Visual Inertial Odometry for a Robotic Navigation Aid for Blind People

He Zhang, Lingqiu Jin, and Cang Ye

Abstract— **This paper presents a new method, called depth-enhanced visual-inertial odometry (DVIO), for real-time pose estimation of a robotic navigation aid (RNA) for assistive wayfinding. The method estimates the device pose by using an RGB-D camera and an inertial measurement unit (IMU). It extracts the floor plane from the camera's depth data and tightly couple the floor plane, the visual features (with depth data from the RGB-D camera or unknown depth), and the IMU's inertial data in a graph optimization framework for 6-DOF pose estimation. Due to use of the floor plane and the depth data from the RGB-D camera, the DVIO method has a better pose estimation accuracy than its VIO counterpart. To enable real-time computing on the RNA, the size of the sliding window for the graph optimization is reduced to trade some accuracy for computational efficiency. Experimental results demonstrate that the method achieved a pose estimation accuracy similar to that of the state of the art VIO but ran at a much faster speed (with a pose update rate of 18 Hz).**

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 people will become blind in the coming decades. Therefore, there is a crucial need in developing navigation aids to help the blind with their daily mobility need and live independent lives. The problem of independent mobility for a blind person 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. In the literature, a number of Robotic Navigation Aids (RNAs) [2]-[3] have been introduced to assist the blind people for wayfinding and/or obstacle avoidance. Among these RNAs, vision-based systems are becoming more popular because the cameras used in these RNAs can provide all of the needed information for navigation, including 6-DOF device pose (position and orientation) estimation and obstacle/object detection. The pose information of an RNA can be used to build a 3D map of the environment, locate the blind traveler in the environment, and guide the traveler to the destination. In the existing literature, monocular camera [4], stereo-cameras [5], [6], RGB-D cameras [7], [8], [9] or 3D time-of-flight (TOF) [10], [11], have been used in these RNAs. Because of the large amount of data to be processed for navigational decision making, these RNAs require an off-board computer, such as a server [6], [8], [10], [11], a laptop [4], [5], [7], [9], or a tablet computer (e.g., Google Tangle with a quad-core Nvidia Tegra K1 processor [12], [3]) to run the compute-intensive navigation software. This approach has hampered the practical use of RNAs. To address this issue, more computationally efficient methods must be developed for real-time computing on the RNA that has limited computing power. This paper concerns itself with real-time and robust RNA pose estimation for the wayfinding application.

In the robotics community, a visual-inertial system (VINS), consisting of a monocular camera and an inertial measurement unit (IMU), has been popularly used for robust motion estimate. A VINS employs a technique that couples the camera's visual data with the inertial data to estimate the camera's ego-motion. Such a technique is termed visual-inertial odometry (VIO) in the literature. VIO uses a filtering/smoothing algorithm to estimate the state vector consisting of the system's pose, velocity, IMU biases, and the depth values of the visual features. The technique requires computing the depth values by structure-from-motion (SFM) for hundreds of visual features to initialize these features and re-computing them over the course of pose estimation. The computation can be time-consuming and the update of the depth values is subject to the pose estimation error. In this paper, we propose a so-called depth-enhanced visual-inertial odometry (DVIO) to estimate the RNA's pose for wayfinding. The DVIO method uses a sensor package including an RGB-D camera and an IMU for pose estimation. The method improves the state-of-the-art VIO approach's performance in terms of pose estimation accuracy and computational time by: 1) using the geometric feature (the floor plane extracted from the camera's depth data) to create additional constraints between nodes of the graph to limit the accumulative pose error; 2) using depth data directly from the RGB-D camera for visual feature initialization to avoid the computation incurred by the SFM and the update of the visual features' depth values. To further reduce the computational cost, the proposed method trades some pose estimation accuracy with computation speed by using a smaller number of nodes for pose graph optimization. This treatment allows the method to attain a pose estimation accuracy equivalent to the state-of-the-art VIO method but run in a much faster speed (with an 18 Hz pose update rate). The computational efficiency allows for real-time computation of the entire wayfinding system, consisting of the DVIO and other modules such as

This work was supported by the NIBIB and the NEI of the NIH under award R01EB018117H. H. Zhang, L. Jin and C. Ye are with Computer Science Department, Virginia Commonwealth University, Richmond, VA 23284, USA. (e-mail: `hzhang8@vcu.edu`, `jinl@mymail.vcu.edu`, `cye@vcu.edu`).

