Tightly-Coupled Aided Inertial Navigation with Point and Plane Features

Yulin Yang* - yuyang@udel.edu
Patrick Geneva†† - pgeneva@udel.edu
Xingxing Zuo† - xingxingzuo@zju.edu.cn
Kevin Eckenhoff* - keck@udel.edu
Yong Liu† - yongliu@iipc.zju.edu.cn
Guoquan Huang* - ghuang@udel.edu

Authors with * are with Department of Mechanical Engineering, University of Delaware
Author with †† is with Department of Computer and Information Sciences, University of Delaware
Authors with † are with the Institute of Cyber-System and Control, Zhejiang University

RPNG

Robot Perception and Navigation Group (RPNG)
Tech Report - RPNG-2018-EXAMPLE
Last Updated - May 17, 2019

Contents

Abstract

This paper presents a tightly-coupled aided inertial navigation system (INS) with point and plane features, a general sensor fusion framework applicable to any visual and depth sensor (e.g., RGBD, LiDAR) configuration, in which the camera is used for point feature tracking and depth sensor for plane extraction. The proposed system exploits geometrical structures (planes) of the environments and adopts the closest point (CP) for plane parameterization. Moreover, we distinguish planar point features from non-planar point features in order to enforce point-on-plane constraints which are used in our state estimator, thus further exploiting structural information from the environment. We also introduce a simple but effective plane feature initialization algorithm for feature-based simultaneous localization and mapping (SLAM). In addition, we perform online spatial calibration between the IMU and the depth sensor as it is difficult to obtain this critical calibration parameter in high precision. Both Monte-Carlo simulations and real-world experiments are performed to validate the proposed approach.

1 Introduction and Related Work

Most of current popular approaches for real time 6DOF sensor position and orientation (pose) estimation rely on inertial measurement units (IMUs), which can provide high frequency but noisy angular velocity and linear acceleration measurements. In particular, direct integration of these noisy inertial readings from a MEMS IMU will result in large estimation drifts even within a short time interval. For this reason, additional measurement information from different sensor modalities (e.g., optical or event cameras [1, 2, 3, 4, 5, 6, 7], imaging sonars [8, 9] and LiDAR [10]) will be fused to improve the estimation consistency and accuracy.

To date, most of aided inertial navigation systems (INS) have been focusing on utilizing point feature measurements for pose estimation or loop closure. Indeed, point features can be easily detected and reliably tracked in both structured and structure-less environments. Moreover, as shown in [11, 12], point features can provide enough geometrical constraints for the pose estimation of aided INS except for global sensor position and yaw. However, with only point features, it is difficult for the estimator to leverage the structural constraints from environments (e.g., the Manhattan world and indoor rooms) for improved performance. Therefore, features (e.g., lines and planes) that contain such structural information should be exploited, which have attracted quite a few research efforts.

In particular, Hesch et al. [13] used a 2D LiDAR to aid INS for indoor localization by estimating orthogonal structural planes of the buildings within an EKF framework. In our recent work [10], however, we proposed a 3D LiDAR aided inertial plane SLAM system (LIPS) within a graph optimization framework by using continuous-time IMU preintegration [14]. The closest point (CP) from the plane to the origin was used for plane parameterization, which was shown to have better performance compared to quaternion plane representation [15]. Unlike LIPS that used only 3D plane features extracted from sparse LiDAR point clouds, Guo et al. [16] employed both point and plane features in the RGBD aided INS, which assumed known global orientations of the planes and modeled the point observations as a direct relative position measurements. Based on these plane and point measurement models, they performed observability analysis and showed that their system still has four unobservable directions as expected in vision-aided INS (VINS). Hsiao et al. [17] recently developed a dense planar inertial SLAM system (DPI-SLAM) with RGBD cameras, within a loosely-coupled graph optimization using the inertial preintegration, dense visual-odometry and planar measurements (in quaternion form) as the error constraints, which is solved by using the iSAM2 [18]. Note that particular geometric planes such as vertical planes are also assumed in [17].

Unlike the aforementioned work, in this paper, we propose a tightly-coupled estimation framework for aided INS with point and plane features, which fuses measurements from a camera and

a generic depth sensor (e.g., RGBD camera or LiDAR). In particular, the camera can be used for acquiring feature tracks through image sequences, while the plane features, which contain more structural information, can be extracted from point cloud generated from the depth sensor. It is important to note that we divide the detected point features into two types: (i) multi-state constraint Kalman filter (MSCKF) [4] feature, and (ii) SLAM point feature, in analogy to [19, 20]. To limit the state vector size, most of the point features will be treated as MSCKF features and linearly marginalized via null space operation [4, 21], while only a few point features that are residing on planes will be kept in state vector as SLAM features. This follows our design idea that, in order to exploit the structural information available in the environment, whenever possible, we want to enforce the point-on-plane constraints to further improve estimation. Moreover, in comparison to extrinsic calibration between the camera and the IMU (for which many algorithms or tools are available), it is not easy to obtain via offline calibration the accurate rigid-body transformation of the IMU and depth sensor, and thus we particularly perform online extrinsic calibration between them. In summary, the main contributions of this work include:

- We develop a tightly-coupled estimator for aided INS with both point and plane features, applicable to a vision sensor along with a generic depth sensor. The camera can be used for point feature tracking while the depth sensor is utilized for plane extraction. In addition, the rigid-body transformation between the IMU and depth sensor is included in the state vector for online spatial extrinsic calibration.
- To exploit the structural information of the environment, whenever possible, we detect planar point features that are kept in the state vector, and then enforce point-on-plane constraints in estimation. In addition, we introduce a simple but effective plane feature initialization method for state estimation.
- Both Monte-Carlo simulations and real-world experiments with a RGBD camera and IMU are performed to validate the proposed approach, where in real tests a new noise model is introduced to better capture the uncertainty of RGBD points.

2 Aided INS with Point and Plane Features

The state vector of the proposed aided INS contains the IMU state $\mathbf{x}_I$, the depth sensor calibration $\mathbf{x}_{calib}$ and the feature state $\mathbf{x}_{feat}$. For simplicity, we consider one point feature and one plane feature in the state vector as:

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_I^\top & \mathbf{x}_{calib}^\top & \mathbf{x}_{feat}^\top \end{bmatrix} \tag{1}$$

where we have defined:

$$\mathbf{x}_I = \begin{bmatrix} {}^I_G\bar{q}^\top & \mathbf{b}_g^\top & {}^G\mathbf{v}_I^\top & \mathbf{b}_a^\top & {}^G\mathbf{p}_I^\top \end{bmatrix}^\top \tag{2}$$

$$\mathbf{x}_{calib} = \begin{bmatrix} {}^D_I\bar{q}^\top & {}^D\mathbf{p}_I^\top \end{bmatrix}^\top \tag{3}$$

$$\mathbf{x}_{feat} = \begin{bmatrix} {}^G\mathbf{p}_\mathbf{f}^\top & {}^G\mathbf{p}_\pi^\top \end{bmatrix}^\top \tag{4}$$

where ${}^I_G\bar{q}$ denotes the JPL quaternion [22] relating to the rotation ${}^I_G\mathbf{R}$ from global frame $\{G\}$ to IMU frame $\{I\}$, ${}^G\mathbf{v}_I$ and ${}^G\mathbf{p}_I$ denote the IMU velocity and position in global frame, $\mathbf{b}_g$ and $\mathbf{b}_a$ represents the random walk biases for gyroscope and accelerometer, respectively. ${}^D_I\bar{q}$ and ${}^D\mathbf{p}_I$ represents the rigid-body orientation and translation between the depth frame $\{D\}$ and the IMU frame $\{I\}$. In addition, ${}^G\mathbf{p}_\mathbf{f}$ and ${}^G\mathbf{p}_\pi$ denote the point and plane feature. Note that in this work we adopt the closest point (CP) representation for planes as advocated in our prior work [10, 23].

