sensors

IMPACT FACTOR 3.4

Indexed in: PubMed

CITESCORE 7.3

Article

A Robust Vehicle Localization Approach Based on GNSS/IMU/DMI/LiDAR Sensor Fusion for Autonomous Vehicles

Xiaoli Meng, Heng Wang and Bingbing Liu

Special Issue

Advances in Multi-Sensor Information Fusion: Theory and Applications 2017

Edited by

Prof. Dr. Xue-Bo Jin, Prof. Dr. Shuli Sun, Prof. Dr. Hong Wei and Prof. Dr. Feng-Bao Yang

MDPI

https://doi.org/10.3390/s17092140

sensors

MDPI

Article

A Robust Vehicle Localization Approach Based on GNSS/IMU/DMI/LiDAR Sensor Fusion for Autonomous Vehicles

Xiaoli Meng [1], Heng Wang [2,*] and Bingbing Liu [1]

[1] Institute for Infocomm Research, Agency for Science, Technology and Research (A*STAR), Singapore; xiaoli.meng09@gmail.com (X.M.); bliu@i2r.a-star.edu.sg (B.L.)
[2] College of Information Science and Engineering, Northeastern University, Shenyang 110819, China
* Correspondence: wangheng@ise.neu.edu.cn; Tel.: +86-024-83681939

Received: 25 August 2017; Accepted: 15 September 2017; Published: 18 September 2017

Abstract: Precise and robust localization in a large-scale outdoor environment is essential for an autonomous vehicle. In order to improve the performance of the fusion of GNSS (Global Navigation Satellite System)/IMU (Inertial Measurement Unit)/DMI (Distance-Measuring Instruments), a multi-constraint fault detection approach is proposed to smooth the vehicle locations in spite of GNSS jumps. Furthermore, the lateral localization error is compensated by the point cloud-based lateral localization method proposed in this paper. Experiment results have verified the algorithms proposed in this paper, which shows that the algorithms proposed in this paper are capable of providing precise and robust vehicle localization.

Keywords: sensor fusion; Unscented Kalman Filter (UKF); vehicle localization

1. Introduction

Automated driving techniques are widely admitted as a promising and challenging way to avoid road crashes and improve traffic conditions [1]. To make an autonomous vehicle drive safely in urban environments, the vehicle needs to know its exact position and orientation. Thus, localization plays a key role in autonomous vehicle applications. Using the popular GNSS (Global Navigation Satellite System) for localization requires a GNSS receiver with an unobstructed line of sight to four or more GNSS satellites. However, even with a high-end GNSS-based system, a vehicle's location may jump up to a few meters as different satellites go in and out of view or obstructions in the environments create multi-path interferences. INS (Inertial Navigation System) is complementary to GNSS as it does not rely on external information sources, which can be blocked or disturbed. INS can provide complete navigation information such as position, velocity and attitude by integrating the accelerometer and gyroscope readings over time. However, as the inertial sensors of an INS are subject to drifts, navigation systems based on stand-alone INS suffer a rapid degradation of position over time. This is particularly true when a low-cost IMU (Inertial Measurement Unit) is employed. In the dead-reckoning integration scheme, wheel encoders are always introduced to slow the rate of growth of IMU integration errors, but are subject to errors due to wheel slip. However, a robust localization solution can be achieved by blending GNSS, INS and DMI (Distance Measuring Instruments) techniques in a way that utilizes the strengths of each individual system and mitigates their weaknesses.

To fuse GNSS, INS and DMI, Extended Kalman Filtering (EKF) is a popular sensor fusion method [2,3], where nonlinear systems are linearized and approximated around current state estimates. However, in the EKF, high-order terms are neglected, which are necessary for some situations. The Particle Filter (PF) [4,5] is another useful method, but the main drawback of this filter is its computational requirement, which makes it not very suitable for real-time applications. In [6],

an INS/GPS sensor fusion scheme based on the State-Dependent Riccati Equation (SDRE) nonlinear filtering method is proposed for Unmanned Aerial Vehicles (UAV), which is widely used in the optimal nonlinear control and filtering literature. Although this method provides an alternative INS/GPS filtering scheme, the real-time performance and application to autonomous vehicles in urban environments are still uncertain. In [7,8], low cost sensors such as cameras are fused with GNSS/INS to improve the localization accuracy of GNSS in dense urban areas where obstacles block satellite signals; however, the improvement of accuracy is limited due to the difficulties in some feature recognition tasks and the calculating of depth information using cameras. Recently, the Unscented Kalman Filter (UKF) has been used for localization based on GPS/INS sensor fusion [9–11] due to the ability to remove the messy Jacobian matrix computation and keep at least a second-order nonlinear function approximation. Although UKF has been proven to be a promising method for GPS/INS fusion, the accuracy and reliability performance still need to be improved for autonomous vehicles under urban environments.

On the other hand, the localization approach based solely on GNSS/IMU/DMI cannot always guarantee a precise location solution due to the existence of the blocking of satellite signals by obstacles (buildings and trees, etc.) and the cumulative errors of IMU and DMI sensors. In order to provide precise localization for positioning an autonomous vehicle reliably, we need to explore other useful information to position an autonomous vehicle. In the urban environment, curbs and lane markings comprise two kinds of useful information for improving the results of GPS/INS/DMI fusion. For example, in [12–15], cameras are fused with other sensors such as GPS and IMU to improve the lateral accuracy by detecting lane markings. However, the detection of lane marking needs to face the challenges of different lighting, poor lane markings, etc. On the other hand, 3D point clouds generated by a 3D LiDAR scanner provide more reliable performance for curb detection, which can also be used to improve the lateral localization accuracy [16–20]. Furthermore, in [21], an eigenvector technique was used to find a line segment corresponding to edges of roads. In [22], a Hough transform was used to find the best fit line to the surface on the road, and points corresponding to the best fit line were used as curb points of the road. In [23], curbs are extracted by using a 1D laser scanner.

In this paper, to further improve the accuracy and reliability of localization for autonomous vehicles in urban environments, we firstly propose a fault-detection-based loosely-coupled GNSS/IMU/DMI localization solution, which can improve the performance of the traditional UKF-based method; then, we correct the lateral localization errors based on curb detection results using a multi-layer LiDAR. The flowchart of overall localization method is summarized in Figure 1.

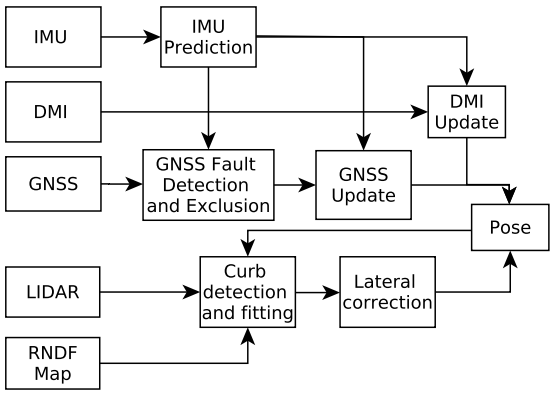

Figure 1. The flowchart of the proposed method. DMI, Distance-Measuring Instrument; RNDF, Route Network Definition File.

The main contribution of this paper is summarized as follows. Firstly, through combining the fault-detection method with the UKF-based GNSS/IMU/DMI fusion algorithm, the localization

accuracy of autonomous vehicles is improved greatly; Secondly, a point cloud-based curb detection and fitting method is proposed to improve the lateral accuracy of the autonomous vehicle further, where the RANSAC algorithm is utilized. The rest of the paper is organized as follows: Section 2 presents the proposed UKF-based localization approach: firstly, the modeling including the process model and the measurement model is introduced, followed by the implementation of the UKF. Details about curb-based lateral localization are provided in Section 3. Section 4 presents the experimental results. Finally, we conclude the paper in Section 5.