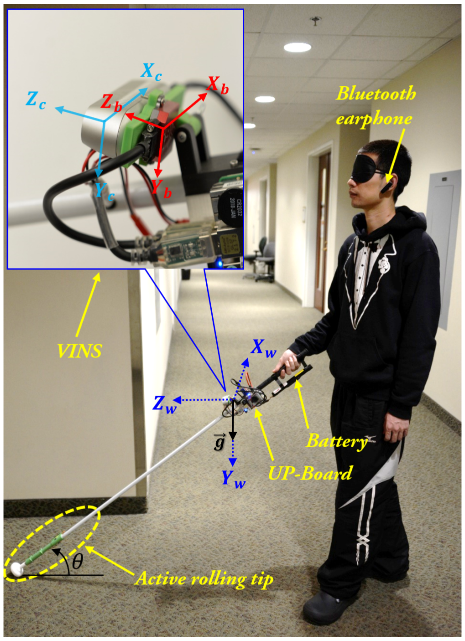

Fig. 1. The RNA prototype. The RNA body, camera, and world coordinate systems are denoted by $\{B\}$, $\{C\}$, $\{W\}$, respectively.

Data Acquisition, Path Planner, Obstacle Avoidance, etc., on a credit-card-sized board computer. As a result, the system's hardware is compact for installation on a traditional white cane and make a highly portable RNA possible.

II. RNA Prototype

As depicted in Fig. 1, the RNA uses an Intel Realsense D435 (RGB-D) Camera and an IMU (VN100 of VectorNav Technologies, LLC) for motion estimate. 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. 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 200 Hz. The RNA uses a mechanism called active rolling tip (ART) [11] to steer the cane to the desired direction of travel to guide the user. The ART consists of a rolling tip, a gearmotor, a motor drive, and a clutch. A custom control board is built to engage and disengage the clutch. When it is engaged, the gearmotor drive the rolling tip and steer the cane. When it is disengaged, the rolling tip is disconnected with the gearmotor and the user can swing the RNA just like using a white cane. Both clutch controller is controlled by the general IO port and the motor drive are controlled by the RS-232 port of the UP-Board.

III. Notations and Task Description

The coordinate systems of the IMU and the camera of the RNA, denoted as $\{B\}\,(X_bY_bZ_b)$ and $\{C\}\,(X_cY_cZ_c)$, are shown in Fig. 1 The initial $\{B\}$ is taken as the world coordinate system $\{W\}\,(X_wY_wZ_w)$ after performing a rotation to make the Z-axis level and align the Y-axis with the gravity vector $\overrightarrow{g}$. In this paper, we use interchangeably rotation matrices $\mathbf{R}$ and Hamilton quaternions $\mathbf{q}$ to describe a rotation. The right subscript k is used to indicate camera frames. b_k and c_k are the body coordinate system and the camera coordinate system when capturing the k^{th} frame. At the time when the k^{th} frame is obtained, the IMU's state is denoted by $\mathbf{x}_{b_k}^{w} = \{\mathbf{t}_{b_k}^{w}, \mathbf{v}_{b_k}^{w}, \mathbf{q}_{b_k}^{w}, \mathbf{b}_a, \mathbf{b}_g\}$, which contains translation, velocity, rotation, and accelerometer bias and gyroscope bias of the IMU. The IMU's 3D pose is denoted as $\xi_{b_k}^{w} = \{\mathbf{t}_{b_k}^{w}, \mathbf{q}_{b_k}^{w}\}$. The transformation from $\{C\}$ to $\{B\}$ is pre-calibrated and denoted as $\mathbf{T}_c^b = [\mathbf{R}_c^b \; \mathbf{t}_c^b]$ is obtained ahead of time by calibration. The camera pose in $\{B\}$ is $\xi_c^b = \{\mathbf{t}_c^b, \mathbf{q}_c^b\}$. The intrinsic parameters of the color camera and the depth camera have been calibrated and the data association between the cameras has been established.

