2018-01-1610 Published 07 Aug 2018

A Localization System for Autonomous Driving: Global and Local Location Matching Based on Mono-SLAM

Zhijun Xu, Sihan Chen, Jie Bai, Libo Huang, and Xin Bi Tongji Univ.

Citation: Xu, Z., Chen, S., Bai, J., Huang, L. et al., "A Localization System for Autonomous Driving: Global and Local Location Matching Based on Mono-SLAM," SAE Technical Paper 2018-01-1610, 2018, doi:10.4271/2018-01-1610.

Abstract

The utilization of the SLAM (Simultaneous Localization and Mapping) technique was extended from the robotics to the autonomous vehicles for achieving the positioning. However, SLAM cannot obtain the global position of the vehicle but a relative one to the start. For sake of this, a fast and accurate system was proposed to obtain both the local position and the global position of vehicles based on mono-SLAM which realized the SLAM by using monocular camera with a lower cost and power consumption. Firstly, the rough latitude and longitude of current position was obtained by using common GPS without differential signal. Then, the Mono-SLAM operated on the consecutive video frames to generate the localization and local trajectory and its accuracy was further improved by utilizing the IMU information. After that, a piece of Map centered in the rough position obtained by common GPS was downloaded from the Open Street Map. Finally, a searching process in the downloaded Map was executed by using chamfer matching algorithm to find a piece of path matched with the constructed trajectory. Consequently, the global position of the vehicle was obtained and the accumulated error can be decreased with cyclical searching. In the test, the performance of this proposed system outperforms current approaches in global location and its error was less than 5 meters, indicating in parallel the potential that mono-SLAM can bring to the global localization task.

Introduction

The development of autonomous vehicles has become a very active research area and many systems have been created towards the goal of self-driving cars over the last few years [1, 2]. Localization is an important component regarding the stability of such systems and thus its precision and robustness is of high importance.

In recent years, SLAM, proposed by Smith R, Self M and Cheeseman P [3] in 1988, is gradually being used in autonomous vehicles. SLAM can be divided into Lidar SLAM and Visual SLAM (VSLAM) based on cameras. Cameras have many advantages, such as, cost less money, consume less power, and provide environmental information more accurate, compared to laser rangefinders, inertial navigation and GPS (Global Positioning System). For the monocular-based SLAM, many scholars conducted in-depth research. MonoSLAM [4] was the first successful monocular SLAM camera system developed by Davison et al. in 2007. In the same year, Davison's masters Murray and Klein published PTAM system (Parallel Tracking and Mapping) [5]. In the architecture of PTAM, the tracking and mapping are separated into two threads in parallel. This design was followed by a later real-time SLAM such as ORB-SLAM and became standard on modern SLAM systems. Newcombe et al. proposed a monocular DTAM system [6] in 2011. In 2013, Engel et al. proposed a visual odometry (VO) system that was also based on the direct method, which was expanded in 2014 to the visual SLAM system LSD-SLAM [7]. In 2016, Engel et al. proposed the DSO system [8]. This system is a new vision odometer based on direct method and sparse method. It combines the minimized photometric error model and the joint optimization of model parameters to achieve high tracking accuracy and robustness.

As monocular SLAM, it can't satisfy both computational speed and computational accuracy, some scholars begin to consider improving the accuracy of monocular visual SLAM by merging IMU information. Inertial Measurement Unit (IMU) is a device that measures the three-axis attitude (or angular velocity) and acceleration of an object, including accelerometer and speed sensor (gyroscope). In 2015, Mur-Artal et al. proposed an open source monocular ORB-SLAM [9], which is currently one of the most effective visual SLAM systems. Then it was expanded to form Visual Inertial ORB-SLAM [10] that integrates IMU information, which applies the pre-integration method proposed by Foster's paper [11]. In 2016, Forster et al. improved SVO to form SVO2.0 [12], which was greatly improved with adding the edge tracking and considering the IMU's prior information of motion. In 2017, the scholar Shen Hongtao et al. of Hong Kong University of Science and Technology proposed the VINS system that integrates IMU and visual information [13].

With the continuous development of SLAM technology, researchers began to consider whether it is possible to combine SLAM precise local positioning technology with global positioning technology to obtain more accurate vehicle global positioning information. In 2013, Georgios Floros et al. [14] proposed combining the visual odometer with the Open Street Map. Particle filtering and Monte Carlo are used to estimate the vehicle pose accurately. Then the vehicle trajectory is matched with the road in the Open Street Map to obtain more accurate vehicle global positioning information.

Based on the researches above, this paper focuses on providing the solution of navigation and positioning in autonomous vehicles of level 4 and above. The main task is to design and construct the positioning system of the autonomous vehicles combining the OSM (Open Street Map) to achieve precise positioning of vehicles in complex road conditions which is utilizing the SLAM technology based on monocular and inertial element (IMU).

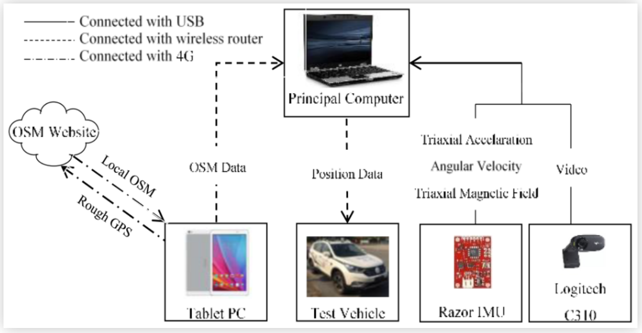
System Platform

The hardware of the experimental verification platform built in this paper is shown in Figure 1, including the IMU module (nine-axis inertial sensor), monocular camera (logitech C310 webcam), tablet PC (Huawei) and principal computer (HP).