2. UKF-Based Localization Approach

For a vehicle localization system, four coordinate systems are defined:

- Earth-Centered-Earth-Fixed (ECEF) coordinate system (E): It has an origin at the center of the Earth. The positive Z-axis goes out the Earth's north pole; the X-axis is along the prime meridian; and the Y-axis completes the right-handed system;
- Global coordinate system (G): The North-East-Down (NED) coordinate system is defined as G with the X-axis pointing north, the Y-axis pointing east and the Z-axis pointing down to construct a right-handed coordinate system;
- Body coordinate system (B): The coordinate system of the vehicle with the X-axis pointing forwards, the Y-axis pointing left and the Z-axis pointing up;
- Sensor coordinate system (S): the three orthogonal axes of the mounted sensors. We assume that S coincides with B after sensor to body alignment calibration [24].

One should note that each sensor defines its own coordinate system. We need to note the difference between the origins for accurate localizations.

We describe the states of the filtering system with the following vector:

$$\mathbf{x} = [\mathbf{p}, \mathbf{v}, \mathbf{q}, \mathbf{b}, \mathbf{d}]^T_{16\times 1} \tag{1}$$

where $\mathbf{p}$ and $\mathbf{v}$ are the position and velocity of the vehicle within the global frame G, respectively. $\mathbf{q}$ is a unit quaternion that represents the rotation from the body frame B to the global frame G. A unit quaternion consists of a vector part $\mathbf{e} = (q_1, q_2, q_3)^T \in \mathbb{R}^3$ and a scalar part $q4 \in \mathbb{R}$ [25]:

$$\mathbf{q} = \left[\mathbf{e}^T, q_4\right]^T = [q_1, q_2, q_3, q_4]^T$$

and its norm equals one, that is:

$$\|\mathbf{q}\| = 1.$$

As accelerometers and gyroscopes have biases, which can be modeled as random walk processes, we have two additional vectors $\mathbf{b}$ and $\mathbf{d}$ in the state vector to represent their biases, respectively. Both variables are given within body frame B.

2.1. Process Model

The process model governs the dynamic relationship between the states of two successive time steps, which can be described by:

$$\mathbf{x}_t = f\left(\mathbf{x}_{t-1}, \mathbf{u}_{t-1}\right) + \mathbf{w}_{t-1} \tag{2}$$

where $\mathbf{x}_t$ is the predicted state after time period δ based on the last known state vector $\mathbf{x}_{t-1}$, $\mathbf{u}_{t-1}$ is the input to the state space models and $\mathbf{w}_{t-1}$ is the process noise. In this study, we use the following process model:

$$\begin{aligned}\mathbf{x}_t &= f\left(\mathbf{x}_{t-1}, \mathbf{u}_{t-1}\right) + \mathbf{w}_{t-1} \\ &= \begin{bmatrix} \mathbf{p}_{t-1} + \mathbf{v}_{t-1}\delta \\ \mathbf{v}_{t-1} + \mathbf{C}(\mathbf{q}_{t-1})(\mathbf{y}_{a,t-1} - \mathbf{b}_{t-1})\delta - \mathbf{g}\delta \\ \exp\left(\frac{1}{2}\Omega\left[\mathbf{y}_{\omega,t-1} - \mathbf{d}_{t-1}\right]\delta\right)\mathbf{q}_{t-1} \\ (1 - 1/\tau_a)\mathbf{b}_{t-1} \\ (1 - 1/\tau_\omega)\mathbf{d}_{t-1} \end{bmatrix} + \begin{bmatrix} \mathbf{w}_{p,t-1} \\ \mathbf{w}_{v,t-1} \\ \mathbf{w}_{q,t-1} \\ \mathbf{w}_{b,t-1} \\ \mathbf{w}_{d,t-1} \end{bmatrix}\end{aligned} \tag{3}$$

where $\mathbf{u}_{t-1} = [\mathbf{y}_{a,t-1}, \mathbf{y}_{\omega,t-1}]$ is the measurement vector from the accelerometer and gyroscope at time step $t-1$, $\mathbf{C}(\mathbf{q}_{t-1})$ is the corresponding rotation matrix of the quaternion $\mathbf{q}_{t-1}$ [26]:

$$\mathbf{C}(\mathbf{q}_{t-1}) = \left(q_{4,t-1}^2 - \mathbf{e}_{t-1}^T\mathbf{e}_{t-1}\right)\mathbf{I}_3 + 2\mathbf{e}_{t-1}\mathbf{e}_{t-1}^T - 2q_{4,t-1}[\mathbf{e}_{t-1}]_\times$$

representing the transformation from the body frame B to the global frame G, $\mathbf{g}$ is the gravitational acceleration vector and $\Omega[\boldsymbol{\omega}]$ is a 4×4 skew symmetric matrix, as in:

$$\Omega[\boldsymbol{\omega}] = \begin{bmatrix} -[\boldsymbol{\omega}]_\times & \boldsymbol{\omega} \\ -\boldsymbol{\omega}^T & 0 \end{bmatrix}$$

where $[\boldsymbol{\omega}]_\times$ is defined by

$$[\boldsymbol{\omega}]_\times = \begin{bmatrix} 0 & -\omega^Z & \omega^Y \\ \omega^Z & 0 & -\omega^X \\ -\omega^Y & \omega^X & 0 \end{bmatrix}$$

where $\left[\omega^X, \omega^Y, \omega^Z\right]$ are the three elements of angular rates on the X-, Y- and Z-axis, respectively, and τ_a and τ_ω are time constants. Process noise is added by the noise vector $\mathbf{w}_{t-1} = \left[\mathbf{w}_{p,t-1}, \mathbf{w}_{v,t-1}, \mathbf{w}_{q,t-1}, \mathbf{w}_{b,t-1}, \mathbf{w}_{d,t-1}\right]^T$. Each noise item is modeled as a zero mean Gaussian noise with covariance matrix $\mathbf{Q}_p, \mathbf{Q}_v, \mathbf{Q}_q, \mathbf{Q}_b, \mathbf{Q}_d$, respectively. The noise items are assumed to be uncorrelated with each other; thus, the process noise covariance matrix has the following expressions:

$$\mathbf{Q} = \operatorname{diag}\left[\mathbf{Q}_p, \mathbf{Q}_v, \mathbf{Q}_q, \mathbf{Q}_b, \mathbf{Q}_d\right]. \tag{4}$$

2.2. Measurement Model

The measurement model governs the relationship between the state vector and sensor measurements, which is:

$$\mathbf{y}_t = g\left(\mathbf{x}_t\right) + \mathbf{n}_t \tag{5}$$

In this paper, measurements from the GNSS receiver and encoder are introduced to bound the errors in estimates of the vehicle position/velocity and attitude. Asynchronous updates are performed within the UKF as measurements become available from the wheel encoder and GNSS receiver. For wheel encoder measurements, a filter update is calculated from the measured speed of the vehicle. For GNSS measurements, a filter update is calculated from the location of the Trimble.