IV. Wayfinding Software

The wayfinding software system of the RNA was developed based on the ROS framework. Each ROS node is an independent function module that communicates with others through a messaging mechanism. The pipeline of the software is depicted in Fig. 2 The Data Acquisition node acquires and publishes the camera's and the IMU's data, which is subscribed by the DVIO node for real-time pose estimation and 3D map building. The 3D map is a point cloud map that is generated by registering the camera's depth data acquired at different positions. The 3D map and the 3D pose are sent to the Path Planner node to calculate the shortest path from the RNA current location to the destination. It uses our path planning method [10] to construct a graph by using the points-of-interest (hallway junctions, entrances, stairways, etc.) and uses the graph to determine the shortest path, based on which the RNA's heading for path tracking is determined. The heading is then adjusted by the Obstacle Avoidance (OA) node. The OA module extracts the floor plane from the 3D point cloud map and treats the point clusters above the floor plane as obstacles. It then projected the point data of the obstacles onto the floor plane to generate a 2D occupancy map and compute numerous candidate directions for obstacle avoidance [13]. The direction that is closest to the path-tracking direction is selected as the desired movement direction for the RNA, from which the needed motion control parameters for the motor drive are determined and used to control the motor to steer the cane. The OA node also generates the navigational message (i.e., the text related to the desired turning angle), which is conveyed to the traveler via the Bluetooth earphone after text-to-speech conversion.

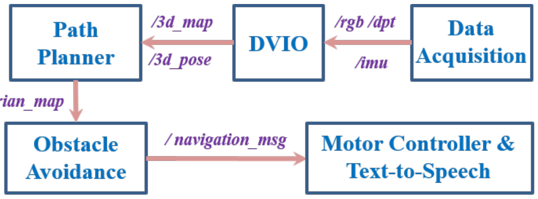

Fig. 2. Software pipeline

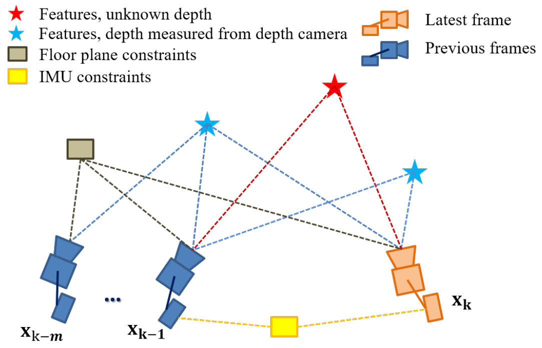

Fig. 3. An example graph of the DVIO method

V. DVIO

The DVIO consists of three components: feature tracker, floor detector and state estimator. The feature tracker extracts visual features from the color image and tracks them across images. The floor detector extracts the floor plane from the depth data. The state estimator estimates the IMU's state using the tracked visual features and their depth data, the extracted floor plane, and the IMU measurements. The IMU's state includes the pose, velocity, and biases. The details of the components are described below.

A. Feature Tracker

The feature tracker detects Harris [14] corner features for each image frame. To retain an affordable computation cost, each image is evenly divided into 8×8 patches. At most 4 features are extracted and tracked for each patch. Therefore, a maximum of 256 feature points are detected for each frame. These features are tracked across image frames by employing the KLT [15] method. A fundamental matrix based RANSAC process is devised to remove the outliers and the inliers are passed to the state estimator for pose estimation.

B. Floor Plane Detector

The floor plane is extracted from the first frame and tracked across frames. In this work, the floor plane is described by its normal and the distance from the origin to the plane. At the time the world coordinate system is initialized, the floor plane's normal is [0, 0, 1] and the distance can be approximated as $\hat{d}_f^w = Lsin(\theta)$. Data points with a z-coordinate value in $[\hat{d}_f^w - \delta, \hat{d}_f^w + \delta](\delta = 10cm)$ are used to extract the floor plane. After the extract, the floor plane will be tracked into the next frame by using the pose estimated by the DVIO for the current frame.

