2D LIDAR Aided INS for Vehicle Positioning in Urban Environments

Sheng Zhao and Jay A. Farrell

***Abstract*— This paper presents a novel method to utilize *2D* LIDAR for INS (Inertial Navigation System) aiding to improve *3D* vehicle position estimation accuracy, especially when GNSS signals are shadowed. In the proposed framework, 2D LIDAR aiding is carried out without imposing any assumptions on the vehicle motion (e.g. we allow full six degree-of freedom motion). To achieve this, a closed-form formula is derived to predict the line measurement in the LIDAR's frame. This makes the feature association, residual formation and GUI display possible. With this formula, the Extended Kalman Filter (EKF) can be employed in a straightforward manner to fuse the LIDAR and IMU data to estimate the full state of the vehicle. Preliminary experimental results show the effectiveness of the LIDAR aiding in reducing the state estimation uncertainty along certain directions, when GNSS signals are shadowed.**

I. Introduction

Accurate vehicle positioning is an essential requirement for next generation intelligent transportation systems. GNSS aided INS is a standard technique that has been widely adopted to provide accurate vehicle position [3]–[5]. The drawback of the GNSS based positioning is that when GNSS signals are shadowed (as in urban environments), the positioning accuracy degrades at a rate determined by the IMU quality. This rate can be rapid for MEM's devices. Without accurate positioning, the performance of other position-dependent vehicle applications, like lane-departure warning and route planning are also be affected. Therefore, there is interest in sensors other than GNSS to aid INS. Camera, LIDAR and RADAR are all potential sensors that can improve INS performance. This paper addresses LIDAR aiding of INS to improve vehicle positioning accuracy. In the commercial market, there are 2D planar and 3D LIDAR's. The 3D LIDAR is able to obtain the 3D point cloud of the surrounding environment, from which the 3D features can be extracted. However, the high cost of 3D LIDAR prohibits the mass deployment in personal vehicles. In contrast, the 2D LIDAR is small and lightweight with rapidly declining cost. Hence, 2D LIDAR aiding is the focus of this paper. The concept of LIDAR aiding, as discussed in this paper, is illustrated in Fig. 1.

The use of 2D LIDAR in robot localization has been extensively investigated by various authors [1], [10]. In some existing approaches, the robot pose is tracked either by matching sequential LIDAR measurements or by matching the most recent LIDAR measurement with a 2D map. However, these approaches assume that the robot only moves in a plane. Unfortunately, the planar motion assumption does not apply to the vehicles driving on roads. Due to bumps, road inclination, and the loaded suspension of the vehicle, the vehicle and sensor motion define 3D trajectories. Motion estimation in 3D is challenging, requiring estimation of the position, velocity, attitude and sensor calibration vectors in three dimensions.

S. Zhao and J. A. Farrell are with the Department of Electrical Engineering, University of California, Riverside, CA 92521 USA, {shzhao,farrell}@ee.ucr.edu

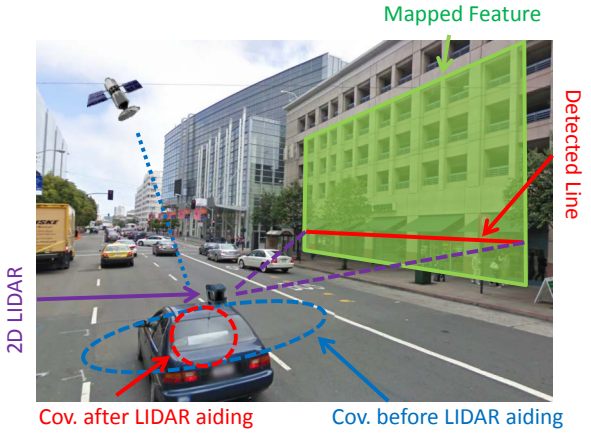

Fig. 1. The typical use of 2D LIDAR aiding for vehicle positioning in urban environments. The uncertainty in the longitude direction can be corrected by GPS, while the LIDAR measurements can provide corrections in the lateral direction where the GPS signals are blocked. Together, LIDAR and GPS can reduce estimated position uncertainty to lane-level accuracy. Note, in practice, the detected line is usually broken into multiple line segments due to occluding objects (e.g., light poles, trees).

The aiding approach taken in this paper falls into the category of *feature-based navigation* wherein the raw LIDAR data is processed to detect certain 2D shapes (e.g., lines) that correspond to 3D features (e.g., planes). We assume that a map of the features is known a priori. The detected shapes are then associated with the mapped features to form the measurement residuals. The measurement residuals are then fed into an Extended Kalman Filter (EKF) to estimate the state of the vehicle. Since the motion of vehicles is 3D, associating a 2D shape to a 3D feature is difficult, as it requires the prediction of the nature of the curve of intersection between the mapped feature and the LIDAR x-y detection plane.

The features that are commonly used in *feature-based navigation* are created by the intersection of the LIDAR x-y plane with common shapes: points (e.g. corners), arcs of ellipses (e.g. cylinder created by trees or poles) and lines (e.g. planes sides of buildings). Since buildings are common in the urban environment, large, and unmoving, they can be easily and reliably detected from 2D LIDAR data, this paper focuses on planar features for LIDAR aiding.

In summary, this paper develops a complete solution that enables the use of 2D LIDAR in aiding the 3D state estimate of a vehicle driving in the urban environment. The presentation of this article focuses on INS aiding, but the results extent directly to encoder based dead-reckoning. Specifically, for the purposes of feature association, residual formation and GUI display, this paper derives an exact closed-form prediction of the location of the line-of-intersection measurement in the LIDAR frame. Due to the simplicity of this formula, feature association and residual formation can be easily solved. Moreover, being able to display the predicted line-of-intersections in the LIDAR's frame makes it easy to visualize the residual and debug the code.

The paper is organized as follows. Section II reviews the related literature. Section III gives an overview of the navigation system. Section IV presents the INS time propagation of the state and its uncertainty. Section V presents the line extraction, feature association, residual formation and EKF LIDAR aiding methods. Section VI presents experimental results. Section VII concludes the paper.

II. Related Work

Feature-based navigation is a very active research area [7], [11]. Cameras are popular due to their low cost and capability to capture 3D features. For aiding purposes, cameras are effectively treated as angle sensors. The performance of camera image processing is not robust to the lighting conditions and is computationally intense.

Compared to cameras, LIDARs are active sensors that are significantly more robust to lighting conditions. Each measurement returns angle, range, and reflection intensity. Utilization of 2D LIDAR in localization has a long history in the robotics community [8], [10]. There are two dominant ways to use LIDAR data: raw point measurements processing [9] and high-level features processing [1]. Often, such work restricts the motion to 2D. Recently, there is a paper using 2D LIDAR aiding IMU to estimate the 3D state of an aggressively flying UAV for an indoor environment [2]. The approach in [2] uses point measurements directly and relies on a 3D occupancy map of the operating environment. This approach would be difficult to extend to outdoor applications because the size of the 3D occupancy map grows cubically in the outdoor environment.

