Vision-Aided Localization For Ground Robots

Mingming Zhang, Yiming Chen and Mingyang Li

***Abstract*— In this paper, we focus on the problem of vision-based localization for ground robotic applications. In recent years, camera only or camera-IMU (inertial measurement unit) based localization methods are widely studied, in terms of theoretical properties, algorithm design, and real-world applications. However, we experimentally find that none of existing methods is able to perform high-precision and robust localization for ground robots in large-scale complicated 3D environments. To this end, in this paper, we propose a novel vision-based localization algorithm dedicatedly designed for ground robots, by fusing measurements from a camera, an IMU, and the wheel odometer. The first contribution of this paper is that we propose a novel algorithm for approximating the motion manifold for ground robots by parametric representation and performing pose integrating via IMU and wheel odometer measurements. Secondly, we propose a complete localization algorithm, by using a sliding-window based estimator. The estimator is designed based on iterative optimization to fuse measurements from multiple sensors on the proposed manifold representation. We show that, based on a variety of real-world experiments, the proposed algorithm outperforms a number of the state-of-the-art vision based localization algorithms by a significant margin, especially when deployed in large-scale complicated environments.**

I. Introduction and Related Work

The localization problem for ground robots has been under active research for about 30 years [1] [2] [3] [4] [5]. The majority of real-world ground robotic applications (e.g., self-driving cars) rely on high-quality GPS-INS systems together with 3D laser range-finders for precise localization [3] [6] [7]. However, those systems are at high manufacturing and maintenance costs, requiring thousands or even tens or hundreds of thousands of dollars, which inevitably prevent their wide applications. Alternatively, low-cost localization approaches have gained increased interests in recent years, especially the ones that rely on cameras [4] [8]. Camera's size, low cost, and 3D sensing capability makes itself a popular sensor. Additionally, recent work shows that, when used together with an IMU, the localization performance can be significantly improved [9] [10] [11] [12]. In fact, camera-IMU localization[1] is widely used in real applications, e.g., smart drones, augmented and virtual reality headsets, or mobile phones.

However, all localization methods mentioned above are *not* optimized for ground robots. Although camera-IMU localization generally performs better than camera only localization by resolving the ambiguities in estimating scale, roll, and pitch [9] [13] [11], it has its own limitations when used for localizing ground robots. Firstly, there are a couple of degenerate cases that can result in large errors when performing motion estimation, e.g., system under static motion, zero rotational velocity motion, constant local linear acceleration motion, and so on [9] [14] [15]. The likelihood of encountering those degenerate cases in ground robots is significantly larger than that in hand-held mobile devices. Secondly, unlike drones or smart headset devices which move freely in 3D environment, ground robots can only move on a manifold (e.g., ground surfaces) due to nature of mechanical design. This makes it possible to use additional low-cost sensors and derive extra mathematical constraints for improving the localization performance [14] [16].

In recent years, there are a couple of low-cost vision-based localization methods designed for ground robots [14] [16] [17]. Specifically, Wu et al. [14] proposed to introduce planar-motion constraints to camera-IMU localization system, and also add wheel odometer measurements for stochastic optimization. The proposed method is shown to improve localization performance in *indoor*

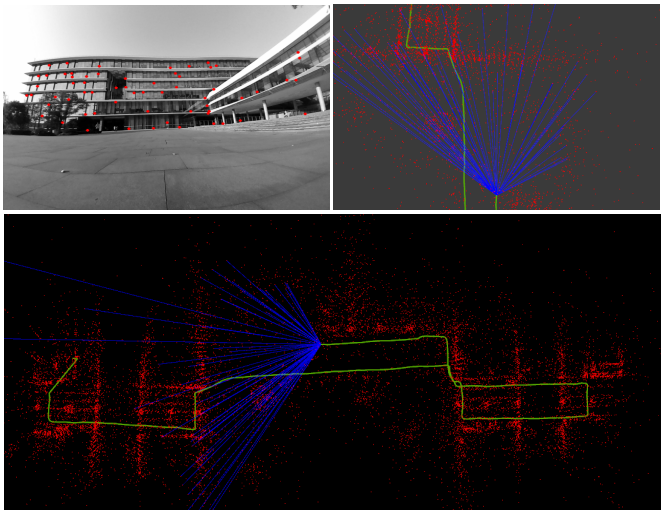

Fig. 1: Representative visualization of the proposed algorithm [1]. Top left: An image used for localization with red dots representing the extracted visual features. Top right and bottom: Different views of localization with pre-built map, green line is the estimated trajectory, red dots are positions of visual landmarks in the pre-built map, and blue lines represent landmark-to-feature correspondences.

The authors are with A.I. Labs, Alibaba Group Inc., Hangzhou, China. { mingmingzhang, yimingchen, mingyangli }@alibaba-inc.com

[1]Camera-IMU localization can also be termed as vision-aided inertial navigation, visual-inertial odometry (VIO), or visual inertial navigation system (VINS) in other papers.

[1]Attached video demonstration online is available at https://youtu.be/_gQ3ky_GTsA