2.2.1. Measurement Model of GNSS

The GNSS receiver could be a differential GNSS receiver or RTK receiver, which measures data at a relatively low frequency (the measurement update rate is up to 20 Hz). When a measurement from the GNSS receiver is available, the GNSS measurement model is given by:

$$\mathbf{y}_p = \mathbf{p} + \mathbf{n}_p \tag{6}$$

where $\mathbf{n}_p$ is the GNSS measurement noise modeled as a Gaussian noise, $N(0,\mathbf{\Sigma}_{GNNS})$, $\mathbf{y}_p = \mathbf{C}_E^G[p^X, p^Y, p^Z]$, and $[p^X, p^Y, p^Z]$ is the position on the X-, Y- and Z-axis in the ECEF frame, calculated by the following equations, respectively:

$$\begin{aligned} p^X &= (N+h)\cos\lambda\cos\phi \\ p^Y &= (N+h)\cos\lambda\sin\phi \\ p^Z &= \left[N\left(1-e^2\right)+h\right]\sin\lambda \end{aligned}$$

where $[\lambda, \phi, h]$ are the latitude, longitude and altitude provided by GNSS fixes. The parameters used above are defined as follows: $N = a/\sqrt{1-e^2\sin\lambda^2}$ is the length from the center of the Earth to the surface; $e = \sqrt{1-b^2/a^2}$ is the Earth eccentricity; and a = 6,378,137 (m), b = 6,356,752.3142 (m) are the Earth ellipsoid semi-major and semi-minor axes, respectively. The transition matrix from the ECEF frame to the global frame G, is denoted by $\mathbf{C}_E^G$ as

$$\mathbf{C}_E^G = \begin{bmatrix} -\sin\lambda\cos\phi & -\sin\lambda\sin\phi & \cos\lambda \\ -\sin\phi & -\cos\phi & 0 \\ -\cos\lambda\cos\phi & -\cos\lambda\sin\phi & -\sin\lambda \end{bmatrix}.$$

The covariance matrix of the GNSS measurement noise is $\mathbf{R} = \mathbf{\Sigma}_{GNNS}$.

2.2.2. Measurement Model of DMI

The measurement model for the encoder is modeled as:

$$\mathbf{y}_v = \mathbf{q}^{-1} \otimes \mathbf{v} \otimes \mathbf{q} + \mathbf{n}_v \tag{7}$$

where $\mathbf{y}_v = [v, 0, 0]$ is the velocity of the vehicle in the body frame, v is the wheel speed measurement from the encoder and $\otimes$ represents the quaternion multiplication [25]. $\mathbf{n}_v$ is the encoder measurement noise modeled as a Gaussian noise, $N(0,\mathbf{\Sigma}_{Encoder})$, and the covariance matrix of the encoder measurement noise is $\mathbf{R} = \mathbf{\Sigma}_{encoder}$.

2.3. Implementation of UKF

In process Model (3), due to the nonlinearity in the quaternion and velocity state functions, unscented transform-based approximation to the optimal filtering solution can be derived by executing two steps of time update and measurement update in turn. In the execution of the two steps, unscented transform is always carried out first to form the sigma points of the state vector.

2.3.1. Time Update

At time step t, sigma points need to be calculated first and followed by performing the time update using time update equations.

- Calculate the sigma points:

$$\bar{\mathbf{\chi}}_{t-1} = \begin{bmatrix} \mathbf{x}_{t-1} & \mathbf{x}_{t-1} \pm \sqrt{(n+\kappa)\,\mathbf{P}_{t-1}} \end{bmatrix} \tag{8}$$

where $\bar{\mathbf{\chi}}_{t-1}$ are the sigma points of state vector $\mathbf{x}$ at previous time step $t-1$, n is the dimension of the state vector $\mathbf{x}$, $\kappa = \alpha^2(n+\gamma) - n$. α determines the spread of the sigma points and γ is a secondary scaling parameter, which is usually set to one. One should note that the initial condition $\mathbf{x}_0 \sim N(\mathbf{x}_0, \mathbf{P}_0)$ should be known.

- Time update process:

$$\bar{\chi}_{t|t-1} = f\left(\chi_{t-1}, \mathbf{u}_{t-1}\right) \tag{9}$$

$$\mathbf{x}_t^- = \sum_{i=0}^{2n} \mathbf{W}_i^m \bar{\chi}_{i,t|t-1} \tag{10}$$

$$\mathbf{P}_t^- = \sum_{i=0}^{2n} \mathbf{W}_i^c \left(\bar{\chi}_{i,t|t-1} - \mathbf{x}_t^-\right)\left(\bar{\chi}_{i,t|t-1} - \mathbf{x}_t^-\right)^T + \mathbf{Q} \tag{11}$$

where $\mathbf{x}_t^-$ and $\mathbf{P}_t^-$ are the predicted mean and covariance, respectively, and $\mathbf{W}_i^m$ and $\mathbf{W}_i^c$ are the weights of mean and covariance, which are associated with the i-th point, given by [27]:

$$\begin{aligned} \mathbf{W}_0^m &= \frac{\kappa}{n+\kappa} \\ \mathbf{W}_0^c &= \frac{\kappa}{n+\kappa} + (1 - \alpha^2 + \beta) \\ \mathbf{W}_i^m &= \mathbf{W}_i^c = \frac{1}{2(n+\kappa)}, i = 1, 2, ..., 2n \end{aligned}$$

where β is a parameter used to incorporate any prior knowledge about the distribution of state $\mathbf{x}$ (for Gaussian distributions, $\beta = 2$ is optimal).

2.3.2. Measurement Update of GNSS

When fix measurement $\mathbf{y} = \mathbf{y}_p$ is available, we can update the nearest state prediction $\mathbf{x}_t^-$ and covariance matrix $\mathbf{P}_t^-$ using the following equations. At first, sigma points need to be calculated and then, measurement update is performed using measurement update equations.

- Calculate the sigma points:

$$\bar{\chi}_t = \left[\begin{array}{cc} \mathbf{x}_t^- & \mathbf{x}_t^- \pm \sqrt{(n+\kappa)\,\mathbf{P}_t^-} \end{array} \right] \tag{12}$$

where $\mathbf{x}_t^-$ and $\mathbf{P}_t^-$ are the predicted mean and covariance from time update at time t, respectively.

- Perform measurement update:

$$\bar{\mathbf{Y}}_t = g\left(\bar{\chi}_t\right) \tag{13}$$

$$\bar{\mathbf{y}}_t = \sum_{i=0}^{2n} \mathbf{W}_i^m \bar{\mathbf{Y}}_{i,t} \tag{14}$$

$$\mathbf{P}_{y_t} = \sum_{i=0}^{2n} \mathbf{W}_i^c \left(\bar{\mathbf{Y}}_{i,t} - \bar{\mathbf{y}}_t\right)\left(\bar{\mathbf{Y}}_{i,t} - \bar{\mathbf{y}}_t\right)^T + \mathbf{R} \tag{15}$$

$$\mathbf{P}_{x_t y_t} = \sum_{i=0}^{2n} \mathbf{W}_i^c \left(\bar{\chi}_{i,t} - \mathbf{x}_t^-\right)\left(\bar{\mathbf{Y}}_{i,t} - \bar{\mathbf{y}}_t\right)^T \tag{16}$$

$$\mathbf{K}_t = \mathbf{P}_{x_t y_t}\left(\mathbf{P}_{y_t}\right)^{-1} \tag{17}$$

$$\boldsymbol{\nu}_t = \mathbf{y}_{p,t} - \bar{\mathbf{y}}_t \tag{18}$$

$$\mathbf{x}_t = \mathbf{x}_t^- + \mathbf{K}_t \boldsymbol{\nu}_t \tag{19}$$

$$\mathbf{P}_t = \mathbf{P}_t^- - \mathbf{K}_t \mathbf{P}_{y_t} \mathbf{K}_t^T \tag{20}$$

where $\bar{\mathbf{Y}}_t$ are the projected sigma points through the measurement function h, $\bar{\mathbf{y}}_t$ is the predicted measurement produced by the weighted sigma points, $\mathbf{P}_{y_t}$ and $\mathbf{P}_{x_t y_t}$ are the predicted measurement covariance and the state-measurement cross-covariance matrix, respectively, $\mathbf{K}_t$ is the Kalman gain, $\boldsymbol{\nu}_t$ is the innovation and $\mathbf{x}_t$ and $\mathbf{P}_t$ are the updated state and covariance at time t, respectively.

2.3.3. Measurement Update of DMI

When wheel speed measurement $\mathbf{y} = \mathbf{y}_v$ is available, we can update the nearest state prediction $\mathbf{x}_t^-$ and covariance matrix $\mathbf{P}_t^-$ using the similar equations with fix measurement update. Similarly, sigma points need to be calculated, and then, measurement update is performed using measurement update equations.