C. State Estimator

State estimation is formulated as a graph optimization problem. Each graph node represents the state vector and the edge between two nodes represent the constraint between them. The state vector is defined as $\chi_k = \{\mathbf{x}_{b_k}^w, \mathbf{x}_{b_{k-1}}^w, ..., \mathbf{x}_{b_{k-m}}^w\}$, where $\mathbf{x}_{b_k}^w = \{\mathbf{t}_{b_k}^w, \mathbf{v}_{b_k}^w, \mathbf{q}_{b_k}^w, \mathbf{b}_a, \mathbf{b}_g\}$ is the IMU's state at the time when the k^{th} image frame is captured, and m ($m = 3$ in our implementation) is the number of nodes used for graph optimization. m is called the size of the sliding window. Similar to Tong's method [16], the visual features with a known depth are used to estimate χ_k in the graph optimization process. According to the works in [17] and [18], the visual features with unknown depth are also useful for state estimation as they contain the information about the rotation and the direction of translation of the VINS. Therefore, we also use these features to create edges in the graph to incorporate them into state estimation. Moreover, the extracted floor plane is incorporated into the graph to further reduce the pose estimation error by using our previous method [19]. Fig. 3 shows one example graph of the DVIO method.

The optimization problem is to find a maximum a posteriori pose estimation that minimizes the sum of the Mahalanobis norms of all measurement residuals given by:

$$\begin{aligned}\chi_k^* \quad = \quad & \underset{\chi_k}{argmin}(\sum_{(k-1,k)\in C_k} \left(\boldsymbol{r}_{k-1,k}^b\right)^2 + \sum_{(f,k)\in C_p} \left(\boldsymbol{r}_{f,k}^p\right)^2 \\ & + \sum_{(i,k)\in C_{v1}} \left(\boldsymbol{r}_{i,k}^v\right)^2 + \sum_{(i,k)\in C_{v2}} \left(\boldsymbol{r}_{i,k}^v\right)^2) \qquad (1)\end{aligned}$$

where $\boldsymbol{r}_{k-1,k}^b$, $\boldsymbol{r}_{f,k}^p$ and $\boldsymbol{r}_{i,k}^v$ are the residual errors relate to the IMU, floor plane, and visual feature measurements, respectively. C_p, C_k, C_{v1}, and C_{v2} represent the set of edges for the floor plane, IMU pre-integration, visual features with a known depth, and visual features with an unknown depth, respectively. We employ the Ceres solver to solve this nonlinear problem. To do so, we need to define the function for each measurement residual and the Jacobian matrix with respect to the variables of χ_k. In this work, $\boldsymbol{r}_{k-1,k}^b$ is defined in the same way as Tong's method [16] while $\boldsymbol{r}_{f,k}^p$ is defined by using our earlier method [19]. The residual functions and the Jacobians related to $\boldsymbol{r}_{i,k}^v$ for visual features with a known depth and unknown depth are described later in this Section. 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 carry out an experiment to characterize the D435 camera. The result is shown in Fig. 4 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 ($\leq$ 2.2 m) visual feature is assigned the depth measurement from the RGB-D camera and a far-range visual feature is assigned an unknown depth.

1) Visual Features with a known depth: According to Tong's method [16], the residual for a visual feature that has been observed in the i^{th} and k^{th} images is computed as

$$\boldsymbol{r}_{i,k}^v = \boldsymbol{p}^{c_k} - \frac{\hat{\boldsymbol{p}}^{c_k}}{|\hat{\boldsymbol{p}}^{c_k}|} \qquad (2)$$

where $\boldsymbol{p}^{c_k} = \boldsymbol{\pi}_c^{-1}\left[u^{c_k}, v^{c_k}\right]^T$, $\hat{\boldsymbol{p}}^{c_k} = (\xi_c^b)^{-1}\hat{\boldsymbol{p}}^{b_k}$, $\hat{\boldsymbol{p}}^{b_k} = \xi_{b_i}^{b_k}\xi_c^b(\boldsymbol{p}^{c_i}\rho_i)$, $\boldsymbol{p}^{c_i} = \boldsymbol{\pi}_c^{-1}\left[u^{c_i}, v^{c_i}\right]^T$. $\boldsymbol{\pi}_c^{-1}$ is the inverse