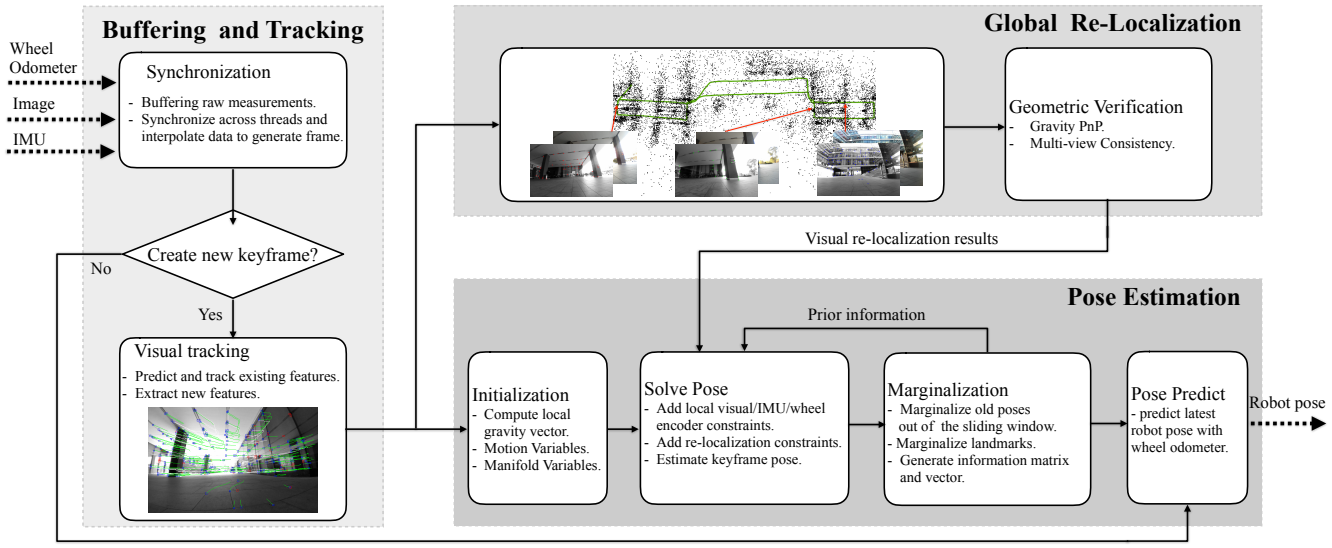

Fig. 2: System overview of the proposed approach.

environments. [17] designed a complete framework for visual-odometer SLAM, in which IMU measurements and wheel odometer measurements were used together for pose integration. Additionally, to better utilize wheel odometer measurements, the corresponding intrinsic parameters can also be calibrated online for performance enhancement [16]. However, [14] [17] [16] only focus on robotic navigation on *a single piece of planar surface*. While this is typically true for most indoor environments, applying those algorithms in complicated outdoor 3D environments is highly risky.

By contrast, in this paper, we propose to approximate motion manifolds for ground robots by a parametric representation, supporting most indoor and outdoor human-made environments. This allows ground robot to conduct general commercial applications without mathematical formulation violation and reduced performance. Additionally, we propose a novel localization framework for fusing measurements from cameras, an IMU, and the wheel odometer, with an iterative sliding-window estimator. The overall localization system using our proposed method is shown in Fig. 2, which is similar to the structure of other state-of-the-art algorithms [10] [11] [18]. By extensive real-world experiments, we show that, the proposed method outperforms other state-of-the-art algorithms, specifically [9] [14] [10], by a significant margin.

II. Notations

In this work, we assume a ground robot navigating with respect to a global reference frame, $\{\mathcal{G}\}$. When a pre-built map is used for persistent localization, its reference frame is denoted as $\{\mathcal{M}\}$. The reference frames for each sensor are denoted by $\{\mathcal{C}\}$, $\{\mathcal{I}\}$, and $\{\mathcal{O}\}$, for camera, IMU, and wheel odometer respectively. The center of frame $\{\mathcal{O}\}$ locates at the center of the robot wheels, with its x-axis pointing forward and z-axis pointing up. Additionally, we use ${}^{\mathbf{A}}\mathbf{p}_{\mathbf{B}}$ and ${}^{\mathbf{A}}_{\mathbf{B}}\bar{\mathbf{q}}$ to represent the position and unit quaternion orientation of frame $\mathcal{B}$ with respect to the frame $\mathcal{A}$. ${}^{\mathbf{A}}_{\mathbf{B}}\mathbf{R}$ is the corresponding rotation matrix of ${}^{\mathbf{A}}_{\mathbf{B}}\bar{\mathbf{q}}$.

III. Localization Algorithm

A. Sensor Measurements and Sensor Data Pre-processing

We describe our localization system by firstly presenting sensor models and the sensor data pre-processing step. Also, in this paper, we assume that, the used sensors are perfectly synchronized by hardware, which is true in our experiments.

1) Wheel Odometer: Similarly to [14], at time t, the measurements for an intrinsically calibrated wheel odometer system are given by:

$$\mathbf{u}_o(t) = \begin{bmatrix} v_o(t) \\ \omega_o(t) \end{bmatrix} = \begin{bmatrix} \mathbf{e}_1^T \cdot {}^{\mathbf{O}(t)}\mathbf{v} + n_v \\ \mathbf{e}_3^T \cdot {}^{\mathbf{O}(t)}\boldsymbol{\omega} + n_w \end{bmatrix} \tag{1}$$

where $\mathbf{e}_i$ is a 3×1 vector, with the ith element to be 1 and other elements to be 0. ${}^{\mathbf{O}(t)}\mathbf{v}$ and ${}^{\mathbf{O}(t)}\boldsymbol{\omega}$ are the local linear velocity and rotational velocity expressed in the frame $\mathcal{O}$ at time t, and n_v and n_w are measurement noises.

2) Inertial Measurement Unit (IMU): On the other hand, the IMU measurements are expressed as:

$$\mathbf{u}_i(t) = \begin{bmatrix} \mathbf{a}_i(t) \\ \boldsymbol{\omega}_i(t) \end{bmatrix} \tag{2}$$

with

$$\mathbf{a}_i(t) = {}^{\mathbf{I}(t)}_{\mathbf{G}}\mathbf{R}\left({}^{\mathbf{G}}\mathbf{a}_I(t) - {}^{\mathbf{G}}\mathbf{g}\right) + \mathbf{b}_{\mathbf{a}} + \mathbf{n}_a \tag{3}$$

$$\boldsymbol{\omega}_i(t) = {}^{\mathbf{I}(t)}\boldsymbol{\omega} + \mathbf{b}_{\mathbf{g}} + \mathbf{n}_g \tag{4}$$

where ${}^{\mathbf{G}}\mathbf{a}_I$ is the IMU's linear acceleration with respect to frame $\mathcal{G}$, ${}^{\mathbf{G}}\mathbf{g}$ is the known gravity vector expressed also in $\mathcal{G}$. $\mathbf{b}_{\mathbf{g}}$ and $\mathbf{b}_{\mathbf{a}}$ are gyroscope and accelerometer biases, and $\mathbf{n}_g$ and $\mathbf{n}_a$ are the corresponding measurement noises, respectively.