2.4. Automatic Detection the Degradation of GNSS Performance

GNSS suffers multi-path errors if the satellite signals are reflected off one or more surfaces before reaching the receiver antenna. A different set of satellites is utilized for fix determination, which can also alter the GNSS fix. Big gaps occur when GNSS signals become available again after short-term GNSS dropouts due to the presence of trees and buildings. Different checks are adopted to make our pose estimator robust to the aforementioned jumps, which are explained as follows:

Zero-Velocity update: As GNSS receivers due to pseudo-random error cannot output fixed position information when the vehicle stays in a stationary position, if the vehicle is not in motion, which can be well detected from the encoder readings, measurements from the receiver are not used for updates. This will restrict the vehicle to a fixed location.

Number of satellites: If the number of satellites visible to the receiver is four or more, the measurements from the receiver pass the check.

Dilution Of Precision (DOP): If Horizontal DOP (HDOP) or Vertical DOP (VDOP) is larger than a threshold, the measurements will be discarded.

Statistical test: During the GNSS measurements update, if an abrupt jump in the GNSS fix occurs, the correction made by the GNSS measurement update will cause the IMU solutions to incorrectly follow these jumps. The chi-squared test is applied once the innovation $\boldsymbol{\nu}$ and the innovation covariance matrix $\boldsymbol{\Sigma}_\nu$ are obtained in the measurement update [28], as in:

$$\boldsymbol{\nu}^T \boldsymbol{\Sigma}_\nu^{-1} \boldsymbol{\nu} \leq \varsigma \tag{21}$$

where the value ς is usually set to reject the innovations exceeding the 95% threshold. During the GNSS measurement update stage, if (21) holds, then the GNSS fix is accepted, and the measurement update proceeds.

Assessing the new horizontal position reading and subtracting it from the current estimate of position: If the difference is much higher than what it should be when compared to the vehicle speed obtained from the encoder, which is assumed to be an accurate quantity, then the measurements are discarded.

Check the altitude component: If the measurements are much higher than the innovation, the updates will be aborted.

Validity of position change: After the GNSS measurement update, the change in position $\Delta\mathbf{p}$ can be calculated by:

$$\Delta\mathbf{p} = \mathbf{p}_t - \mathbf{p}_{t-1} \tag{22}$$

The change in position is invalid if it satisfies:

$$|\Delta\mathbf{p}| > v\,(1+\eta)\Delta t + \epsilon \vee \left(|\Delta\mathbf{p}| > \epsilon \wedge \frac{\Delta\mathbf{p}}{|\Delta\mathbf{p}|} \cdot \begin{bmatrix} \cos\psi \\ \sin\psi \end{bmatrix} > \tau \right) \tag{23}$$

where ψ represents the travel heading, calculated from quaternion $\mathbf{q}$ [25]. η, ϵ and τ are three constants representing the anticipated percentage velocity error, allowed position jitter and allowable travel direction error, respectively. When $\Delta\mathbf{p}$ is rejected, a predicted position is calculated based on heading and wheel speed.

The accuracy of the estimated location of the vehicle is about 1–3 cm or 1 m depending on RTK or if the differential mode is operating.The resulting lateral offset will not guarantee safe driving for autonomous vehicles. In the following section, we are going to introduce the lateral localization based on LiDAR measurements to improve the accuracy of the localization results.

3. Correction of Lateral Localization Errors

After we get the localization result from the fusion of GNSS/IMU/encoder, the lateral localization error can be calculated as follows. The detection of curbs using LiDAR (Light Detection And Ranging) provides an accurate lateral distance between the vehicle and curb (denote it by d_1); at the same time, the same distance can also be calculated using the localization result from GNSS/encoder and RNDF (Route Network Definition File) information (denote it by d_2). The difference of d_1 and d_2, i.e., $d_1 - d_2$, is the lateral localization error, which can readily be corrected.

Assume that the RNDF (Route Network Definition File) information is accurate enough: given the vehicle position provided by the GNSS/IMU/DMI fusion system, a lateral distance from the vehicle to the curb can always be obtained through a simple geometry calculation; see the yellow line segment in Figure 2.

However, as known to all, the result of the GNSS/IMU/DMI fusion system is always affected by drifts due to the signal failure of GNSS caused by the complexity of the urban environment. This drift may cause the vehicle to hit the curb or rush to the next lane, which may cause a serious accident. In Figure 3, the pink line is the curb line that is fitted by the point cloud collected by a 3D LiDAR, whose accuracy is less than 10 cm, and the orange line is the curb line of the RNDF; obviously, there is a large difference between the vehicle to these two lines, which is caused by the drift of the localization result. Thus, a lateral correction is very necessary for the localization of the autonomous vehicle. To accomplish the goal of LiDAR-based lateral error correction, we first detect longitudinal curbs and use them as measurements to estimate the autonomous vehicle's lateral distance, then compare it with the distance calculated from RNDF, e.g., the yellow line as shown in Figure 2. The difference of these two distances is then used to correct the lateral error of the autonomous vehicle.

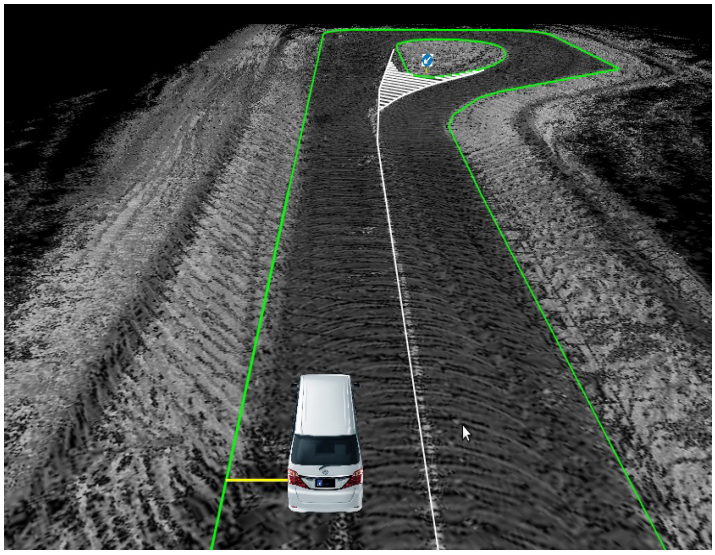

Figure 2. Lateral distance from the vehicle to the curb.

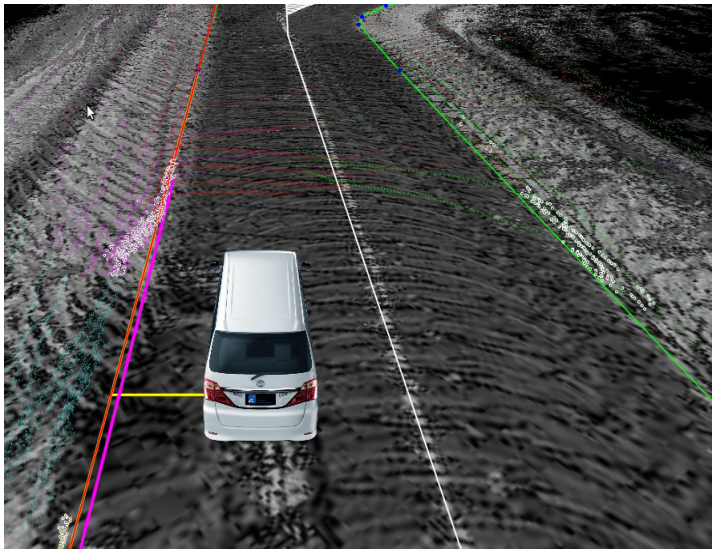

Figure 3. Curb estimated by fitting.