2.1 IMU Motion Model

With IMU measurements, the system motion model can be described as [22]:

$$\begin{aligned}
{}^{I}_{G}\dot{\bar{q}}(t) &= \frac{1}{2}\mathbf{\Omega}\left({}^{I}\boldsymbol{\omega}(t)\right){}^{I}_{G}\bar{q}(t) \\
{}^{G}\dot{\mathbf{p}}_{I}(t) &= {}^{G}\mathbf{v}_{I}(t), \quad {}^{G}\dot{\mathbf{v}}_{I}(t) = {}^{G}\mathbf{a}(t) \\
\dot{\mathbf{b}}_{g}(t) &= \mathbf{n}_{wg}, \quad \dot{\mathbf{b}}_{a}(t) = \mathbf{n}_{wa}(t) \\
\dot{\mathbf{x}}_{calib}(t) &= \mathbf{0}_{6\times 1}, \quad \dot{\mathbf{x}}_{feat}(t) = \mathbf{0}_{6\times 1}
\end{aligned} \tag{5}$$

where $\boldsymbol{\omega}$ and $\mathbf{a}$ represent angular velocity and linear acceleration, respectively. With denoting $\lfloor\cdot\rfloor$ as skew symmetric matrix, we have $\mathbf{\Omega}(\boldsymbol{\omega}) = \begin{bmatrix} -\lfloor\boldsymbol{\omega}\rfloor & \boldsymbol{\omega} \\ -\boldsymbol{\omega}^{\top} & 0 \end{bmatrix}$. The biases $\mathbf{b}_g$ and $\mathbf{b}_a$ are driven by the white Gaussian noises $\mathbf{n}_{wg}$ and $\mathbf{n}_{wa}$, respectively. In addition, given the true state $\mathbf{x}$ and the estimated state $\hat{\mathbf{x}}$, the error state is defined as $\tilde{\mathbf{x}} = \mathbf{x} - \hat{\mathbf{x}}$. Note that the quaternion takes on a different error state $\delta\boldsymbol{\theta}$ as:

$$\delta\bar{q} = \bar{q} \otimes \hat{\bar{q}}^{-1} \simeq \begin{bmatrix} \frac{1}{2}\delta\boldsymbol{\theta}^{\top} & 1 \end{bmatrix}^{\top} \tag{6}$$

where $\otimes$ represents the multiplication for JPL quaternions [22]. Therefore, the linearized system model (5) can be written as:

$$\dot{\tilde{\mathbf{x}}}(t) = \mathbf{F}_{c}(t)\tilde{\mathbf{x}}(t) + \mathbf{G}_{c}(t)\mathbf{n}(t) \tag{7}$$

where $\mathbf{F}_c(t)$ and $\mathbf{G}_c(t)$ represents the continuous-time error state Jacobians and noise Jacobians, while $\mathbf{n}_g$ and $\mathbf{n}_a$ are white Gaussian noises contaminating the IMU angular velocity and linear acceleration readings. $\mathbf{n}(t) = \begin{bmatrix} \mathbf{n}_g^{\top} & \mathbf{n}_{wg}^{\top} & \mathbf{n}_a^{\top} & \mathbf{n}_{wa}^{\top} \end{bmatrix}^{\top}$ represents the system noises modeled as a zero-mean white Gaussian process with autocorrelation $\mathbb{E}\left[\mathbf{n}(t)\mathbf{n}^{\top}(t)\right] = \mathbf{Q}_c\delta(t-\tau)$.

To propagate the covariance $\mathbf{P}_{k|k}$ at time step k, the state transition matrix $\mathbf{\Phi}_{(k+1,k)}$ from time t_k to t_{k+1} can be computed by solving $\dot{\mathbf{\Phi}}_{(k+1,k)} = \mathbf{F}_c(t_k)\mathbf{\Phi}_{(k+1,k)}$ with identity initial condition. Thus, the discrete-time noise covariance and the propagated covariance can be written as:

$$\mathbf{Q}_k = \int_{t_k}^{t_{k+1}} \mathbf{\Phi}_{(k,\tau)}\mathbf{G}_c(\tau)\mathbf{Q}_c\mathbf{G}_c^{\top}(\tau)\mathbf{\Phi}_{(k,\tau)}^{\top}\mathbf{d}\tau \tag{8}$$

$$\mathbf{P}_{k+1|k} = \mathbf{\Phi}_{(k+1,k)}\mathbf{P}_{k|k}\mathbf{\Phi}_{(k+1,k)}^{\top} + \mathbf{Q}_k \tag{9}$$

2.2 Point Feature Measurements

The perspective projection which maps a point feature ${}^{G}\mathbf{p_f}$ onto the camera's image plane is given by:

$$\mathbf{z}^{(p)} = \begin{bmatrix} \frac{{}^{C}x}{{}^{C}z} & \frac{{}^{C}y}{{}^{C}z} \end{bmatrix}^{\top} + \mathbf{n}^{(p)} \tag{10}$$

$${}^{C}\mathbf{p_f} = {}^{C}_{I}\mathbf{R}\,{}^{I}_{G}\mathbf{R}\left({}^{G}\mathbf{p_f} - {}^{G}\mathbf{p}_I\right) + {}^{C}\mathbf{p}_I \tag{11}$$

where ${}^{C}\mathbf{p_f} = [{}^{C}x, {}^{C}y, {}^{C}z]^{\top}$ denotes the point in camera frame $\{C\}$. ${}^{C}_{I}\mathbf{R}$ and ${}^{C}\mathbf{p}_I$ represents the known extrinsic calibration between the camera and IMU. Hence, the Jacobians of the point measurement $\mathbf{z}^{(p)}$ with respect to the state vector (1) can be computed as follows:

$$\mathbf{H}_C = \frac{\partial\tilde{\mathbf{z}}^{(p)}}{\partial{}^{C}\tilde{\mathbf{p}}_{\mathbf{f}}}\begin{bmatrix} \frac{\partial{}^{C}\tilde{\mathbf{p}}_{\mathbf{f}}}{\partial\tilde{\mathbf{x}}_I} & \frac{\partial{}^{C}\tilde{\mathbf{p}}_{\mathbf{f}}}{\partial\tilde{\mathbf{x}}_{calib}} & \frac{\partial{}^{C}\tilde{\mathbf{p}}_{\mathbf{f}}}{\partial\tilde{\mathbf{x}}_{feat}} \end{bmatrix} \tag{12}$$

where we have:

$$\frac{\partial \tilde{\mathbf{z}}^{(p)}}{\partial {}^{C}\tilde{\mathbf{p}}_{\mathbf{f}}} = \frac{1}{{}^{C}\hat{z}^{2}} \begin{bmatrix} {}^{C}\hat{z} & 0 & {}^{C}\hat{x} \\ 0 & {}^{C}\hat{z} & {}^{C}\hat{y} \end{bmatrix} \tag{13}$$

$$\frac{\partial {}^{C}\tilde{\mathbf{p}}_{\mathbf{f}}}{\partial \tilde{\mathbf{x}}_{I}} = \begin{bmatrix} {}^{C}_{I}\hat{\mathbf{R}} \lfloor {}^{I}_{G}\hat{\mathbf{R}}({}^{G}\hat{\mathbf{p}}_{\mathbf{f}} - {}^{G}\hat{\mathbf{p}}_{I}) \rfloor & \mathbf{0}_{3\times 9} & -{}^{C}_{I}\hat{\mathbf{R}}\,{}^{I}_{G}\hat{\mathbf{R}} \end{bmatrix} \tag{14}$$

$$\frac{\partial {}^{C}\tilde{\mathbf{p}}_{\mathbf{f}}}{\partial \tilde{\mathbf{x}}_{calib}} = \mathbf{0}_{3\times 6}, \quad \frac{\partial {}^{C}\tilde{\mathbf{p}}_{\mathbf{f}}}{\partial \tilde{\mathbf{x}}_{feat}} = \begin{bmatrix} {}^{C}_{I}\hat{\mathbf{R}}\,{}^{I}_{G}\hat{\mathbf{R}} & \mathbf{0}_{3} \end{bmatrix} \tag{15}$$

2.3 Plane Feature Measurements

For a plane $\mathbf{p}_{\pi}$, given the normal direction $\mathbf{n}_{\pi}$ and the distance d_{π} from the plane to origin, $\mathbf{p}_{\pi}$ can be represented by the closest point [10, 23] from the plane to origin as:

$$\mathbf{p}_{\pi} = d_{\pi}\mathbf{n}_{\pi} \tag{16}$$

Since plane features can be directly extracted from the point clouds acquired by the depth sensor (e.g., RGBD and LiDAR), we assume a direct plane measurement:

$$\mathbf{z}^{(\pi)} = {}^{D}\mathbf{p}_{\pi} + \mathbf{n}^{(\pi)} = {}^{D}d_{\pi}\,{}^{D}\mathbf{n}_{\pi} + \mathbf{n}^{(\pi)} \tag{17}$$

$$\begin{bmatrix} {}^{D}\mathbf{n}_{\pi} \\ {}^{D}d_{\pi} \end{bmatrix} = \begin{bmatrix} {}^{D}_{I}\mathbf{R} & \mathbf{0}_{3\times 1} \\ {}^{D}\mathbf{p}_{I}^{\top}\,{}^{D}_{I}\mathbf{R} & 1 \end{bmatrix} \begin{bmatrix} {}^{I}_{G}\mathbf{R} & \mathbf{0}_{3\times 1} \\ -{}^{G}\mathbf{p}_{I}^{\top} & 1 \end{bmatrix} \begin{bmatrix} {}^{G}\mathbf{n}_{\pi} \\ {}^{G}d_{\pi} \end{bmatrix} \tag{18}$$

where $\mathbf{n}^{(\pi)}$ is the plane measurement noise with covariance $\mathbf{R}_{\pi}$. Note that it is not trivial to model this measurement noise $\mathbf{n}^{(\pi)}$ inferred from point-cloud measurements which we will explain in detail next.