3) Data Interpolation: A couple of operations in the proposed system require performing pose (i.e., position and orientation) integration between images (e.g., the operation in Sec. III-A.4). However, due to the nature of multi-sensor system, we might not have IMU and wheel odometer measurements at exactly the same time when images are captured. To this end, we propose to compute extra 'virtual' IMU and wheel odometer measurements by interpolating the closest measurements before and after the image's timestamp. Since IMU and wheel odometer measurements are at relatively high frequencies (e.g., 200Hz and 30Hz in our case) and ground robots in human-made environments are typical under smoothed motion, we choose to apply linear interpolation.

4) Image Processing: Once a new image is received, we proceed to perform pose integration to compute the corresponding predicted pose by wheel odometer measurements (see Sec. III-B). Once enough translational or rotational displacement is detected by pose prediction (e.g., 20 centimeters and 3 degrees in our tests), the new image will be processed, otherwise it will be dropped. For feature processing, FREAK [19] feature is computed due to its efficiency on low-cost processors, which is followed by feature matching and RANSAC outlier rejection steps. Note that, a couple of IMU-camera localization algorithms rely on IMU-based pose integration for detecting displacement to decide keyframes [9]. However, if a device is under slow motion or 'stop-and-go-style' motion for a long time, the long-time IMU integration becomes inaccurate. Due to the availability of wheel odometer, we simply decide keyframes by wheel odometry based pose integration, which is more temporally robust.

B. State Vector and Iterative Optimization

To illustrate the localization algorithm, we first introduce the state vector. At timestamp t_k, the state vector is [2]:

$$\mathbf{x}_k = \begin{bmatrix} \mathbf{O}_k{}^T & \mathbf{s}_k^T & \mathbf{d}_k^T \end{bmatrix}^T, \tag{5}$$

where

$$\mathbf{d}_k = \begin{bmatrix} \mathbf{b}_{\mathbf{g}_k}^T & \mathbf{b}_{\mathbf{a}_k}^T & {}^G\mathbf{v}_I^T & {}^{\mathbf{G}}\mathbf{p}_{\mathbf{O}_i}^T & {}^{\mathbf{G}}_{\mathbf{O}_i}\bar{\mathbf{q}}^T \end{bmatrix}^T \tag{6}$$

and $\mathbf{O}_k$ is the sliding-window pose at timestamp k:

$$\mathbf{O}_k = \begin{bmatrix} \mathbf{x}_{\mathbf{O}_{k-N+1}}^T & \cdots & \mathbf{x}_{\mathbf{O}_{k-1}}^T \end{bmatrix}^T, \mathbf{x}_{\mathbf{O}_i} = \begin{bmatrix} {}^{\mathbf{G}}\mathbf{p}_{\mathbf{O}_i}^T & {}^{\mathbf{G}}_{\mathbf{O}_i}\bar{\mathbf{q}}^T \end{bmatrix}^T \tag{7}$$

for $i = k-N+1, \cdots, k-1$. $\mathbf{s}_k$ represents the motion manifold parameters at the timestamp k, which will be explained in details in Sec. III-C.

When a new image at t_{k+1} is recorded, pose integration is performed to compute $\mathbf{d}_{k+1}$ (see Sec. III-D). Subsequently, we employ the following cost function to refine our state:

$$\mathcal{C}_{k+1}(\mathbf{x}_k, \mathbf{d}_{k+1}, \mathbf{f}_{k+1}) = 2\,\boldsymbol{\eta}_k^T(\mathbf{x}_k - \hat{\mathbf{x}}_k) + ||\mathbf{x}_k - \hat{\mathbf{x}}_k||_{\boldsymbol{\Sigma}_k} + \sum_{i,j \in \mathbf{S}_{i,j}} \boldsymbol{\gamma}_{i,j} + \sum_{i=k-N+1}^{k+1} \boldsymbol{\psi}_i + \beta_i(\mathbf{d}_k, \mathbf{d}_{k+1}) \tag{8}$$

where $\boldsymbol{\eta}_k$ and $\boldsymbol{\Sigma}_k$ are estimated prior information vector and matrix from the previous timestamp respectively, $\hat{\mathbf{x}}_k$ is the estimate of $\mathbf{x}_k$, $||\mathbf{a}||_{\boldsymbol{\Sigma}}$ is computed by $\mathbf{a}^T\boldsymbol{\Sigma}\mathbf{a}$, $\mathbf{f}_{k+1}$ is the set of visual landmarks involved in the optimization process, $\mathbf{S}_{i,j}$ represents the set of pairs between keyframes and observed features, and $\boldsymbol{\gamma}_{i,j}$ is the computed camera reprojection residual vector. Additionally, $\boldsymbol{\psi}_i$ is the residual corresponding to the motion manifold, and $\boldsymbol{\beta}_i$ to the pose prediction cost by IMU and odometer measurements between time t_k and t_{k+1}. Specifically, the camera cost function is:

$$\boldsymbol{\gamma}_{i,j} = \sum_{i,j \in \mathbf{S}_{i,j}} \left\| \mathbf{z}_{ij} - h\big(\mathbf{x}_{\mathbf{O}_i}, \mathbf{f}_j\big) \right\|_{\mathbf{R}_C} \tag{9}$$

where $\mathbf{z}_{ij}$ represents camera measurement corresponding to the pose i and visual landmark $\mathbf{f}_j$, $\mathbf{R}_C$ is the measurement information matrix, and the function $h(\cdot)$ is the model of a calibrated perspective camera [13].

To model $h(\cdot)$, $\mathbf{f}_j$ is required. Recent state-of-the-art localization algorithms can choose to estimate $\mathbf{f}_j$ online [10] or not model it in the state vector [9], or a hybrid method between them [12]. In this work, we choose to not model $\mathbf{f}_j$ in our state vector (see Eq. 5), similar to [9]. To achieve that, during optimization process, we first compute features' positions via Gauss-Newton based multi-view triangulation method, which are used as initial values in minimizing $\mathcal{C}_{k+1}$. We note that, to avoid using the same information multiple times, at each timestamp, we choose *un-processed* features only [14]. With computed positions of features, all prerequisites of solving $\mathcal{C}_{k+1}$ are prepared.