The most closely related work to this paper is [6] which uses 2D LIDAR IMU aiding to estimate the 3D position of a person walking inside a building. The present paper differs from [6] in the formation of residuals, and residual error model equations. In [6], they use geometric constraints to form the residuals, resulting in residual equations that are nonlinear to the line measurement (ϕ and ρ, see Section V) noise components. Herein, the measurement residuals are formed directly resulting in equations that are linear in the line measurement noise components. In addition, the residual formulation herein is more natural – the error between the measured line and the predicted line. Hence, the measurements, predicted measurements and residuals can be easily visualized for a GUI and for debugging.

III. Navigation System Overview

The system considered in this paper includes three sensors: an Inertial Measurement Unit (IMU), a planar LIDAR, and a Global Navigation Satellite System (GNSS) receiver. The main focus of this paper is on LIDAR aiding of the IMU system. But in a realistic application, all available sensors would be fused to optimally estimate the state vector.

Herein we use $\{G\}$ to denote the global frame which is considered to be a non-accelerating and non-rotating frame. It is also the navigation frame in which the estimated position and velocity will be resolved. We use $\{I\}$ to denote the IMU frame, and $\{L\}$ to denote the LIDAR frame.

IV. INS Description

This section is divided into four parts. Subsection IV-A introduces the IMU measurements. Subsection IV-B presents the model of the actual system. Subsection IV-C presents the navigation system time propagation equations. Subsection IV-D presents the error dynamic model.

A. IMU

The IMU provides measurements of rotational velocity ($\boldsymbol{\omega}_m(t)$) and specific force ($\mathbf{a}_m(t)$), both are expressed in IMU frame:

$$\boldsymbol{\omega}_m(t) = {}^I\boldsymbol{\omega}_{GI}(t) + {}^I\mathbf{b}_g(t) + \mathbf{n}_g(t) \tag{1}$$

$$\mathbf{a}_m(t) = {}^I_G\mathbf{R}({}^G\mathbf{a}_{GI}(t) - {}^G\mathbf{g}) + {}^I\mathbf{b}_a(t) + \mathbf{n}_a(t) \tag{2}$$

where $\boldsymbol{\omega}_{GI}$ and $\mathbf{a}_{GI}$ are the angular rate and the acceleration vectors of the $\{I\}$-frame with respect to the $\{G\}$-frame, ${}^G\mathbf{g}$ is the gravity vector in $\{G\}$-frame, ${}^I\mathbf{b}_g(t)$ and ${}^I\mathbf{b}_a(t)$ are time correlated sensor errors that we will refer to as biases, and $\mathbf{n}_g(t)$ and $\mathbf{n}_a(t)$ represent white Gaussian measurement noise processes with power spectral densities (PSD) $\mathbf{Q}_a$ and $\mathbf{Q}_g$, respectively.

B. Nature

The kinematic model of the IMU motion in the $\{G\}$-frame based on the inputs ${}^I\boldsymbol{\omega}_{GI}$ and ${}^G\mathbf{a}_{GI}$ is

$${}^G\dot{\mathbf{p}}_I(t) = {}^G\mathbf{v}_I(t), \quad {}^G\dot{\mathbf{v}}_I(t) = {}^G\mathbf{a}_{GI}(t) \tag{3}$$

$${}^G_I\dot{\mathbf{R}} = {}^G_I\mathbf{R}\lfloor {}^I\boldsymbol{\omega}_{GI}(t)\times\rfloor \tag{4}$$

where $\lfloor\boldsymbol{\omega}\times\rfloor \triangleq \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}$.

The state vector is $\mathbf{x}_I = \begin{bmatrix} {}^G\mathbf{p}_I^\top & {}^G\mathbf{v}_I^\top & {}^G\boldsymbol{\theta}_I^\top \end{bmatrix}^\top$, where ${}^G\boldsymbol{\theta}_I$ represents any one of the equivalent (attitude) representations (e.g., Euler angles or quaternion) for ${}^G_I\mathbf{R}$.

C. Navigation System

The navigation system maintains the estimated state vector $\hat{\mathbf{x}} = \begin{bmatrix} {}^G\hat{\mathbf{p}}_I^\top & {}^G\hat{\mathbf{v}}_I^\top & {}^G\hat{\boldsymbol{\theta}}_I^\top & {}^I\hat{\mathbf{b}}_a^\top & {}^I\hat{\mathbf{b}}_g{}^\top \end{bmatrix}^\top$ by numerically integrating the following equations:

$${}^G\dot{\hat{\mathbf{p}}}_I(t) = {}^G\hat{\mathbf{v}}_I(t), \quad {}^G\dot{\hat{\mathbf{v}}}_I(t) = {}^G\hat{\mathbf{a}}_{GI}(t) \tag{5}$$

$${}^I\dot{\hat{\mathbf{b}}}_g(t) = \mathbf{0}_{3\times1}, \quad {}^I\dot{\hat{\mathbf{b}}}_a(t) = \mathbf{0}_{3\times1} \tag{6}$$

$${}^G_I\dot{\hat{\mathbf{R}}} = {}^G_I\hat{\mathbf{R}}\lfloor {}^I\hat{\boldsymbol{\omega}}_{GI}(t)\times\rfloor \tag{7}$$

where ${}^G\hat{\mathbf{a}}_{GI}$ and ${}^I\hat{\boldsymbol{\omega}}_{GI}(t)$ are computed from the IMU measurements as

$$ {}^G\hat{\mathbf{a}}_{GI}(t) = {}^G_I\hat{\mathbf{R}}\left(\mathbf{a}_m(t) - {}^I\hat{\mathbf{b}}_a(t)\right) + {}^G\mathbf{g} \tag{8} $$

$$ {}^I\hat{\boldsymbol{\omega}}_{GI}(t) = \boldsymbol{\omega}_m(t) - {}^I\hat{\mathbf{b}}_g(t). \tag{9} $$

The numerical integration is performed at the IMU sample rate to propagate the INS state vector between the time instants when aiding measurements are available.

D. Error Analysis

This section summarizes the model for the INS error state temporal dynamics (see e.g., Section 11.4 in [3]).

Let $\mathbf{x} = \begin{bmatrix}\mathbf{x}_I^\top & {}^I\tilde{\mathbf{b}}_a{}^\top & {}^I\tilde{\mathbf{b}}_g{}^\top\end{bmatrix}^\top$. The IMU biases are modeled as random walk processes

$$ {}^I\dot{\mathbf{b}}_a = \mathbf{n}_{\omega a}, \quad {}^I\dot{\mathbf{b}}_g = \mathbf{n}_{\omega g} \tag{10} $$

where the power spectral densities of $\mathbf{n}_{\omega a}$ and $\mathbf{n}_{\omega g}$ are the positive definite matrices $\mathbf{Q}_{\omega a}$ and $\mathbf{Q}_{\omega g}$, respectively.