2.3.1 Plane Extraction from Point Clouds

Given a point cloud, ${}^{D}\mathbf{p}_{\mathbf{f}i}, i = 1 \dots m$ corresponding to a plane, we define each point measurements as:

$${}^{D}\mathbf{p}_{\mathbf{f}mi} = {}^{D}\mathbf{p}_{\mathbf{f}i} + \mathbf{n}_{\mathbf{f}i}, \;\; \mathbf{n}_{\mathbf{f}i} \sim \mathcal{N}(\mathbf{0}_{3\times 1}, \mathbf{R}_{\mathbf{f}i}) \tag{19}$$

where ${}^{D}\mathbf{p}_{\mathbf{f}i}$ is the true value of point i's position in the depth sensor's frame. Note that $\mathbf{R}_{\mathbf{f}i}$ is the point measurement covariance which is crucial for modeling the uncertainty of the plane feature. We will explain our choice for $\mathbf{R}_{\mathbf{f}i}$ based on the chosen sensor in our experiments [see (50)]. We can define the distance from the point ${}^{D}\mathbf{p}_{\mathbf{f}i}$ to plane ${}^{D}\mathbf{p}_{\pi}$ as:

$$d_i = \frac{{}^{D}\mathbf{p}_{\pi}^{\top}\,{}^{D}\mathbf{p}_{\mathbf{f}mi}}{\left\| {}^{D}\mathbf{p}_{\pi} \right\|} - \left\| {}^{D}\mathbf{p}_{\pi} \right\| \tag{20}$$

where ${}^{D}\mathbf{p}_{\pi} (= {}^{D}d_{\pi}\,{}^{D}\mathbf{n}_{\pi})$ is in CP form. As in our prior work [10], we can formulate a maximum likelihood estimation (MLE) to extract the plane ${}^{D}\mathbf{p}_{\pi}$ as:

$$\arg\min_{{}^{D}\mathbf{p}_{\pi}} \sum_{i=1}^{m} \| d_i \|^{2}_{\mathbf{R}_{di}^{-1}} \tag{21}$$

The linearizion of (20) and the covariance $\mathbf{R}_{di}$ can be written as:

$$\tilde{d}_i \simeq \mathbf{H}_{ri}{}^{D}\tilde{\mathbf{p}}_\pi + \mathbf{H}_{ni}\mathbf{n}_{\mathbf{f}i} \tag{22}$$

$$\mathbf{H}_{di} = \frac{\partial \tilde{d}_i}{\partial {}^{D}\tilde{\mathbf{p}}_\pi} = \frac{1}{{}^{D}\hat{d}_\pi}{}^{D}\hat{\mathbf{p}}_{\mathbf{f}i}^\top \left(\mathbf{I}_3 - {}^{D}\hat{\mathbf{n}}_\pi {}^{D}\hat{\mathbf{n}}_\pi^\top\right) - {}^{D}\hat{\mathbf{n}}_\pi^\top \tag{23}$$

$$\mathbf{H}_{ni} = \frac{\partial \tilde{d}_i}{\partial \mathbf{n}_{\mathbf{f}i}} = {}^{D}\mathbf{n}_\pi^\top, \quad \mathbf{R}_{di} = \mathbf{H}_{ni}\mathbf{R}_{\mathbf{f}i}\mathbf{H}_{ni}^\top \tag{24}$$

With these Jacobians and residuals, we can solve (21) by Levenberg-Marquardt algorithm and obtain the plane estimate ${}^{D}\hat{\mathbf{p}}_\pi$ and its covariance by:

$$\mathbf{R}_\pi = \left(\sum_{i=1}^{m} \mathbf{H}_{di}^\top \left(\mathbf{H}_{ni}\mathbf{R}_{\mathbf{f}i}\mathbf{H}_{ni}^\top\right)^{-1}\mathbf{H}_{di}\right)^{-1} \tag{25}$$

2.3.2 Plane Measurement Jacobians

In order to perform the plane measurement update, we also need to compute the plane measurement Jacobians with respect to the state vector (including extrinsic calibration between the depth sensor and IMU):

$$\mathbf{H}_\pi = \frac{\partial \tilde{\mathbf{z}}^{(\pi)}}{\partial \tilde{\mathbf{x}}} = \begin{bmatrix} \frac{\partial \tilde{\mathbf{z}}^{(\pi)}}{\partial \tilde{\mathbf{x}}_I} & \frac{\partial \tilde{\mathbf{z}}^{(\pi)}}{\partial \tilde{\mathbf{x}}_{calib}} & \frac{\partial \tilde{\mathbf{z}}^{(\pi)}}{\partial \tilde{\mathbf{x}}_{feat}} \end{bmatrix} \tag{26}$$

where based on the chain rule of differentiation, we have:

$$\frac{\partial \tilde{\mathbf{z}}^{(\pi)}}{\partial \tilde{\mathbf{x}}_I} = \frac{\partial \tilde{\mathbf{z}}^{(\pi)}}{\partial \begin{bmatrix} {}^{D}\tilde{\mathbf{n}}_\pi \\ {}^{D}\tilde{d}_\pi \end{bmatrix}} \frac{\partial \begin{bmatrix} {}^{D}\tilde{\mathbf{n}}_\pi \\ {}^{D}\tilde{d}_\pi \end{bmatrix}}{\partial \begin{bmatrix} {}^{I}\tilde{\mathbf{n}}_\pi \\ {}^{I}\tilde{d}_\pi \end{bmatrix}} \frac{\partial \begin{bmatrix} {}^{I}\tilde{\mathbf{n}}_\pi \\ {}^{I}\tilde{d}_\pi \end{bmatrix}}{\partial \tilde{\mathbf{x}}_I} \tag{27}$$

$$\frac{\partial \tilde{\mathbf{z}}^{(\pi)}}{\partial \tilde{\mathbf{x}}_{calib}} = \frac{\partial \tilde{\mathbf{z}}^{(\pi)}}{\partial \begin{bmatrix} {}^{D}\tilde{\mathbf{n}}_\pi \\ {}^{D}\tilde{d}_\pi \end{bmatrix}} \frac{\partial \begin{bmatrix} {}^{D}\tilde{\mathbf{n}}_\pi \\ {}^{D}\tilde{d}_\pi \end{bmatrix}}{\partial \tilde{\mathbf{x}}_{calib}} \tag{28}$$

$$\frac{\partial \tilde{\mathbf{z}}^{(\pi)}}{\partial {}^{G}\tilde{\mathbf{x}}_{feat}} = \begin{bmatrix} \frac{\partial \tilde{\mathbf{z}}^{(\pi)}}{\partial {}^{G}\tilde{\mathbf{p}}_{\mathbf{f}}} & \frac{\partial \tilde{\mathbf{z}}^{(\pi)}}{\partial {}^{G}\tilde{\mathbf{p}}_\pi} \end{bmatrix} = \begin{bmatrix} \mathbf{0}_3 & \frac{\partial \tilde{\mathbf{z}}^{(\pi)}}{\partial {}^{G}\tilde{\mathbf{p}}_\pi} \end{bmatrix} \tag{29}$$

$$\frac{\partial \tilde{\mathbf{z}}^{(\pi)}}{\partial {}^{G}\tilde{\mathbf{p}}_\pi} = \frac{\partial \tilde{\mathbf{z}}^{(\pi)}}{\partial \begin{bmatrix} {}^{D}\tilde{\mathbf{n}}_\pi \\ {}^{D}\tilde{d}_\pi \end{bmatrix}} \frac{\partial \begin{bmatrix} {}^{D}\tilde{\mathbf{n}}_\pi \\ {}^{D}\tilde{d}_\pi \end{bmatrix}}{\partial \begin{bmatrix} {}^{G}\tilde{\mathbf{n}}_\pi \\ {}^{G}\tilde{d}_\pi \end{bmatrix}} \frac{\partial \begin{bmatrix} {}^{G}\tilde{\mathbf{n}}_\pi \\ {}^{G}\tilde{d}_\pi \end{bmatrix}}{\partial {}^{G}\tilde{\mathbf{p}}_\pi} \tag{30}$$

The detailed derivations can be found in Appendix A.

2.3.3 Data Association for Plane Features

When a new plane measurement comes in, we employ the Mahalanobis distance test to decide whether it corresponds to a new plane or a currently estimated one. Given the current covariance $\mathbf{P}_{k|k}$, the Mahalanobis distance for the new plane measurement is computed as:

$$r_m = \left(\tilde{\mathbf{z}}^{(\pi)}\right)^\top \left(\mathbf{H}_\pi \mathbf{P}_{k|k}\mathbf{H}_\pi^\top + \mathbf{R}_\pi\right)^{-1}\tilde{\mathbf{z}}^{(\pi)} \tag{31}$$

where r_m subjects to $\boldsymbol{\chi}^2$ distribution. If r_m is smaller than a lower threshold λ_{min}, this plane will be considered as an existing plane in the state vector. If it is larger than a higher threshold λ_{max}, this plane will be treated as a new plane and initialized in the state vector.