At each timestamp, we use iterative nonlinear optimization to solve $\mathcal{C}_{k+1}$, for obtaining posterior estimates $\hat{\mathbf{x}}_{k+1}^{\star}$ and $\hat{\mathbf{f}}_{k+1}^{\star}$. To keep the computational cost constrained, after the optimization process, we marginalize oldest pose, biases and velocity term in $\mathbf{d}_k$, and features $\mathbf{f}_{k+1}$. We also marginalize the IMU and wheel odometer measurements between poses at t_k and t_{k+1}. This process computes new prior terms $\boldsymbol{\eta}_{k+1}$ and $\boldsymbol{\Sigma}_{k+1}$. We here denote the gradient and Hessian matrix of $\mathcal{C}_k$ with respect to $[\mathbf{x}_k^T, \mathbf{d}_k^T, \mathbf{f}_{k+1}^T]^T$ as $\boldsymbol{\psi}$ and $\boldsymbol{\Omega}$:

$$\boldsymbol{\psi} = \begin{bmatrix} \boldsymbol{\psi}_r \\ \boldsymbol{\psi}_m \end{bmatrix}, \boldsymbol{\Omega} = \begin{bmatrix} \boldsymbol{\Omega}_{rr} & \boldsymbol{\Omega}_{mr}^T \\ \boldsymbol{\Omega}_{mr} & \boldsymbol{\Omega}_{mm} \end{bmatrix} \tag{10}$$

where $\boldsymbol{\psi}_m$ and $\boldsymbol{\Omega}_{mm}$ are corresponding to the terms that are going to be marginalized, e.g., oldest pose and features, $\boldsymbol{\psi}_r$ and $\boldsymbol{\Omega}_{rr}$ to the terms to be kept, and $\boldsymbol{\Omega}_{mr}$ is the cross term. Similarly to other marginalization methods [10], we compute:

$$\boldsymbol{\eta}_{k+1} = \boldsymbol{\psi}_r - \boldsymbol{\Omega}_{mr}^T\boldsymbol{\Omega}_{mm}^{-1}\boldsymbol{\psi}_m \tag{11}$$

$$\boldsymbol{\Sigma}_{k+1} = \boldsymbol{\Omega}_{rr} - \boldsymbol{\Omega}_{mr}^T\boldsymbol{\Omega}_{mm}^{-1}\boldsymbol{\Omega}_{mr} \tag{12}$$

[2]For simpler representation, we ignore sensor extrinsic parameters in our presentation in this section. However, those parameters are explicitly modeled in our formulation and used in experiments.

The remaining question is how to represent motion manifold and perform pose prediction, which will be discussed in details in the next section.

C. Manifold Representation

We focus on ground robot navigating in outdoor environments, especially in large-scale human-made scenes (e.g., company and university campuses, streets, residential areas, etc.). In those scenarios, the terrains are typically smooth and without sharp slope changes. Based on that, we choose to approximate the motion manifold at any 3D location $\mathbf{x}$ by quadratic polynomial:

$$\mathcal{M}(\mathbf{x}) = z + c + \mathbf{B}^T \begin{bmatrix} x \\ y \end{bmatrix} + \frac{1}{2} \begin{bmatrix} x \\ y \end{bmatrix}^T \mathbf{A} \begin{bmatrix} x \\ y \end{bmatrix}, \mathbf{x} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{13}$$

with

$$\mathbf{B} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}, \mathbf{A} = \begin{bmatrix} a_1 & a_2 \\ a_2 & a_3 \end{bmatrix}. \tag{14}$$

The manifold parameters are:

$$\mathbf{s} = \begin{bmatrix} c & b_1 & b_2 & a_1 & a_2 & a_3 \end{bmatrix}^T. \tag{15}$$

For any local $\mathbf{x}'$ that is close to $\mathbf{x}$, the manifold equation can be expressed as:

$$\mathcal{M}(\mathbf{x}') = 0, \text{ if } ||\mathbf{x}' - \mathbf{x}|| < \epsilon \tag{16}$$

where ϵ is a distance constraint. Since we use a limited number of polynomial parameters to represent the motion manifold, ϵ is necessary by limiting the representation in a local region. As mentioned in Sec. III-A.4, in our localization formulation, we choose keyframes of the state vector based on geometrical pose displacement. Therefore, by defining a local motion manifold at ${}^{\mathbf{G}}\mathbf{p}_{\mathbf{O}_k}$, for any keyframe index i, the following equation holds:

$$\mathcal{M}({}^{\mathbf{G}}\mathbf{p}_{\mathbf{O}_k}) = 0, \text{ if } ||{}^{\mathbf{G}}\mathbf{p}_{\mathbf{O}_k} - {}^{\mathbf{G}}\mathbf{p}_{\mathbf{O}_i}|| < \epsilon \tag{17}$$

To use motion manifold as optimization cost functions in Eq. 8, we express Eq. 13 and 17 as stochastic constraints. Specifically, we define the residuals of $\boldsymbol{\psi}_i$ for any valid keyframe (see Eq. 8) as $\boldsymbol{\psi}_i = [f_{p_i}, f_{q_i}]^T$, with

$$f_{p_i} = \frac{1}{\boldsymbol{\sigma}_p^2} \left(\mathcal{M}\left({}^{G}\mathbf{p}_{\mathbf{O}_i}\right) \right)^2 \tag{18}$$

and $\boldsymbol{\sigma}_p^2$ is the corresponding noise variance. Additionally, we apply an orientation cost as:

$$f_q = \left\| \lfloor \left({}^{\mathbf{G}}_{\mathbf{O}_i}\mathbf{R} \cdot \mathbf{e}_3 \right) \times \rfloor_{12} \cdot \left. \frac{\partial\, \mathcal{M}}{\partial\, \mathbf{p}} \right|_{\mathbf{p}={}^{G}\mathbf{p}_{\mathbf{O}_i}} \right\|_{\mathbf{R}_q} \tag{19}$$

where $\lfloor \mathbf{a} \times \rfloor_{12}$ are the first two rows of the cross-product matrix of $\mathbf{a}$, $\mathbf{R}_q$ is the measurement noise information matrix, and

$$\left. \frac{\partial\, \mathcal{M}}{\partial\, \mathbf{p}} \right|_{\mathbf{p}={}^{G}\mathbf{p}_{\mathbf{O}_i}} = \begin{bmatrix} \mathbf{B} + \mathbf{A} \begin{bmatrix} {}^{G}x_{\mathbf{O}_i} \\ {}^{G}y_{\mathbf{O}_i} \end{bmatrix} \\ 1 \end{bmatrix} \tag{20}$$

