Cost-effective Camera Localization Aided by Prior Point Clouds Maps for Level 3 Autonomous Driving Vehicles

Yan-Tung Leung[1], Xi Zheng[1], Hiu-Yi Ho[1], Weisong Wen[1], Li-Ta Hsu[1]
[1]Department of Aeronautical and Aviation Engineering, The Hong Kong Polytechnic University – (nikkie-yt.leung, zheng-xi.zheng, queenie-hy.ho)@connect.polyu.hk, (welson.wen, lt.hsu)@polyu.edu.hk

KEY WORDS: Visual localization, Prior Point Clouds Maps, Autonomous Driving Vehicles, 3D LiDAR maps, Matching geometry, Image reconstruction.

ABSTRACT:

Precise and robust localization is critical for many navigation tasks, especially autonomous driving systems. The most popular localization approach is global navigation satellite systems (GNSS). However, it has several shortcomings such as multipath and non-line-of-sight reception. Vision-based localization is one of the approaches without using GNSS which is based on vision. This paper used visual localization with a prior 3D LiDAR map. Compared to common methods for visual localization using camera-acquired maps, this paper used the method that tracks the image feature and poses of a monocular camera to match with prior 3D LiDAR maps. This paper reconstructs the image feature to several sets of 3D points by a local bundle adjustment-based visual odometry system. Those 3D points matched with the prior 3D point cloud map to track the globe pose of the user. The visual localization approach has several advantages. (1) Since it only relies on matching geometry, it is robust to changes in ambient luminosity appearance. (2) Also, it uses the prior 3D map to provide viewpoint invariance. Moreover, the proposed method only requires users to use low-cost and lightweight camera sensors.

1. INTRODUCTION

Accurate positioning is an important factor for many navigation tasks, especially for autonomous driving systems. Although GNSS has been popularly used for many years, it still has some limitations. GNSS systems precision is limited within a couple of meters and lacks information on orientation. In particular, GNSS positioning performance has a tolerance of 100 m in urban canyons due to non-line-of-sight (NLOS) and multipath effects (Gu et al., 2015). This major limitation is caused by different factors occurring in the high density of human structures areas, which include buildings, obstacles, and high demand for navigation services.

The recently developed point cloud map matching-based localization method (Zou et al., 2022) attracted lots of attention due to its high accuracy and robustness. The key idea is to match the real-time point clouds captured by the 3D Light Detection and Ranging (LiDAR) with the pre-built point cloud map, therefore, it can estimate the position of the vehicle within the map (Yurtsever et al., 2020). Akai et al. (2017) propose a road marking detection method that uses LiDAR reflective intensity data to build a pre-built map and match it with the NDT approach. However, this method requires a sufficient number of landmarks to make the system successful.

Autonomous driving vehicles which possess Society of Automotive Engineers (SEA) Level 3 (Sae International, 2018) or above requirements remain a marginal part of the market due to the cost of LiDAR being expensive. Toyota operated a potential SEA Level 4 service in the Tokyo 2020 Olympic Village. However, it still did not popular in the market. The vehicle used the LiDAR sensor which required a higher cost than the monocular camera. Using LiDAR in consumer autonomous driving vehicles will increase the operation cost thus the company is prohibitive. Using monocular camera base localization to replace LiDAR-based localization could become popular. The monocular camera is widely available on a low-cost and small platform. Although the monocular camera did not directly provide range information, it provided rich visual information to establish correspondences with the reference images. Therefore, it is promising to investigate the efficient and robust camera-based localization solution within the pre-built point cloud map.

The key idea in this research is using matching geometry for visual localization which the user only requires a monocular camera. This paper proposes a cost-effective camera localization solution aided by the prior 3D point cloud map. The prior map was built with LiDAR data. However, the point cloud data have a distortion problem which is an important element of building prior 3D point cloud maps. The key to removing the point cloud distortion is to estimate the trajectory of the LiDAR in a scanning period. This paper uses ground-truth data to provide the real position with angular velocity and acceleration information for LiDAR. Also, the LiDAR data provides motion information. The real location was measured by using linear interpolation based on the time difference and location change. First, the integration of the visual/inertial is employed to reconstruct the local environment described with sparse but representative 3D feature points, so-called the local points map (LPM). Thanks to the relative motion estimation from the Visual-Inertial integrated system, a good initial guess can be obtained simultaneously. Second, given the initial guess of the pose estimation together with the generated local points map the iterative closest point (ICP) (Segal et al., 2009) is adopted to match the LPM with the prior 3D point cloud map to get the pose of the system within the map. However, the 3D point of LPM with the prior map has a scale problem due to the point estimated by the motion of the camera. This paper provides alignment with a 7-DoF transformation which is measured by a non-linear least squares minimization problem with g2o (Kümmerle et al., 2011) and the Levenberg-Marquardt algorithm (Moré, 1978).

2. RELATED WORKS

2.1 LiDAR distortion

LiDAR is an important sensor, especially for SLAM (Simultaneous Localization and Mapping). LiDAR provides precise distance measurements. LiDAR lacks the effect of visual features such as low light conditions. It is ideal for creating highly accurate maps and for detecting obstacles in the environment. However, LiDAR has a problem with distortion when it moves at the same time as starting the scanning process. It impacts the

accuracy of the map and the difficulty of localization when mapping using LiDAR data. Distortion in LiDAR point clouds is typically categorized into two types: ego-motion distortion and object-motion distortion. This paper mainly focuses on ego-motion distortion.

LOAM (Zhang and Sanjiv Singh, 2014) is one of the examples using SLAM-based approaches which achieve efficient and accurate scan matching in odometry and mapping. Another method is based on iterative closest point (ICP) which matches with consecutive scans to correction of the point (Schneider et al., 2010). However, it affects the result by moving objects in the environment such as vehicles (Hong et al., 2010). One of the corrections methods uses the information of IMU or odometry measurements to correct the point. However, IMU or odometry has the problem of cumulative error (BROSSARD and BONNABEL, 2019). Therefore, this paper uses GNSS/INS unit to overcome the cumulative error.