After computing measurement Jacboians and residuals, the standard EKF update [24] can be used to update state estimate and covariance.

3 Plane Feature Initialization

Inspired by [11, 25], we propose a simple but effective plane feature initialization algorithm within the EKF framework. Given the current state $\mathbf{x}_k$ and its covariance $\mathbf{P}_{k|k}$, a new plane ${}^G\mathbf{p}_\pi$ is observed and needs to be added into the state. The plane measurement and its linearizion can be re-written as:

$$\mathbf{z}^{(\pi)} = \mathbf{h}\left(\begin{bmatrix}\mathbf{x}_k \\ {}^G\mathbf{p}_\pi\end{bmatrix}\right) + \mathbf{n}^{(\pi)} \tag{32}$$

$$\tilde{\mathbf{z}}^{(\pi)} \simeq \mathbf{H}_x\tilde{\mathbf{x}}_k + \mathbf{H}_\mathbf{f}{}^G\tilde{\mathbf{p}}_\pi + \mathbf{n}^{(\pi)} \tag{33}$$

where $\mathbf{H}_x$ and $\mathbf{H}_\mathbf{f}$ represents the Jacobians w.r.t. the current state $\mathbf{x}_k$ and the new plane feature. Before initialization, the covariance for the new plane feature is treated as ∞ and has no correlation with existing state. Thus, the augmented prior covariance for $\mathbf{x}_k$ and ${}^G\mathbf{p}_\pi$ can be written as:

$$\mathbf{P}_{k+1|k} = \begin{bmatrix}\mathbf{P}_{k|k} & \mathbf{0} \\ \mathbf{0} & \infty\end{bmatrix} \tag{34}$$

The initialization problem can be reformulated as maximum likelihood estimation (MLE):

$$\min_{\mathbf{x}_k, {}^G\mathbf{p}_\pi} \left\|\mathbf{z}^{(\pi)} - \mathbf{h}\left(\begin{bmatrix}\mathbf{x} \\ {}^G\mathbf{p}_\pi\end{bmatrix}\right)\right\|^2_{\mathbf{R}_\pi^{-1}} + \left\|\begin{bmatrix}\tilde{\mathbf{x}} \\ {}^G\tilde{\mathbf{p}}_\pi\end{bmatrix}\right\|^2_{\mathbf{P}^{-1}_{k+1|k}} \tag{35}$$

By taking first order derivative and setting it zero, the optimal state correction can be solved as:

$$\boldsymbol{\Lambda}\begin{bmatrix}\tilde{\mathbf{x}} \\ {}^G\tilde{\mathbf{p}}_\pi\end{bmatrix} = \begin{bmatrix}\mathbf{H}_x^\top \\ \mathbf{H}_\mathbf{f}^\top\end{bmatrix}\mathbf{R}_\pi^{-1}\tilde{\mathbf{z}}^{(\pi)} \tag{36}$$

where $\boldsymbol{\Lambda}$ denotes the information matrix of the new state (including the plane) and is computed by:

$$\boldsymbol{\Lambda} = \begin{bmatrix}\mathbf{H}_x^\top\mathbf{R}_\pi^{-1}\mathbf{H}_x + \mathbf{P}_{k|k}^{-1} & \mathbf{H}_x^\top\mathbf{R}_\pi^{-1}\mathbf{H}_\mathbf{f} \\ \mathbf{H}_\mathbf{f}^\top\mathbf{R}_\pi^{-1}\mathbf{H}_x & \mathbf{H}_\mathbf{f}^\top\mathbf{R}_\pi^{-1}\mathbf{H}_\mathbf{f}\end{bmatrix} \tag{37}$$

Therefore, the covariance $\mathbf{P}_{k+1|k+1}$ of the new state can be written as:

$$\mathbf{P}_{k+1|k+1} = \boldsymbol{\Lambda}^{-1} = \begin{bmatrix}\mathbf{P}_{xx} & \mathbf{P}_{x\pi} \\ \mathbf{P}_{x\pi}^\top & \mathbf{P}_{\pi\pi}\end{bmatrix} \tag{38}$$

where $\mathbf{P}_{xx}$ and $\mathbf{P}_{\pi\pi}$ denote the covariances of current state and plane ${}^G\mathbf{p}_\pi$, respectively. $\mathbf{P}_{x\pi}$ denotes the correlation between the current state and the plane. Since we get the plane measurements from the point cloud, $\mathbf{H}_\mathbf{f}$ is square and invertible. Hence, we can continue to simplify the above equation based on the block matrix inversion:

$$\mathbf{P}_{\pi\pi} = \mathbf{H}_\mathbf{f}^{-1}\left(\mathbf{R} + \mathbf{H}_x\mathbf{P}_{k|k}\mathbf{H}_x^\top\right)\mathbf{H}_\mathbf{f}^{-\top} \tag{39}$$

$$\mathbf{P}_{x\pi} = -\mathbf{P}_{k|k}\mathbf{H}_x^\top\mathbf{H}_\mathbf{f}^{-\top}, \quad \mathbf{P}_{xx} = \mathbf{P}_{k|k} \tag{40}$$

Hence, the covariance matrix can be finally written as:

$$\mathbf{P}_{k+1|k+1} = \begin{bmatrix} \mathbf{P}_{k|k} & -\mathbf{P}_{k|k}\mathbf{H}_x^\top \mathbf{H}_\mathbf{f}^{-1} \\ -\mathbf{H}_\mathbf{f}^{-\top}\mathbf{H}_x\mathbf{P}_{k|k} & \mathbf{H}_\mathbf{f}^{-1}\left(\mathbf{R}_\pi + \mathbf{H}_x\mathbf{P}_{k|k}\mathbf{H}_x^\top\right)\mathbf{H}_\mathbf{f}^{-\top} \end{bmatrix} \tag{41}$$

And the updated state correction can be written as:

$$\begin{bmatrix} \tilde{\mathbf{x}} \\ {}^G\tilde{\mathbf{p}}_\pi \end{bmatrix} = \left(\mathbf{P}_{k+1|k+1}\right)^{-1} \begin{bmatrix} \mathbf{H}_x^\top \\ \mathbf{H}_\mathbf{f}^\top \end{bmatrix} \mathbf{R}_\pi^{-1}\tilde{\mathbf{z}}^{(\pi)} = \begin{bmatrix} \mathbf{0} \\ \mathbf{H}_\mathbf{f}^{-1}\tilde{\mathbf{z}}^{(\pi)} \end{bmatrix} \tag{42}$$

It is important to note that as compared to [11], for the new plane feature initialization, we only need to compute the inverse of the plane feature Jacobian,a 3×3 matrix. If $\mathbf{H_f}$ is invertible, from both (41) and (42), the initialization will not update the existing state. Instead, only the plane feature covariance and correlation with existing state are created during the feature initialization.

4 Point-on-Plane Constraints

For those point features in the state vector, we wish to exploit the structure of the environment by enforcing a point-on-plane constraint whenever possible. Specifically, assuming a point $\mathbf{p_f}$ is on the plane $\mathbf{p}_\pi$, we have the following point-on-plane constraint:

$$\mathbf{g}(\mathbf{x}) := \frac{\mathbf{p_f}^\top \mathbf{p}_\pi}{\|\mathbf{p}_\pi\|} - \|\mathbf{p}_\pi\| = 0 \tag{43}$$

Instead of hard constraints, we treat it as a probabilistic compensation for the feature uncertainty of the planar model. The cost term used by the estimator is given by:

$$\min_{\mathbf{x}} \ \|\mathbf{g}(\mathbf{x})\|^2_{\sigma_g^{-2}} \tag{44}$$

where σ_g is the variance we assign to the point-on-plane constraints, which is set to be $0.01m$ in our experiments.