Laptop computer, which CPU is duo @2.5GHz, is the processing unit of the whole system. It accepts the environmental information detected by the sensor unit and carries out analysis and processing results. The Logitech C310 monocular camera and the nine-axis inertial sensor razor IMU are the system's sensing units. The Logitech C310 monocular camera delivers up to 30 frames of 1280*720 pixels per second. The razor IMU contains a triaxial gyroscope (ITG3200), a triaxial accelerometer (ADXL345) and a triaxial magnetometer (HMC5883L) that detect real-time information of the angular velocity, acceleration and magnetic field under three-dimensional coordinates. The tablet PC, which processor version is MSM8953 @2.0GHz, is used to provide rough GPS signals and interact with principal computer through a wireless router. And the vehicle also interacts with principal computer wirelessly to receive location information.

The structure of this platform system is shown below.

FIGURE 1 The schematic of system structure. The principal computer, tablet PC, razor IMU and Logitech C310 are all mounted on the test vehicle



SLAM Based on IMU and Monocular

This section mainly introduces the integration process of data from IMU and monocular camera. The conversion relations among various coordinate systems is presented at first. Then the method of obtaining vehicle pose through IMU data and monocular SLAM is discussed. The fused vehicle pose data and vehicle trajectory are obtained finally after the fusion of data from IMU and monocular camera.

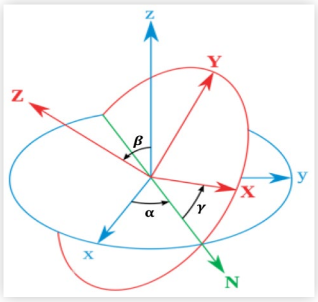
Coordinate Conversion

The coordinates involved in this system are the global coordinate system, the vehicle coordinate system, the monocular camera coordinate system, and the IMU coordinate system. In this paper, the center of the vehicle at the initial position is set as the origin of the global coordinate system and the vehicle's vertical axis is set as the x-axis of the global coordinate, the horizontal axis is set as the y-axis. The data obtained from each sensor is based on its own coordinate system, so the primary task of merging multiple sensor data is transformation of the coordinate system.

All coordinate systems are uniformly converted into a global coordinate system. In the quiescent state, the Euler angles (α,β,γ), shown in Figure 2, between the vehicle coordinate system (or the camera coordinate system) and the global coordinate system are measured from VSLAM. The Euler angles of the IMU can be calculated from the sensor. The transformation matrix C_{xyz} between the coordinate systems is as follows:

$$C_{xyz} = C(\gamma)\cdot C(\beta)\cdot C(\alpha) = \begin{bmatrix} cos(\gamma) & sin(\gamma) & 0 \\ -sin(\gamma) & cos(\gamma) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\beta) & sin(\beta) \\ 0 & -sin(\beta) & cos(\beta) \end{bmatrix} \begin{bmatrix} cos(\alpha) & sin(\alpha) & 0 \\ -sin(\alpha) & cos(\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (1)$$

FIGURE 2 The Euler angles between the vehicle coordinate system (in red) and the global coordinate system (in blue)



IMU Pose Estimation

Gesture Estimation Gesture estimation in IMU is based on the angular velocity, which is measured by the triaxial gyroscope. Calculate the static quaternion by the factored quaternion algorithm (FQA) and estimate the dynamic quaternion by the differential relation between the angular velocity and the quaternion, and merge the static quaternion with dynamic quaternion together by Kalman Filter, and then achieve pose estimation.

Factored quaternion algorithm [15, 16] is a static and quasi-static attitude estimation algorithm, which only needs single-step calculation. When the IMU is static, the acceleration $a = [0 \ \ 0 \ \ g]T$ is measured. If the IMU is rotated about the y-axis by angle θ, the x-axis by an angle of φ, and the heading angle is ψ, the quaternions that characterizes these angles are:

$$q_\theta = cos\left(\frac{\theta}{2}\right)[1000] + sin\left(\frac{\theta}{2}\right)[0010] \tag{2}$$

$$q_\varphi = cos\left(\frac{\varphi}{2}\right)[1000] + sin\left(\frac{\varphi}{2}\right)[0100] \tag{3}$$

$$q_\psi = cos\left(\frac{\psi}{2}\right)[1000] + sin\left(\frac{\psi}{2}\right)[0001] \tag{4}$$

The above quaternion components are combined to obtain the quaternion:

$$q_m = q_\psi \otimes q_\theta \otimes q_\varphi \tag{5}$$

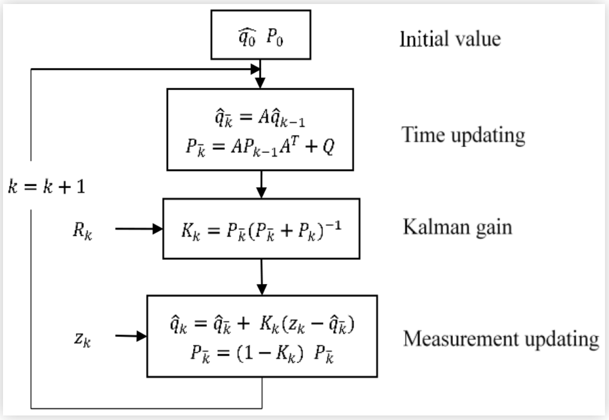
In general, there must be filter processing due to the gyro drift and measurement noise. Common filter includes Complementary Filter (CF), Kalman Filter (KF), Extended Kalman Filter (EKF), etc. This paper uses Kalman Filter in gesture estimation of IMU, since it has already met the accuracy requirement. The quaternion is set as state vector$x = q$ and the measurement variable is provided by the decomposition quaternion algorithm $z = q_m$, then the discrete Kalman Filter equation is:

$$\begin{cases} q_k = A_{k-1}q_{k-1} + w_{k-1}, p(w) \sim N(0,Q) \\ z_k = H_{k-1}q_k + v_k, \quad\ p(v) \sim N(0,R) \end{cases} \tag{6}$$