The error state vector is defined as

$$ \tilde{\mathbf{x}} = \begin{bmatrix}{}^G\tilde{\mathbf{p}}_I^\top & {}^G\tilde{\mathbf{v}}_I^\top & {}^G\delta\boldsymbol{\theta}^\top & {}^I\tilde{\mathbf{b}}_a{}^\top & {}^I\tilde{\mathbf{b}}_g{}^\top\end{bmatrix}^\top \tag{11} $$

where $\tilde{\mathbf{x}} = \mathbf{x} - \hat{\mathbf{x}}$ for position, velocity and biases vectors. For rotation error, the error angle vector ${}^G\delta\boldsymbol{\theta}$ represents the small angle rotation from the actual global frame $\{G\}$ to the estimated global frame $\{\hat{G}\}$ (see page 359 in [3]).

$$ {}^G_I\hat{\mathbf{R}} = (\mathbf{I} - \lfloor {}^G\delta\boldsymbol{\theta}\times\rfloor){}^G_I\mathbf{R} \tag{12} $$

$$ {}^I_G\mathbf{R} = {}^I_G\hat{\mathbf{R}}(\mathbf{I} - \lfloor {}^G\delta\boldsymbol{\theta}\times\rfloor) \tag{13} $$

$$ {}^G_I\mathbf{R} = (\mathbf{I} + \lfloor {}^G\delta\boldsymbol{\theta}\times\rfloor){}^G_I\hat{\mathbf{R}}. \tag{14} $$

The linearized differential equation for the INS error state is

$$ \dot{\tilde{\mathbf{x}}} = \mathbf{F}_c\tilde{\mathbf{x}} + \mathbf{G}_c\mathbf{n} \tag{15} $$

where $\mathbf{n} = [\mathbf{n}_a \ \mathbf{n}_g \ \mathbf{n}_{\omega a} \ \mathbf{n}_{\omega g}]^\top$ with PSD matrix $\mathbf{Q} = diag([\mathbf{Q}_a, \mathbf{Q}_g, \mathbf{Q}_{\omega a}, \mathbf{Q}_{\omega g}])$. The linearization process and the matrices $\mathbf{F}_c$ and $\mathbf{G}_c$ are described in Section 11.4 in [3]. The error covariance is propagated through time according to

$$ \mathbf{P}_{k+1} = \boldsymbol{\Phi}_k\mathbf{P}_k\boldsymbol{\Phi}_k^\top + \mathbf{Qd}_k \tag{16} $$

where $\boldsymbol{\Phi}_k$ is the discrete-time state transition matrix and $\mathbf{Qd}_k$ is the process noise covariance matrix computed from $\mathbf{F}_c$, $\mathbf{G}_c$, and the PSD matrix $\mathbf{Q}$, see Section 7.2.5.2 in [3].

V. LIDAR Processing

The vehicle is assumed to be operating in a known environment where certain plane features have been previously mapped. The planes may represent, for example, signs or sides of buildings. The map database is known *a priori* and stored onboard the vehicle. For the i-th plane feature, we store two parameters: ${}^G\boldsymbol{\pi}_i \in \Re^3$ and ${}^Gd_i \in \Re^+$. Hence, we have a library of mapped plane features, following the notation in [6]:

$$ {}^G\Pi = \{{}^G\boldsymbol{\pi}_i, {}^Gd_i\}_{i=1,\ldots,N_\pi}. $$

The set

$$ \Pi_i = \{{}^G\mathbf{x} \in \Re^3 | {}^G\boldsymbol{\pi}_i \cdot {}^G\mathbf{x} = {}^Gd_i\} $$

represents the i-th 2D-plane feature in the $\{G\}$-frame where ${}^G\boldsymbol{\pi}_i$ is the unit normal vector to the plane and ${}^Gd_i$ is the shortest distance to the plane from the $\{G\}$-frame origin.

The intersection, if it exists, between the LIDAR measurement plane (i.e., x-y) and any other plane is a line. Such lines must be detected and tested for association with planes in the feature library. When a detected line is associated with a plane in the feature library, then 2D LIDAR based state correction is possible using an appropriately formed residual between the two lines.

The k-th scan of the LIDAR at time t_k returns a set of data $\mathcal{D}_k = \{(\theta_i, R_i)\}_{i=1}^{N_l}$ where for the Hokuyo LIDAR, $N_l = 760$, $\theta_i \in [-\frac{\pi}{2}, \frac{\pi}{2}]$ and R_i is the range measurement. This section discusses the processing of each LIDAR data scan $\mathcal{D}_k$ to extract lines, to associate extracted lines with plane features to form line measurements, to compute the covariance of line measurements, to form the residual measurement between each predicted and extracted line, and to aid IMU in an EKF framework.

Any line in the LIDAR x-y plane that does not pass through the origin is uniquely defined by the shortest vector from the origin of $\{L\}$-frame to the line. The shortest vector is represented as ${}^L\mathbf{x} = \rho\,{}^L\boldsymbol{\ell}$, where ${}^L\boldsymbol{\ell} = [\cos\phi \ \ \sin\phi \ \ 0]^\top$ is a unit vector and ρ is the magnitude of ${}^L\mathbf{x}$. Because the line-of-intersection cannot pass through the LIDAR origin, it can be represented by two parameters: ϕ and ρ. Let $\boldsymbol{\chi} = [\phi \ \rho]^\top$.

In this paper, we assumes all the points in one LIDAR scan are taken simultaneously. This assumption is reasonable at low speeds. At higher speeds, the method can be extended, using the IMU data, to compensate for vehicle motion.

A. Measurement Prediction

For various purposes (e.g., association of detected lines with mapped planar features, measurement residual formation, and graphical display), it is useful to have formulas to compute $\hat{\boldsymbol{\chi}}_i = [\hat{\phi}_i, \ \hat{\rho}_i]^\top$, when Π_i, ${}^G\hat{\mathbf{p}}_I$, ${}^G_I\hat{\mathbf{R}}$ and the LIDAR extrinsic calibration parameters ${}^I_L\mathbf{R}$, ${}^I\mathbf{p}_L$, are given.

The problem can be solved in an optimization framework with two constraints. The first constraint, that the vector ${}^L\mathbf{x}_i$ must be in the LIDAR x-y plane, is

$$ {}^L\mathbf{z}_L \cdot {}^L\mathbf{x}_i = 0 \tag{17} $$

where ${}^L\mathbf{z}_L = [0 \ 0 \ 1]^\top$. The second constraint is that the end of the vector ${}^L\mathbf{x}_i$ must be on the plane Π_i:

$$ ({}^L_G\mathbf{R}{}^G\boldsymbol{\pi}_i) \cdot {}^L\mathbf{x}_i = {}^L\hat{d}_i \tag{18} $$

where ${}^L\hat{d}_i = {}^Gd_i - {}^G\boldsymbol{\pi}_i \cdot ({}^G\hat{\mathbf{p}}_I + {}^G_I\hat{\mathbf{R}}\ {}^I\mathbf{p}_L)$. The problem is to find the shortest vector ${}^L\hat{\mathbf{x}}_i$ satisfying constraints (17) and (18):

$$ {}^L\hat{\mathbf{x}}_i = \arg\min_{{}^L\mathbf{x}_i}(\|{}^L\mathbf{x}_i\|^2) \tag{19} $$

$$ s.t. \text{ eqn. (17) and (18).} \tag{20} $$

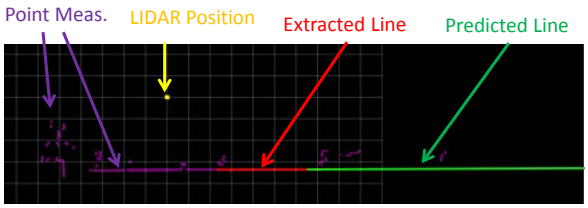

Fig. 2. LIDAR's GUI showing raw measurements (purple dots), the extracted line (red) and the predicted line (green).