We identify the correspondence between point feature ${}^G\mathbf{p_f}$ and the new coming plane based on Mahalanobis distance test (44). For clarity, we denote ${}^D\mathbf{p}_{\pi m} \triangleq \mathbf{z}^{(\pi)}$ as the new plane measurement. Then the distance from the $\mathbf{p_f}$ to this new plane $\mathbf{p}_\pi$ can be defined in the depth frame as:

$$d_m = \frac{{}^D\mathbf{p}_{\pi m}^\top {}^D\mathbf{p_f}}{\|{}^D\mathbf{p}_{\pi m}\|} - \left\|{}^D\mathbf{p}_{\pi m}\right\| \tag{45}$$

$${}^D\mathbf{p_f} = {}^D_I\mathbf{R}^I_G\mathbf{R}\left({}^G\mathbf{p_f} - {}^G\mathbf{p}_I\right) + {}^D\mathbf{p}_I \tag{46}$$

With that, we compute the Mahalanobis distance as:

$$r_p = d_m^\top \left(\mathbf{H}_{mx}\mathbf{P}_{k|k}\mathbf{H}_{mx}^\top + \mathbf{H}_{mn}\mathbf{R}_\pi\mathbf{H}_{mn}^\top\right)^{-1} d_m \tag{47}$$

where $\mathbf{H}_{mx}$ and $\mathbf{H}_{mn}$ are Jacobans w.r.t. the state and the plane noise $\mathbf{n}_\pi$, respectively. The detail derivation can be found Appendix B. Also, based on $\boldsymbol{\chi}^2$ distribution, we can set a threshold. If r_p is smaller than the threshold, we will accept this point-on-plane constraint. Then, similar to [26], this constraint (43) can be treated as additional measurement and be used to update the state estimate and covariance.

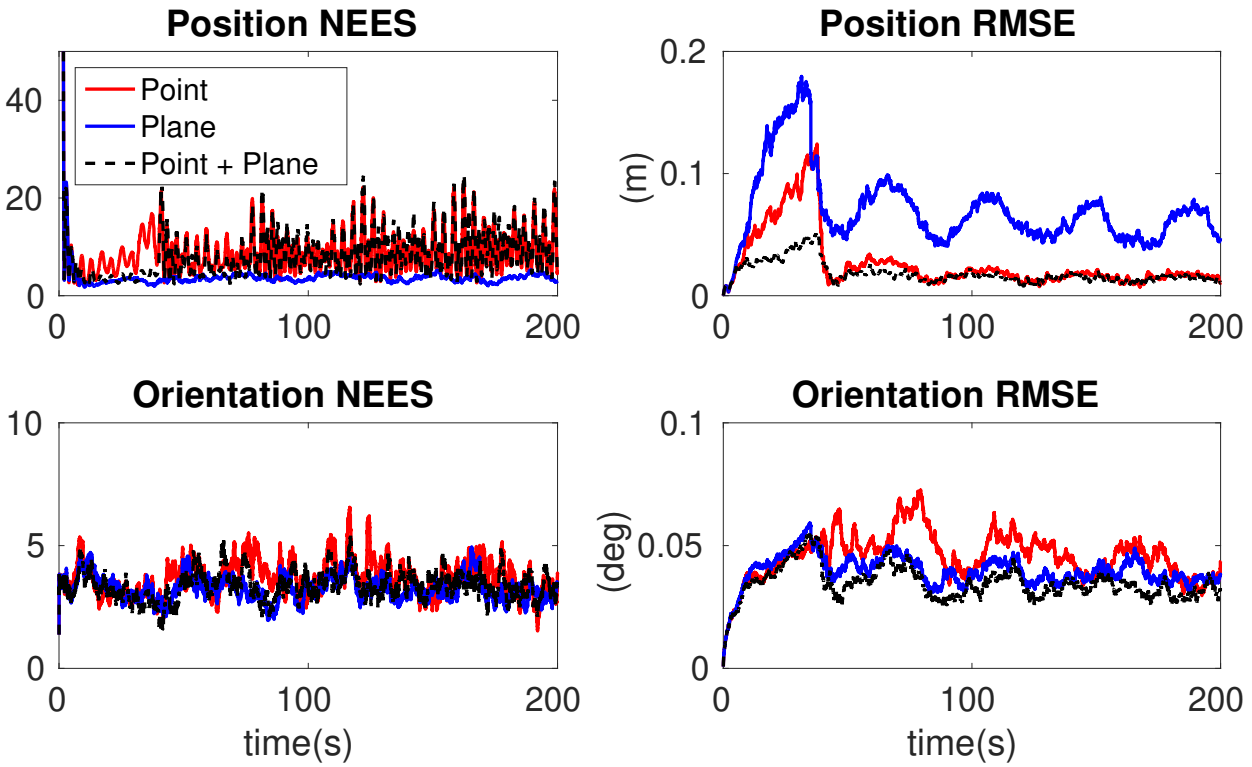

Figure 1: Simulation results: Averaged NEES and RMSE values of the IMU poses (orientation and position) of the proposed aided INS with point and/or plane features.

5 Simulation Results

We first validate the proposed tightly-coupled INS with point and plane features in Monte-Carlo simulations. Similar to our previous work [12], we simulate a 3D sinusoidal trajectory that an IMU-stereo camera sensor rig travels along. The sensor rig collects IMU readings, projective point measurements as well as direct CP plane measurements from a pre-generated map. We run Monte-Carlo simulations with the proposed estimator for the following scenarios: (i) using point features only, (ii) using plane features only, and (iii) using both features. The average normalized estimation error square (NEES) and root mean square error (RMSE) [27] are used to evaluate the accuracy and consistency of IMU pose estimation, which are shown in Fig. 1. These results clearly demonstrate that the system performs better when both point and plane features are used, compared to other two cases when only one type of feature is used. Note that in this simulation, the online calibration and point-on-plane constraints are not applied, while they will be validated in the real-world experiments presented next.

6 Experimental Results

To further validate the proposed system, we perform proof-of-concept experiments using a RGBD camera (Intel Realsense ZR300[1]) which can provide IMU readings, monocular images and dense point clouds. Note that we did not use the depth correspondence between the optical image pixel and the depth image. Instead, we treat the images and point clouds as separate independent

[1] https://software.intel.com/en-us/realsense/zr300

Figure 2: Proof-of-concept indoor experiment setup for the proposed aided INS with point and plane features. ArUco makers [31] are placed in the workspace to serve as planar point features and thus to create point-on-plane constraints.

measurements. FAST [28] features from the monocular image were extracted and then tracked with optical flow [29]. Plane features were extracted through segmentation of the acquired point clouds [30]. Then these point and plane features were fed into our estimator. During the tests, the ZR300 sensor traveled through an indoor environment (see Fig. 2 and 3).

Since the point cloud of ZR300 is generated by compounding the measurements from infrared cameras and color cameras, it is not trivial to model the point cloud noise covariance $\mathbf{R}_{\mathbf{f}i}$ in (19). A point ${}^{D}\mathbf{p}_{\mathbf{f}i} = \begin{bmatrix} {}^{D}x_i & {}^{D}y_i & {}^{D}z_i \end{bmatrix}^\top$ from the point cloud is generated by fusing the bearing information ($u_i = \frac{{}^{D}x_i}{{}^{D}z_i}$, $v_i = \frac{{}^{D}y_i}{{}^{D}z_i}$) from the color camera and depth information ${}^{D}z_i$ from the infrared cameras.

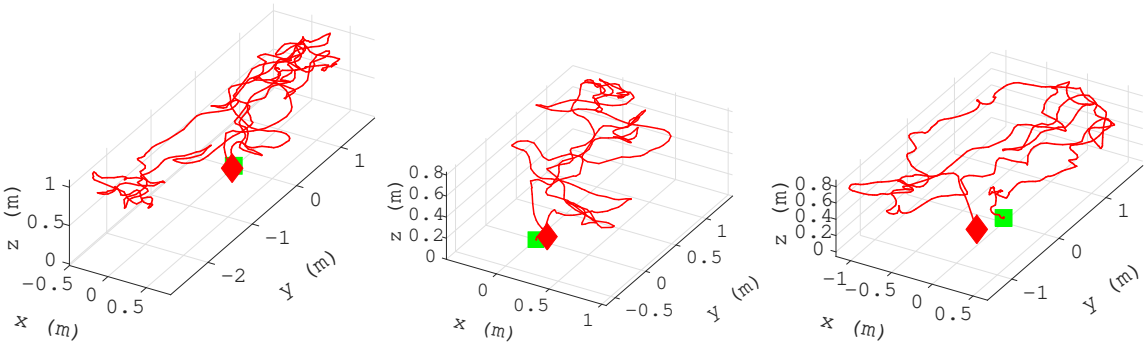

Figure 3: Estimated trajectories of three real-world experiments. From left to right is: Trajectory 1 (37m), Trajectory 2 (20m), and Trajectory 3 (28.5m). The green square and red diamond represent the starting and ending point, respectively. Note that the trajectory length is estimated by accumulating the position changes between every two consecutive frames.