2.2 LiDAR and Visual SLAM

SLAM is the approach when the robot localizes in an unknown environment, at the same time, constructs a map of its surroundings (Leonard and Durrant-Whyte, 1991). Recently, researchers have proposed an integrated approach of LiDAR-SLAM and Visual-SLAM. This method solves the visual-based localization problem by combining with a LiDAR prior map which provides accurate range measurements.

In the mapping process, scan-matching used LiDAR data to estimate the motion information and generate the 3D map (Debeunne and Vivet, 2020). Typically, the process used Iterative Closest Point (ICP) algorithm to register 3D point clouds and alignment (Besl and McKay, 1992; Tam et al., 2013). The graph-based optimization is used to reduce local errors by representing the robot trajectories and map (Konolige et al., 2010). Also, it used feature-based methods performing loop closure to improve the global map consistency (Martín et al., 2014; Steder et al., 2011). This approach can generate highly accurate 3D maps due to the accurate range of information provided by LiDAR.

The LiDAR sensor is expensive which limits application on low-cost and small platforms. Therefore, Visual SLAM is much available on those platforms. Visual SLAM is a method of visual localization that use images as the source of information (Taketomi et al., 2017). The most common method matches the image feature to estimate the robot's motion and build a feature map. Most early visual SLAM approaches use filtering frameworks such as the Extended Kalman filter (EKF) or particle filter to build probability models. Chiuso et al. (2002) developed a real-time reconstructing Structure from Motion (SFM) with monocular images. Mono-SLAM (Davison, 2003; Davison et al., 2007) developed a similar method with used an Extended Kalman filter and added a local loop closure process to estimate the feature position and the post of the camera. EKF has a linearization issue due to inconsistencies happen. Therefore, researchers have proposed enhancing the parameterization such as the approach of Eade and Drummond (2007) which use a local filter to build sub-maps. Bundle adjustment (BA) (Triggs et al., 2000) is a method used in SFM as global optimization. Therefore, parallel tracking and mapping (PTAM) (Eade and Drummond, 2007) is based on keyframe BA to perform the tracking and mapping process. Strasdat et al. (2010) compare those performances and show that the keyframe BA has efficient accuracy and computational cost.

The most typical method of visual localization is structure-based which relies on 3D reconstructions (3D point cloud map) and localizes in 3D point cloud map (Moulon et al., 2013; Torii et al., 2015). This method compares the image by the local feature such as SIFT descriptor (Lowe, 2004). However, the feature is affected by the illumination conditions and the changes in the season, which are not suitable in changing environments to locate the vehicle's position (Moulon et al., 2013). Meanwhile, using a monocular camera have a scale problem that limits its applications.

Recently, researchers have proposed several machine learning methods (Dusmanu et al., 2019; Sarlin et al., 2020), such as end-to-end learning architectures, to relieve the problem (Csurka et al., 2019). However, those end-to-end methods did not prove as steady as the geometric and probabilistic approaches. On the other hand, using image retrieval-based methods directly searches the most relevant images from the map which extracts all the information in the area of the small gradient (Zhou et al., 2020). Therefore, it prefers better than a structure-based method in the texture, motion blur, and defocuses of the image. It requires high computing power (GPUs) for real-time performance.

However, most of the approaches also focus on matching the optical feature of the environment, fewer approaches focus on using a 3D point cloud map to extract geometry information and match it with the camera image. Wolcott et al. (2014) propose an approach to localize the vehicle using a monocular camera in a prior point cloud map. It uses a prior 3D point cloud map to create 2D synthetic images by image rendering and matching with the maximum normalized mutual information from real-time images. Pascoe et al. (2015) propose an approach to localize the vehicle by minimizing the normalized information distance. It uses real-time camera images and the rendered image from a prior map which combines data from LiDAR and cameras. Those approaches focus on matching in 2D space and using GPU for rendering images. Therefore, Caselitz et al. (2016) propose an approach to avoid the use of GPU for image rendering, it directly matches 3D geometry.

3. METHODOLOGY

This paper proposes a cost-effective camera localization solution with reliable initialization aided by the prior 3D point cloud map. Based on the method introduced by Caselitz et al. (2016) and focuses on the urban condition to find out the scientific problem. Figure 1 shows the proposed method's flow chart, which aims to use a monocular camera to localize within a prior 3D point cloud map. First, the visual feature used to reconstruct the local environment is described with sparse but representative 3D feature points, known as a local points map (LPM). The relative motion estimation from the visual odometers proposed a good initial guess can be obtained simultaneously. Second, given the initial guess of the pose estimation together with the generated local points map, the ICP-based point could registration method (Segal et al., 2009) is adopted to match the LPM with the prior 3D point cloud map to determine the pose of the system within the map.

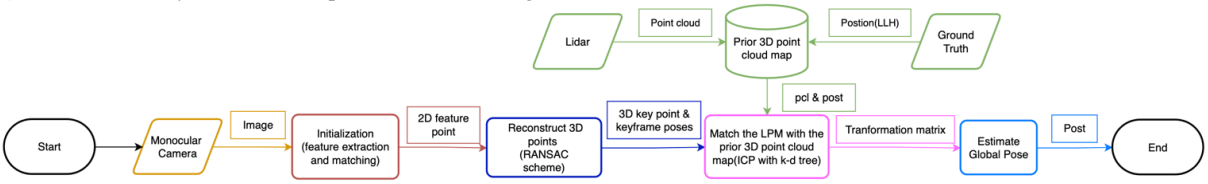


Figure 1. Proposed method's flow chart

3.1 Prior 3D point cloud map

The prior 3D point cloud map is developed using LiDAR data and the group truth data. The inputs are the image and the GNSS position data, the output is a point cloud message with a camera pose. However, the LiDAR data has a distortion problem related to the LiDAR scanning method. Traditionally, the LiDAR scanned 360 degrees from points around its center. When the vehicle with a LiDAR sensor is static, the coordinates of LiDAR origin without change, the start point, and the end point of LiDAR scanning are the same. However, if the vehicle is moving at the same time the LiDAR is operating, the distortion will occur as the longer scanning time caused by the moving of LiDAR coordinates origin. Figure 2 explains the reason for LiDAR distortion.