Where w is the process noise, v is the observed noise, and Q and R are the covariance matrix of A and H, respectively. Through the differential relationship between angular velocity and quaternion, A is given by:

$$A = I + \frac{1}{2}R(\omega_b)\text{h} \tag{7}$$

FIGURE 3 The Flow Chart of Quaternion Based Kalman Filter Algorithm


$\widehat{q_0}$ P_0
Initial value
$\hat{q}_{\bar{k}} = A\hat{q}_{k-1}$
$P_{\bar{k}} = AP_{k-1}A^T + Q$
Time updating
$k = k + 1$
R_k
$K_k = P_{\bar{k}}(P_{\bar{k}} + P_k)^{-1}$
Kalman gain
z_k
$\hat{q}_k = \hat{q}_{\bar{k}} + K_k(z_k - \hat{q}_{\bar{k}})$
$P_{\bar{k}} = (1 - K_k)\ P_{\bar{k}}$
Measurement updating
© SAE International

Where, h is sampling update interval, $R(\omega_b)$ is:

$$R(\omega_b) = \begin{bmatrix} 0 & -\omega_b \\ \omega_b & \omega_\times \end{bmatrix}, \omega_\times = \begin{bmatrix} 0 & \omega_{bz} & -\omega_{by} \\ \omega_{bz} & 0 & \omega_{bx} \\ \omega_{by} & -\omega_{bx} & 0 \end{bmatrix} \tag{8}$$

So the flow chart of quaternion-based Kalman Filter algorithm is shown in Figure 3.

Displacement Estimation The accelerometer is used to construct the measurement model. The state vector is $x = (x_{car}, y_{car}, v_x, v_y, a_{nx}, a_{ny})$. The displacement, speed and acceleration of z-axis are not involved in the calculation because the movement of the vehicle is generally on the oxy plane. a_{nx}, a_{ny} are the acceleration offset, including the component and offset of the gravity's acceleration. The decomposition for the measured value of the acceleration is:

$$a^g = a^I \cdot C_{Ig} - a_n - w_a \tag{9}$$

Where ag is the acceleration component of the global coordinate system, aI is the measured value of the accelerometer, C_{Ig} is the conversion matrix from the IMU coordinate system to the global coordinate system, which is affected by the pose solution of the IMU. w_a is noise. And the observation model constructed in this paper is:

$$x_k = f_k(x_{k-1}, u_k) + w_k = f_k x_{k-1} + b_k u_k + w_k \tag{10}$$

Where, $f_k = \begin{bmatrix} 1 & 0 & \Delta t_k & 0 & 0 & 0 \\ 0 & 1 & 0 & \Delta t_k & 0 & 0 \\ 0 & 0 & 1 & 0 & -\Delta t_k & 0 \\ 0 & 0 & 0 & 0 & 0 & -\Delta t_k \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, b_k = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \Delta t_k & 0 \\ 0 & \Delta t_k \\ 0 & 0 \\ 0 & 0 \end{bmatrix},$

$u_k = (a_x\ a_y)^T$, w_k satisfies the Gaussian white noise model with zero mean.

In practice, when the IMU moves, the acceleration offset (a_{nx}, a_{ny}) is not constant but can be regarded as a constant in short-term calculation.

Monocular Camera Pose Estimation

This paper uses ORB-SLAM algorithm to be the VSLAM prototype. It applies monocular camera positioning and three-dimensional composition contains the following steps:

(1) Initialization: find the initial correspondence between the two ORB feature points $\{R_{ini}\}$.
(2) Pose Estimation: The constant speed motion model is a conventional method for generating a priori pose estimation. ORB-SLAM generates a priori pose estimationq_{mono} using a constant velocity motion model and uses the prior estimate to track map points that match the previous frame.
(3) Map generation: The ORB-SLAM map generation module runs key frame extraction, map point triangulation, key frame and map point maintenance.
(4) Closed-loop detection: ORB-SLAM completes global position recognition and closed-loop detection through "Word Bag Model" [17].

Through the above steps, the best pose estimate and trajectory are obtained. The key point of pose estimation by monocular camera is finding the transformation (R, T) such that the cost function $f(R, T)$ has the minimum value

$$f(R,T)=\sum_i \left\|Rp_i+T-q_i\right\|^2 \tag{11}$$

In the above formula, p_i is the matching point measured before the camera moves, q_i is the corresponding matching point measured after the camera moves, R is the rotation matrix, and T is the translation vector.

The current pose could be obtained by transform calculation:

$$(x_k, y_k, z_k)^T=(x_{k-1}, y_{k-1}, z_{k-1})^T+T_k$$

$$(\theta_k, \phi_k, \psi_k)=R_k(\theta_{k-1}, \phi_{k-1}, \psi_{k-1})$$

$$v_k=T_k/\Delta t_k \tag{12}$$

The observation model could be obtained based on the state vectors:

$$z_k=h_k x_k+v_k \tag{13}$$

Where, $h_k = I$, v_k is noise, satisfying the Gaussian white noise distribution, $v_k \sim N(0, R_v(vo_{trust}))$. The value of noise covariance$(R_v(vo_{trust}))$ [18] depends on the credibility of the data obtained by the camera.

Algorithm Framework Based on IMU-Monocular

It is theoretically feasible to obtain the location information through dead reckoning by IMU, removing the gravity component from the acceleration value, and obtaining the displacement by second integral. However, the error due to noise and jitter makes the remove of the acceleration component can't be accurate and there will be accumulated error. It is a common practice to combine inertial navigation systems with other systems to improve positioning accuracy.