Hence, we can have:

$$^{D}\mathbf{p}_{\mathbf{f}mi} = (^{D}z_i + n_z)\begin{bmatrix} u_i + n_u \\ v_i + n_v \\ 1 \end{bmatrix} \tag{48}$$

$$\simeq \begin{bmatrix} ^{D}\hat{x}_i \\ ^{D}\hat{y}_i \\ ^{D}\hat{z}_i \end{bmatrix} + \underbrace{\begin{bmatrix} ^{D}\hat{z}_i & 0 & \hat{u}_i \\ 0 & ^{D}\hat{z}_i & \hat{v}_i \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{H}_{\mathbf{f}ni}} \begin{bmatrix} n_u \\ n_v \\ n_z \end{bmatrix} \tag{49}$$

where $n_u, n_v \sim \mathcal{N}(0, \sigma^2_{pixel})$ represent normalized image pixel noise with variance σ^2_{pixel}, $n_z \sim \mathcal{N}(0, \sigma^2_{zi})$ represents the depth measurement noise with $\sigma_{zi} = \alpha^{D}z_i$. Note that in our experiments, we take $\alpha = 0.04$ since the point cloud from ZR300 is too noisy. Hence, the point measurement covariance for RGBD sensor can be modeled as:

$$\mathbf{R}_{\mathbf{f}i} = \mathbf{H}_{\mathbf{f}ni}\begin{bmatrix} \sigma^2_{pixel} & 0 & 0 \\ 0 & \sigma^2_{pixel} & 0 \\ 0 & 0 & \sigma^2_{zi} \end{bmatrix}\mathbf{H}^{\top}_{\mathbf{f}ni} \tag{50}$$

In our experiments, we implemented two types of point features: 1) the MSCKF features which were marginalized and, thus only pose information from these feature measurements was fused into the estimator; 2) SLAM point features which were extracted from the fiducial tags (see Fig. 2) and kept in the state vector.

We denote the starting and ending points of the trajectory as $^{G}\mathbf{p}_s$ and $^{G}\mathbf{p}_e$, respectively. Then, the distance d_{se} between the two points can be computed as:

$$d_{se} = \left\| ^{G}\mathbf{p}_s - ^{G}\mathbf{p}_e \right\|_2 \tag{51}$$

Since we ran 3 different trajectories and for each trajectory, the sensor returned to approximately the same position. Therefore, d_{se} is used to evaluate the accuracy of the proposed algorithm. For each trajectory, we ran the proposed algorithm with 3 different setups[2]: i) with SLAM plane features only; ii) with both SLAM point and plane features; iii) with both SLAM point and plane features and point-on-plane constraints. We ran 10 times for each setup and computed the average d_{se}, which are shown in Table 1. The introduction of SLAM point features greatly improved the performance than the case with the SLAM plane features only. In addition, adding the point-on-plane geometrical constraints further improved the system estimation accuracy.

7 Conclusions and Future Work

In this paper, we have presented a tightly-coupled EKF based aided INS with point and plane features in order to better exploit the available geometrical information in structured environments. In the proposed system, a camera is used for point feature tracking and a depth sensor for plane feature extraction, while online spatial calibration between the IMU and the depth sensor is also performed. In particular, we detect point features which reside on the plane features and enforce point-on-plane constraints in the EKF update in order to further exploit the structure information of the environment. In addition, a plane SLAM feature initialization scheme is proposed and compared to existing work, and we analytically show that given full plane measurement from point

[2] All these setups use MSCKF point features.

Table 1: Experiment results for 3 trajectories. The values in table represent the distances from the ending point to the starting point of estimated trajectories.

Unit (m)	Trajectory 1	Trajectory 2	Trajectory 3
MSCKF+Plane	0.2682	0.2607	0.8432
MSCKF+Pt+Plane	0.0539	0.1113	0.3608
MSCKF+Pt-On-Plane	0.0461	0.1095	0.3363

cloud, the plane initialization will not update the existing states. Both Monte-Carlo simulations and real-world experiments with a RGBD camera were performed to verify our algorithm. In the experiments, we also introduced a point noise model which can better capture the uncertainty of the RGBD points. In the future we will integrate online temporal (time offset) calibration between IMU and the depth sensor. Moreover, we will develop more robust algorithms for plane feature data association without relying on the VIO.

Appendix A: Plane Measurement Jacobians

The plane measurement Jacobians can be computed as:

$$\frac{\partial \tilde{\mathbf{z}}^{(\pi)}}{\partial \tilde{\mathbf{x}}_I} = \frac{\partial \tilde{\mathbf{z}}^{(\pi)}}{\partial \begin{bmatrix} {}^D\tilde{\mathbf{n}}_\pi \\ {}^D\tilde{d}_\pi \end{bmatrix}} \frac{\partial \begin{bmatrix} {}^D\tilde{\mathbf{n}}_\pi \\ {}^D\tilde{d}_\pi \end{bmatrix}}{\partial \tilde{\mathbf{x}}_I} \tag{52}$$

$$= \begin{bmatrix} {}^D\hat{d}\mathbf{I}_3 & {}^D\hat{\mathbf{n}}_\pi \end{bmatrix} \begin{bmatrix} {}^D_I\hat{\mathbf{R}} \lfloor {}^I_G\hat{\mathbf{R}} {}^G\hat{\mathbf{n}}_\pi \rfloor & \mathbf{0}_{3\times 9} & \mathbf{0}_3 \\ -{}^I\hat{\mathbf{p}}_D^\top \lfloor {}^I_G\hat{\mathbf{R}} {}^G\hat{\mathbf{n}}_\pi \rfloor & \mathbf{0}_{1\times 9} & -{}^G\hat{\mathbf{n}}_\pi^\top \end{bmatrix}$$

$$\frac{\partial \tilde{\mathbf{z}}^{(\pi)}}{\partial \tilde{\mathbf{x}}_{calib}} = \frac{\partial \tilde{\mathbf{z}}^{(\pi)}}{\partial \begin{bmatrix} {}^D\tilde{\mathbf{n}}_\pi \\ {}^D\tilde{d}_\pi \end{bmatrix}} \frac{\partial \begin{bmatrix} {}^D\tilde{\mathbf{n}}_\pi \\ {}^D\tilde{d}_\pi \end{bmatrix}}{\partial \tilde{\mathbf{x}}_{calib}} \tag{53}$$

$$= \begin{bmatrix} {}^D\hat{d}\mathbf{I}_3 & {}^D\hat{\mathbf{n}}_\pi \end{bmatrix} \begin{bmatrix} \lfloor {}^D_I\hat{\mathbf{R}} {}^I\hat{\mathbf{n}}_\pi \rfloor & \mathbf{0}_3 \\ -{}^I\hat{\mathbf{n}}_\pi^\top {}^D_I\hat{\mathbf{R}}^\top \lfloor {}^D\hat{\mathbf{p}}_I \rfloor & {}^D\hat{\mathbf{n}}_\pi^\top {}^D_I\hat{\mathbf{R}}^\top \end{bmatrix}$$

$$\frac{\partial \tilde{\mathbf{z}}^{(\pi)}}{\partial {}^G\tilde{\mathbf{p}}_\pi} = \frac{\partial \tilde{\mathbf{z}}^{(\pi)}}{\partial \begin{bmatrix} {}^D\tilde{\mathbf{n}}_\pi \\ {}^D\tilde{d}_\pi \end{bmatrix}} \frac{\partial \begin{bmatrix} {}^D\tilde{\mathbf{n}}_\pi \\ {}^D\tilde{d}_\pi \end{bmatrix}}{\partial \begin{bmatrix} {}^G\tilde{\mathbf{n}}_\pi \\ {}^G\tilde{d}_\pi \end{bmatrix}} \frac{\partial \begin{bmatrix} {}^G\tilde{\mathbf{n}}_\pi \\ {}^G\tilde{d}_\pi \end{bmatrix}}{\partial {}^G\tilde{\mathbf{p}}_\pi} \tag{54}$$

$$= \frac{\partial \tilde{\mathbf{z}}^{(\pi)}}{\partial \begin{bmatrix} {}^D\tilde{\mathbf{n}}_\pi \\ {}^D\tilde{d}_\pi \end{bmatrix}} \begin{bmatrix} {}^D_G\hat{\mathbf{R}} & \mathbf{0}_{3\times 1} \\ -{}^G\hat{\mathbf{p}}_D^\top & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{{}^G\hat{d}_\pi} \left(\mathbf{I}_3 - {}^G\hat{\mathbf{n}}_\pi {}^G\hat{\mathbf{n}}^\top{}_\pi \right) \\ {}^G\hat{\mathbf{n}}_\pi^\top \end{bmatrix}$$

Appendix B: Jacobians for Point-on-Plane Constraints

We have two cases for the point-on-plane constraints: (i) ${}^D\mathbf{p}_\pi$ is a new plane and will be initialized, and (ii) ${}^D\mathbf{p}_\pi$ corresponds to a plane feature already in the state.

