COST-EFFECTIVE CAMERA LOCALIZATION AIDED BY PRIOR POINT CLOUDS MAPS FOR LEVEL 3 AUTONOMOUS DRIVING VEHICLES

LEUNG Yan Tung

MPhil

THE HONG KONG POLYTECHNIC UNIVERSITY

2024

The Hong Kong Polytechnic University

Department of Aeronautical and Aviation Engineering

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

Leung Yan Tung

A thesis submitted in partial fulfilment of the requirements

for the degree of Master of Philosophy

Aug 2023

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

___________ ________ (Signed)

_____Leung Yan Tung______(Name of student)

ABSTRACT

For navigation tasks, particularly within autonomous driving systems, accurate and robust localization is the critical aspect. While global navigation satellite systems (GNSS) are a widespread choice for localization, it has exhibited drawbacks such as susceptibility to issues like multipath and non-line-of-sight reception. Vision-based localization offers an alternative by relying on visual cues, circumventing the use of GNSS signals. In this study, we proposed a visual localization method aided by a prior 3D LiDAR map. Our approach involves reconstructing image features into multiple sets of 3D points using a localized bundle adjustment-based visual odometry system. Subsequently, these reconstructed 3D points are aligned with the prior 3D point cloud map, enabling the tracking of the user's global pose. The proposed visual localization methodology boasts several advantages. Firstly, the aided prior maps contribute to improving the robustness in the face of variations in ambient lighting and appearance. Additionally, it capitalizes on the prior 3D map to confer viewpoint invariance. The key idea of point cloud registration for the proposed approach determines geometric matching to establish the accurate position and orientation of a camera within its surroundings. This is achieved by contrasting the geometric features present in the camera's image with those stored in a reference map. The method identifies and aligns the geometric points between the camera image and the prior 3D point cloud map. Notably, our method is also conducive to the utilization of cost-effective and lightweight camera sensors by end-users. The experiment results show the proposed methods are accurate and frame rates without the need for supplementary information.

KEYWORDS: Autonomous Driving Vehicles, Visual localization, 3D LiDAR maps, Image reconstruction, Matching geometry.

ACKNOWLEDGEMENT

Firstly, I would like to begin by expressing my sincere gratitude to my advisor, Dr. Li-Ta Hsu, for his dedicated and meticulous guidance throughout my academic journey. His insightful advice and unwavering support have truly enriched my research experience during my M.Phil studies.

Additionally, I extend heartfelt appreciation to my co-supervisor, Dr. Weisong Wen, whose instructive ideas and valuable suggestions significantly contributed to the depth of my M.Phil studies.

I am also deeply thankful to the members of the Intelligent Positioning and Navigation Laboratory (IPNL) for their generous support during my academic pursuit. A special mention goes to Xi Zheng for her invaluable guidance on the intricacies of visual SLAM.

Furthermore, I wish to acknowledge the Department of Aeronautical and Aviation Engineering at the Hong Kong Polytechnic University for providing excellent facilities and financial support, which greatly facilitated my research endeavours.

Lastly, I would like to express my profound gratitude to my friends and family for their unwavering encouragement, support, and love throughout this journey. Their presence and belief in me have been the driving force that kept me determined and focused.