The algorithm used in this paper takes the IMU prediction model into account after the pose estimation in VSLAM, and combines the pose to obtain the positioning result. Algorithm framework is shown in Figure 4.

Pose fusion consists of two parts: displacement and posture. The fusion of displacement is: EKF fusion according to estimated model equation (10) and observation equation (13). The fusion of posture is: when the difference between the two measurements is less than the threshold, the result would be obtained by weighted average; and when the difference between the two measurements is greater than the threshold, the extended Kalman Filter would be applied. The map construction would be achieved and the driving trajectory would be draw after the integration of pose data. The results are shown below:

It can be seen from Figure 5 that the trajectory obtained by merging IMU data is not very smooth, but its error was much smaller. And the angle of yaw changing is more accurate in IMU-monocular SLAM.

FIGURE 4 Algorithm Framework Based on IMU - monocular Vision

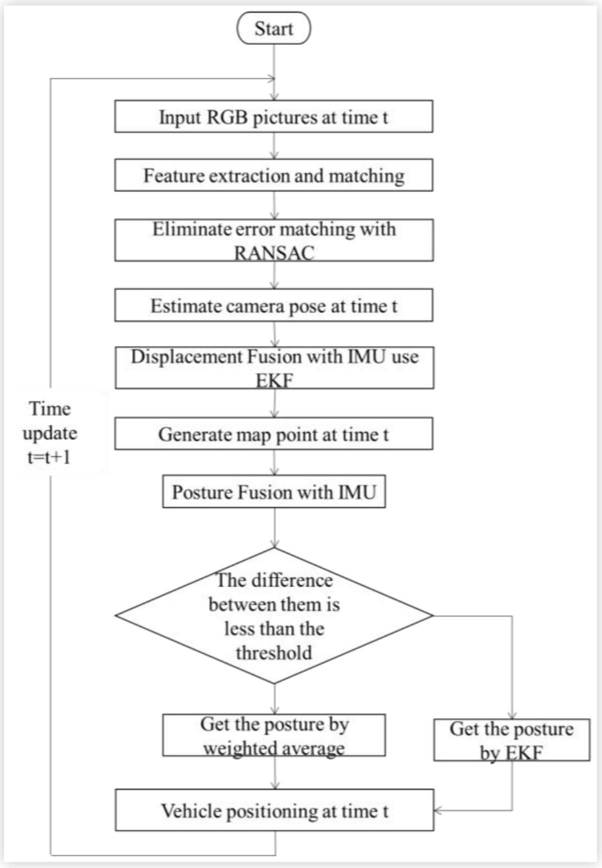
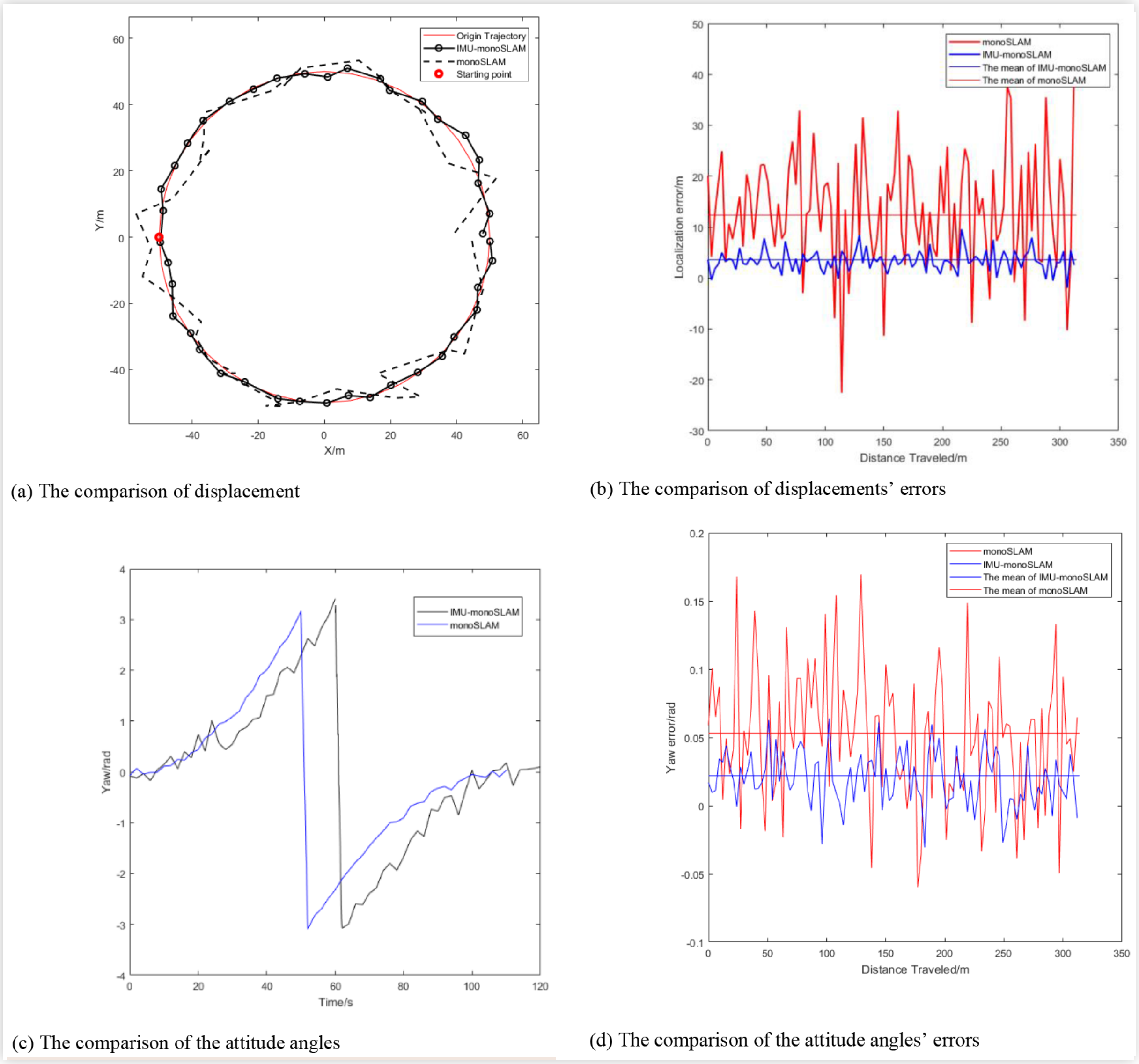