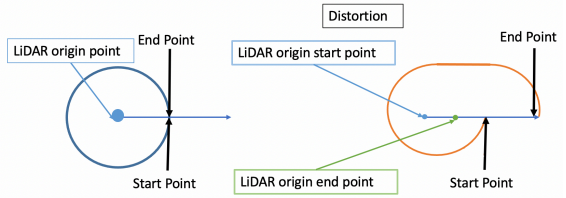


Figure 2. Illustration of the LiDAR distortion phenomenon.

The distortion problem can be solved by using ground truth data provided by GNSS. The ground truth data provided the message of Latitude, Longitude, and Height position (LLH) and quaternion. LLH is used to define the ENU coordinate system for local processing. The quaternion is used to find the rotation between different coordinate systems.

$$\mathbf{D} = \{t_{D_n}, \mathbf{p}_{\mathbf{D_n}}, \boldsymbol{q}_{\boldsymbol{D_n}}, n = 1, \dots, k\}, \quad (1)$$

$$\mathbf{G} = \begin{Bmatrix} t_{G_1} & \cdots & t_{G_m} \\ \mathbf{p}_{\mathbf{G_1}} & \cdots & \mathbf{p}_{\mathbf{G_m}} \\ \mathbf{q}_{\mathbf{G_1}} & \cdots & \mathbf{q}_{\mathbf{G_m}} \end{Bmatrix}, \quad (2)$$

$$\mathbf{P} = \{t_{P_n}, \mathbf{p}_{\mathbf{P_n}}, \mathbf{q}_{\mathbf{P_n}}, n = 1, \dots, k\}, \quad (3)$$

Where **D** is a set of the total number of point cloud data with distortion in one frame. **G** is a set of the number of ground truth data. **P** is a set of the total number of point cloud data without distortion in one frame. t is message time. **p** is coordinated in the ENU coordinate system. **q** is the quaternion.

To calculate the actual position of each point cloud data, including the ENU position and quaternion, linear interpolation is used for ENU position, while spherical linear interpolation is used for the quaternion.

$$\frac{t_{D_n} - t_{G_i}}{t_{G_j} - t_{G_i}} = \frac{\mathbf{p}_{\mathbf{P_n}} - \mathbf{p}_{\mathbf{G_i}}}{\mathbf{p}_{\mathbf{G_j}} - \mathbf{p}_{\mathbf{G_i}}}, \quad (4)$$

$$\mathbf{q}_{\mathbf{P_n}} = \frac{\sin(1-r)\theta}{\sin(\theta)} \mathbf{q}_{\mathbf{G_i}} + \frac{\sin(r\theta)}{\sin(\theta)} \mathbf{q}_{\mathbf{G_j}}, \quad (5)$$

Where *i,j* is the near ground truth message time with t_{D_n}. *r* is the interpolation coefficient. θ is the angle between q_{G_i} and q_{G_j}.

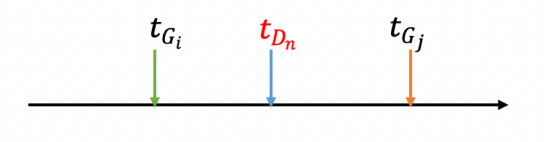


Figure 3. Timeline of chosen data

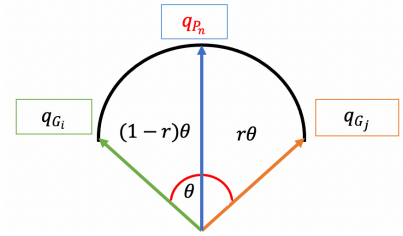


Figure 4. Spherical linear interpolation

3.2 Feature Matching

This process utilizes the ORB algorithm which is combined with the Oriented Features from Accelerated and Segments Test (FAST) algorithm and Rotated Binary Robust Independent Elementary Features (BRIEF) algorithm (Rublee et al., 2011) , to extract image features. The oriented FAST algorithm is employed for feature extraction.

The FAST algorithm computes the feature point by comparing the pixel's intensity with the surrounding pixels. However, the FAST feature without considering the orientation and multi-scale feature. Therefore, the ORB algorithm uses an image pyramid with a Harris corner detector to locate each key point at a different scale. To estimate the orientation, it assumed the corner intensity is offset from the center. Therefore, the algorithm calculates the orientation of the patch by calculating the image moments.

The Rotated BRIEF algorithm improves the original BRIEF algorithm(Calonder et al., 2010) by considering the rotation of feature points, which is important for matching between images. It computes binary feature vectors as descriptors for each key point detected by the oriented FAST algorithm. This paper uses a 7x7 Gaussian kernel used to smooth image patches, and the features vector ($\mathbf{f_n(B)}$) for each patch is defined by n binary tests.

$$\tau(B; x, y) = \begin{cases} 1, & B(x)_l < B(y)_l \\ 0, & B(x)_l \geq B(y)_l \end{cases}, \quad (10)$$

$$\mathbf{f_n(B)} = \sum_{1<i<n} 2^{i-1} \tau(B; \mathrm{x}_i, y_i), \quad (11)$$

Where τ is the binary test. n is vector length. $B(x)_l$ is the intensity of pixel x in patch B.

However, the BRIEF algorithm without consider the orientation of the feature point. To address this limitation, the Rotated BRIEF algorithm was introduced, which incorporates the orientation information by multiplying the cosine and sine values with a set of feature points rotated according to the key point orientation.

$$\mathbf{S} = \begin{pmatrix} u_1 & \dots & u_n \\ v_1 & \dots & v_n \end{pmatrix}, \quad (12)$$

$$\mathbf{R_\theta} = \begin{bmatrix} cos\,\theta & -sin\,\theta \\ sin\,\theta & cos\,\theta \end{bmatrix}, \quad (13)$$

$$\mathbf{S_\theta} = \mathbf{R_\theta S}, \quad (14)$$

$$\mathbf{g_n(p, \theta)} := \mathbf{f_n(p)} | (\mathrm{x}_i, y_i) \in \mathbf{S_\theta}, \quad (15)$$

Where θ is the key point orientation from oriented FAST. **S** is a feature set of n binary tests at location (u_i, v_i). R_θ is the corresponding rotation matrix**.** $\mathbf{S_\theta}$ is a set of feature points rotated according to the key point orientation.