The physical interpretation of Eq. 19 is that, the motion manifold $\mathcal{M}$ has explicitly defined roll and pitch of a ground robot which should be consistent with ${}^{\mathbf{G}}_{\mathbf{O}_i}\mathbf{R}$.

It is important to point out that, in complicated outdoor environments, the 'true' local motion manifold will be changing over time. However, for most human-made environments, e.g., university campuses, streets, residential areas, and etc., the terrain is typically smooth and changing slowly. To this end, we define the manifold parameter $\mathbf{s}$ as:

$$\mathbf{s}_{k+1} = \mathbf{s}_k + \mathbf{n}_{s_k},\ \mathbf{e}_i^T \mathbf{n}_{s_k} \sim \mathcal{N}(0, \sigma_{s_k}^2) \text{ with } i = 1, \cdots, 6 \tag{21}$$

where $\mathcal{N}(0, \sigma_{i,s_k}^2)$ represents Gaussian distribution with mean 0 and variance $\sigma_{s_k}^2$. Specifically, σ_{s_k} is defined by:

$$\sigma_{s_k} = \alpha_p ||{}^{G}\mathbf{p}_{\mathbf{O}_{k+1}} - {}^{G}\mathbf{p}_{\mathbf{O}_k}|| + \alpha_q ||{}^{G}_{\mathbf{O}_k}\bar{\mathbf{q}}^{-1} \otimes {}^{G}_{\mathbf{O}_{k+1}}\bar{\mathbf{q}}|| \tag{22}$$

where $\otimes$ represents quaternion multiplication, α_p and α_q are control parameters.

It is important to point out that the work similarly to ours on parametric manifold approximation is [20]. However the lack of a sequential statistical estimator makes [20] *not* suitable for high-precision localization. We also note that in Eq. 13 we choose to fix the coefficient of z to be 1, which means that our manifold representation is *not* generic. However, this perfectly fits our applications, since most ground robots can *not* climb vertical walls.

D. Pose Prediction

To present the details of odometry based pose prediction, we assume there are κ wheel odometer measurements between timestamps t_k and t_{k+1}. The timestamps of those measurements are denoted by $t_k(1), \cdots, t_k(\kappa)$, with $t_k(1) = t_k$ and $t_k(\kappa) = t_{k+1}$. Starting from $t_k(1)$, the pose prediction can be computed by integrating:

$${}^{\mathbf{G}}_{\mathbf{O}}\dot{\mathbf{R}}(t) = {}^{\mathbf{G}}_{\mathbf{O}}\mathbf{R}(t) \left\lfloor {}^{\mathbf{O}(t)}\boldsymbol{\omega}(t) \right\rfloor \tag{23}$$

$${}^{\mathbf{G}}\dot{\mathbf{p}}_{\mathbf{O}}(t) = {}^{\mathbf{G}}\mathbf{v}_{\mathbf{O}}(t) = {}^{\mathbf{G}}_{\mathbf{O}}\mathbf{R}(t) {}^{\mathbf{O}(t)}\mathbf{v}(t) \tag{24}$$

where $\lfloor \mathbf{a} \rfloor$ represents the skew-symmetric matrix of vector $\mathbf{a}$. However, the nature of wheel odometer measurement makes it difficult for integrating Eq. 23 and 24, since it only provides estimates of the first element in ${}^{\mathbf{O}(t)}\mathbf{v}(t)$ and the third element in ${}^{\mathbf{O}(t)}\boldsymbol{\omega}(t)$, shown in Eq. 1. If pose integration is performed with odometer measurement only (let other values in ${}^{\mathbf{O}(t)}\mathbf{v}(t)$ and ${}^{\mathbf{O}(t)}\boldsymbol{\omega}(t)$ to be zero), it is equivalent to say the pose integration is calculated on a planar surface, which is the tangent plane of the manifold at the integration starting point. From $t_k(1)$ to $t_k(2)$, if the odometry-only integrated poses are denoted by ${}^{\mathbf{G}}_{\mathbf{O}}\breve{\mathbf{R}}(t_k(2))$ and ${}^{\mathbf{G}}\breve{\mathbf{p}}_{\mathbf{O}}(t_k(2))$, we can write:

$${}^{\mathbf{G}}_{\mathbf{O}}\mathbf{R}(t_k(2)) = {}^{\mathbf{G}}_{\mathbf{O}}\breve{\mathbf{R}}(t_k(2)) \cdot \delta\mathbf{R} \tag{25}$$

$${}^{\mathbf{G}}\mathbf{p}_{\mathbf{O}}(t_k(2)) = {}^{\mathbf{G}}\breve{\mathbf{p}}_{\mathbf{O}}(t_k(2)) + {}^{\mathbf{G}}_{\mathbf{O}}\mathbf{R}(t_k(1)) \cdot \delta\mathbf{p} \tag{26}$$

where $\delta\mathbf{R}$ and $\delta\mathbf{p}$ are correction terms from the tangent plane of manifold to the manifold itself. Since the tangent plane can be considered as the manifold's first-order approximation, $\delta\mathbf{R}$ and $\delta\mathbf{p}$ should be small. Thus, we here

adopt the odometry-only integration described above, and model $\delta\mathbf{R}$ and $\delta\mathbf{p}$ as Gaussian noises. We also note that, this approximation is similar to [21], in which unknown external force is modeled as Gaussian noises for flying drones.