Start
Input RGB pictures at time t
Feature extraction and matching
Eliminate error matching with RANSAC
Estimate camera pose at time t
Displacement Fusion with IMU use EKF
Time update t=t+1
Generate map point at time t
Posture Fusion with IMU
The difference between them is less than the threshold
Get the posture by weighted average
Get the posture by EKF
Vehicle positioning at time t

FIGURE 5 The comparison of vehicle positioning results between mono-SLAM and IMU-monocular SLAM when the vehicle travels along a curve drawn from the ground in a test site



(a) The comparison of displacement

(b) The comparison of displacements' errors

(c) The comparison of the attitude angles

(d) The comparison of the attitude angles' errors

Matching with Open Street Map Data

Open Street Map Data

Open Street Map (OSM) is a collaborative project to create a free editable map of the world created by Steve Coast in the UK in 2004. Since then, its registered users have grown to over 2 million. It can collect data through manual survey, GPS devices, aerial photography, and other free sources. This crowdsourced data would be available under the Open Database Licence [19].

The data of Open Street Map (OSM) could be obtained through the Internet and users could download the specific area of the map. The size of the map area could be presented by $D = (lat_{min}, lon_{min}, lat_{max}, lon_{max},)$. The feedback map is an XML formatted topology data structure file that contains three core elements: nodes, ways and relations [20]. Nodes n are points with a geographic position, stored as coordinates: $n = (lat, lon)$. Ways w are ordered lists of nodes, representing a polyline, or possibly a polygon if they form a closed loop: $w = \{n_i\}_{i = 1\ldots k}$. The relations r are used for representing the relationship of existing nodes and ways.

In the OSM data format, the information of roads and buildings that are required for vehicle positioning is usually stored in the form of way. Therefore, in order to simplify subsequent data processing, the ways in the XML format map are extracted for matching of the maps.

Path Similarity: Fréchet Distance

Fréchet distance is a description of the path spatial similarity proposed by the French mathematician Maurice René Fréchet in 1906. This description also considers the factors of the spatial distance of the path [21]. In other words, to obtain the most similar path to the target path is to find the path whose maximum distance with the target path is the smallest of all paths. The Fréchet metric takes into account the flow of the two curves because the pairs of points whose distance contributes to the Fréchet distance sweep continuously along their respective curves. This makes the Fréchet distance a better measure of similarity for curves than alternatives, such as the Hausdorff distance, for arbitrary point sets.

Let S be a metric space, A and B be two continuous curves on S. That is, $A : [0, 1] \to S$, $B : [0, 1] \to S$. η, μ is two re-parameterization functions for the unit interval. That is, $\eta : [0, 1] \to S$, $\mu : [0, 1] \to S$. The Fréchet distance $F(A, B)$ of the curves A and B is then defined as

$$F(A,B) = \inf_{\eta,\mu} \max_{t\in[0,1]} \left\{ d\left(A(\eta(t)), B(\mu(t))\right) \right\} \tag{14}$$

Where, d is the measure function on S.

This method is first used to filter out the path that is most similar to the path to be matched and prepare for the next match conversion operation.

Localization by Chamfer Matching with OSM

Chamfer Matching (CM) is a more commonly used method to match two maps. In Chamfer Matching, $Q = \{q_i\}$ is the map to be matched, $T = \{t_j\}$ is the template map, and the conversion relation between two maps is $E = (\theta, t_x, t_y)$, $E \in SE(2)$. In order to obtain the final results, the above problem can be expressed by the following optimization formula:

$$\hat{E} = \arg\min_{E\in SE(2)} d_{CM}\left(W(Q; E), T\right) \tag{15}$$

Where,

$$W(Q; E) = E \cdot Q$$

$$d_{CM}(Q, T) = \frac{1}{|Q|} \sum_{q_i\in Q} \min_{t_j\in T} \left| q_i - t_j \right| \tag{16}$$

CM algorithm has two advantages: Firstly, you can find the best alignment conversion matrix, the second is to quickly calculate the results.

In the process of map matching, the local Open Street Map data is demanded firstly. In order to ensure that the map contains the initial location of the vehicle, a rough GPS data of the vehicle location should be uploaded. With the rough GPS data as the center, the latitude and longitude informatio n(lat_{min}, lon_{min}, lat_{max}, lon_{max},) of the rectangular area D could be calculated and uploaded to the Open Street Map website. The partial map could be downloaded and is shown in Figure 6(a). The road map could be stripped through a simple algorithm and is shown in Figure 6(b).