Table of Contents

Table of figures

1. INTRODUCTION

1.1 Research Background

The industry and academics have become interested in autonomous driving systems topic as the development of robotics has grown rapidly in recent decades. For many navigation and localization tasks, accurate positioning is an essential factor, specifically for autonomous driving systems. Localization entails the machine's acquisition of its own motion state, whereas navigation involves the machine's perception of the environment and independent traversal to the intended destination [1]. Once localized, the robot gains the ability to autonomously chart its movement trajectory. To achieve consistent navigation, it is imperative to establish a foundation of robust and dependable localization. The primary challenge lies in ensuring enduring stability and precise localization under challenging environmental circumstances and during instances of extreme motion. Global Navigation Satellite Systems (GNSS) is the traditional methods to provide the position of a robot. While GNSS have enjoyed widespread usage for many years, they still exhibit certain limitations. GNSS systems offer precision within a few meters but lack orientation information. Notably, their positioning accuracy faces restrictions within urban canyons, with a tolerance of approximately 100 meters due to the influence of non-line-of-sight (NLOS) and multipath effects [2], which are caused by the signal occlusion and reflection. However, the error is unacceptable for vehicle navigation. The major limitation derives from various factors prevailing in densely built areas, encompassing structures like buildings, obstacles, and the high demand for navigation services [3].

Therefore, Simultaneous Localization and Mapping (SLAM) is one of the efficient alternative techniques which estimate the vehicle's post at the same time build the map of the surrounding environment. Camera and Light Detection and Ranging (LiDAR) is the most common sensor of SLAM which are the visual SLAM and LiDAR SLAM, respectively.

LiDAR is a popular sensor for mapping as it provides the accurate 3D points (point clouds) of the surrounding environment. The LiDAR-based navigation solution is mainly executed by the point cloud registration methods, such as the representative iterative closest point (ICP) [4] and Normal Distribution Transform (NDT) approach [5]. The emergence of the point cloud map matching-based localization method [6] has garnered considerable attention owing to its impressive accuracy and robustness. The fundamental concept involves aligning real-time point clouds obtained from LiDAR scans with prior point cloud maps, enabling the estimation of vehicle positions within the map [7]. Akai et al. [8] introduce a road marking detection approach employing LiDAR reflective intensity data to construct a pre-built map, which is then matched using the NDT approach. However, this method necessitates a substantial number of distinctive landmarks for successful system operation.

LiDAR sensors provide the larger Field of View (FOV) of the information of a 3D map. However, the unaffordable cost of LiDAR sensors leads to the LIDAR sensors being difficult to popularize. Autonomous driving vehicles adhering to Society of Automotive Engineers (SAE) Level 3 [9] or higher standards remain a niche sector due to the costliness of LiDAR technology. Although Toyota showcased a potential SAE Level 4 service during the Tokyo 2020 Olympic Village operations, it has not yet gained widespread traction in the market. This service employed LiDAR sensors, which incur higher costs compared to cameras. Integrating LiDAR into consumer-oriented autonomous driving vehicles escalates operational expenses, making it financially impractical for companies. By contrast, visual SLAM, which uses a camera as the main sensor, provide a more lightweight and cheaper approach [10].Consequently, the adoption of monocular camera-based localization as a substitute for LiDAR-based localization presents an attractive avenue. Monocular cameras are widely accessible on low-cost and compact platforms. While monocular cameras do not directly provide range information, they furnish abundant visual data that can be used to establish correspondences with reference images. Therefore, there exists promise in investigating an efficient and robust camera-based localization solution within the framework of a prior point cloud map.

1.2 Objectives

The core concept explored in this study around geometric matching for visual localization, a process facilitated by the utilization of a monocular camera and a prior map. This paper introduces an economical camera-based localization solution which complemented by a prior 3D point cloud map established through LiDAR data. However, the point cloud data presents distortion, a pivotal aspect in constructing accurate prior 3D point cloud maps. The essence of correcting this distortion lies in estimating the LiDAR's trajectory during its scanning phase. This research employs ground-truth data to furnish actual positions, inclusive of angular velocity and acceleration data for LiDAR. Moreover, motion information is gleaned from LiDAR data. The actual location is determined via linear interpolation based on temporal variations and positional shifts.

Firstly, the system reconstructs the localized environment by the visual data which provides the sparse and representative 3D feature points, known as the local points map (LPM). The visual information provides a robust initial estimation concurrently. Subsequently, armed with the initial pose estimate and the generated local points map, the iterative closest point (ICP)-based [4] method is adopted. This ICP process aligns the LPM with the prior 3D point cloud map, yielding the system's pose within the map. However, a scale issue occurs between the 3D points of the LPM and the prior map due to the motion estimation based on monocular camera. To address this, the paper presents a 7-DoF transformation alignment achieved through non-linear least squares minimization using the g2o [11] framework and the Levenberg-Marquardt algorithm [12].