The closed form solution is (see the appendix)

$$\hat{\phi}_i = \arctan\left(\text{sgn}({}^L\hat{d}_i)\frac{a_2}{a_1}\right), \quad \hat{\rho}_i = \frac{|{}^L\hat{d}_i|}{\sqrt{a_1^2+a_2^2}} \tag{21}$$

where ${}^L\mathbf{a}_i = [a_1\ a_2\ a_3]^\top$ and ${}^L\mathbf{a}_i = {}^L_G\mathbf{R}\,{}^G\boldsymbol{\pi}_i$. The computed variable $\hat{\boldsymbol{\chi}}_i = [\hat{\phi}_i\ \hat{\rho}_i]^\top$ allows prediction of the line-of-intersection in the LIDAR frame, which is required both for aiding and for a LIDAR frame GUI. The LIDAR GUI is shown in Fig. 2.

B. Line Extraction & Feature Association

This section discusses the method to obtain line measurements, represented by $\boldsymbol{\chi}$, from $\mathcal{D}_k$ and how to associate mapped features to the line measurements.

The starting point is a set of 2D points ${}^L\mathbf{p}_i = [x_i\ y_i]^\top$, $i = 1\dots n$ extracted from the k-th LIDAR data scan $\mathcal{D}_k$ and represented in $\{L\}$-frame that are assumed to be associated with a line. This set of points could be found, for example, by the Hough transform or the *Split-and-Merge* algorithm [8]. The LIDAR raw data R_i and θ_i for the i-th point are related to the $\{L\}$-frame rectangular coordinates by ${}^L\mathbf{p}_i = R_i[\cos\theta_i\ \sin\theta_i]^\top$. We define the range and angle measurement noise to be $\bar{\mathbf{n}} = [n_R\ n_\theta]^\top$, and assume n_R and n_θ are uncorrelated zero mean Gaussian noises with standard deviation σ_R and σ_θ respectively. The covariance of $\bar{\mathbf{n}}$ is

$$\mathbf{P}_{\bar{\mathbf{n}}} = \begin{bmatrix}\sigma_R^2 & 0\\ 0 & \sigma_\theta^2\end{bmatrix}. \tag{22}$$

1) Line Fitting: If the set of points was exactly on the line $\boldsymbol{\chi}$, then they would each satisfy the equation

$$0 = {}^L\mathbf{N}\cdot{}^L\mathbf{p}_i - \rho \tag{23}$$

which is equivalent to $\bar{\alpha}x + \bar{\beta}y = \rho$. For our set of points $\{{}^L\mathbf{p}_i\}_{i=1}^n$ the distance ρ and the unit vector ${}^L\mathbf{N}^\top = [\bar{\alpha}\ \bar{\beta}] = [\cos\phi\ \sin\phi]$ are unknown. In addition, the point locations, are computed from the noise corrupted raw LIDAR data.

For any hypothesized $\boldsymbol{\chi}$, due to the measurement noise $\bar{\mathbf{n}}$, the distance of each point ${}^L\mathbf{p}_i$ from the line

$$r_{l_i} = {}^L\mathbf{N}\cdot{}^L\mathbf{p}_i - \rho \tag{24}$$

is a random variable. The (linearized) covariance of r_{l_i} is $P_{r_{l_i}} = {}^L\bar{\mathbf{M}}_i\mathbf{P}_{\bar{\mathbf{n}}}\left({}^L\bar{\mathbf{M}}_i\right)^\top$, where

$${}^L\bar{\mathbf{M}}_i = {}^L\mathbf{N}^\top\begin{bmatrix}\cos\theta_i & -R_i\sin\theta_i\\ \sin\theta_i & R_i\cos\theta_i\end{bmatrix}. \tag{25}$$

The linearized covariance is accurate when n_θ is small, which is typically the case for LIDAR applications. Under this assumption, the distribution of $\bar{\mathbf{M}}_i$ is accurately approximated as Gaussian. For the optimal line, the sequence r_{l_i} is zero mean and white.

With the above discussions, the goal of the line fitting algorithm is to find the line parameters $\boldsymbol{\chi}$ that maximizes the likelihood function

$$\mathcal{L}(\{r_{l_i}\}_{i=1}^n \mid \boldsymbol{\chi}) = \mathcal{L}(r_{l_1}\mid\boldsymbol{\chi})\dots\mathcal{L}(r_{l_n}\mid\boldsymbol{\chi}) \tag{26}$$

$$= exp\left(-\frac{1}{2}\sum_{i=1}^n r_{l_i}^\top P_{r_{l_i}}^{-1} r_{l_i}\right) \tag{27}$$

where we have used the fact that r_{l_i} is independent of r_{l_j} for $i \neq j$ and dropped the constant of normalization. Because $\mathcal{L}(\{r_{l_i}\}_{i=1}^n \mid \boldsymbol{\chi})$ is a nonlinear function of $\boldsymbol{\chi}$, there is no closed-form solution to the above problem. However, the minimum of the cost function can be found rapidly by an iterative algorithm.

Algorithm Setup: To fit a line, we reparameterize it using $[\alpha,\beta] = [\bar{\alpha},\bar{\beta}]/\rho$ as

$$\alpha x + \beta y - 1 = 0. \tag{28}$$

This representation is appropriate for LIDAR applications because any detected line cannot pass through the origin of the LIDAR frame (i.e., $\rho \neq 0$).

Define $\boldsymbol{\eta} = [\alpha\ \beta]^\top$, then, from eqn. (24), we can form a linear estimation problem as

$$\mathbf{A}\boldsymbol{\eta} = \mathbf{b} + \frac{1}{\rho}\mathbf{r}_l \tag{29}$$

where $\mathbf{A} = [{}^L\mathbf{p}_1,\dots,{}^L\mathbf{p}_n]^\top$, $\mathbf{b} = [1,\dots,1]^\top$ and $\mathbf{r}_l = [r_{l_1},\ \dots,\ r_{l_n}]^\top$. The symbol $\mathbf{r}_l \in \Re^n$ represents error in units of meters with $cov(\mathbf{r}_l) = \mathbf{P}_{\mathbf{r}_l} = diag(P_{r_{l_1}},\dots,P_{r_{l_n}})$ as discussed in eqn. (25), while $\mathbf{n}_L \triangleq \frac{1}{\rho}\mathbf{r}_l$ is a dimensionless quantity with $cov(\mathbf{n}_L) = \mathbf{P}_{\mathbf{n}_L} = \frac{1}{\rho^2}\mathbf{P}_{\mathbf{r}_l}$.