When the VSLAM is running, there will be a cumulative drift in a short distance although the IMU data fusion has been amended. After the vehicle has traveled a specific distance of k meters and accumulated the minimum turning angle θ_{min}, an initial trajectory path is created that is used to match the processed OSM map. Use CM to make a detailed match for all possible position and direction, and then calculate a set of possible matching results. As shown in Figure 7,

FIGURE 6 Partial OSM map and processed road map. (a) The partial map downloaded from the OSM, (b) The road map after stripped

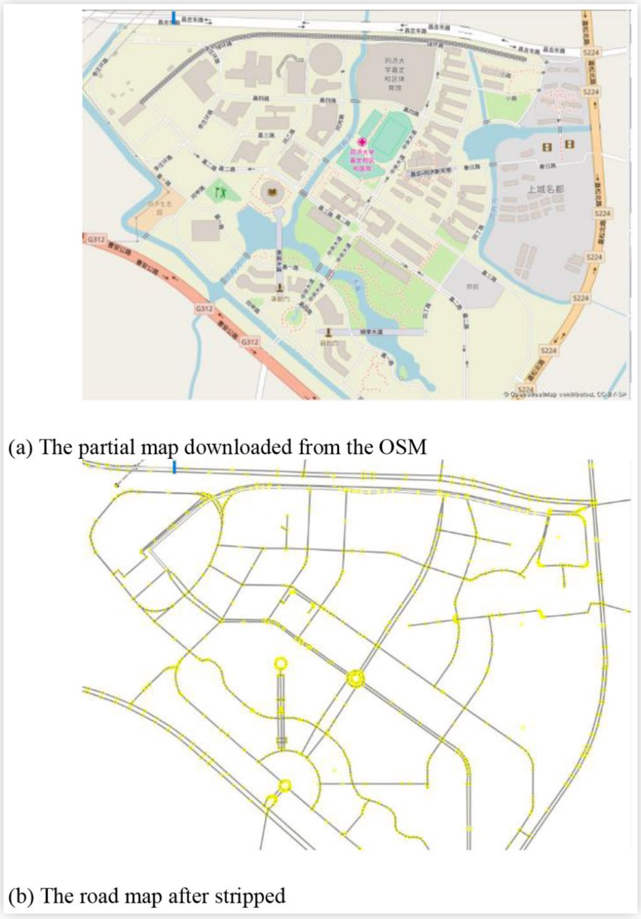
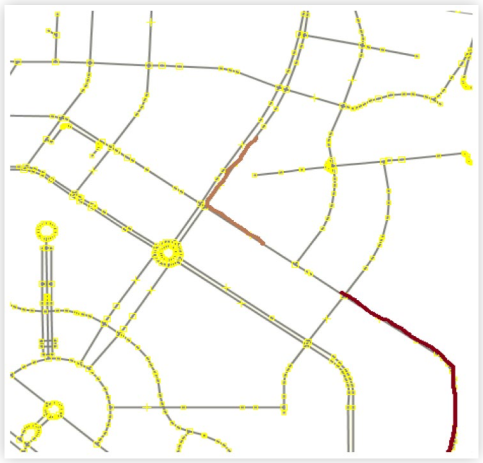

(a) The partial map downloaded from the OSM
(b) The road map after stripped
© SAE International

FIGURE 7 Preliminary map matching results



since the results produced by the VSLAM and the matching result are both in error, a variety of optimization is demanded in order to improve the accuracy of positioning as much as possible. The detail of optimization is discussed in the next section.

Localization Correction

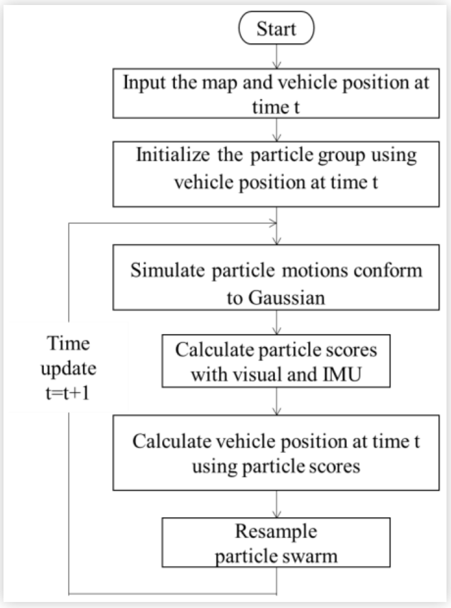
Particle Filter Positioning

Particle Filter Positioning is also known as Monte Carlo Localization (MCL) [22]. This algorithm is used to accurately determine robots' location through sensor information when they know the map information. The essence of the algorithm is to approximately represent the posterior probability density of any state by a finite set of weighted random samples (particles). When a robot can't run in a completely predictable way, it will produce a lot of random guesses that are called particles. Each particle contains a complete description of the state of the future. When the robot observes the environment, it discards particles that are not consistent with the observations and, eventually, the particles should converge to the robot's actual position. Therefore, Monte Carlo positioning algorithm is divided into four steps, as shown in Figure 8.

When Monte Carlo localization algorithm is used to estimate vehicle pose, we first select a set of random samples $X_t = \left\{x_t^{[1]}, x_t^{[2]}, \ldots, x_t^{[M]}\right\}$ containing M particles, initializing the MCL module through the map transformation relationship obtained from Chamfer matching and distributing particles based on assumed weights.

The final attitude of the vehicle is composed of latitude and longitude and heading angle of the vehicle: $\Sigma = (x_{car}, y_{car}, \psi)$. The standard odometer model is used as the motion model

FIGURE 8 Monte Carlo positioning algorithm


Start
Input the map and vehicle position at time t
Initialize the particle group using vehicle position at time t
Simulate particle motions conform to Gaussian
Calculate particle scores with visual and IMU
Calculate vehicle position at time t using particle scores
Resample particle swarm
Time update t=t+1

and the Chamfer matching cost index model is used as the observation model:

$$w^{[m]} = \lambda\, exp^{-\lambda \cdot d_{CM}^{[m]}}, m \in 1 \ldots M \tag{17}$$

The value of λ is determined experimentally.