After computing the feature descriptors, key point feature descriptors in two consecutive images are matched using the Brute-Force Matcher with Hamming distance, which estimates the closest distance between descriptors. To increase the accuracy, it executes the cross-check.

$$D(b_1, b_2) = b_1 \oplus b_2, \quad (16)$$

Where D is Hamming distance. b1, b2 are feature descriptors of two different image

3.3 Local Points Map Reconstruct

After obtaining the paired 2D feature points, this module calculates the feature point into a 3D map point and keyframe poses, which call a local point maps, solving the local Bundle Adjustment problem. First, it calculates the fundamental (**F**) matrix and homograph (**H**) matrix with an eight-point algorithm. To improve the stability and accuracy of the solution, it normalizes the coordinates of the input point set.

$$\mathbf{p_1}^{\mathbf{T}}\mathbf{F}\mathbf{p_2} = 0, \tag{17}$$

Where $\mathbf{p_1}, \mathbf{p_2}$ is a pair of matching points in two Frames. **F** is a fundamental matrix.
Second, it scores random sample consensus (RANSAC) results using reprojection error. It is assumed that the reprojection from the current frame to the reference frame will generate a straight line (l_2).

$$l_2 = \boldsymbol{F}^{-1}\begin{bmatrix} u_2 \\ v_2 \\ 1 \end{bmatrix}, \tag{18}$$

Where $\begin{bmatrix} u_2 \\ v_2 \\ 1 \end{bmatrix}$ is the coordinate of the current frame
Ideally, the point should be exactly on a straight line, but it has a projection error. The RANSAC score is accumulated based on RANSAC score by the projection error of the current matrix.
At the same time, it computes the **H** matrix with the same process as the **F** matrix. Third, the score ratio of the two matrices is calculated to determine which model to choose. Then, recovering the rotation and translation from the chosen matrix is achieved by singular value decomposition (SVD) of the essential matrix. Finally, it calculates the 3D point by the triangulation algorithm.

3.4 Localization in Prior Map

This process uses 3D Euclidean spaces and Lie group SE(3) to describe the variables.

$$\mathbf{R} = \{\mathbf{r}_i \in \mathbb{R}^3, i = 1, \dots, n\}, \tag{19}$$
$$\mathbf{P} = \{\mathbf{p}_j \in \mathbb{R}^3, j = 1, \dots, m\}, \tag{20}$$
$$\mathbf{T} = \{\mathbf{T}_i \in \mathrm{SE}(3), i = 1, \dots, t\}, \tag{21}$$
$$\mathbf{F} = \{\mathbf{F}_i \in \mathrm{SE}(3), i = 1, \dots, f\}, \tag{22}$$

Where **R** is a set of reconstruction 3D points. **P** is a set of 3D points in the prior 3D point cloud map. **T** is a set of transformations between **R** and **P**. **F** is a set of keyframe poses.
To compute the correspondences between the reconstructed points and the point of the 3D point cloud map, this paper utilizes the ICP algorithm with KD-tree to perform nearest neighbor search. The method utilizes the local points map and the prior point cloud map to determine correspondences. The correspondences are updated iteratively by estimating the transformation between the two point clouds. The prior point cloud map constructed a KD-tree, which is then utilized to find the nearest neighbor of each point within a specified search radius.

$$\mathbf{R}_j = argmin\left|\left|\mathbf{P}_k - \mathbf{R}_i\right|\right|, for\ i = 1, \dots, M, k = 1, \dots, N, \tag{23}$$

Where $\mathbf{R}_j$ is the closest neighbor point of $\mathbf{P}_k$. N is the number of points in **P**. M is the number of points in **R**.
The correspondence set (C) is comprised of pairs (i, j), representing the correspondences between point i in the prior point cloud map and point j in the reconstruction of 3D points.

$$\mathbf{C} = \{(i, j) |\ i \in \mathbf{R}, j \in \mathbf{P}\}, \tag{24}$$

This paper estimates the process of alignment by the given set of correspondences. It estimates the alignment between the reconstructed local point map with the prior 3D point cloud map by using the similarity transformation.
It estimates the transformation with the reference frame and current keyframe by solving the non-linear least squares minimization problem with g2o (Kümmerle et al., 2011).
First, this paper uses the VertexSim3Expmap class of g2o to estimate the transformation between two 3D point clouds. It uses exponential map parametrization for the transformation factor to account for differences between the two point clouds.
The equation for the VertexSim3Expmap can be written as:

$$\mathbf{T} = [s\mathbf{r} \mid \boldsymbol{t}] \in sim(3), \tag{25}$$

Where **r** is rotation matrix (3x3). **t** is translation vector (3x1). *s* is scaling factor (scalar)
Second, this paper uses sparse optimizer class of g2o to find the optimal values of **T** that minimize the sum of squared errors between the transformed points in the prior 3D point cloud map and their corresponding points in the reconstructed local point map. This optimization problem can be written as:

$$minimize\ \sum w_i * \|\mathbf{T} * \mathbf{P}_i - \mathbf{R}_i\|^2, \tag{26}$$

Where w_i is the weight assigned to each correspondence
The sparse optimizer class uses the Levenberg-Marquardt algorithm to iteratively update the transformation matrix (**T**) until the sum of squared errors is minimized. At each iteration, the algorithm computes the Jacobian matrix (**J**) of the objective function with respect to the parameters of **T**, and uses it to compute the update to **T**:

$$\mathbf{T}_k = \mathbf{T}_{k-1} + (\mathbf{J}^T\mathbf{J} + \lambda\mathbf{I})^{-1}\mathbf{J}^T\mathbf{r}, for\ k = 1 \dots . N\ , \tag{27}$$

where **J** is the Jacobian matrix of the error function with respect to **T**. **r** is the residual vector. λ is a damping parameter. **I** is identity matrix. N is number of iteratively.
The optimal values of the relative transformation between the two-point clouds are obtained by minimizing the cost function. Then, joining all the similarity transformations when estimating all the iterations, it estimates the reference frame pose of the map. Finally, it estimates the global pose of the vehicle.