We also define a function $\boldsymbol{\chi} = h(\boldsymbol{\eta})$ to extract the line parameter $\boldsymbol{\chi}$ from $\boldsymbol{\eta}$. The function $h(\cdot)$ is defined as

$$\phi = \arctan\left(\frac{\beta}{\alpha}\right), \quad \rho = \frac{1}{\|\boldsymbol{\eta}\|}. \tag{30}$$

Initialization: At the initialization, because $\boldsymbol{\chi}$ is not yet available, $\mathbf{P}_{\mathbf{n}_L}$ and $\mathbf{P}_{\mathbf{n}_l}$ cannot be computed; therefore, minimization of eqn. (27) is not possible. Instead, we approximate $\mathbf{P}_{\mathbf{n}_l} = \mathbf{I}$ and minimize $\sum_{i=1}^n r_{l_i}^\top r_{l_i}$ using $\boldsymbol{\eta}^0$ that is the solution of $\left(\mathbf{A}^\top\mathbf{A}\right)\boldsymbol{\eta}^0 = \mathbf{A}^\top\mathbf{b}$. Then we have $\boldsymbol{\chi}^0 = h(\boldsymbol{\eta}^0)$.

Step 1: Let the superscript k denote the k-th iteration. Use eqn. (25) with $\boldsymbol{\chi}^{k-1}$ to compute $\mathbf{P}_{\mathbf{n}_L}$. Then we re-solve (29) as $(\mathbf{A}^\top\mathbf{P}_{\mathbf{n}_L}^{-1}\mathbf{A})\boldsymbol{\eta}^k = \mathbf{A}^\top\mathbf{P}_{\mathbf{n}_L}^{-1}\mathbf{b}$ for $\boldsymbol{\eta}^k$.

Step 2: Compute $\boldsymbol{\chi}^k$ using $\boldsymbol{\chi}^k = h(\boldsymbol{\eta}^k)$.

Step 3: If $\left\|\boldsymbol{\chi}^k - \boldsymbol{\chi}^{k-1}\right\| > \delta_c$, then return to **Step1**; otherwise the algorithm ends, having computed $\boldsymbol{\chi}^k$. The parameter δ_c is chosen to trade off the accuracy and speed.

At the conclusion of the iteration, the covariance of $\boldsymbol{\eta}^k$ is $\mathbf{P}_{\boldsymbol{\eta}} = (\mathbf{A}^\top\mathbf{P}_{\mathbf{n}_L}^{-1}\mathbf{A})^{-1}$. The covariance of $\boldsymbol{\chi}^k$ can be

computed by linearizing $h(\cdot)$ around $\boldsymbol{\eta}^k$ to get $\tilde{\boldsymbol{\chi}} = \mathbf{H}\tilde{\boldsymbol{\eta}}$, where

$$\mathbf{H} = \begin{bmatrix} -\frac{\beta}{\alpha^2+\beta^2} & \frac{\alpha}{\alpha^2+\beta^2} \\ -\frac{\alpha}{\|\boldsymbol{\eta}\|^3} & -\frac{\beta}{\|\boldsymbol{\eta}\|^3} \end{bmatrix}. \tag{31}$$

Hence the linearized covariance of $\boldsymbol{\chi}^k$ is

$$\mathbf{P}_{\boldsymbol{\chi}} = \mathbf{H}\mathbf{P}_{\boldsymbol{\eta}}\mathbf{H}^\top. \tag{32}$$

Note: In practice, this algorithm converges rapidly. With δ_c chosen to be 10^{-5}, the algorithm converges in 2 or 3 iterations.

Note: In practice, QR decomposition is used to compute $\boldsymbol{\eta}^k$ to improve the computational efficiency.

Note: As the number of points n associated with the line increases, the variance of the estimated line parameter $\boldsymbol{\eta}$ decreases. This can be seen from

$$\begin{aligned} \mathbf{P}_{\boldsymbol{\eta}} &= (\mathbf{A}^\top \mathbf{P}_{\mathbf{n}_L}^{-1}\mathbf{A})^{-1} && (33)\\ &= ((\mathbf{P}_{\mathbf{n}_L}^{-\top/2}\mathbf{A})^\top(\mathbf{P}_{\mathbf{n}_L}^{-\top/2}\mathbf{A}))^{-1} && (34)\\ &= (\bar{\mathbf{A}}^\top\bar{\mathbf{A}})^{-1} && (35)\\ &= ({}^L\bar{\mathbf{p}}_1\,{}^L\bar{\mathbf{p}}_1^\top + \ldots + {}^L\bar{\mathbf{p}}_n\,{}^L\bar{\mathbf{p}}_n^\top)^{-1}. && (36) \end{aligned}$$

The diagonal elements of each ${}^L\bar{\mathbf{p}}_i\,{}^L\bar{\mathbf{p}}_i^\top$ are positive. Hence, the more points used in the line fitting, the smaller the diagonal elements of $\mathbf{P}_{\boldsymbol{\eta}}$ will be.

2) Line Merging: Several lines may be extracted from $\mathcal{D}_k$. The i-th and j-th lines can be merged if their *Mahalanobis Distance* passes the chi-squared test:

$$\left\|\boldsymbol{\chi}_i - \boldsymbol{\chi}_j\right\|^2_{(\mathbf{P}_{\boldsymbol{\chi}_i}+\mathbf{P}_{\boldsymbol{\chi}_j})} < \delta_d \tag{37}$$

where the threshold δ_d is computed from the chi-squared distribution.

After we decide to merge the i-th and j-th line, we group the set of points together from two lines and then go back to the line fitting step to obtain the merged line parameters.

3) Feature Association: After the line merging step, the j-th line measurement will be associated to no more than one of the mapped features to form its residual. Feature association uses the *Mahalanobis Distance*: find the k-th mapped feature that minimizes $\left\|\boldsymbol{\chi}_j - \hat{\boldsymbol{\chi}}_k\right\|_{\mathbf{P}_{\boldsymbol{\chi}_j}}$, where $\hat{\boldsymbol{\chi}}_k$ is the predicted line-of-intersection for the k-th mapped feature computed from eqn. (21). If the k-th mapped feature satisfies $\left\|\boldsymbol{\chi}_j - \hat{\boldsymbol{\chi}}_k\right\|_{\mathbf{P}_{\boldsymbol{\chi}_j}} < \delta_a$, then we associate the k-th mapped feature to the j-th line masurement. Otherwise no measurement is associated to this feature. The threshold δ_a is obtained from the chi-squared distribution.

C. LIDAR Residual Formation

To use measurements in the EKF, we define the measurement equation and the measurement prediction as:

$$\mathbf{y} = h(\mathbf{x}, \mathbf{n}), \quad \hat{\mathbf{y}} = h(\hat{\mathbf{x}}, \mathbf{0}) \tag{38}$$

where $h(\cdot,\cdot)$ is a nonlinear function of the current state $\mathbf{x}$ and measurement noise $\mathbf{n}$, $\mathbf{y}$ is the measurement vector and $\hat{\mathbf{y}}$ is the predicted measurement. Then the residual is defined as $\mathbf{r} = \mathbf{y} - \hat{\mathbf{y}}$. The measurement equation $h(\mathbf{x}, \mathbf{n})$ linearized around the estimated state $\hat{\mathbf{x}}$ is

$$\mathbf{r} = \mathbf{H}\tilde{\mathbf{x}} + \boldsymbol{\Gamma}\mathbf{n} \tag{39}$$

where $\tilde{\mathbf{x}} = \mathbf{x} - \hat{\mathbf{x}}$.