B.1: Case I

When the plane is first time observed, the Jacobians of the d_m w.r.t. the state vector can be written as:

$$\mathbf{H}_{mx} = \frac{\partial \tilde{d}_m}{\partial \tilde{\mathbf{x}}} = \begin{bmatrix} \frac{\partial \tilde{d}_m}{\partial \tilde{\mathbf{x}}_I} & \frac{\partial \tilde{d}_m}{\partial \tilde{\mathbf{x}}_{calib}} & \frac{\partial \tilde{d}_m}{\partial \tilde{\mathbf{x}}_{feat}} \end{bmatrix} \tag{55}$$

Where we have:

$$\frac{\partial \tilde{d}_m}{\partial \tilde{\mathbf{x}}_I} = \frac{\partial \tilde{d}_m}{\partial {}^D\tilde{\mathbf{p}}_\mathbf{f}} \frac{\partial {}^D\tilde{\mathbf{p}}_\mathbf{f}}{\partial \tilde{\mathbf{x}}_I} \tag{56}$$

$$= {}^D\hat{\mathbf{n}}_\pi^\top {}^D_I\hat{\mathbf{R}} \begin{bmatrix} \lfloor {}^I_G\hat{\mathbf{R}} \left({}^G\hat{\mathbf{p}}_\mathbf{f} - {}^G\hat{\mathbf{p}}_I \right) \rfloor & \mathbf{0}_{3\times 9} & -{}^I_G\hat{\mathbf{R}} \end{bmatrix} \tag{57}$$

$$\frac{\partial \tilde{d}_m}{\partial \tilde{\mathbf{x}}_{calib}} = \frac{\partial \tilde{d}_m}{\partial {}^D\tilde{\mathbf{p}}_\mathbf{f}} \frac{\partial {}^D\tilde{\mathbf{p}}_\mathbf{f}}{\partial \tilde{\mathbf{x}}_{calib}} \tag{58}$$

$$= {}^D\hat{\mathbf{n}}_\pi^\top \begin{bmatrix} \lfloor {}^D_I\hat{\mathbf{R}} {}^I_G\hat{\mathbf{R}} \left({}^G\hat{\mathbf{p}}_\mathbf{f} - {}^G\hat{\mathbf{p}}_I \right) \rfloor & \mathbf{I}_3 \end{bmatrix} \tag{59}$$

$$\frac{\partial \tilde{d}_m}{\partial \tilde{\mathbf{x}}_{calib}} = \frac{\partial \tilde{d}_m}{\partial {}^D\tilde{\mathbf{p}}_\mathbf{f}} \frac{\partial {}^D\tilde{\mathbf{p}}_\mathbf{f}}{\partial \tilde{\mathbf{x}}_{feat}} = \begin{bmatrix} {}^D\hat{\mathbf{n}}_\pi^\top {}^D_I\hat{\mathbf{R}} {}^I_G\hat{\mathbf{R}} & \mathbf{0}_{1\times 3} \end{bmatrix} \tag{60}$$

The Jacobians for the noise can be described as:

$$\mathbf{H}_{mn} = \frac{\partial \tilde{d}_m}{\partial \mathbf{n}^{(\pi)}} = \frac{1}{{}^D\hat{d}_\pi} {}^D\hat{\mathbf{p}}_\mathbf{f}^\top \left(\mathbf{I}_3 - {}^D\hat{\mathbf{n}}_\pi {}^D\hat{\mathbf{n}}_\pi^\top \right) - {}^D\hat{\mathbf{n}}_\pi^\top \tag{61}$$

B.2: Case II

If plane ${}^G\mathbf{p}_\pi$ is already in the state vector, we still have (61). The components of (55) need to be computed as:

$$\frac{\partial \tilde{d}_m}{\partial \delta \boldsymbol{\theta}_I} = \frac{\partial \tilde{d}_m}{\partial {}^D\tilde{\mathbf{p}}_\mathbf{f}} \frac{\partial {}^D\tilde{\mathbf{p}}_\mathbf{f}}{\partial \delta \boldsymbol{\theta}_I} + \frac{\partial \tilde{d}_m}{\partial {}^D\tilde{\mathbf{p}}_\pi} \frac{\partial {}^D\tilde{\mathbf{p}}_\pi}{\partial \delta \boldsymbol{\theta}_I} \tag{62}$$

$$= {}^D\hat{\mathbf{n}}_\pi^\top {}^D_I\hat{\mathbf{R}} \lfloor {}^I_G\hat{\mathbf{R}} \left({}^G\hat{\mathbf{p}}_\mathbf{f} - {}^G\hat{\mathbf{p}}_I \right) \rfloor + \frac{\partial \tilde{d}_m}{\partial {}^D\tilde{\mathbf{p}}_\pi} \begin{bmatrix} {}^D\hat{d}_\pi \mathbf{I}_3 & {}^D\hat{\mathbf{n}}_\pi \end{bmatrix} \begin{bmatrix} {}^D_I\hat{\mathbf{R}} \lfloor {}^I_G\hat{\mathbf{R}} {}^G\hat{\mathbf{n}}_\pi \rfloor \\ -{}^I\hat{\mathbf{p}}_D^\top \lfloor {}^I_G\hat{\mathbf{R}} {}^G\hat{\mathbf{n}}_\pi \rfloor \end{bmatrix} \tag{63}$$

$$\frac{\partial \tilde{d}_m}{\partial {}^G\tilde{\mathbf{p}}_I} = \frac{\partial \tilde{d}_m}{\partial {}^D\tilde{\mathbf{p}}_\mathbf{f}} \frac{\partial {}^D\tilde{\mathbf{p}}_\mathbf{f}}{\partial {}^G\tilde{\mathbf{p}}_I} + \frac{\partial \tilde{d}_m}{\partial {}^D\tilde{\mathbf{p}}_\pi} \frac{\partial {}^D\tilde{\mathbf{p}}_\pi}{\partial {}^G\tilde{\mathbf{p}}_I} \tag{64}$$

$$= -{}^D\hat{\mathbf{n}}_\pi^\top {}^D_I\hat{\mathbf{R}} {}^I_G\hat{\mathbf{R}} + \frac{\partial \tilde{d}_m}{\partial {}^D\tilde{\mathbf{p}}_\pi} \frac{\partial {}^D\tilde{\mathbf{p}}_\pi}{\partial \begin{bmatrix} {}^D\tilde{\mathbf{n}}_\pi \\ {}^D\tilde{d}_\pi \end{bmatrix}} \begin{bmatrix} \mathbf{0}_3 \\ -{}^G\hat{\mathbf{n}}_\pi^\top \end{bmatrix}$$

$$\frac{\partial \tilde{d}_m}{\partial \delta \boldsymbol{\theta}_D} = \frac{\partial \tilde{d}_m}{\partial {}^D\tilde{\mathbf{p}}_\mathbf{f}} \frac{\partial {}^D\tilde{\mathbf{p}}_\mathbf{f}}{\partial \delta \boldsymbol{\theta}_D} + \frac{\partial \tilde{d}_m}{\partial {}^D\tilde{\mathbf{p}}_\pi} \frac{\partial {}^D\tilde{\mathbf{p}}_\pi}{\partial \delta \boldsymbol{\theta}_D} \tag{65}$$

$$= {}^D\hat{\mathbf{n}}_\pi^\top \lfloor {}^D_I\hat{\mathbf{R}} {}^I\hat{\mathbf{p}}_\mathbf{f} \rfloor + \frac{\partial \tilde{d}_m}{\partial {}^D\tilde{\mathbf{p}}_\pi} \frac{\partial {}^D\tilde{\mathbf{p}}_\pi}{\partial \begin{bmatrix} {}^D\tilde{\mathbf{n}}_\pi \\ {}^D\tilde{d}_\pi \end{bmatrix}} \begin{bmatrix} \lfloor {}^D_I\hat{\mathbf{R}} {}^I\hat{\mathbf{n}}_\pi \rfloor \\ -{}^I\hat{\mathbf{n}}_\pi^\top {}^D_I\hat{\mathbf{R}}^\top \lfloor {}^D\hat{\mathbf{p}}_I \rfloor \end{bmatrix}$$

$$\frac{\partial \tilde{d}_m}{\partial {}^D\tilde{\mathbf{p}}_I} = \frac{\partial \tilde{d}_m}{\partial {}^D\tilde{\mathbf{p}}_\mathbf{f}} \frac{\partial {}^D\tilde{\mathbf{p}}_\mathbf{f}}{\partial {}^D\tilde{\mathbf{p}}_I} + \frac{\partial \tilde{d}_m}{\partial {}^D\tilde{\mathbf{p}}_\pi} \frac{\partial {}^D\tilde{\mathbf{p}}_\pi}{\partial {}^D\tilde{\mathbf{p}}_I} \tag{66}$$

$$= {}^D\hat{\mathbf{n}}_\pi^\top \mathbf{I}_3 + \frac{\partial \tilde{d}_m}{\partial {}^D\tilde{\mathbf{p}}_\pi} \frac{\partial {}^D\tilde{\mathbf{p}}_\pi}{\partial \begin{bmatrix} {}^D\tilde{\mathbf{n}}_\pi \\ {}^D\tilde{d}_\pi \end{bmatrix}} \begin{bmatrix} \mathbf{0}_3 \\ {}^I\hat{\mathbf{n}}_\pi^\top {}^D_I\hat{\mathbf{R}}^\top \end{bmatrix}$$