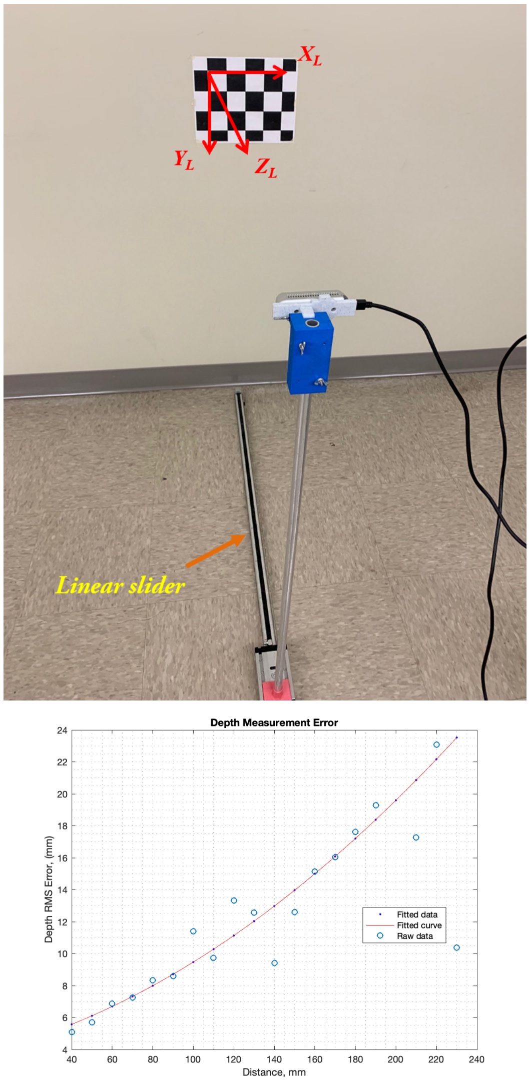

Fig. 4. Characterization of the D435's depth measurement: the linear motion table moves the D435 from 40 mm to 240 mm with a step-size of 10 mm. At each position, 300 frames of depth data were captured and used to compute the mean and RMS of the measurement error. The method in [20] was employed to estimate the ground truth depth, which is then refined by using the known camera movement (10 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.

projection that maps a pixel from the image onto the normalization plane (defined by $z^{c_*} = 1$). u and v represent the coordinates of the visual feature on the image plane while superscript k and i denote the k^{th} and i^{th} images, respectively. ρ_i is the depth of the visual feature on the i^{th} camera frame. The Jacobian matrices of $\boldsymbol{r}^v_{i,k}$ with respect to the IMU poses, $\xi^w_{b_k}$ and $\xi^w_{b_i}$, can be found in the open-source implementation [16]. Apparently, the error of ρ_i plays a critical role in attaining a good pose estimation accuracy for DVIO. For this reason, the depth data of a visual feature is used only if it is no greater than 2.2 meters. In addition, the depth data remains constant during the graph optimization process. This is advantageous over a monocular VIO method that needs to update the depth value throughout the pose estimation process. The re-computation of the depth can introduce error and degrade the pose estimation accuracy. Unlike a monocular VIO method that uses the depth estimate of a feature point at its first observation [16], DVIO uses the smallest depth value of the frames within the sliding window for ρ_i. In addition, if the feature is tracked into the next frame with a smaller depth, then ρ_i is updated with that depth value. These treatments are to minimize measurement error for ρ_i.

2) *Visual Feature Measurement with unknown depth:* The residual function for the visual features with an unknown depth is based on epipolar geometry. A feature point is observed as visual features $\boldsymbol{X}^{c_i} = [u^{c_i}, v^{c_i}, 1]^T$ and $\boldsymbol{X}^{c_k} = [u^{c_k}, v^{c_k}, 1]^T$ on the i^{th} and k^{th} images, respectively. Given the ground truth camera motion, $\boldsymbol{X}^{c_i}$, $\boldsymbol{X}^{c_k}$, $\boldsymbol{O}_{c_i}$, and $\boldsymbol{O}_{c_k}$ should stay on the epipolar plane, where $\boldsymbol{O}_{c_i}$ and $\boldsymbol{O}_{c_k}$ are the camera focus points for image frames i and k, respectively. Since the camera motion is estimated, $\boldsymbol{X}^{c_k}$ is off the plane. The distance between $\boldsymbol{X}^{c_k}$ and the epipolar plane is computed as the residual error