4. EXPERIMENT

4.1 Experiment Setup

This paper evaluated the performance of the proposed method by using the UrbanNav dataset (Hsu et al., 2021) which data was collected in Hong Kong as a typical urban canyon. This paper developed the 3D point cloud map by using NovAtel SPAN-CPT + Inertial Explorer (IE) (1 Hz), HDL 32E Velodyne (10 Hz), and ZED2 Stereo (15 Hz) to provide ground truth, 3D LiDAR, and camera image, respectively. During the data collection, we collect raw GPS measurements by a commercial-level u-blox F9P GNSS receiver (1 Hz). Figure 5 shows the full experiment setup.

Figure 5. Experiment setup

This paper collects image data by using the ZED2 Stereo mounted on our vehicle. The cameras were calibrated using the MATLAB method. All the data were collected and synchronized using a robot operation system (ROS) (Quigley et al., 2009). During the evaluation, we compare the following pipelines:

1) Using the 3D local point by ORB-SLAM to evaluate the accuracy of our proposed method.

- ORB-SLAM (Campos et al., 2021): Using the ORB method to extract features and descriptions then calculate the 3D local point.

2) Using different datasets to evaluate our proposed method by comparing the estimated camera trajectory and ground truth trajectory with the prior map.

- KITTI odometry dataset (Geiger et al., 2012): Provide outdoor data in rural areas by using a vehicle.
- UrbanNav dataset (Hsu et al., 2021): Provide outdoor data in urban areas by using a vehicle.

4.2 Experimental Evaluation

4.2.1 Prior 3D point cloud map: This paper uses the prior 3D point cloud map to match with the 3D feature point. However, the LiDAR data have a distortion problem. Figure 6 shows the prior 3D point cloud map with a distortion problem using the UrbanNav dataset.

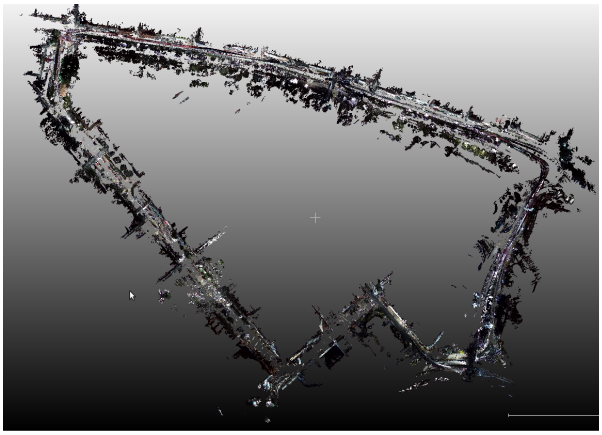
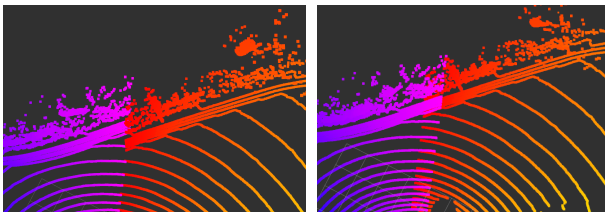


Figure 6. Prior 3D point cloud map with distortion (UrbanNav dataset)

Figure 7 shows that the problem of distortion has been solved. It shows the situation of the wall which should show as a continuous line. The colors red and purple has represented the start and end times of the LiDAR scan. However, the left-hand side image shows the line is cut off. After solving the distortion problem, the right-hand side image shows the line overlap which means the problem has been solved.



(a) (b)

Figure 7. Visualize one frame of LiDAR data from the UrbanNav dataset (a) with distortion and (b) without distortion.

4.2.2 Initialization: It uses the camera data as an input source of images. In the initialization process, the ORB algorithm is used to extract and describe the detected feature point of the input data. Then, the detected feature point of each image is used to match the previous image. Figure 8 shows the results of matching the feature point in two constant images.

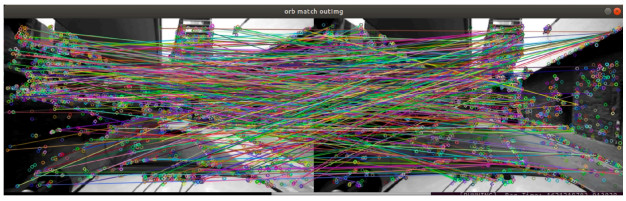
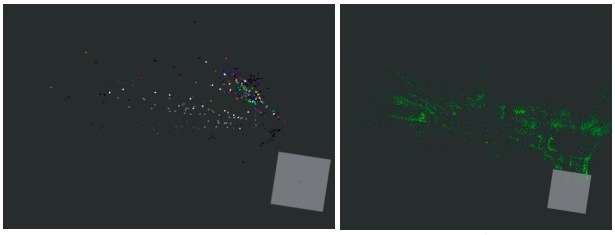


Figure 8. Example of Feature Matching.

4.2.3 Reconstruct: In the reconstruction process, it used the pair feature point to reconstruct the 3D point in each frame. The reconstructed feature points are then combined into a 3D map point and keyframe poses, forming the local point map. Figure 9 shows an example of the local point map in one frame and the combination of several frames in RViz (Kam et al., 2015).



(a) (b)

Figure 9. Example of (a) a local point map in one frame and (b) a combination of several frames in RViz.