1.3 Overview

This research provides a method to localize the user positioning in a 3D LiDAR map with only the monocular camera. Therefore, this research first reviews the difficulties and existing solution of the methodology of SLAM, Scene representations, and LiDAR distortion via literature reviews (Section 2). Then, this paper states the methodology of this research about mapping and localization (Section 3). After that, states the experiment results with finding and practical issues related to the finding of the proposed research method performance (Section 4). Finally, summarizes all the chapters and discusses possible future research (Section 5).

2. LITERATURE REVIEW

2.1. Simultaneous Localization and Mapping (SLAM)

SLAM is the approach when the robot localizes in an unknown environment and simultaneously constructs a map of its surroundings [13]. The sensor on the robot is used for relative observations of several unknown landmarks as the robot is moving in the environment, which is shown in Figure 1. The process of SLAM is a recursive estimation process which considered a probability problem[14]. The problem can be written as a probability distribution:

$$\mathrm{P}(\mathbf{x_k}, \mathbf{m} \,|\mathbf{Z_{0:k}}, \mathbf{u_{0:k}}, \mathbf{x_0}) \; , \tag{1}$$

k : Time instant

$\mathbf{x_k}$: Vector of vehicle location and orientation

$\mathbf{u_k}$: Control vector of state $\mathbf{x_k}$ between time k-1 to k

$\mathbf{Z_{k,i}}$: Observation of the i^{th} landmark taken at time k

$\mathbf{m}$: Set of landmarks

$\mathbf{Z_{0:k}}$: Set of observation form time 0 to k

$\mathbf{u_{0:k}}$: Set of control vector form time 0 to k

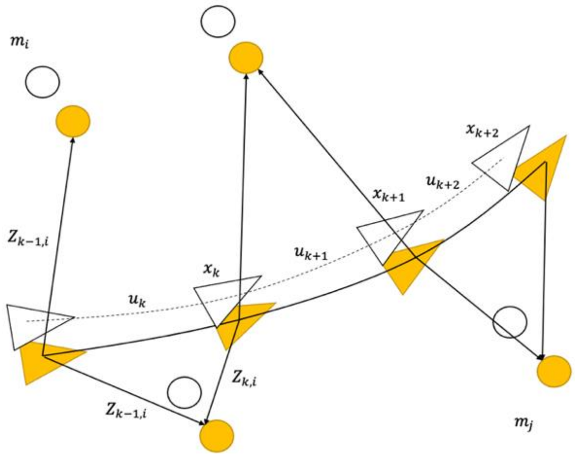


Figure 1. Process of SLAM: Estimated map and trajectory (yellow) and ground truth (white)

SLAM mainly include data processing, mapping analysis and loop closure detection [15]. Data processing mainly uses the data from the sensor to estimate the pose of the robot. Mapping analysis uses the robot's pose to generate or optimize the map and the trajectory. Loop closure detection considers whether the robot's position is passed position or not. Therefore, the process of data processing can further optimize the robot's pose. The standard architecture of SLAM is illustrated in Figure 2.

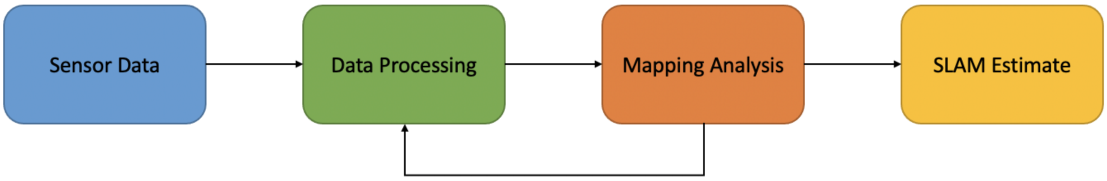


Figure 2. The standard architecture of SLAM[16]

Different sensors, such as camera, LiDAR and sonar, can be used for SLAM (Figure 3). Most autonomous driving systems use LiDAR SLAM or Visual SLAM which mainly use LiDAR or camera sensors to provide data.