$$\frac{\partial \tilde{d}_m}{\partial {}^G\tilde{\mathbf{p}}_\mathbf{f}} = \frac{\partial \tilde{d}_m}{\partial {}^D\tilde{\mathbf{p}}_\mathbf{f}} \frac{\partial {}^D\tilde{\mathbf{p}}_\mathbf{f}}{\partial {}^G\tilde{\mathbf{p}}_\mathbf{f}} = {}^D\hat{\mathbf{n}}_\pi^\top {}^D_I\hat{\mathbf{R}} {}^I_G\hat{\mathbf{R}} \tag{67}$$

$$\frac{\partial \tilde{d}_m}{\partial {}^G\tilde{\mathbf{p}}_\pi} = \frac{\partial \tilde{d}_m}{\partial {}^D\tilde{\mathbf{p}}_\pi} \frac{\partial {}^D\tilde{\mathbf{p}}_\pi}{\partial {}^G\tilde{\mathbf{p}}_\pi} \tag{68}$$

$$\frac{\partial \tilde{d}_m}{\partial {}^D\tilde{\mathbf{p}}_\pi} = \frac{1}{{}^D\hat{d}_\pi} \left({}^D\hat{\mathbf{p}}_\mathbf{f}^\top - {}^D\hat{\mathbf{n}}_\pi^\top {}^D\hat{\mathbf{p}}_\mathbf{f} {}^D\hat{\mathbf{n}}_\pi^\top - {}^D\hat{d}_\pi {}^D\hat{\mathbf{n}}_\pi^\top \right) \tag{69}$$

$$\frac{\partial {}^D\tilde{\mathbf{p}}_\pi}{\partial {}^G\tilde{\mathbf{p}}_\pi} = \frac{\partial {}^D\tilde{\mathbf{p}}_\pi}{\partial \begin{bmatrix} {}^D\tilde{\mathbf{n}}_\pi \\ {}^D\tilde{d}_\pi \end{bmatrix}} \begin{bmatrix} {}^D_G\hat{\mathbf{R}} & \mathbf{0}_{3\times 1} \\ -{}^G\hat{\mathbf{p}}_D^\top & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{{}^G\hat{d}_\pi} \left(\mathbf{I}_3 - {}^G\hat{\mathbf{n}}_\pi {}^G\hat{\mathbf{n}}_\pi^\top \right) \\ {}^G\hat{\mathbf{n}}_\pi^\top \end{bmatrix} \tag{70}$$

References

[1] Zheng Huai and Guoquan Huang. "Robocentric Visual-Inertial Odometry". In: *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*. Madrid, Spain, 2018.

[2] Kevin Eckenhoff, Patrick Geneva, and Guoquan Huang. "Direct Visual-Inertial Navigation with Analytical Preintegration". In: *Proc. of the IEEE International Conference on Robotics and Automation*. Singapore, 2017, pp. 1429–1435.

[3] Alex Zihao Zhu, Nikolay Atanasov, and Kostas Daniilidis. "Event-based visual inertial odometry". In: *IEEE Conference on Computer Vision and Pattern Recognition*. Honolulu, Hi, USA, 2017, pp. 5391–5399.

[4] A. I. Mourikis and S. I. Roumeliotis. "A multi-state constraint Kalman filter for vision-aided inertial navigation". In: *International Conference on Robotics and Automation*. Rome, Italy, 2007, pp. 3565–3572.

[5] C. Forster et al. "On-Manifold Preintegration for Real-Time Visual–Inertial Odometry". In: *IEEE Transactions on Robotics* 33.1 (2017), pp. 1–21. ISSN: 1552-3098. DOI: 10.1109/TRO.2016.2597321.

[6] Tong Qin, Peiliang Li, and Shaojie Shen. "VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator". In: *arXiv preprint arXiv:1708.03852* (2017).

[7] Teng Zhang et al. "An invariant-EKF VINS algorithm for improving consistency". In: *arXiv preprint arXiv:1702.07920* (2017).

[8] Yulin Yang and Guoquan Huang. "Acoustic-Inertial Underwater Navigation". In: *Proc. of the IEEE International Conference on Robotics and Automation*. Singapore, 2017, pp. 4927–4933.

[9] Sharmin Rahman, Alberto Quattrini Li, and Ioannis Rekleitis. "Sonar Visual Inertial SLAM of Underwater Structures". In: *IEEE International Conference on Robotics and Automation*. Brisbane, Australia, 2018.

[10] Patrick Geneva et al. "LIPS: LiDAR-Inertial 3D Plane SLAM". In: *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*. Madrid, Spain, 2018.

[11] J.A. Hesch et al. "Consistency Analysis and Improvement of Vision-aided Inertial Navigation". In: *IEEE Transactions on Robotics* PP.99 (2013), pp. 1–19. DOI: 10.1109/TRO.2013.2277549.

[12] Yulin Yang and Guoquan Huang. "Aided Inertial Navigation with Geometric Features: Observability Analysis". In: *Proc. of the IEEE International Conference on Robotics and Automation*. Brisbane, Australia, 2018, pp. 2334–2340.

[13] J. A. Hesch et al. "A Laser-Aided Inertial Navigation System (L-INS) for human localization in unknown indoor environments". In: *International Conference on Robotics and Automation*. Anchorage, Alaska, 2010, pp. 5376–5382. DOI: 10.1109/ROBOT.2010.5509693.

[14] Kevin Eckenhoff, Patrick Geneva, and Guoquan Huang. "High-Accuracy Preintegration for Visual-Inertial Navigation". In: *Proc. of the International Workshop on the Algorithmic Foundations of Robotics*. San Francisco, CA, 2016.

[15] M. Kaess. "Simultaneous localization and mapping with infinite planes". In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. Seattle, Washington, 2015, pp. 4605–4611. DOI: 10.1109/ICRA.2015.7139837.

[16] Chao X Guo and Stergios I Roumeliotis. "IMU-RGBD camera navigation using point and plane features". In: *International Conference on Intelligent Robots and Systems*. Tokyo, Japan, 2013, pp. 3164–3171.

[17] Ming Hsiao, Eric Westman, and Michael Kaess. "Dense Planar-Inertial SLAM with Structural Constraints". In: *Proc. of the IEEE International Conference on Robotics and Automation*. Brisbane, Australia, 2018, pp. 6521–6528.

[18] M. Kaess et al. "iSAM2: Incremental Smoothing and Mapping Using the Bayes Tree". In: *International Journal of Robotics Research* 31 (Feb. 2012), pp. 217–236.

[19] Dimitrios G Kottas and Stergios I Roumeliotis. "An iterative Kalman smoother for robust 3D localization on mobile and wearable devices". In: *International Conference on Robotics and Automation*. Seattle, WA, 2015, pp. 6336–6343.

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

[21] Yulin Yang, James Maley, and Guoquan Huang. "Null-Space-based Marginalization: Analysis and Algorithm". In: *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*. Vancouver, Canada, 2017.

[22] Nikolas Trawny and Stergios I. Roumeliotis. *Indirect Kalman Filter for 3D Attitude Estimation*. Tech. rep. University of Minnesota, Dept. of Comp. Sci. & Eng., Mar. 2005.

[23] Yulin Yang and Guoquan Huang. *Observability Analysis for Aided INS with Heterogeneous features of points, lines and planes*. Tech. rep. udel.edu/∼yuyang/downloads/tr_observabilityIII.pdf. RPNG, University of Delaware, 2018.

[24] Peter S. Maybeck. *Stochastic Models, Estimation, and Control*. Vol. 1. London: Academic Press, 1979.

[25] Kejian Wu et al. "A Square Root Inverse Filter for Efficient Vision-aided Inertial Navigation on Mobile Devices". In: *Robotics: Science and Systems*. 2015.

[26] Kejian J Wu et al. "VINS on wheels". In: *International Conference on Robotics and Automation*. IEEE. Singapore, 2017, pp. 5155–5162.

[27] Y. Bar-Shalom and T. E. Fortmann. *Tracking and Data Association*. New York: Academic Press, 1988.

[28] Edward Rosten, Reid Porter, and Tom Drummond. "Faster and Better: A Machine Learning Approach to Corner Detection". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32.1 (2010), pp. 105–119.

[29] Bruce D. Lucas and Takeo Kanade. "An Iterative Image Registration Technique with an Application to Stereo Vision". In: *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2*. Vancouver, BC, Canada, 1981, pp. 674–679.

[30] Pedro F Proença and Yang Gao. "Fast Cylinder and Plane Extraction from Depth Cameras for Visual Odometry". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 6813–6820.

[31] OpenCV Developers Team. *Open Source Computer Vision (OpenCV) Library*. Available: http://opencv.org.