It is also important to point out that our method of approximating odometry based pose integration is *not* the only solution. However, based on our high-FPS odometer measurements and testing environments, we experimentally find that the proposed one already achieves high accuracy and outperforms competing state-of-the-art methods. Exploring and comparing the performance of different pose integration methods will be our focus in future work.

On the other hand, pose prediction using IMU is performed similar to other work on camera-IMU localization [9] [10] [3], and thus we omit the details. In this paper, based on pose integration of wheel odometer and IMU, the cost β in localization optimization function Eq. 8 can be computed, consisting of both wheel odometer term and IMU term. We also point out pose prediction for keyframe selection (see Sec. III-A.4) is performed by wheel odometer based integration. This is due to the fact that errors of wheel odometer integration will not grow as a function of time (especially important for stop-and-go motion), leading to bounded errors of prior estimates for keyframes.

E. Localization with Pre-built Map

The method described in Sec. III-B defines an open-loop localization algorithm, in which long-term drift is inevitable. To cope with this problem, online simultaneous localization and mapping (SLAM) [8] or localization with pre-built map [11] [18] can be performed to generate pose estimates with bounded localization errors. In this paper, we focus on the latter approach. Note that, to enable localization with pre-built map, any type of visual map can be used, if the following conditions can be satisfied. Firstly and most importantly, the generated map must be accurate. Secondly, when building the localization map, the used feature detector and descriptor should match that is used for our localization algorithm.

Our framework of persistent re-localization is similar to that of [11] [18], and we here briefly go over the steps. To be able to use a pre-built map, the position and orientation between the map frame and global frame, i.e., ${}^{\mathbf{G}}\mathbf{p}_{\mathbf{M}}$ and ${}^{\mathbf{G}}\bar{\mathbf{q}}_{\mathbf{M}}$, need to be modeled into state vector (Eq. 5) and be estimated online. In our persistent re-localization framework, all operations of normal open-loop localization remain the same, with one additional operation added: associate features detected from incoming image to visual 3D landmarks in the map and use that for formulating extra cost functions. This is achieved by combination of appearance based loop closure query and geometrical consistent verification [11].

[3]Minor difference exists due to our pose parameterization on frame $\mathcal{O}$ instead of $\mathcal{I}$. As a result, extrinsic parameters between $\mathcal{O}$ and $\mathcal{I}$ will be used during the integration.

TABLE I: Localization errors of different motion manifold representation methods: using 0th-order, 1st-order, 2nd-order (proposed) representation, and the one without modeling the manifold.

	2nd-order	**1st-order**	**0th-order**	**w/o manifold**
Dataset 01				
pos. err. (m)	**0.4993**	3.7528	3.2917	2.1186
rot. err. (deg.)	**2.78423**	3.8094	4.22689	2.95411
Dataset 02				
pos. err. (m)	**0.7330**	2.6927	2.3419	2.3178
rot. err. (deg.)	**2.59439**	4.32612	4.44717	3.51778

TABLE II: Position and rotation (yaw only) errors of the proposed approach, compared to other state-of-the-art methods VINS-Wheel[14], Hybrid-MSCKF[12] and VINS-Mono[10], on different indoor and outdoor datasets.

	Proposed	**[14]**	**[12]**	**[10]**
INDOOR 01				
position err. (m)	1.229	**1.070**	2.849	5.236
rotation err. (deg.)	**1.032**	1.164	1.340	2.437
INDOOR 02				
position err. (m)	**0.783**	0.825	0.950	4.629
rotation err. (deg.)	2.251	**2.022**	2.382	9.281
INDOOR 03				
position err. (m)	1.026	2.760	2.764	**0.986**
rotation err. (deg.)	**2.351**	2.841	2.849	9.360
INDOOR 04				
position err. (m)	**0.626**	0.773	0.794	0.762
rotation err. (deg.)	**1.591**	2.413	2.738	4.314
OUTDOOR 01				
position err. (m)	**1.141**	2.988	2.829	2.44
rotation err. (deg.)	3.225	5.564	6.101	**2.032**
OUTDOOR 02				
position err. (m)	**0.653**	1.479	2.001	6.552
rotation err. (deg.)	**3.153**	8.298	4.906	6.591
OUTDOOR 03				
position err. (m)	**3.119**	3.496	—	—
rotation err. (deg.)	**4.229**	4.513	—	—
OUTDOOR 04				
position err. (m)	**2.004**	13.272	12.361	10.069
rotation err. (deg.)	**1.426**	11.364	10.419	10.378
OUTDOOR 05				
position err. (m)	**0.834**	5.804	4.174	21.583
rotation err. (deg.)	**2.987**	6.034	6.261	9.526
OUTDOOR 06				
position err. (m)	**3.156**	12.179	11.301	23.514
rotation err. (deg.)	**2.14636**	13.739	11.784	15.716

[a] —: Fails due to severe drift.

IV. EXPERIMENTS

To the best of the authors' knowledge, there does *not* exist any proper public dataset with camera, IMU, and wheel odometer measurements for large scale ground robots[4]. Therefore, we collected more than 60 datasets from July 2018 to Feb. 2019, for our experiments. During the data collection, the ground robots were operated in both indoor and outdoor environments, under various types of motion, and under different light and weather conditions. In our experiments, the images were recorded at 10Hz with 640×400-pixels resolution, IMU measurements were at 200Hz,

[4]This statement is at the time of the paper submission of this work.