In LIDAR aiding, we use two measurements: ϕ and ρ. Here we define ϕ and ρ to be the measurements obtained from the line extraction step. The measurement noise is defined as $\mathbf{n} = [\tilde{\phi} \;\; \tilde{\rho}]^\top$, and we assume $\mathbf{n}$ to be zero-mean, white Gaussian with covariance matrix $\mathbf{R} \triangleq \mathbf{P}_{\boldsymbol{\chi}}$, where $\mathbf{P}_{\boldsymbol{\chi}}$ is given in eqn. (32). The measurement equations of ϕ and ρ are given in eqn. (21), and are rewritten here for clarity:

$$\phi = h_1(\mathbf{x}, \mathbf{n}) = \arctan\left(\text{sgn}({}^L d_i)\frac{a_2}{a_1}\right) + \tilde{\phi}, \tag{40}$$

$$\rho = h_2(\mathbf{x}, \mathbf{n}) = \frac{|{}^L d_i|}{\sqrt{a_1^2 + a_2^2}} + \tilde{\rho} \tag{41}$$

where ${}^L d_i = {}^G d_i - {}^G\boldsymbol{\pi}_i \cdot ({}^G\mathbf{p}_I + {}^G_I\mathbf{R}\,{}^I\mathbf{p}_L)$, ${}^L\mathbf{a}_i = [a_1 \; a_2 \; a_3]^\top$ and ${}^L\mathbf{a}_i = {}^L_G\mathbf{R}\,{}^G\boldsymbol{\pi}_i$. The prediction $\hat{\phi}$ and $\hat{\rho}$ are computed by $\hat{\phi} = h_1(\hat{\mathbf{x}}, \mathbf{0})$ and $\hat{\rho} = h_2(\hat{\mathbf{x}}, \mathbf{0})$. Hence we can define two residuals r_1 and r_2 to be $r_1 = \phi - \hat{\phi}$ and $r_2 = \rho - \hat{\rho}$. Since r_1 is an angle residual, it is normalized into $[-\pi, \pi]$ in practice. In the following we will form the residual model equations in the form of eqn. (39) for ϕ and ρ, respectively.

1) Residual of ϕ: For ϕ, we have $\frac{\partial h_1(\cdot)}{\partial {}^L\mathbf{a}_i} = \frac{1}{\mu}\boldsymbol{\lambda}^\top$, where $\mu = a_1^2 + a_2^2$ and $\boldsymbol{\lambda}^\top = \text{sgn}({}^L\hat{d}_i)\begin{bmatrix} -a_2 & a_1 & 0 \end{bmatrix}$. To compute $\frac{\partial {}^L\mathbf{a}_i}{\partial(\delta\boldsymbol{\theta})}$, we linearize ${}^L\mathbf{a}_i$ using the estimated state to obtain

$$\begin{aligned} {}^L\mathbf{a}_i &= {}^L_G\mathbf{R}\,{}^G\boldsymbol{\pi}_i && (42)\\ &= {}^L_I\mathbf{R}\,{}^I_G\mathbf{R}\,{}^G\boldsymbol{\pi}_i && (43)\\ &= {}^L_I\mathbf{R}\,{}^I_G\hat{\mathbf{R}}(\mathbf{I} - \lfloor\delta\boldsymbol{\theta}\times\rfloor)\,{}^G\boldsymbol{\pi}_i && (44)\\ &= {}^L\hat{\mathbf{a}}_i + {}^L_I\mathbf{R}\,{}^I_G\hat{\mathbf{R}}\lfloor{}^G\boldsymbol{\pi}_i\times\rfloor\delta\boldsymbol{\theta}. && (45) \end{aligned}$$

The above equation yields

$$\frac{\partial {}^L\mathbf{a}_i}{\partial(\delta\boldsymbol{\theta})} = {}^L_I\mathbf{R}\,{}^I_G\hat{\mathbf{R}}\lfloor{}^G\boldsymbol{\pi}_i\times\rfloor. \tag{46}$$

Thus we can write

$$\begin{aligned} \frac{\partial h_1(\cdot)}{\partial(\delta\boldsymbol{\theta})} &= \frac{\partial h_1(\cdot)}{\partial {}^L\mathbf{a}_i}\cdot\frac{\partial {}^L\mathbf{a}_i}{\partial(\delta\boldsymbol{\theta})} && (47)\\ &= \frac{1}{\mu}\boldsymbol{\lambda}^\top\,{}^L_I\mathbf{R}\,{}^I_G\hat{\mathbf{R}}\lfloor{}^G\boldsymbol{\pi}_i\times\rfloor. && (48) \end{aligned}$$

Hence the linearized residual model for ϕ is

$$r_1 = \mathbf{h}_1^\top\tilde{\mathbf{x}} + \tilde{\phi} \tag{49}$$

where $\mathbf{h}_1^\top = \begin{bmatrix} \mathbf{0}_{1\times 6} & \frac{\partial h_1(\cdot)}{\partial(\delta\boldsymbol{\theta})} & \mathbf{0}_{1\times 6} \end{bmatrix}$.

2) Residual of ρ: For ρ, we have

$$\begin{aligned} \frac{\partial h_2(\cdot)}{\partial {}^G\mathbf{p}_I} &= \frac{1}{\sqrt{\mu}}\text{sgn}({}^L\hat{d}_i)\frac{\partial {}^L d_i}{\partial {}^G\mathbf{p}_I} + {}^L d_i\left(\boldsymbol{\kappa}^\top\frac{\partial {}^L\mathbf{a}_i}{\partial {}^G\mathbf{p}_I}\right) && (50)\\ &= \frac{1}{\sqrt{\mu}}\text{sgn}({}^L\hat{d}_i)(-{}^G\boldsymbol{\pi}_i^\top) + {}^L d_i(\boldsymbol{\kappa}^\top\cdot\mathbf{0}) && (51)\\ &= -\text{sgn}({}^L\hat{d}_i)\frac{{}^G\boldsymbol{\pi}_i^\top}{\sqrt{\mu}} && (52) \end{aligned}$$

where $\boldsymbol{\kappa}^\top \triangleq \frac{\partial(1/\sqrt{\mu})}{\partial {}^L\mathbf{a}_i} = -\mu^{-\frac{3}{2}}\begin{bmatrix} a_1 & a_2 & 0\end{bmatrix}$. In addition, we have

$$\frac{\partial h_2(\cdot)}{\partial(\delta\boldsymbol{\theta})} = \frac{1}{\sqrt{\mu}}\text{sgn}({}^L\hat{d}_i)\frac{\partial {}^Ld_i}{\partial(\delta\boldsymbol{\theta})} + {}^Ld_i\left(\boldsymbol{\kappa}^\top\frac{\partial {}^L\mathbf{a}_i}{\partial(\delta\boldsymbol{\theta})}\right). \quad (53)$$