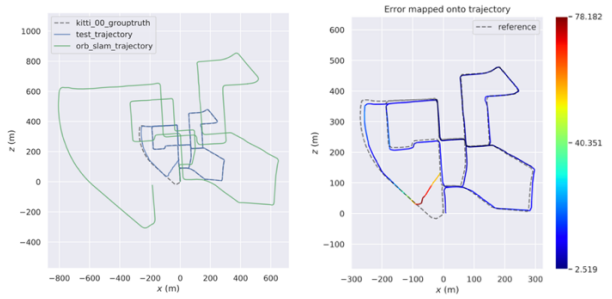
4.2.4 Matching: This paper evaluated the proposed method in two datasets with different environments. First, we evaluated the KITTI odometry dataset to evaluate the accuracy of our proposed method. Second, we use the UrbanNav dataset to evaluate our proposed method in Urban conditions by comparing the estimated camera trajectory and ground truth trajectory with the prior map.

4.2.4.1 Evaluate the accuracy: We use the KITTI odometry dataset to evaluate the accuracy of our proposed method. KITTI dataset provides LiDAR data, stereo image, and ground truth data. We only use the image of the left camera.

In this experiment, we chose the raw dataset which is sequence 00. The group truth data of the KITTI odometry dataset are provided by KITTI Vision Benchmark Suite. We assume the group truth data is the correct pose of the camera. Therefore, we can compute the 6-DoF camera position error.

We use ORB-SLAM to provide the 3D point from the left-side camera image. Figure 10(a) shows the trajectory of the group truth data of the KITTI odometry dataset, the visual odometry of ORB-SLAM and our method. It shows that the trajectory of the patterns of ORB-SLAM is similar to other results, but the drift

happened. Figure 10(b) shows the localization result of our proposed method compared with the group truth data of the KITTI odometry dataset. Although some parts did not match with the group truth data, it shows that most of the drift is corrected.



(a) (b)

Figure 10. The (a) trajectory and the (b) error of the KITTI group truth data, ORB-SLAM, and our method.

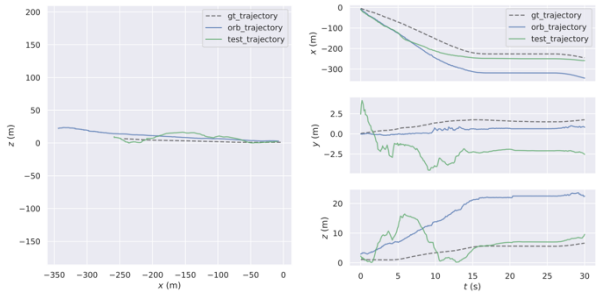
The results were run 6 times to reduce the randomness. Table 1 shows the absolute trajectory error (ATE) of each dataset. The results include the error between our method trajectory with the KITTI ground truth trajectory and the ORB-SLAM trajectory with the KITTI ground truth trajectory. The result shows that our method outperforms the visual-only method.

ATE(Average)		Our method	ORB-SLAM
Rotation (Degree)	RMSE	117.6	127.8
	SD	10.5	13.7
Translation (m)	RMSE	20.9	309.8
	SD	16.6	181.3
Transformation	RMSE	21.0	309.8
	SD	16.5	181.3

Table 1. The results of average ATE for 6 runs

4.2.4.2 Evaluate in Urban conditions: We use the data collected by our lab which is the UrbanNav dataset team. The data was collected in Kowloon Tong, Hong Kong. The group truth data was computed by the data of NovAtel SPAN-CPT + IE (1 Hz). We only use the image of the left camera provided by the ZED 2 camera. Also, we assume the group truth data is the correct pose of the camera. Therefore, we can compute the 6-DoF camera position error.

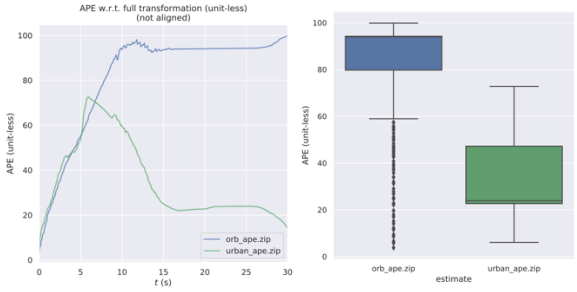
In this experiment, we chose a straight road in the urban environment. The result of the trajectory of ORB-SLAM data, and data from our method compared with the trajectory of ground truth data, as shown in Figure 11(a). Figure 11(b) shows the trajectory error in the translation part.



(a) (b)

Figure 11. The trajectory results of ORB-SLAM and our method compared with the ground truth data. (a) Multiple trajectories are plotted in a line chart with XZ axis. (b) Multiple trajectories are plotted in a line chart with the X-axis representing time, Y-axis representing time and Z-axis representing time.

In the X-axis, those have similar patterns, and our method is close to the ground truth data. In the Y-axis and Z-axis, our method has more drift than the orb-slam trajectory in the first 15 seconds as the scenes of the first 15 seconds are more complicated. Figure 12(a) shows the absolute trajectory error (ATE) of our trajectory and ORB-SLAM trajectory. Figure 12(b) shows the box plot of the absolute trajectory error. Although our method initially exhibits greater drift in the first 6 seconds, it subsequently corrects the camera pose resulting in a better interquartile range compared to the ORB-SLAM method. This indicates that our method is able to solve for accumulated errors.



(a) (b)

Figure 12. Absolute trajectory error (ATE) of our trajectory and ORB-SLAM trajectory shows in (a) line chart and (b) box plot.

5. CONCLUSION

This paper proposed the method of using vision-based localization with prior 3D LiDAR maps to track the camera pose. We combined the benefits of LiDAR and the camera to estimate the transformation of the local point map and prior 3D map. It continues to track the 6-DoF camera pose. To evaluate the performance of the system, we use open-source data. It demonstrated the accuracy of this system through real-world experiments, which produced notable outcomes. However, the challenge of the urban environment still occurs. Our future work will improve our method in more dynamic and challenging scenarios.