3.1. Curb Detection

3.1.1. Curb Detection Principle

The first step for lateral correction is the curb detection. Figure 4 shows the curb detection principle. Assume that A, B, C, D, E are part of the adjacent points collected by one beam of a 3D LiDAR; the vectors $\overrightarrow{AB}$, $\overrightarrow{BC}$, $\overrightarrow{CD}$, $\overrightarrow{DE}$ can be calculated, and B, C can be selected as the curb point since the angle between $\overrightarrow{BC}$ and the ground is very large, the same for vector $\overrightarrow{CD}$.

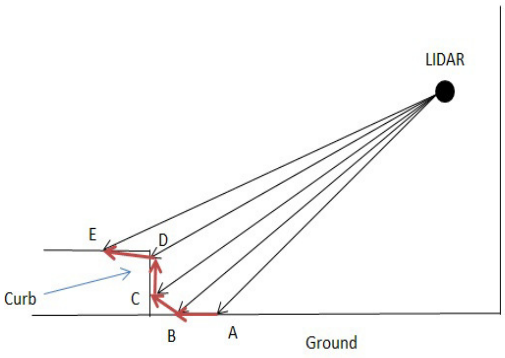

Figure 4. Points selected on the curb. A, B, C, D are 3D points obtained by LiDAR. B and C are selected as points on the curb since the vectors $\vec{BC}$ and $\vec{CD}$ are more "upcast".

3.1.2. Algorithm for Curb Detection

This subsection provides Algorithm 1 which is used for curb detection.

The following Figure 5 presents an example of the curb detection result where curb points are marked as white points.

Algorithm 1 Framework of curb detection.

Require:
Point clouds collected by a 3D LiDAR;

Ensure:
Step 1: Given input point cloud, select the area of interest;

Step 2: Calculate the vector difference of adjacent points in each beam;

Step 3: Select curb-like points and filter out noises;

Step 4: Separate higher obstacles by comparing the height with a threshold;

return Curb points.

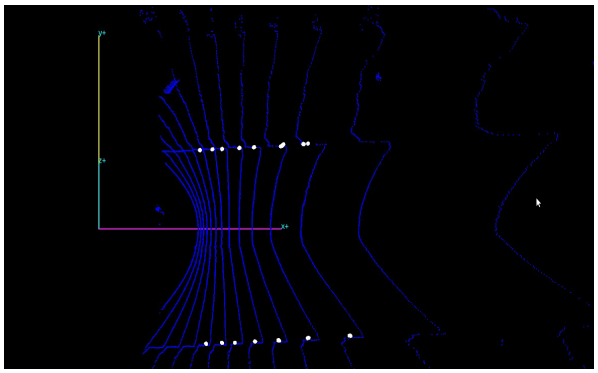

Figure 5. Output to the curb detection module.

3.2. Curb Line Fitting Using RANSAC

After detecting the curb, the next step is to fit the curb line using the RANSAC algorithm, which is an iterative method to estimate the parameters of a curve from detected curb points. The details are presented in the following Algorithm 2.

Denote S as a set of curb points that are separated from higher obstacles, then we have

Algorithm 2 Framework of curb fitting.

Require:
Detected curb points;

Ensure:
Step 1: Randomly select a sample of s curb points from S, and instantiate the model from this subset;

Step 2: Determine the set of curb points S_i that are within a distance threshold t of the model. The set S_i is the consensus set of samples and defines the inliers of S;

Step 3: If the subset of S_i is greater than some threshold T, re-estimate the model using all of the points in S_i and terminate;

Step 4: If the size of S_i is less than T, select a new subset and repeat the above;

Step 5: After N trials, the largest consensus set S_i is selected, and the model is re-estimated using all of the points in the subset S_i.

return Curb model.

In Algorithm 2, the calculation of the distance is a key point, since there are two curb cases: straight line and curve; the distance calculation is different for these two cases. Fortunately, given the curb map obtained offline and the vehicle position, the shape of the curb can be known beforehand since the curb shape does not change much within a certain area. The following are the individual methods to calculate the distance.

A. Straight line curb case: In Figure 6, the positive direction of the X coordinate denotes the heading direction of the vehicle; the positive direction of the Y coordinate denotes the left side of the vehicle; the straight line denotes the curve, and its algebraic expression is:

$$ax + by + c = 0 \tag{24}$$

assume the coordinate of one curb point is (x_i, y_i) and the distance d_i is defined as the minimum distance from (x_i, y_i) to the straight line, which can be calculated as:

$$d_i = \frac{|ax_i + by_i + c|}{\sqrt{a^2 + b^2}} \tag{25}$$

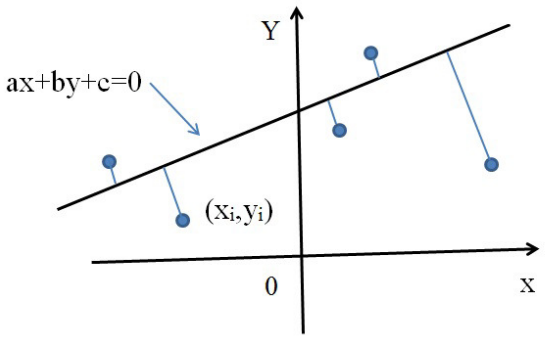

Figure 6. Straight line curb.

B. Curve line curb case: Similar to Figure 6, in Figure 7, the positive direction of the X coordinate denotes the heading direction of the vehicle; the positive direction of the Y coordinate denotes the left side of the vehicle. The curve line denotes the curb, and its algebraic expression is:

$$y = ax^2 + bx + c \tag{26}$$

In addition, each dashed line denotes the tangent line of the curve, which is perpendicular to the line joining the point of tangency and the curb point (x_i, y_i). Assume that the coordinate of one curb point is (x_i, y_i); the distance d_i is also defined as the minimum distance from (x_i, y_i) to the curve, which can be calculated as follows:

Firstly, we need to find one point on the curve that is perpendicular to the line joining the point of tangency and the curb point (x_i, y_i). Denote this point as $(x, ax^2 + bx + c)$; the slope of the tangent line across this point can be calculated as:

$$k_1 = 2a + b$$

and the slope joining this point and the curb point (x_i, y_i) is:

$$k_2 = \frac{y_i - (ax^2 + bx + c)}{x_i - x}$$

From $k_1 k_2 = -1$, we have:

$$A_i x^3 + B_i x^2 + C_i x + D_i = 0 \tag{27}$$

where:

$$\begin{aligned} A_i &= 2a^2, \\ B_i &= 3ab \\ C_i &= 2ac + b^2 + 1 - 2ay_i, \\ D_i &= bc - by_i - x_i \end{aligned}$$

The real root of (27) is:

$$x^* = -\frac{1}{3A_i}(B_i + C_i + \frac{\delta_0}{\delta_2}) \tag{28}$$

where:

$$\begin{aligned} \delta_0 &= B_i^2 - 3A_i C_i \\ \delta_1 &= 2B_i^3 - 9A_i B_i C_i + 27A_i^2 D_i \\ \delta_2 &= (0.5(\delta_1 + \sqrt{\delta_1^2 - 4\delta_0^3}))^{\frac{1}{3}} \end{aligned}$$

Then, we have:

$$y^* = a{x^*}^2 + bx^* + c \tag{29}$$

The minimum distance from curb point (x_i, y_i) to the curve is:

$$d_i = \sqrt{(x_i - x^*)^2 + (y_i - y^*)^2} \tag{30}$$

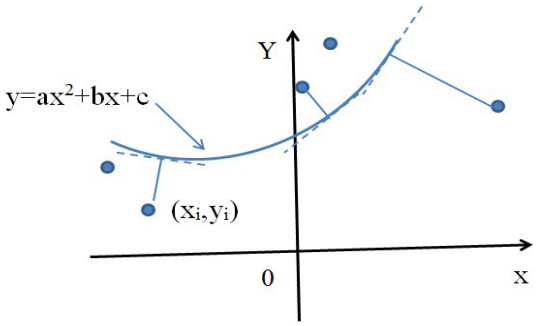

Figure 7. Curve line curb.