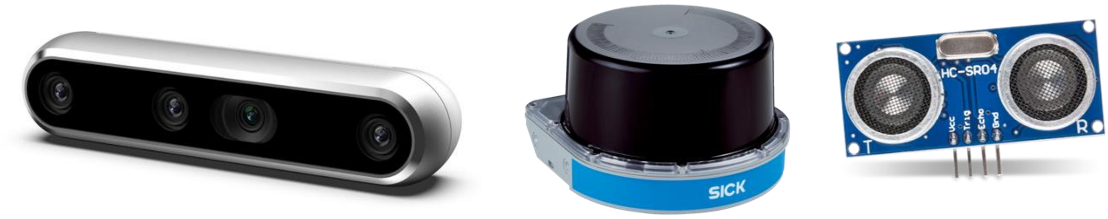


Figure 3. Stereo camera, LiDAR, SONAR

2.1.1. LiDAR SLAM

In LiDAR SLAM, the primary data source comprises LiDAR sensors that supply point cloud data. The robot's pose estimation is achieved through consecutive scans.

During the mapping process, the LiDAR data is employed for scan-matching, which calculates motion estimation and the creation of a comprehensive 3D map [17]. The commonly adopted method involves utilizing the Iterative Closest Point (ICP) algorithm for registering and aligning 3D point clouds [18], [19]. Nevertheless, this process often incurs higher computational costs due to the search for point correspondences which is coupled with a sensitivity to minimization. To address this, KD-tree structures are employed to expedite the search for the nearest point [17]. Graph-based optimization techniques are further employed to mitigate local errors by representing robot trajectories and maps [20]. Additionally, feature-based methods are applied for loop closure, enhancing the global consistency of the map [21], [22]. This approach proves particularly adept at generating highly accurate 3D maps, capitalizing on the precise range information furnished by the LiDAR sensor.

2.1.2. Visual SLAM

Due to the costliness of LiDAR sensors, their application is limited, particularly on low-cost and compact platforms. Consequently, Visual SLAM emerges as a viable alternative for these platforms. Visual SLAM constitutes a methodology for localizing through visuals, utilizing images as the primary information source [10]. Among the prevalent techniques, the most widely adopted involves matching image features to estimate the robot's motion and constructing a feature map.

In the early stages, most visual SLAM methods relied on filtering frameworks such as the particle filter or Extended Kalman filter (EKF) to formulate probability models [18]. Chiuso et al. [19] proposed a method using monocular images to reconstruct the 3D feature points map by Structure from Motion (SFM) in real-time. Mono-SLAM [20] and OpenVINS [21] devised a similar approach, employing an Extended Kalman filter (EKF) while incorporating a local loop closure process for estimating feature positions and camera poses. However, EKF is susceptible to linearization challenges stemming from inconsistencies. In response, researchers have proposed enhancements in parameterization. For instance, Eade and Drummond [22] employed a local filter to build sub-maps. For global optimization in SFM, Bundle Adjustment (BA) [23] is a widely employed technique. This principle forms the foundation of parallel tracking and mapping (PTAM) [22] , which leverages keyframe BA for simultaneous tracking and mapping. Strasdat et al. [24] conducted a comparative analysis of these approaches, revealing that keyframe BA achieves an optimal balance between accuracy and computational efficiency.