The partial $\frac{\partial {}^L\mathbf{a}_i}{\partial(\delta\boldsymbol{\theta})}$ is given in eqn. (46). To compute $\frac{\partial {}^Ld_i}{\partial(\delta\boldsymbol{\theta})}$, we have

$$\begin{aligned} {}^Ld_i &= {}^Gd_i - {}^G\boldsymbol{\pi}_i\cdot({}^G\hat{\mathbf{p}}_I + {}^G_I\mathbf{R}^I\mathbf{p}_L) && (54)\\ &= {}^Gd_i - {}^G\boldsymbol{\pi}_i\cdot{}^G\hat{\mathbf{p}}_I - {}^G\boldsymbol{\pi}_i\cdot(\mathbf{I}+\lfloor\delta\boldsymbol{\theta}\times\rfloor){}^G_I\hat{\mathbf{R}}^I\mathbf{p}_L && (55)\\ &= {}^L\hat{d}_i + {}^G\boldsymbol{\pi}_i\cdot\lfloor{}^G_I\hat{\mathbf{R}}^I\mathbf{p}_L\times\rfloor\delta\boldsymbol{\theta}. && (56)\end{aligned}$$

So

$$\frac{\partial {}^Ld_i}{\partial(\delta\boldsymbol{\theta})} = {}^G\boldsymbol{\pi}_i^\top\lfloor{}^G_I\hat{\mathbf{R}}^I\mathbf{p}_L\times\rfloor. \quad (57)$$

Substituting eqn. (46) and (57) into eqn. (53) yields

$$\begin{aligned}\frac{\partial h_2(\cdot)}{\partial(\delta\boldsymbol{\theta})} = \text{sgn}({}^L\hat{d}_i)\frac{{}^G\boldsymbol{\pi}_i^\top}{\sqrt{\mu}}\lfloor{}^G_I\hat{\mathbf{R}}^I\mathbf{p}_L\times\rfloor\\ + {}^Ld_i({\boldsymbol{\kappa}}^\top{}^L_I\mathbf{R}^I_G\hat{\mathbf{R}}\lfloor{}^G\boldsymbol{\pi}_i\times\rfloor). \quad (58)\end{aligned}$$

Hence the linearized residual model of ρ is

$$r_2 = \mathbf{h}_2^\top\tilde{\mathbf{x}} + \tilde{\rho} \quad (59)$$

where $\mathbf{h}_2^\top = \begin{bmatrix}\frac{\partial h_2(\cdot)}{\partial {}^G\mathbf{p}_I} & \mathbf{0}_{1\times3} & \frac{\partial h_2(\cdot)}{\partial(\delta\boldsymbol{\theta})} & \mathbf{0}_{1\times6}\end{bmatrix}$.

3) Stacked Residual Dynamics: Stacking (49) and (59) together, we obtain the residual dynamics in the form shown in (39), with

$$\mathbf{H} = \begin{bmatrix}\mathbf{h}_1 & \mathbf{h}_2\end{bmatrix}^\top, \quad \boldsymbol{\Gamma} = \mathbf{I} \quad (60)$$

and the residual vector $\mathbf{r} = \begin{bmatrix}r_1 & r_2\end{bmatrix}^\top$.

D. Measurement Aiding

The Kalman filter gain is computed as

$$\mathbf{K} = \mathbf{P}_{k+1|k}\mathbf{H}^\top(\mathbf{H}\mathbf{P}_{k+1|k}\mathbf{H}^\top + \mathbf{R})^{-1}. \quad (61)$$

The state and covariance measurement update equations are

$$\begin{aligned}\hat{\mathbf{x}}_{k+1|k+1} &= \hat{\mathbf{x}}_{k+1|k} + \mathbf{K}\mathbf{r},\\ \mathbf{P}_{k+1|k+1} &= (\mathbf{I}-\mathbf{K}\mathbf{H})\mathbf{P}_{k+1|k}(\mathbf{I}-\mathbf{K}\mathbf{H})^\top + \mathbf{K}\mathbf{R}\mathbf{K}^\top.\end{aligned}$$

VI. Experimental Results

The IMU used in this project is a MEM's device with 60 Hz bandwidth that provides measurements at 200 Hz. The LIDAR scan rate is 20 Hz, but we only use its measurements to aid the INS at 1.0 Hz. The effective LIDAR range is 0.3 to 30 m, and the angular resolution is 0.25 degree. The scanning angle is configured to be 180 degrees. GPS aiding occurs at 1.0 Hz. It provides range, Doppler, phase measurements. With carrier phase differential GPS aided INS, the positioning accuracy will be centimeter to decimeter level when we have an open sky.

In typical urban environments, GPS signals are blocked by buildings in the lateral direction while the longitude direction remains relatively open. As our GNSS+INS system uses a tightly coupled design, the positioning uncertainty will therefore be larger in the lateral direction. The goal is to use the LIDAR to aid state estimation by detecting features along the road in the lateral direction. Therefore, the LIDAR is mounted on the right hand side of the vehicle, pointing to the right. This mounting also facilitates detection of features such as the sides of buildings in the urban environment. The IMU/GPS unit is mounted on top center of the vehicle.

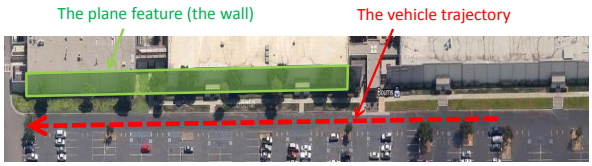

Fig. 3. The vehicle trajectory and the plane feature. The red line points due east.

Only one feature Π_1 is mapped for this preliminary experimental demonstration. The parameters of this feature are ${}^G\boldsymbol{\pi}_1 = [1\ 0\ 0]^\top$ and the ${}^Gd_1 = 2771.93$. The method extends directly to larger numbers of mapped planes.

We simulate the urban environment by driving the car parallel to a building in a parking lot (see Fig. 3). In this experiment, LIDAR aiding is on for the entire time. The GPS aiding is on in the beginning and turned off at time 70 sec to simulate the loss of GPS signals. The plane feature can be detected by LIDAR roughly after 65 sec. In Fig. 4, the measurement residuals and the number of points used to fit the line measurement are shown. The standard deviation of the estimated position and yaw angle are given in Fig. 5. In Fig. 4, we can see that even with GPS aiding off after 70 sec, the use of the LIDAR residuals by the EKF keeps the LIDAR residuals small and within the three standard deviation prediction for the residuals (blue curve), which is computed in part using the covariance of the line parameters. We plot the predicted 3σ of residuals (blue) using the theory presented in Section V, which is time varying as a function of the number of LIDAR points used for line fitting. This is consistent with the time variation of the observed residuals (red). The variation in the number of points used for line fitting is due to the trees in front of the building cutting the line-of-intersection into multiple line segments. In spite of the interference from trees, our LIDAR processing methods can still merge the line segments into a single line measurement to aid the INS. Fig. 5 shows that the estimation accuracy in the north direction and the yaw angle remains accurate. This is due to the LIDAR aiding using a wall with its normal vector in the north direction. In contrast, the uncertainty along east and down directions grow because these directions are not observable from (i.e., are orthogonal to) the LIDAR measurement. The increase in the standard deviation in the north direction near $t = 80$ sec is due to the decreased number of points used in the line fitting during that period of time.