When estimating the attitude of the vehicle Σ_t, the current particle is first used as a starting point, and then each particle is evaluated under the observation model. The particle is assigned a weight, and finally the sample is re-sampled according to the weight. The final attitude Σ_t at time t could be obtained through assigning weight to the particles. When estimating the vehicle's attitude at the next moment, the MCL begins to iterate and sample each particle for a new position based on the local mileage estimate and the current particle position, continuing to evaluate each of the sampled particles in the observed model, assigning weights, resampling, and obtaining the final position at $t + 1$ moment.

The Monte Carlo localization algorithm is used to correct the pose of the vehicle to reduce the cumulative error caused by VSLAM and the matching error generated by the matching process.

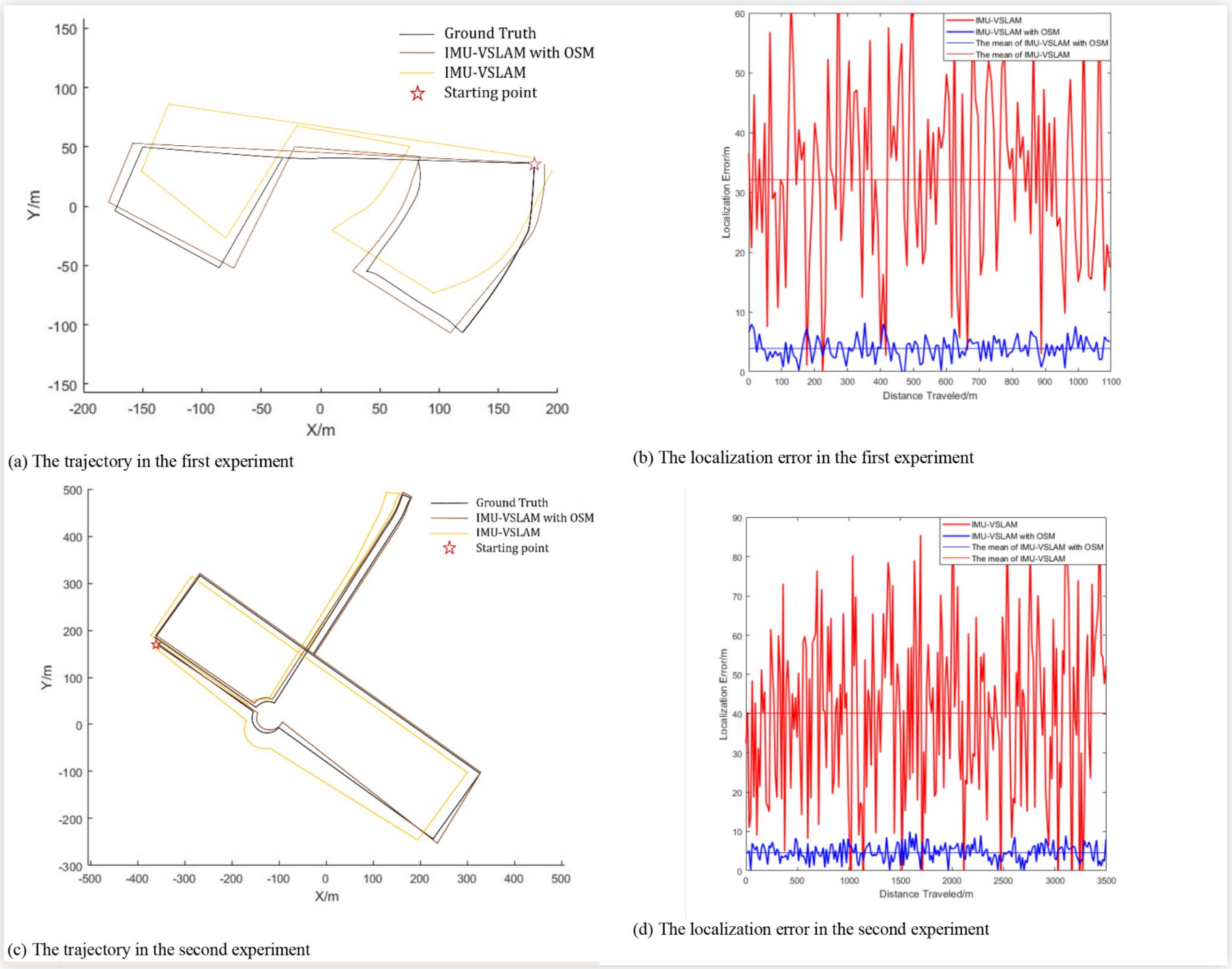
Map Data Update

Due to the drift error of VSLAM, the method of updating map data is proposed to reduce the error. After a certain interval Δt, the system will re-set the current position as the initial point and send the rough GPS signal to the OSM website to download the new local map *D1*, and then match again.

Experimental Results

In order to verify whether the vehicle positioning accuracy of the integrated OSM information has been improved, two experiments were conducted. The first experiment was short in distance, with simpler path. While the second path was longer, and the complexity increased. During the experiments, the vehicle equipped with IMU and monocular camera was driven at a lower speed, and the mono-SLAM incorporating IMU data tracked the vehicle trajectory. In Figure 9(a) and (c), the errors of IMU-VSLAM are larger, and the maximum error is about 90 m, far away from the actual trajectory. While the result of IMU-VSLAM with OSM data is more accurate, the map information compensates for the possible drift of the IMU-VSLAM algorithm and therefore significantly reduces the error. In the first experiment, the average error of IMU-VSLAM trajectory is about 32.13 m, and the average error is about 3.89 m with OSM information. In the second experiment, the average value of IMU-VSLAM error increased to 40.15 m, and it is about 4.58 m after integration OSM information. Therefore, the integration of map information not only obtains the global positioning information, but also improves the positioning accuracy.