The following Algorithm 3 presents the pseudo-code of curb fitting.

Algorithm 3 Pseudo-code of curb fitting.

Require:
M 3D points (only the X and Y coordinates are used);

Ensure:
Step 1: Initialize parameters of the algorithm. Let $N = 1000$; $T = 0.6 \times M$;

Step 2: Repeat for N iterations:

a. Select two points randomly from the M points;

b. Compute the parameters (a, b, c) that define the line passing through those two points;

c. Count the number of inliers for the current line;

d. If the number of inliers is greater than or equal to T, terminate;

e. Keep the line having a maximum number of inliers;

Step 3: Draw the line having the maximum number of inliers.

return Line/curve parameters of curbs.

3.3. Lateral Correction Based on the Kalman Filter

3.3.1. Lateral Error Estimation

After fitting the curb into straight lines or curves (roundabout area) by using the candidate points selected from the point cloud of a 3D LiDAR, the next step is to calculate the lateral distance from the vehicle to the curb. Assume the curb is straight. The curve case is similar, as shown in Figure 8. The solid line denotes the RNDF curb. The dash-dotted line denotes the curb estimated. The origin denotes the position of the vehicle. In the lateral direction, i.e., the positive direction of the Y coordinate, there is a gap e, which is the lateral error caused by the localization error. Obviously, in order to correct this error, we need to adjust the vehicle position from the origin to the position of the rectangle on the dashed line, which means move the vehicle a distance e along the positive direction of the Y coordinate.

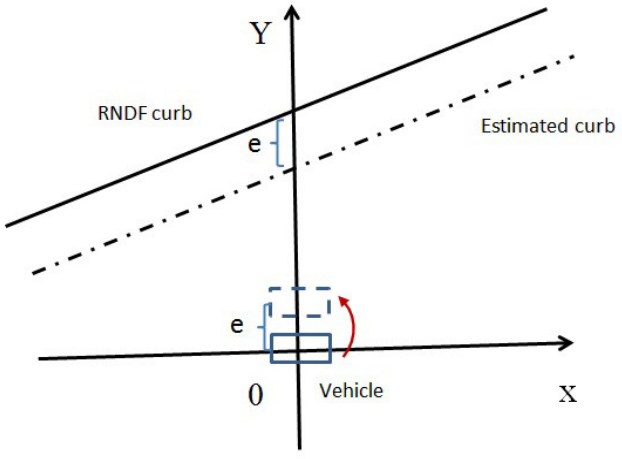

Figure 8. Lateral distances.

In order to filter the noise caused by sensor and measurement noise, etc., and smooth the lateral adjustment, a one-dimensional Kalman filter [29] is used to estimate the difference of lateral distances from the vehicle to the estimated curb and RNDF curb. Define the measurement vector y_t as the

difference e and the state vector x_t as the systematic part of this difference. Assume the change of x in time to be random. Set the system transition matrix as the identity matrix. We have that the system of equation takes the form:

$$x_t = x_{t-1} + w_t \tag{31}$$

the observation equation is given by:

$$y_t = x_t + v_t \tag{32}$$

where w_t, v_t are scalar variables of zero mean. The initial value x_0 is assumed to be zero. Then, our Kalman filter algorithm has the following form:

- State space: The space of real number $\mathbf{R}$
- State Vector: x_t
- System equation: $x_t = x_{t-1} + w_t$
- Observation: $y_t = x_t + v_t$
- Prediction equation: $x_{t/t-1} = x_t, P_{t/t-1} = P_{t-1} + W_t$
- Updating equations: $x_t = x_{t/t-1} + K_t(y_t - x_{t/t-1})$,

$$K_t = \frac{P_{t/t-1}}{P_{t/t-1}+V_t}$$
$$P_t = (1 - K_t)P_{t/t-1}$$

3.3.2. Lateral Adjustment

After we obtain the lateral difference e, which is the output of the one-dimensional Kalman filter as stated in the last subsection, next, we need to calculate the adjustment of the vehicle position in the global frame. Here we denote the adjustment as $(\Delta x, \Delta y)$.

Assume the global pose of the vehicle is (x_v, y_v, ϕ_v), and the intersection point coordinate of the estimated curb line and Y coordinate in the local coordinate frame of the vehicle is (x_l, y_l), while the global coordinate of this intersection point is denoted as (x_g, y_g), then we have:

$$\Delta x = \frac{e}{d_{RNDF}}(x_g - x_v)$$
$$\Delta y = \frac{e}{d_{RNDF}}(y_g - y_v)$$

where e is the lateral distance error, d_{RNDF} is the lateral distance from vehicle to the RNDF curb line and:

$$x_g = x_v + x_l \cos(\phi_v) - y_l \sin(\phi_v)$$
$$y_g = y_v + x_l \sin(\phi_v) + y_l \cos(\phi_v)$$

Then, $(\Delta x, \Delta y)$ are added to the vehicle global position to correct the lateral error, i.e.,

$$x'_v = x_v + \Delta x$$
$$y'_v = y_v + \Delta y$$

where (x'_v, y'_v) is the vehicle coordinate after adjustment, which corresponds to the $[p^X, p^Y]$ of the vehicle position $\mathbf{p}$ in (1).

4. Experimental Results

Result of GNSS/INS/DMI-Based Localization

In this subsection, we evaluate the performance of GNSS/IMU/DMI-based localization quantitatively using the publicly available dataset proposed in [30], where the ground truths of poses are provided to be 1.0 cm and 0.5°. Since the dataset does not include encoder data, artificial encoder data are generated from the RTK GNSS speed measurements, which are used as the ground truth. They are upsampled to get the output rate of 100 Hz. The proposed approach is tested on the experimental Campus-0L (5.563 min) in which the trajectory totally covers about 1143 m in distance with two sharp turns. In this paper, we compare the pose estimates from two UKF-based methods with the reference: one is the UKF method with the jump detection strategies, represented by 'UKF'; the other one is the UKF method without jump detection involved, represented by 'UKF w/o'. In what follows, we use 'REF' to represent the reference trajectory.

Figure 9 illustrates the estimated vehicle trajectories in the horizontal plane provided by the two methods against the reference. In the plot, the solid black line represents the reference trajectory; the dots indicate the GNSS measurements; the red dashed line shows the estimated trajectory by 'UKF'; and the blue dash-dotted line demonstrates the trajectories estimated by 'UKF w/o'. As we can see from the figure, the estimated trajectories from both methods follow the reference well, as there is no obvious jumps in this dataset.

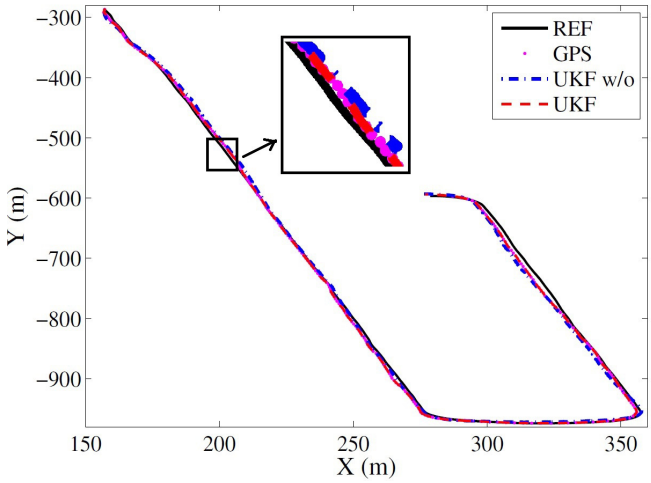

Figure 9. The estimated trajectories by the UKF-based methods with or without (w/o) the jump detection strategies against the reference.