VII. Conclusions and Future Work

This paper presents a novel solution utilizing *2D* LIDAR to aid an INS to maintain or improve the *3D* vehicle positioning

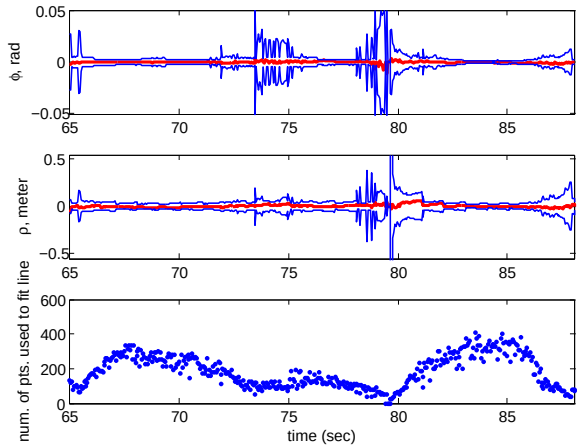

Fig. 4. Feature residuals (red), 3σ standard deviation (blue) and the number of points used to fit the line (blue dots).

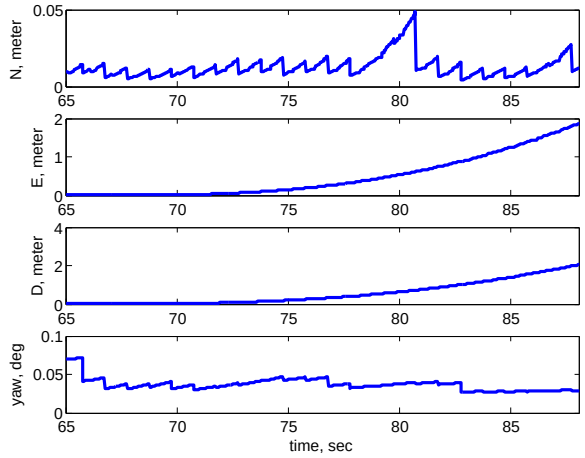

Fig. 5. The standard deviation (1σ) of the estimated position and yaw.

accuracy, especially when the GNSS signals are shadowed. An analytic formula is derived to predict the line measurement in the LIDAR frame and based on this formula, new solutions are proposed for the feature association, residual formation and GUI display. Preliminary experimental results show the effectiveness of the LIDAR aiding to reduce the lateral direction uncertainty along the road when the GPS signals are unavailable.

REFERENCES

[1] E. Brunskill and N. Roy, "SLAM using incremental probabilistic PCA and dimensionality reduction," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*. IEEE, 2005, pp. 342–347.

[2] A. Bry, A. Bachrach, and N. Roy, "State estimation for aggressive flight in GPS-denied environments using onboard sensing," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 1–8.

[3] J. Farrell, *Aided navigation: GPS with high rate sensors*. McGraw-Hill New York, NY, USA:, 2008.

[4] J. A. Farrell, T. Givargis, and M. Barth, "Real-time differential carrier phase GPS-aided INS," *IEEE Transactions on Control Systems Technology*, vol. 8, no. 4, pp. 709–721, 2000.

[5] J. A. Farrell, H. Tan, and Y. Yang, "Carrier phase GPS-aided INS based vehicle lateral control," *ASME Journal of Dynamics Systems, Measurement, & Control*, vol. 125, no. 3, pp. 339–353, 2003.

[6] J. Hesch, F. Mirzaei, G. Mariottini, and S. Roumeliotis, "A laser-aided inertial navigation system (L-INS) for human localization in unknown indoor environments," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 5376–5382.

[7] K. Lingemann, A. Nüchter, J. Hertzberg, and H. Surmann, "High-speed laser localization for mobile robots," *Robotics and Autonomous Systems*, vol. 51, no. 4, pp. 275–296, 2005.

[8] V. Nguyen, A. Martinelli, N. Tomatis, and R. Siegwart, "A comparison of line extraction algorithms using 2D laser rangefinder for indoor mobile robotics," in *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*. IEEE, 2005, pp. 1929–1934.

[9] S. Pfister, K. Kriechbaum, S. Roumeliotis, and J. Burdick, "Weighted range sensor matching algorithms for mobile robot displacement estimation," in *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, vol. 2. IEEE, 2002, pp. 1667–1674.

[10] S. Thrun, D. Fox, W. Burgard, and F. Dellaert, "Robust Monte Carlo localization for mobile robots," *Artificial intelligence*, vol. 128, no. 1, pp. 99–141, 2001.

[11] A. Vu, A. Ramanandan, A. Chen, J. A. Farrell, and M. Barth, "Real-time computer vision/dgps-aided inertial navigation system for lane-level vehicle navigation," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 13, no. 2, pp. 899–913, 2012.

APPENDIX

This section derives eqn. (21) which is a closed-form solution to the optimization problem of eqn. (19). The solution of this optimization problem is

$$^{L}\hat{\mathbf{x}}_i = \mathbf{A}^\top(\mathbf{A}\mathbf{A}^\top)^{-1}\mathbf{b} \tag{62}$$

where $\mathbf{A} = \begin{bmatrix} ^{L}\mathbf{z}_L & {}^{L}_{G}\mathbf{R}\,{}^{G}\boldsymbol{\pi}_i \end{bmatrix}^\top$ and $\mathbf{b} = \begin{bmatrix} 0 & {}^{L}\hat{d}_i \end{bmatrix}^\top$. Let $\mathbf{a} = {}^{L}_{G}\mathbf{R}\,{}^{G}\boldsymbol{\pi}_i = \begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix}^\top$ which is a unit vector. The solution procedes as follows:

$$(\mathbf{A}\mathbf{A}^\top)^{-1} = \frac{\begin{bmatrix} 1 & -a_3 \\ -a_3 & 1 \end{bmatrix}}{1 - a_3^2}. \tag{63}$$

Substituting eqn. (63) into eqn. (62) yields

$$^{L}\hat{\mathbf{x}}_i = \mathbf{A}^\top(\mathbf{A}\mathbf{A}^\top)^{-1}\mathbf{b} = \begin{bmatrix} a_1 \\ a_2 \\ 0 \end{bmatrix} \frac{^{L}\hat{d}_i}{a_1^2 + a_2^2}. \tag{64}$$

Therefore, we have

$$\hat{\phi}_i = \arctan\left(\text{sgn}(^{L}\hat{d}_i)\frac{a_2}{a_1}\right), \quad \hat{\rho}_i = \frac{|^{L}\hat{d}_i|}{\sqrt{a_1^2 + a_2^2}}. \tag{65}$$