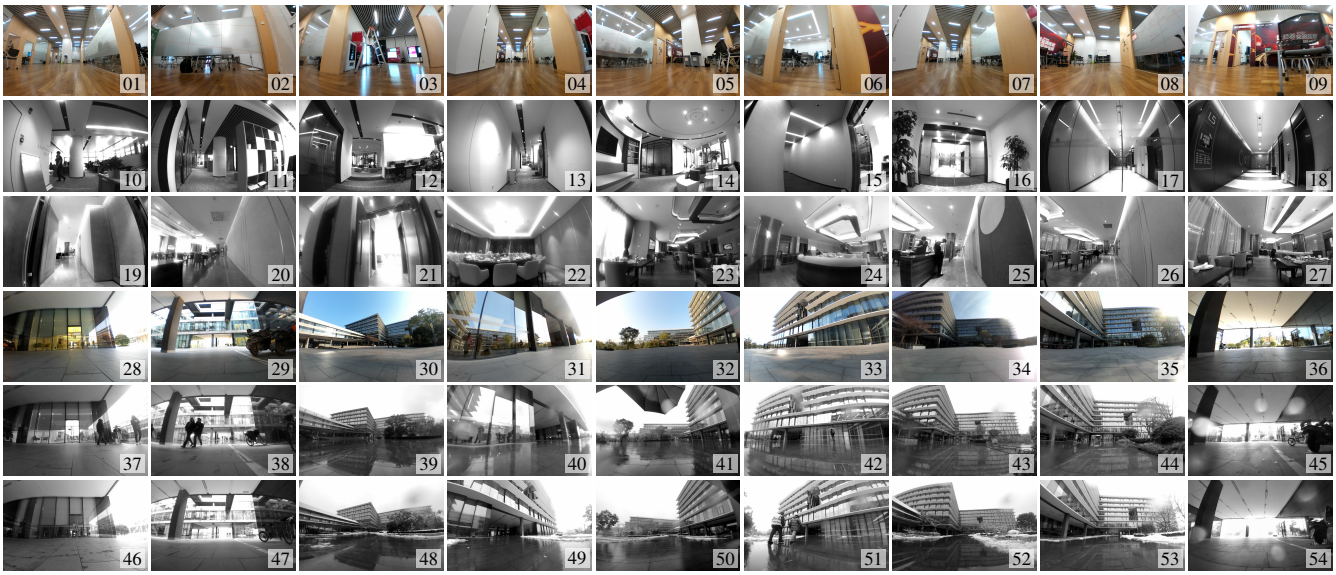

Fig. 3: Sample images for long-term deployment scenarios. **Hall** (1st row), **office** (2nd row), **hotel** (3rd row) are all indoor scenarios, while other three rows are different weather conditions for outdoors from **sunny** (4th row), **rainy** (5th row) to **snowy** (6th row).

and wheel odometer measurements were at 30Hz. Based on the motion during our data collection process, the median frequency of placed keyframes of the proposed localization algorithm was at about 3.5Hz.

A. Evaluation of Manifold Representation

The first experiment is to evaluate the necessity and performance of manifold representation. Specifically, we implemented the proposed localization algorithm in four different modes: 1) removing manifold constraint in Eq. 8, 2) using zero-order approximation for motion manifold (only using c in Eq. 15), 3) using first-order approximation (using c, b_1, and b_2 in Eq. 15), and 4) using the proposed quadratic representation.

Table I shows localization errors for four different modes, evaluated by two different outdoor datasets in which ground robot started and ended at the same location. Due to the lack of full-trajectory ground truth, we computed the localization errors by evaluating the final drifts. A couple of conclusions can be made from the results. First, by using 'poor' parameterization for motion manifold, the localization accuracy can even be reduced. In fact, when approximated by zero-th order or first order polynomials, using motion manifold for localization will be worse than not using it. This is due to the fact that outdoor environment is typically complicated and simple representation is not enough. However, when proper modeling (i.e. the proposed method) is used, better localization precision can be achieved. This experiment validates the most important assumption in our paper: with properly designed motion manifold representation the localization accuracy can be improved. In fact, for the two datasets involved, the improvement of position precision is at least 69%, which is significant.

B. Overall Localization Performance

The next experiment is to evaluate the overall localization performance of the proposed method, compared to a couple of the state-of-the-art algorithms, i.e., VINS-Wheel [14], Hybrid-MSCKF [12] and VINS-Mono [10]. Ten experiments were conducted, with four in indoor 2D environments and six in outdoor 3D environments. Similarly to the previous experiment, the final errors were computed as the metric for different methods, on open-loop estimation setup.

Table II shows the localization errors for all methods in all testing datasets. In indoor cases, the proposed method is able to obtain best overall performance, but the difference between the proposed method and VINS-Wheel is not large. This is due to the fact that, in indoor environments, the planar surface assumption of VINS-Wheel is true and this helps with the localization accuracy. However, in outdoor datasets, the proposed method outperforms all competing methods by a wide margin. In fact, except for the third dataset, the position error of the proposed algorithm is at least 62% better than VINS-Wheel, 60% than Hybrid-MSCKF, and 53% than VINS-Mono. We also note that in the third dataset a lot of rotation-only time periods were involved, which was known to cause degenerate cases for camera-IMU localization. In fact, both Hybrid-MSCKF and VINS-Mono diverges in this case, however, the proposed method performs relatively well. This experiment shows that when used in vision-based localization for ground robots, the proposed algorithm performs significantly better than alternative state-of-the-art methods, in terms of both accuracy and robustness.

C. Re-localization Tests

The proposed re-localization method is also tested with different ground robots platforms in 60 different environmental conditions. The involved testing environments contain

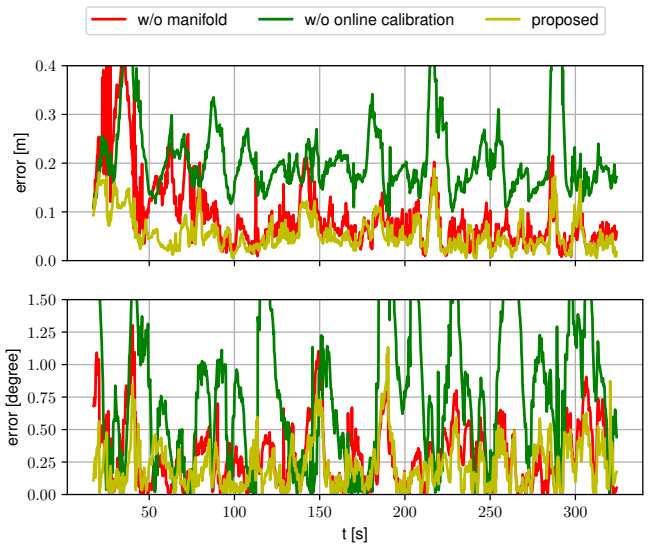

Fig. 4: Translation error and rotation error of the 'full' proposed approach, the one without manifold representation, and the one without online sensor extrinsic calibration.

ones that are highly challenging, e.g., in front of fully-glass-covered building, with low light conditions, in low texture area, and so on (see Fig. 3). Note that, we do *not* aim to seek for difficult cases on purpose, those are cases common for real robotic applications.