The camera is the main sensor of Visual SLAM such as RGB-D camera, monocular camera and stereo camera. Monocular Cameras are cheaper and more common on robots and different SLAM technology. However, the monocular camera has the biggest challenge of 3D mapping which has the inherent scale ambiguity. The scale problem is the main error effect as the scale drifts over time and cannot be observed [25]. The benefits of the monocular camera are can seamlessly switch between environments of different scales, such as an indoor environment and a large outdoor environment. The stereo camera and RGB-D camera is the camera sensor which can provide the scale information of each image pixel. The stereo camera uses two different-angle cameras to provide different views of images (called the multiple view geometry) which can estimate the depth of each pixel which is shown in Figure 4. The RGB-D camera includes a monocular camera, IR transmitters and IR receivers which use infrared to provide depth Information. However, the range of detection is limited. The monocular camera provides more flexibility in range detection.

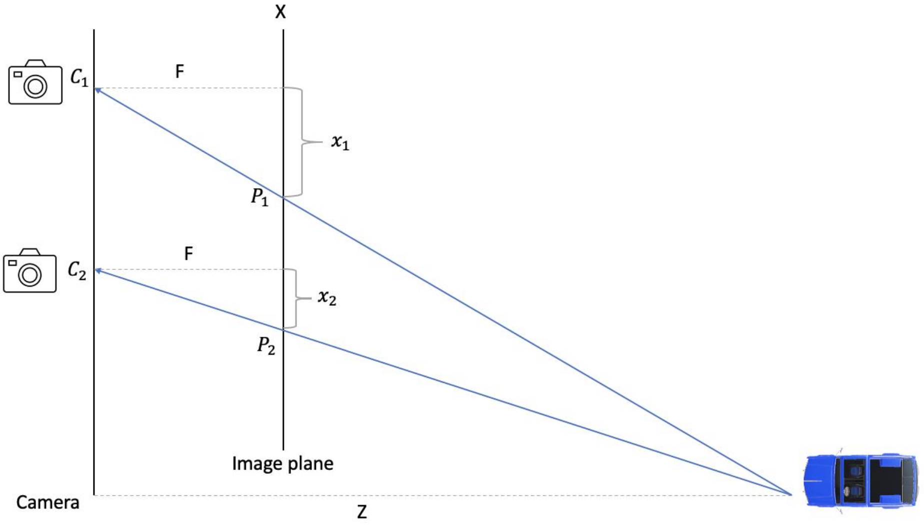


Figure 4.Geometry of stereo camera

To estimate the scale of the monocular camera, Knorr et al. [26] proposed a method to determine the scale by using the front cameras to track the user's faces and the back camera to track the reconstruction features. However, this method is difficult to use in autonomous driving systems as the driver environment is different from the road environment. Another method which focuses on autonomous driving systems is to assume the local planarity and the height of the camera between the ground is known [27]–[29]. Therefore, this method simplifies the process of scale estimation. Strasdat et al. [30] proposed the monocular SLAM method based on the keypoint featured which uses the Lie group and Lie algebra of similarity transformations to estimate the motion and map structure.

2.1.3. Visual Localization within Prior Point Cloud Map

Recently, researchers have introduced an integrated approach that combines LiDAR-SLAM and Visual-SLAM. This innovative method addresses the limitations of purely visual approaches, which lack direct range information due to the passive nature of cameras. By merging Visual-SLAM with a prior LiDAR map, this approach reduces the challenge of visual-based localization by incorporating accurate range measurements.

The structure-based method is the prevalent technique for visual localization, it relies on the 3D reconstruction point cloud maps to estimate the position [31]–[33]. This technique involves comparing local features such as the SIFT and ORB descriptor [34]. However, the features extracted from the image are susceptible to variations in illumination and seasonal changes, making them unsuitable for precise vehicle positioning in dynamic environments [32]. Moreover, the use of monocular cameras introduces scale-related issues that restrict their applications.