$$\boldsymbol{r}^v_{i,k} = \left(\mathbf{R}^{c_i}_{c_k} \boldsymbol{X}^{c_k}\right)^T \left(\left[\mathbf{t}^{c_i}_{c_k}\right]_\times \boldsymbol{X}^{c_i}\right) \tag{3}$$

where $\mathbf{R}^{c_i}_{c_k} = \left(\mathbf{R}^w_{c_i}\right)^{-1} \left(\mathbf{R}^w_{c_k}\right)$ and $\mathbf{t}^{c_i}_{c_k} = \left(\mathbf{R}^w_{c_i}\right)^{-1} \left(\mathbf{t}^w_{c_k} - \mathbf{t}^w_{c_i}\right)$. The Jacobian matrices of $\boldsymbol{r}^v_{i,k}$ with respect to poses $\xi^w_{b_k}$ and $\xi^w_{b_i}$ are computed using the chain rule as follows: $\frac{\partial \boldsymbol{r}^v_{i,k}}{\partial \xi^w_{b_i}} = \frac{\partial \boldsymbol{r}^v_{i,k}}{\partial \xi^{c_i}_{c_k}} \frac{\partial \xi^{c_i}_{c_k}}{\partial \xi^w_{b_i}}$, $\frac{\partial \boldsymbol{r}^v_{i,k}}{\partial \xi^w_{b_k}} = \frac{\partial \boldsymbol{r}^v_{i,k}}{\partial \xi^{c_i}_{c_k}} \frac{\partial \xi^{c_i}_{c_k}}{\partial \xi^w_{b_k}}$

$$\frac{\partial \boldsymbol{r}^v_{i,k}}{\partial \xi^{c_i}_{c_k}} = \begin{bmatrix} \left(\left[\mathbf{R}^{c_i}_{c_k} \boldsymbol{X}^{c_k}\right]_\times \boldsymbol{X}^{c_i}\right)^T & -\left(\left[\mathbf{t}^{c_i}_{c_k}\right]_\times \boldsymbol{X}^{c_i}\right)^T \mathbf{R}^{c_i}_{c_k} \left[\boldsymbol{X}^{c_k}\right]_\times \end{bmatrix} \tag{4}$$

$$\frac{\partial \xi^{c_i}_{c_k}}{\partial \xi^w_{b_i}} = \begin{bmatrix} -\left(\mathbf{R}^w_{c_i}\right)^{-1} & \left[\mathbf{t}^{c_i}_{c_k}\right]_\times \left(\mathbf{R}^b_c\right)^{-1} + \left(\mathbf{R}^w_{c_i}\right)^{-1} \mathbf{R}^w_{b_i} \left[\mathbf{t}^b_c\right]_\times \\ \mathbf{0}_{3\times3} & -\left(\mathbf{R}^{c_i}_{c_k}\right)^{-1} \left(\mathbf{R}^b_c\right)^{-1} \end{bmatrix} \tag{5}$$

$$\frac{\partial \xi^{c_i}_{c_k}}{\partial \xi^w_{b_k}} = \begin{bmatrix} \left(\mathbf{R}^w_{c_i}\right)^{-1} & -\left(\mathbf{R}^w_{c_i}\right)^{-1} \mathbf{R}^w_{b_k} \left[\mathbf{t}^b_c\right]_\times \\ \mathbf{0}_{3\times3} & \left(\mathbf{R}^b_c\right)^{-1} \end{bmatrix} \tag{6}$$

In this way, the feature points whose depth measurements are greater than 2.2 meters are also added to the graph for pose estimation without using their depth data (which incur large error). These feature points contribute to the estimation for the IMU's rotation and the direction of the IMU's translation.