6. REFERENCES

Akai, N., Morales, L.Y., Takeuchi, E., Yoshihara, Y., Ninomiya, Y., 2017. Robust localization using 3D NDT scan matching with experimentally determined uncertainty and road marker matching, in: 2017 IEEE Intelligent Vehicles Symposium (IV). Presented at the 2017 IEEE Intelligent Vehicles Symposium (IV), pp. 1356–1363. https://doi.org/10.1109/IVS.2017.7995900

Besl, P.J., McKay, N.D., 1992. Method for registration of 3-D shapes, in: Sensor Fusion IV: Control Paradigms and Data Structures. Spie, pp. 586–606.

BROSSARD, M., BONNABEL, S., 2019. Learning Wheel Odometry and IMU Errors for Localization, in: 2019 International Conference on Robotics and Automation (ICRA). Presented at the 2019 International Conference on Robotics and Automation (ICRA), pp. 291–297. https://doi.org/10.1109/ICRA.2019.8794237

Calonder, M., Lepetit, V., Strecha, C., Fua, P., 2010. BRIEF: Binary Robust Independent Elementary Features, in: Daniilidis, K., Maragos, P., Paragios, N. (Eds.), Computer Vision – ECCV 2010, Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, pp. 778–792. https://doi.org/10.1007/978-3-642-15561-1_56

Campos, C., Elvira, R., Rodríguez, J.J.G., Montiel, J.M.M., Tardós, J.D., 2021. ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial and Multi-Map SLAM. IEEE Trans. Robot. 37, 1874–1890. https://doi.org/10.1109/TRO.2021.3075644

Caselitz, T., Steder, B., Ruhnke, M., Burgard, W., 2016. Monocular camera localization in 3D LiDAR maps, in: 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Presented at the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1926–1931. https://doi.org/10.1109/IROS.2016.7759304

Chiuso, A., Favaro, P., Jin, H., Soatto, S., 2002. Structure from motion causally integrated over time. IEEE transactions on pattern analysis and machine intelligence 24, 523–535.

Csurka, G., Dance, C.R., Humenberger, M., 2019. From handcrafted to deep local features. arXiv:1807.10254 [cs].

Davison, A.J., 2003. Real-time simultaneous localisation and mapping with a single camera, in: Computer Vision, IEEE International Conference On. IEEE Computer Society, pp. 1403–1403.

Davison, A.J., Reid, I.D., Molton, N.D., Stasse, O., 2007. MonoSLAM: Real-Time Single Camera SLAM. IEEE Transactions on Pattern Analysis and Machine Intelligence 29, 1052–1067. https://doi.org/10.1109/TPAMI.2007.1049

Debeunne, C., Vivet, D., 2020. A Review of Visual-LiDAR Fusion based Simultaneous Localization and Mapping. Sensors 20, 2068. https://doi.org/10.3390/s20072068

Dusmanu, M., Rocco, I., Pajdla, T., Pollefeys, M., Sivic, J., Torii, A., Sattler, T., 2019. D2-net: A trainable cnn for joint description and detection of local features, in: Proceedings of the Ieee/Cvf Conference on Computer Vision and Pattern Recognition. pp. 8092–8101.

Eade, E., Drummond, T., 2007. Monocular SLAM as a graph of coalesced observations, in: 2007 IEEE 11th International Conference on Computer Vision. IEEE, pp. 1–8.

Geiger, A., Lenz, P., Urtasun, R., 2012. Are we ready for autonomous driving? the kitti vision benchmark suite, in: 2012 IEEE Conference on Computer Vision and Pattern Recognition. IEEE, pp. 3354–3361.

Gu, Y., Hsu, L.-T., Kamijo, S., 2015. Passive Sensor Integration for Vehicle Self-Localization in Urban Traffic Environment. Sensors (Basel) 15, 30199–30220. https://doi.org/10.3390/s151229795

Hong, S., Ko, H., Kim, J., 2010. VICP: Velocity updating iterative closest point algorithm, in: 2010 IEEE International Conference on Robotics and Automation. Presented at the 2010 IEEE International Conference on Robotics and Automation, pp. 1893–1898. https://doi.org/10.1109/ROBOT.2010.5509312

Hsu, L.-T., Kubo, N., Wen, W., Chen, W., Liu, Z., Suzuki, T., Meguro, J., 2021. UrbanNav:An Open-Sourced Multisensory Dataset for Benchmarking Positioning Algorithms Designed for Urban Areas. Presented at the Proceedings of the 34th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2021), pp. 226–256. https://doi.org/10.33012/2021.17895

Kam, H.R., Lee, S.-H., Park, T., Kim, C.-H., 2015. RViz: a toolkit for real domain data visualization. Telecommun Syst 60, 337–345. https://doi.org/10.1007/s11235-015-0034-5

Konolige, K., Grisetti, G., Kümmerle, R., Burgard, W., Limketkai, B., Vincent, R., 2010. Efficient sparse pose adjustment for 2D mapping, in: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, pp. 22–29.

Kümmerle, R., Grisetti, G., Strasdat, H., Konolige, K., Burgard, W., 2011. G2o: A general framework for graph optimization, in: 2011 IEEE International Conference on Robotics and Automation. Presented at the 2011 IEEE International Conference on Robotics and Automation, pp. 3607–3613. https://doi.org/10.1109/ICRA.2011.5979949

Leonard, J.J., Durrant-Whyte, H.F., 1991. Simultaneous map building and localization for an autonomous mobile robot, in: Proceedings IROS '91:IEEE/RSJ International Workshop on Intelligent Robots and Systems '91. Presented at the Proceedings IROS '91:IEEE/RSJ International Workshop on Intelligent Robots and Systems '91, pp. 1442–1447 vol.3. https://doi.org/10.1109/IROS.1991.174711

Lowe, D.G., 2004. Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision 60, 91–110. https://doi.org/10.1023/B:VISI.0000029664.99615.94

Martín, F., Triebel, R., Moreno, L., Siegwart, R., 2014. Two different tools for three-dimensional mapping: DE-based scan matching and feature-based loop detection. Robotica 32, 19–41.