During this experiment, no jump exists in the DGNSSmeasurements. To verify the efficiency of jump detection, some random noises were intentionally induced, which can be seen from the GNSS measurements represented by dots in Figure 10. Figure 10 illustrates the estimated trajectories in the horizontal plane against the reference with simulated GNSS jumps. The results are provided by the 'UKF w/o' and 'UKF' methods. As we can see from the figure, even with the GNSS jumps, the trajectory estimated by the 'UKF' method still follows the reference very well; however, the trajectory from the 'UKF w/o' method deviates from the reference when GNSS jumps are introduced.

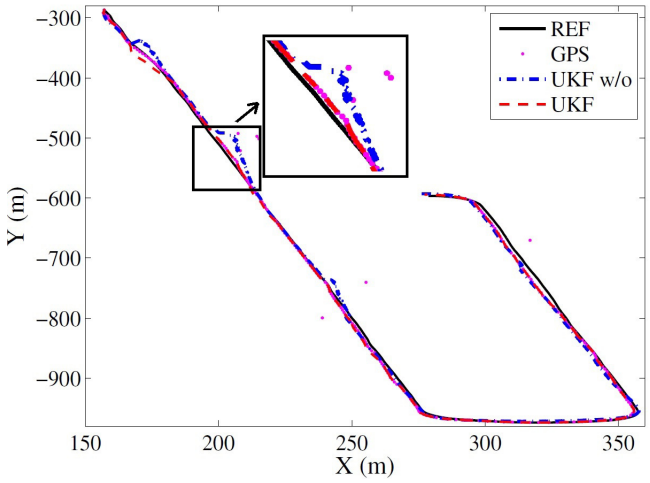

Figure 10. The estimated trajectories by the UKF-based methods with or without the jump detection strategies with random noises in the GNSS measurements against the reference.

From Figures 9 and 10, it can be seen that though the accuracy of position using UKF has been improved, lateral localization errors still exist, which may need to be corrected. To verify the lateral error correction method proposed in this paper, we apply it to the autonomous vehicle of the Institute for Infocomm Research, Agency for Science, Technology and Research (A*STAR), Singapore, which is equipped with GPS/IMU/DMI/LiDAR [31]; see Figure 11.

Figure 11. Autonomous vehicle equipped with GPS/IMU/DMI/LiDAR.

Point clouds of curb-like obstacles around the vehicle are detected by the 3D LiDAR mounted on the vehicle. After separating curbs from other obstacles and fitting them into straight lines and curve lines, lateral distances from vehicle to the curbs are calculated, then we apply the lateral correction method proposed in Section 3 to correct the lateral error of the GPS/IMU/DMI localization results. Figures 12 and 13 show the localization results of the autonomous vehicle without and with lateral error correction, respectively, where pink lines denote the curb detected using the method proposed in this paper, and the long yellow line segment in each figure is the curb line obtained from RNDF, while the short yellow line segment denotes the lateral distance between the vehicle to RNDF curbs. Figure 14 shows the lateral errors before and after lateral correction during a period of time. From Figures 12–14, it can seen that lateral localization errors have been improved greatly.

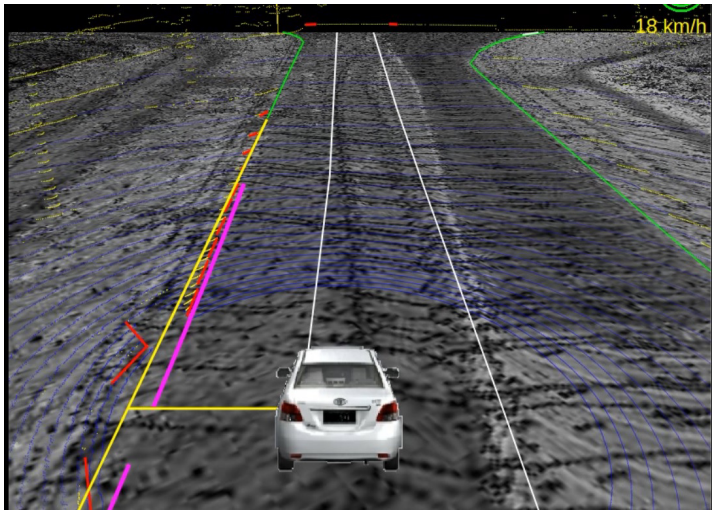

Figure 12. Localization result without lateral correction.

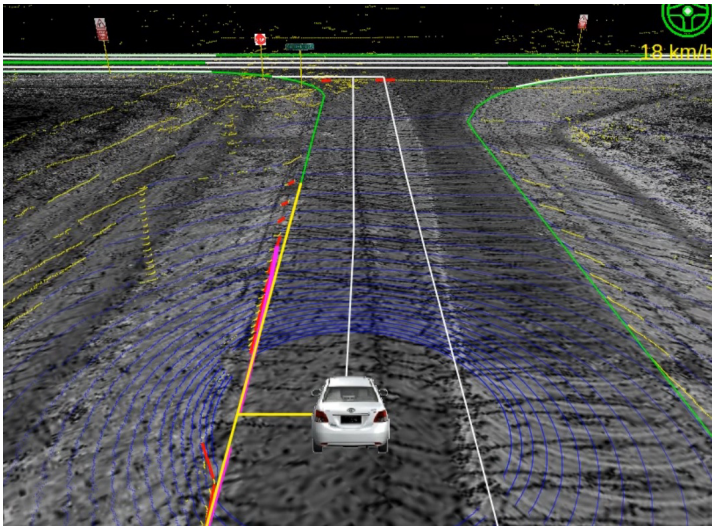

Figure 13. Localization result with lateral correction.

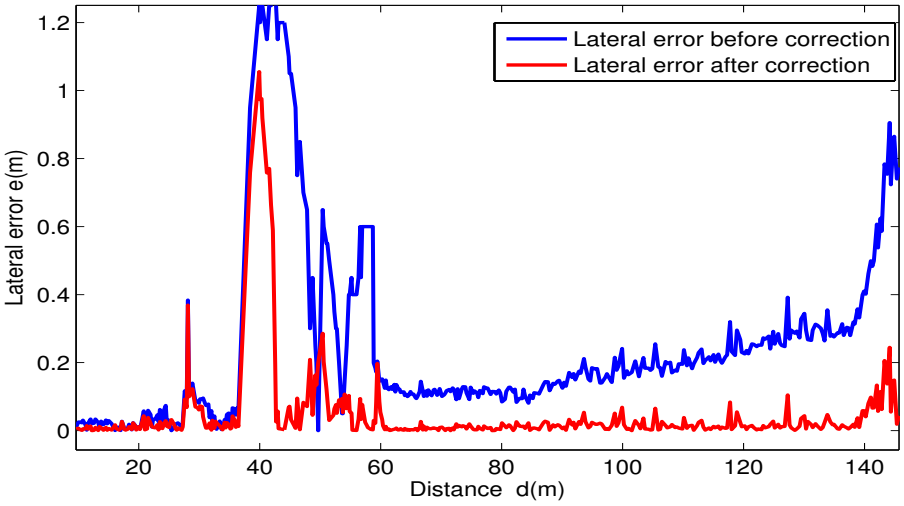

Figure 14. Lateral errors before and after lateral correction.

5. Conclusions

In this paper, to further improve the localization accuracy of an autonomous vehicle using the UKF-based GNSS/IMU/DMI fusion method, a multi-constraint fault-detection approach is proposed to handle the GNSS jumps. In addition, point cloud-based lateral correction is also proposed, where curbs are estimated in real time using a 3D LiDAR. A one-dimensional Kalman filter is adopted to estimate the lateral errors, which are used to improve lateral localization. Our method suppresses falsely-detected curb points by using the RANSAC algorithm. In future work, after the selection of point clouds for curbs, machine learning and deep learning algorithms may be used to further process the point clouds, such that curbs can be separated from other obstacles, such as vehicles and pedestrians.