On the other hand, several researchers have proposed SLAM approaches integrated with machine learning methods [35]–[37], including end-to-end learning architectures, to mitigate these challenges [38], [39]. Nevertheless, these end-to-end approaches have demonstrated less stability compared to geometric and probabilistic methods. Another method is the image retrieval-based method that directly searches for relevant images from the map, extracting all information within regions of subtle gradients [40]. Consequently, they outperform structure-based methods in handling texture, motion blur, and image defocus. However, the real-time performance of such methods demands substantial computing power (such as GPUs).

Despite various methodologies primarily focusing on optical feature matching within the environment, fewer approaches emphasize the utilization of a 3D point cloud map extract geometric information for alignment with camera images. For instance, Wolcott et al. [41] propose a technique that employs a prior 3D point cloud map to generate synthetic 2D images through rendering and then matches them with real-time images using maximum normalized mutual information. Similarly, Pascoe et al. [42] present an approach that minimizes normalized information distance by using real-time camera images and rendered images from a combined LiDAR and camera map. These methods predominantly operate in a 2D space and involve GPU-based image rendering. In contrast, Caselitz et al. [42] introduced a method that eliminates the need for GPU-based image rendering. Instead, it directly aligns 3D geometries for improved accuracy and efficiency.

2.2. Scene representations

The scene used in SLAM is reconstructed by the sensor which is essential for mapping, localization, visualization or planning. Different purposes of SLAM use different types of maps. The most common types of maps: (a) voxel maps[43] ; (b) point cloud maps[44]; (c) feature point maps[45].

2.2.1. Voxel maps

The voxel maps divide the environment into several 3D volume cell which is voxels. The voxel is similar to a 2D pixel, but it provides occupancy, colour, or other attributes of the environment's 3D structure [46]. Truncated Signed Distance Function (TSDF) is popularly used in the direct SLAM method which focuses on generating a detailed and accurate 3D map [47], [48]. Those researchers proposed a method which can systematically regularize noise and model continuous surfaces. Occupancy maps are popularly used in the navigation task which is mainly used for basic navigation and obstacle avoidance. Occupancy maps are used to represent the probability of each cell being occupied by an obstacle or object. GMapping [49] and FastSLAM [50] proposed the method for each grid of cells which includes the information of occupancy maps and the trajectory of the robot. Hornung et al. [51] proposed a method to decrease the memory requirement of saving cells information when using occupancy maps. To capture the continuous distance information to obstacles, Euclidean Signed Distance Function (ESDF) is used to present the occupied obstacles and the distances information [52]. Lau et al. [53] proposed the method to use ESDF with occupancy maps which make use of the sensor data to provide a piece of local information. This method has better performance than ESDF construction strategies [52].

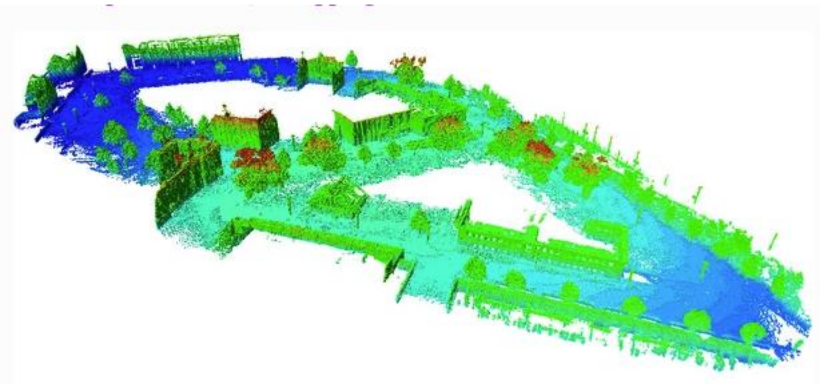


Figure 5.OctoMap generated by occupancy maps [51].