VI. Experiments

A. DVIO Performance Evaluation

The performance of the DVIO method is compared with three state-of-the-art VIO methods, ROVIO [21], OKVIS [22], and VINS-Mono [23], by experiments. Ten datasets were collected by holding the RNA and walking along a 20-meters straight at a speed of ∼0.7 m/s. The ground truth position of the endpoint is [0, 0, 20]. We use the end-point-error-norm (EPEN) to evaluate the accuracy of the pose estimation methods. The 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 on the RNA, we use a small window size (4 pose-nodes), which trades some accuracy for speed. The results on pose estimation accuracy are tabulated in Table

TABLE I
COMPARISON FOR THE FINAL END POSITION ERRORS

Dataset	ROVIO	OKVIS	VINS-Mono	DVIO
1	1.70	2.24	1.24	0.57
2	1.40	3.54	0.99	0.55
3	1.80	2.24	0.51	1.24
4	1.14	1.44	0.97	0.82
5	1.18	1.19	0.73	0.79
6	0.72	1.37	0.69	1.23
7	1.94	3.05	1.23	1.07
8	3.51	6.83	0.24	1.06
9	4.70	6.59	1.39	0.78
10	4.06	6.54	0.56	1.04
Mean, Std (meters)	2.22, 1.30	3.50, 2.18	0.85, 0.35	0.92, 0.24

I and the run times on the Up Board are plotted in Fig. 5 From Table I, we can see that the DVIO achieves a pose estimation accuracy that is comparable to (slightly worse than) VINS-Mono and much better than OKVIS and ROVIO. (OKVIS and ROVIO cannot be used for our application due to their much worse accuracies.) Fig. 5 shows that DVIO is about two times faster than VINS-Mono. It is run time performance is equal to OKVIS but worse than ROVIO. The low computational cost of DVIO allows for the real-time implementation (∼18 fps) of the RNA's wayfinding software (see Fig. 2). It is noted that our VINS does not have hardware level time synchronization between the camera and the IMU data. VINS-Mono calibrated the time offset between the camera and the IMU in its optimization process [23] and thus resulted in a more accurate pose estimation result. Although DVIO does not consider this time offset, it achieves comparable accuracy to that of VINS-Mono.

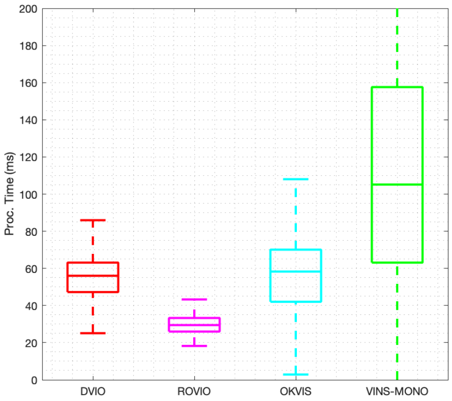

Fig. 5. Run time on UP-Board

B. Wayfinding experiment of the RNA

To test the practicability of the DVIO in assistive wayfinding of RNA, a sighted human subject (blindfolded) is recruited to perform a navigation task (Fig. 6) from Room 2264 to Room 2253 in the West Engineering building five times. Seven obstacles are placed evenly along the 30-meter path. The human subject stopped at a point when the RNA indicated that the destination had been reached. If the user stopped at a point close enough to the destination (within 1.5 meters), the test was regarded as a successful one. Otherwise, it was a failure. In each test, the number of times that the RNA hits an obstacle is recorded. The experimental results show that: 1) out of five tests, four were successful; 2) the RNA succeed in guiding the user to avoid the obstacles thirty-one times and failed four times in obstacle avoidance. This means that the RNA is able to provide assistance in indoor wayfinding and obstacle avoidance. One test case of the RNA is given in the attached video clip.

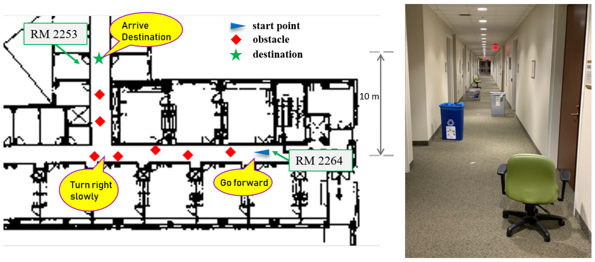

Fig. 6. Left: RNA Test; Right: Scenario snapshot at start point

VII. CONCLUSIONS AND FUTURE WORK

In this paper, a new DVIO method is introduced to estimate the pose of an RNA for assistive wayfinding. The method produces in real-time accurate pose estimation, which is used for 3D map-building and locating the RNA in a floorplan for wayfinding. The experimental results demonstrate the efficacy of the proposed method. In terms of future work, re-localization method, such as [3], will be employed to eliminate the accumulative pose error and reset the user's location in the floorplan. This will allow human subject study in a much larger scale environment.