Moré, J.J., 1978. The Levenberg-Marquardt algorithm: Implementation and theory, in: Watson, G.A. (Ed.), Numerical Analysis, Lecture Notes in Mathematics. Springer, Berlin, Heidelberg, pp. 105–116. https://doi.org/10.1007/BFb0067700

Moulon, P., Monasse, P., Marlet, R., 2013. Global Fusion of Relative Motions for Robust, Accurate and Scalable Structure from Motion. Presented at the Proceedings of the IEEE International Conference on Computer Vision, pp. 3248–3255.

Pascoe, G., Maddern, W., Newman, P., 2015. Direct Visual Localisation and Calibration for Road Vehicles in Changing City Environments. Presented at the Proceedings of the IEEE International Conference on Computer Vision Workshops, pp. 9–16.

Quigley, M., Gerkey, B., Conley, K., Faust, J., Foote, T., Leibs, J., Berger, E., Wheeler, R., Ng, A., 2009. ROS: an open-source Robot Operating System. ICRA workshop on open source software 3, 5.

Revaud, J., Weinzaepfel, P., De Souza, C., Pion, N., Csurka, G., Cabon, Y., Humenberger, M., 2019. R2D2: Repeatable and Reliable Detector and Descriptor. arXiv:1906.06195 [cs].

Rublee, E., Rabaud, V., Konolige, K., Bradski, G., 2011. ORB: An efficient alternative to SIFT or SURF, in: 2011 International Conference on Computer Vision. Ieee, pp. 2564–2571.

Sae International, 2018. Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles. SAE international 4970, 1–5.

Sarlin, P.-E., DeTone, D., Malisiewicz, T., Rabinovich, A., 2020. SuperGlue: Learning Feature Matching With Graph Neural Networks. Presented at the Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4938–4947.

Sattler, T., Leibe, B., Kobbelt, L., 2011. Fast image-based localization using direct 2D-to-3D matching, in: 2011 International Conference on Computer Vision. Presented at the 2011 International Conference on Computer Vision, pp. 667–674. https://doi.org/10.1109/ICCV.2011.6126302

Schneider, S., Himmelsbach, M., Luettel, T., Wuensche, H.-J., 2010. Fusing vision and LIDAR - Synchronization, correction and occlusion reasoning, in: 2010 IEEE Intelligent Vehicles Symposium. Presented at the 2010 IEEE Intelligent Vehicles Symposium, pp. 388–393. https://doi.org/10.1109/IVS.2010.5548079

Schonberger, J.L., Hardmeier, H., Sattler, T., Pollefeys, M., 2017. Comparative Evaluation of Hand-Crafted and Learned Local Features. Presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1482–1491.

Segal, A.V., Haehnel, D., Thrun, S., 2009. Generalized-ICP. Robotics: science and systems 2, 435.

Steder, B., Ruhnke, M., Grzonka, S., Burgard, W., 2011. Place recognition in 3D scans using a combination of bag of words and point feature based relative pose estimation, in: 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems. Presented at the 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1249–1255. https://doi.org/10.1109/IROS.2011.6094638

Strasdat, H., Montiel, J.M.M., Davison, A.J., 2010. Real-time monocular SLAM: Why filter?, in: 2010 IEEE International Conference on Robotics and Automation. Presented at the 2010 IEEE International Conference on Robotics and Automation, pp. 2657–2664. https://doi.org/10.1109/ROBOT.2010.5509636

Taketomi, T., Uchiyama, H., Ikeda, S., 2017. Visual SLAM algorithms: A survey from 2010 to 2016. IPSJ Transactions on Computer Vision and Applications 9, 1–11.

Tam, G.K.L., Cheng, Z.-Q., Lai, Y.-K., Langbein, F.C., Liu, Y., Marshall, D., Martin, R.R., Sun, X.-F., Rosin, P.L., 2013. Registration of 3D Point Clouds and Meshes: A Survey from Rigid to Nonrigid. IEEE Transactions on Visualization and Computer Graphics 19, 1199–1217. https://doi.org/10.1109/TVCG.2012.310

Torii, A., Arandjelović, R., Sivic, J., Okutomi, M., Pajdla, T., 2015. 24/7 place recognition by view synthesis, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Presented at the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1808–1817. https://doi.org/10.1109/CVPR.2015.7298790

Triggs, B., McLauchlan, P.F., Hartley, R.I., Fitzgibbon, A.W., 2000. Bundle Adjustment — A Modern Synthesis, in: Triggs, B., Zisserman, A., Szeliski, R. (Eds.), Vision Algorithms: Theory and Practice, Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, pp. 298–372. https://doi.org/10.1007/3-540-44480-7_21

Wolcott, R.W., Eustice, R.M., 2014. Visual localization within LIDAR maps for automated urban driving, in: 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems. Presented at the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 176–183. https://doi.org/10.1109/IROS.2014.6942558

Yurtsever, E., Lambert, J., Carballo, A., Takeda, K., 2020. A Survey of Autonomous Driving: Common Practices and Emerging Technologies. IEEE Access 8, 58443–58469. https://doi.org/10.1109/ACCESS.2020.2983149

Zhang, J., Sanjiv Singh, 2014. LOAM: Lidar Odometry and Mapping in Real-time. Robotics: Science and Systems, pp. 1–9.

Zhou, Q., Sattler, T., Pollefeys, M., Leal-Taixé, L., 2020. To Learn or Not to Learn: Visual Localization from Essential Matrices, in: 2020 IEEE International Conference on Robotics and Automation (ICRA). Presented at the 2020 IEEE International Conference on Robotics and Automation (ICRA), pp. 3319–3326. https://doi.org/10.1109/ICRA40945.2020.9196607

Zou, Q., Sun, Q., Chen, L., Nie, B., Li, Q., 2022. A Comparative Analysis of LiDAR SLAM-Based Indoor Navigation for Autonomous Vehicles. IEEE Transactions on Intelligent Transportation Systems 23, 6907–6921. https://doi.org/10.1109/TITS.2021.3063477