In all experiments, the proposed algorithm always worked well, i.e., globally re-localized in the end of the dataset and jumps in pose estimates did not show up during the entire trajectory. We note that this is achieved by both accurate local pose estimation and global loop closure estimation. In fact, in some challenging datasets, it is possible that there were 30 meters' trajectory or 40 seconds' time without good loop closure query (also partially due to the FREAK feature we use [19]). The accurate local pose estimate ensures limited local drift which makes it possible for later global re-localization correction without 'jumps'.

Additionally, based on a representative re-localization dataset, we also evaluated the effectiveness of the proposed manifold-based localization approach, by using or not using it during the test. The result is shown in Fig. 4, which clearly demonstrates that with the proposed manifold representation the re-localization errors can also be largely reduced. Although it is not the focus of this paper, it is mentioned in III-B that the proposed system is implemented with online sensor extrinsic calibration. Fig. 4 also shows that if this is not implemented, the errors will be increased.

V. CONCLUSIONS

In this paper, we propose a vision-aided localization algorithm dedicatedly designed for ground robots. Specifically, a novel method is proposed to approximate the motion manifold and use that to derive localization cost function. Additionally, an optimization-based sliding window estimator is designed to fuse camera, IMU, and wheel odometer measurements together, to perform precise localization tasks. By extensive experimental results, we show that the proposed method outperforms competing state-of-the-art localization algorithms by a significant margin.

REFERENCES

[1] Teddy Yap, Mingyang Li, Anastasios I Mourikis, and Christian R Shelton. A particle filter for monocular vision-aided odometry. In *IEEE International Conference on Robotics and Automation*, 2011.

[2] Ji Zhang and Sanjiv Singh. Low-drift and real-time lidar odometry and mapping. *Autonomous Robots*, 41(2):401–416, 2017.

[3] Jesse Levinson, Michael Montemerlo, and Sebastian Thrun. Map-based precision vehicle localization in urban environments. In *Robotics: Science and Systems*, volume 4, page 1, 2007.

[4] Ryan W Wolcott and Ryan M Eustice. Visual localization within lidar maps for automated urban driving. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 176–183, 2014.

[5] Mingming Zhang, Yiming Chen, and Mingyang Li. SDF-loc: Signed distance field based 2D relocalization and map update in dynamic environments. In *American Control Conference*, 2019.

[6] Guowei Wan, Xiaolong Yang, Renlan Cai, Hao Li, Yao Zhou, Hao Wang, and Shiyu Song. Robust and precise vehicle localization based on multi-sensor fusion in diverse city scenes. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[7] Jesse Levinson, Jake Askeland, Jan Becker, Jennifer Dolson, David Held, Soeren Kammel, J Zico Kolter, Dirk Langer, Oliver Pink, Vaughan Pratt, et al. Towards fully autonomous driving: Systems and algorithms. In *Intelligent Vehicles Symposium (IV)*, 2011.

[8] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. ORB-SLAM: a versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[9] Mingyang Li and Anastasios I Mourikis. High-precision, consistent ekf-based visual-inertial odometry. *The International Journal of Robotics Research*, 32(6):690–711, 2013.

[10] Tong Qin, Peiliang Li, and Shaojie Shen. VINS-Mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics*, 34(4):1004–1020, 2018.

[11] Simon Lynen, Torsten Sattler, Michael Bosse, Joel A Hesch, Marc Pollefeys, and Roland Siegwart. Get out of my lab: Large-scale, real-time visual-inertial localization. In *Robotics: Science and Systems*, 2015.

[12] Mingyang Li and Anastasios I Mourikis. Optimization-based estimator design for vision-aided inertial navigation. In *Robotics: Science and Systems*, pages 241–248, 2013.

[13] Mingyang Li and Anastasios I Mourikis. Online temporal calibration for camera–imu systems: Theory and algorithms. *The International Journal of Robotics Research*, 33(7):947–964, 2014.

[14] Kejian J Wu, Chao X Guo, Georgios Georgiou, and Stergios I Roumeliotis. Vins on wheels. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 5155–5162, 2017.

[15] Dimitrios G Kottas, Kejian J Wu, and Stergios I Roumeliotis. Detecting and dealing with hovering maneuvers in vision-aided inertial navigation systems. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3172–3179. IEEE, 2013.

[16] Andrea Censi, Antonio Franchi, Luca Marchionni, and Giuseppe Oriolo. Simultaneous calibration of odometry and sensor parameters for mobile robots. *IEEE Transactions on Robotics*, 29(2), 2013.

[17] Meixiang Quan, Songhao Piao, Minglang Tan, and Shi-Sheng Huang. Tightly-coupled monocular visual-odometric slam using wheels and a mems gyroscope. *arXiv preprint arXiv:1804.04854*, 2018.

[18] Thomas Schneider, Marcin Dymczyk, Marius Fehr, Kevin Egger, Simon Lynen, Igor Gilitschenski, and Roland Siegwart. maplab: An open framework for research in visual-inertial mapping and localization. *IEEE Robotics and Automation Letters*, 3(3):1418–1425, 2018.

[19] Alexandre Alahi, Raphael Ortiz, and Pierre Vandergheynst. Freak: Fast retina keypoint. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 510–517, 2012.

[20] Bhoram Lee, Kostas Daniilidis, and Daniel D Lee. Online self-supervised monocular visual odometry for ground vehicles. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 5232–5238, Seattle, WA, May 2015.

[21] Barza Nisar, Philipp Foehn, Davide Falanga, and Davide Scaramuzza. VIMO: Simultaneous visual inertial model-based odometry and force estimation. *IEEE Robotics and Automation Letters*, 2019.