REFERENCES

[1] R. R. Bourne, S. R. Flaxman, T. Braithwaite, M. V. Cicinelli, A. Das, J. B. Jonas, J. Keeffe, J. H. Kempen, J. Leasher, H. Limburg *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," *The Lancet Global Health*, vol. 5, no. 9, pp. e888–e897, 2017.

[2] J. A. Hesch and S. I. Roumeliotis, "Design and analysis of a portable indoor localization aid for the visually impaired," *The International Journal of Robotics Research*, vol. 29, no. 11, pp. 1400–1415, 2010.

[3] B. Li, J. P. Muñoz, X. Rong, Q. Chen, J. Xiao, Y. Tian, A. Arditi, and M. Yousuf, "Vision-based mobile indoor assistive navigation aid for blind people," *IEEE transactions on mobile computing*, vol. 18, no. 3, pp. 702–714, 2018.

[4] S. Treuillet, E. Royer, T. Chateau, M. Dhome, J.-M. Lavest *et al.*, "Body mounted vision system for visually impaired outdoor and indoor wayfinding assistance," in *CVHI*, 2007.

[5] J. M. Saez, F. Escolano, and A. Penalver, "First steps towards stereo-based 6dof slam for the visually impaired," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)-Workshops*, 2005, pp. 23–23.

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

[7] B. Li, X. Zhang, J. P. Muñoz, J. Xiao, X. Rong, and Y. Tian, "Assisting blind people to avoid obstacles: An wearable obstacle stereo feedback system based on 3d detection," in *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2015, pp. 2307–2311.

[8] Y. H. Lee and G. Medioni, "Rgb-d camera based navigation for the visually impaired," in *Proceedings of the RSS*, 2011.

[9] H. Takizawa, S. Yamaguchi, M. Aoyagi, N. Ezaki, and S. Mizuno, "Kinect cane: An assistive system for the visually impaired based on three-dimensional object recognition," in *IEEE/SICE international symposium on system integration (SII)*, 2012, pp. 740–745.

[10] H. Zhang and C. Ye, "An indoor navigation aid for the visually impaired," in *IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 2016, pp. 467–472.

[11] 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*, vol. 2, no. 2, pp. 33–42, 2016.

[12] R. Jafri, R. L. Campos, S. A. Ali, and H. R. Arabnia, "Visual and infrared sensor data-based obstacle detection for the visually impaired using the google project tango tablet development kit and the unity engine," *IEEE Access*, vol. 6, pp. 443–454, 2017.

[13] C. Ye, "Navigating a mobile robot by a traversability field histogram," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 37, no. 2, pp. 361–372, 2007.

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

[15] B. D. Lucas, T. Kanade *et al.*, "An iterative image registration technique with an application to stereo vision," 1981.

[16] 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.

[17] J.-P. Tardif, Y. Pavlidis, and K. Daniilidis, "Monocular visual odometry in urban environments using an omnidirectional camera," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2008, pp. 2531–2538.

[18] J. Zhang, M. Kaess, and S. Singh, "Real-time depth enhanced monocular odometry," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2014, pp. 4973–4980.

[19] H. Zhang and C. Ye, "Plane-aided visual-inertial odometry for pose estimation of a 3d camera based in-door blind navigation," in *28th British Machine Vision Conference*, 2017.

[20] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, 2000.

[21] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, "Robust visual inertial odometry using a direct ekf-based approach," in *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2015, pp. 298–304.

[22] S. Leutenegger, P. Furgale, V. Rabaud, M. Chli, K. Konolige, and R. Siegwart, "Keyframe-based visual-inertial slam using nonlinear optimization," *Proceedings of Robotis Science and Systems (RSS) 2013*, 2013.

[23] T. Qin and S. Shen, "Online temporal calibration for monocular visual-inertial systems," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 3662–3669.