FIGURE 9 The experiment results



Summary

In this paper, the conversion is achieved from vehicle's local positioning information to global by integration of OSM information, with the positioning accuracy being greatly improved. Although the error range is still large, this integration of embedded positioning is a new development in the direction of vehicle positioning. Optimization and tracking is not perfect in the currently proposed method for integration of embedded positioning. Therefore, more work is needed in optimizing the algorithm to reduce the positioning error.

References

1. Thrun, S., "Toward Robotic Cars," *Communications of the ACM* 53(4):99-106, 2010.
2. Levinson, J., Askeland, J., Becker, J., Dolson, J., Held, D., Kammel, S., Kolter, J., Langer, D., Pink, O., Pratt, Y., et al., "Towards Fully Autonomous Driving: Systems and Algorithms," in *IEEE Intelligent Vehicles Symposium*, 2011.
3. Smith, R., Self, M., and Cheeseman, P., "Estimating Uncertain Spatial Relationships in Robotics," *Machine Intelligence & Pattern Recognition* 5(5):435-461, 1988.
4. Davison, A.J., Reid, I.D., Molton, N.D. et al., "MonoSLAM: Real-Time Single Camera SLAM," *IEEE Transactions on Pattern Analysis & Machine Intelligence* 29(6):1052, 2007.
5. Klein, G. and Murray, D., "Parallel Tracking and Mapping for Small AR Workspaces," *IEEE and ACM International Symposium on Mixed and Augmented Reality*, 2008, 1-10. IEEE.
6. Newcombe, R.A., Lovegrove, S.J., and Davison, A.J., "DTAM: Dense Tracking and Mapping in Real-Time," *International Conference on Computer Vision*, 2011, 2320-2327. IEEE Computer Society.
7. Engel, J., Schöps, T., and Cremers, D., "LSD-SLAM: Large-Scale Direct Monocular SLAM," *European Conference on Computer Vision*, Vol. 8690, 2014, 834-849.
8. Engel, J., Koltun, V., and Cremers, D., "Direct Sparse Odometry," *IEEE Transactions on Pattern Analysis & Machine Intelligence* 99:1-1, 2017.

9. Mur-Artal, R., Montiel, J.M.M., and Tardós, J.D., "ORB-SLAM: A Versatile and Accurate Monocular SLAM System," *IEEE Transactions on Robotics* 31(5):1147-1163, 2017.
10. Mur-Artal, R. and Tardós, J.D., "Visual-Inertial Monocular SLAM with Map Reuse," *IEEE Robotics & Automation Letters* 2(2):796-803, 2017.
11. Forster, C., Carlone, L., Dellaert, F. et al. "IMU Preintegration on Manifold for Efficient Visual-Inertial Maximum-A-Posteriori Estimation," Georgia Institute of Technology, 2015.
12. Forster, C., Zhang, Z., Gassner, M. et al., "SVO: Semidirect Visual Odometry for Monocular and Multicamera Systems," *IEEE Transactions on Robotics* 33(2):249-265, 2017.
13. Li, P., Qin, T., Hu, B et al., "Monocular Visual-Inertial State Estimation for Mobile Augmented Reality," *2017 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, 2017, 11-21. *IEEE*.
14. Georgios, F., Benito Van Der, Z., and Bastian, L., "OpenStreetSLAM: Global Vehicle Localization Using Open Street Maps," *IEEE International Conference on Robotics and Automation*, 2013, 1054-1059. *IEEE*.
15. Conrado, A., "Implementation of a Quaternion-Based Kalman Filter for Human Body Motion Tracking Using MARG Sensors."
16. Yun, X.P., Bachmann, E.R., Mcghee, R.B., "A Simplified Quaternion-Based Algorithm for Orientation Estimation from Earth Gravity and Magnetic Field Measurements."
17. Galvez-López, D. and Tardos, J.D., "Bags of Binary Words for Fast Place Recognition in Image Sequences," *IEEE Trans on Robotics* 28(5):1188-1197, 2012.
18. Hervier T, Bonnabel S, Goulette F. "Accurate 3D Maps from Depth Images and Motion Sensors via Nonlinear Kalman Filtering," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, 5291-5297. IEEE.
19. https://en.wikipedia.org/wiki/OpenStreetMap.
20. Haklay, M. and Weber, P., "OpenStreetMap: User-Generated Street Maps," *IEEE Pervasive Computing* 7(4):12-18, 2008.
21. Fréchet, M. and Maurice, "Sur Quelques Points du Calcul Fonctionnel," *Rendiconti del Circolo Matematico di Palermo (1884-1940)* 22.1:1-72, 1906.
22. Ioannis, R., "A Particle Filter Tutorial for Mobile Robot Localization," *Centre for Intelligent Machines* 2(2):481-488, 2003.

Contact Information

Zhijun Xu
Tongji University
No.4800 Cao'An Road, Jiading District
Shanghai, Republic of China
xzj1623@tongji.edu.cn

Acknowledgments

National Key R&D Program of China 2016YFB0101101.

Abbreviations

IMU - Inertial Measurement Unit is a device that measures the three-axis attitude (or angular velocity) and acceleration of an object

OSM - Open Street Map is a collaborative project to create a free editable map of the world

FQA - Factored quaternion algorithm is a static and quasi-static attitude estimation algorithm which only needs single-step calculation

EKF - Extended Kalman Filter is an efficient recursive filter

MCL - Monte Carlo Localization algorithm approximately represents the posterior probability density of any state by a finite set of weighted random samples

ISSN 0148-7191