Acknowledgments: This work was supported by the Funds of National Natural Science of China (Grant No. 61673098).

Author Contributions: Xiaoli Meng conceived and designed the experiments; Heng Wang wrote the paper and analyzed the data; Bingbing Liu performed the experiments.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Montemerlo, M.; Becker, J.; Bhat, S.; Dahlkamp, H.; Dolgov, D.; Ettinger, S.; Haehnel, D.; Hilden, T.; Hoffmann, G.; Huhnke, B.; et al. Junior: The Stanford entry in the urban challenge. *J. Field Robot.* **2008**, *25*, 569–597.
2. Ronnback, S. Development of an INS/GPS Navigation Loop for an UAV. Master's Thesis, Lulea Tekniska Universitet, Lulea, Sweden, 2000.
3. Zhao, Y.M. *GPS/IMU Integrated System for Land Vehicle Navigation Based on MEMS*; Royal Institute of Technology: Stockholm, Sweden, 2011.
4. Pitt, M.K.; Shephard, N. Filtering via simulation: Auxiliary particle filters. *J. Am. Stat. Assoc.* **1999**, *94*, 590–599.
5. Arulampalam, S.; Maskell, S.; Gordon, N.; Clapp, T. A tutorial on particle filter for on-line nonlinear/non-Gaussian Bayesian tracking. *IEEE Trans. Signal Process.* **2002**, *50*, 174–188.
6. Nemra, A.; Aouf, N. Robust INS/GPS sensor fusion for UAV localization using SDRE nonlinear filtering. *IEEE Sens. J.* **2010**, *10*, 789–797.
7. Vu, A.; Ramanandan, A.; Chen, A.; Farrell, J.A.; Barth, M. Real-time computer vision/DGPS-aided inertial navigation system for lane-level vehicle navigation. *IEEE Trans. Intell. Transp. Syst.* **2012**, *13*, 899–913.
8. Suhr, J.K.; Jang, J.; Min, D.; Jung, H.G. Sensor fusion-based low-cost vehicle localization system for complex urban environments. *IEEE Trans. Intell. Transp. Syst.* **2017**, *18*, 1078–1086.
9. Oh, S.M. Multisensor fusion for autonomous UAV navigation based on the Unscented Kalman Filter with Sequential Measurement Updates. In Proceedings of the 2010 IEEE Conference Multisensor Fusion and Integration for Intelligent Systems (MFI), Salt Lake City, UT, USA, 5–7 September 2010; pp. 217–222.
10. Zhou, J.C.; Yang, Y.H.; Zhang, J.Y.; Edwan, E.; Loffeld, O. Tightly-coupled INS/GPS using Quaternion-based Unscented Kalman filter. In Proceedings of the AIAA Guidance, Navigation, and Control Conference, Portland, OR, USA, 8–11 August 2011.
11. Enkhtur, M.; Cho, S.Y.; Kim, K.H. Modified Unscented Kalman Filter for a multirate INS/GPS integrated navigation system. *ETRI J.* **2013**, *35*, 943–946.
12. Tao, Z.; Bonnifait, P.; Fremont, V.; Ibanez-Guzman, J. Mapping and localization using gps, lane markings and proprioceptive sensors. In Proceedings of the IEEE International Conference on Intelligent Robots and Systems, Tokyo, Japan, 3–7 November 2013; pp. 406–412.
13. Schindler, A. Vehicle self-localization with high-precision digital maps. In Proceedings of the IEEE Intelligent Vehicle Symposium, Gold Coast, Australia, 23–26 June 2013; pp. 141–146.
14. Gruyer, D.; Belaroussi, R.; Revilloud, M. Map-aided localization with lateral perception. In Proceedings of the 2014 IEEE Intelligent Vehicles Symposium (IV), Dearborn, MI, USA, 8–11 June 2014; pp. 674–680.

15. Pandey, G.; Mcbride, J.R.; Eustice, R.M. Ford campus vision and LiDAR data set. *Int. J. Rob. Res.* **2011**, *30*, 1543–1552.
16. Zhao, G.Q.; Yuan, J.S. Curb detection and tracking using 3D-LIDAR scanner. In Proceedings of the 19th IEEE International Conference on Image Processing (ICIP), Orlando, FL, USA, 30 September–3 October 2012.
17. Huang, A.S.; Teller, S. Lane boundary and curb estimation with lateral uncertainties. In Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, St. Louis, MO, USA, 10–15 October 2009; pp. 1729–1734.
18. Hervieu, A.; Soheilian, B. Road side detection and reconstruction using LIDAR sensor. In Proceedings of the 2013 IEEE on the Intelligent Vehicles Symposium (IV), Gold Coast, Australia, 23–26 June 2013; pp. 1247–1252.
19. Yeonsik, K.; Chiwon, R.; Seung-Beum, S.; Bongsob, S. A Lidar-based decision-making method for road boundary detection using multiple Kalman filters. *IEEE Trans. Ind. Electron.* **2012**, *59*, 4360–4368.
20. Kim, Z. Robust lane detection and tracking in challenging scenarios. *IEEE Trans. Intell. Transp. Syst.* **2008**, *9*, 16–26.
21. Kodagoda, K.; Wijesoma, W.; Balasuriya, A. CuTE: Curb tracking and estimation. *IEEE Trans. Control Syst. Technol.* **2006**, *14*, 951–957.
22. Siegemund, J.; Pfeiffer, D.; Franke, U.; Forstne, W. Curb reconstruction using conditional random fields. In Proceedings of the IEEE Intelligent Vehicles Symposium, San Diego, CA, USA, 21–24 June 2010.
23. Shin, Y.; Jung, C.; Chung, W. Drivable road region detection using a single laser range finder for outdoor patrol robots. In Proceedings of the IEEE Intelligent Vehicles Symposium(IV), San Diego, CA, USA, 21–24 June 2010.
24. Meng, X.; Zhang, Z.; Sun, S.; Wu, J.; Wong, W. Biomechanical model-based displacement estimation in micro-sensor motion capture. *Meas. Sci. Technol.* **2012**, *23*, 055101.
25. Kuipers, J. *Quaternions and Rotation Sequences: A Primer with Applications to Orbits, Aerospace, and Virtual Reality*; Princeton University Press: Princeton, NJ, USA, 1999.
26. Choukroun, D.; Bar-Itzhack, I.; Oshman, Y. Novel quaternion Kalman filter. *IEEE Trans. Aerosp. Electron. Syst.* **2006**, *42*, 174–190.
27. Wan, E.A.; der Merwe, R.V. The Unscented Kalman Filter for Nonlinear Estimation. In Proceedings of the 2000 IEEE Adaptive Systems for Signal Processing, Communications, and Control Symposium (AS-SPCC'2000), Lake Louise, AB, Canada, 4 October 2000; pp. 153–158.
28. Sukkarieh, S.; Nebot, E.; Durrant-Whyte, H. A high integrity IMU/GPS navigation loop for autonomous land vehicle applications. *IEEE Trans. Robot. Autom.* **1999**, *15*, 572–578.
29. Galanis, G.; Anadranistakis, M. A one-dimensional Kalman filter for the correction of near surface temperature forecasts. *Meteorol. Appl.* **2002**, *9*, 437–441.
30. Blanco, J.; Moreno, F.; Gonzalez, J. A Collection of Outdoor Robotic Datasets with centimeter-accuracy Ground Truth. *Auton. Robots* **2009**, *27*, 174–190.
31. Wang, H.; Wang, B.; Liu, B.B.; Meng, X.L.; Yang, G.H. Pedestrian recognition and tracking using 3D LiDAR for autonomous vehicle. *Robot. Auton. Syst.* **2017**, *88*, 71–78.