2.2.2. Point cloud maps

A point cloud can be generated by the LiDAR and RGB-D camera and combine different point clouds to provide a map. The point cloud in the point cloud map is individual. Moosmann and Stille [54] proposed a method which uses LiDAR to generate the point cloud map for estimating the vehicle trajectory. Most of the SLAM with point cloud map use Unscented Kalman Filters [55], Rao-blackwellized Particle Filter [56], Extended Kalgnman Filters [57], or Sparse Extended Information Filters [58] to process the scan-matching. However, those methods are difficult to match when a larger number of landmarks. Holz et al. [59] proposed a scan-matching method to match point cloud data with the point cloud map which is a fast and accurate method.

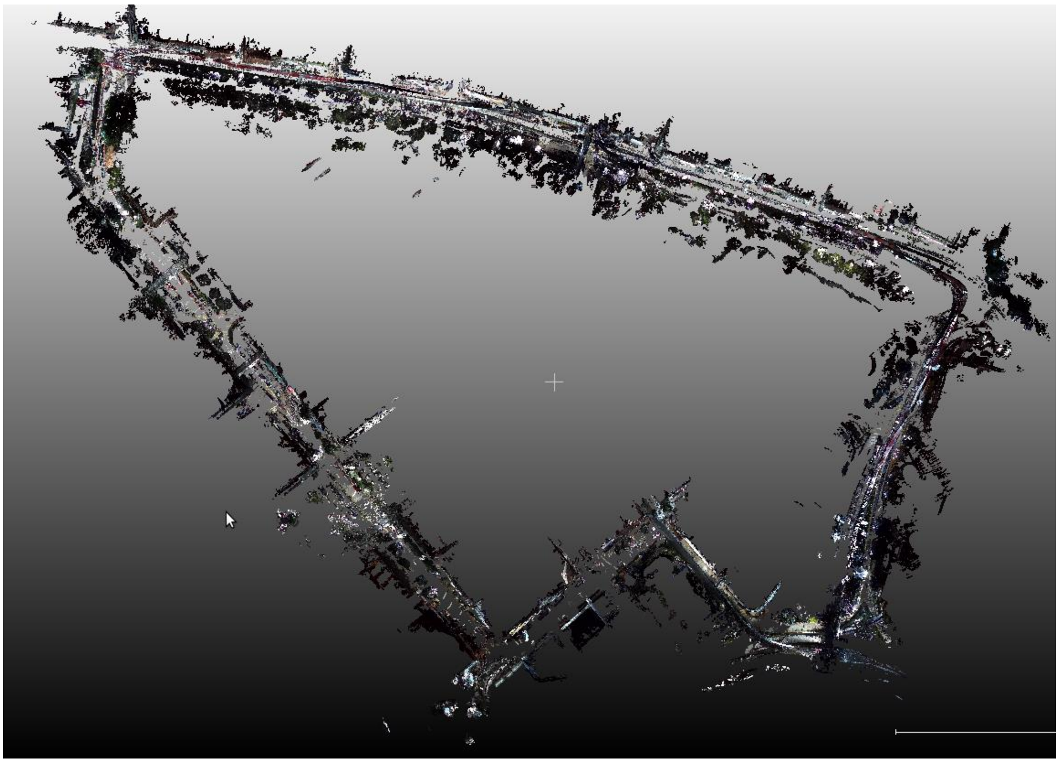


Figure 6. Point cloud map of Hong Kong environment (UrbanNav dataset [60])

2.2.3. Feature point maps

A feature point map can be generated by the feature point from the camera image and the camera poses. In the map, it includes the feature point with the relative pixel locations and the index of camera pose. Therefore, this map is popular with the feature point-based SLAM methods as the map provides a fast approach for re-localization and bundle adjustment. Klein and Murray [61] proposed a method of camera tracking system based on a feature point map which is PTAM. Mur-Artal et al. [45] proposed a method of combining the direct and feature-based methods based on a feature point map which is ORB-SLAM. Qin et al. [62] proposed a method of combining the Visual system and inertial systems in real-time which estimates the results with a feature point map which is VINS-Mono.

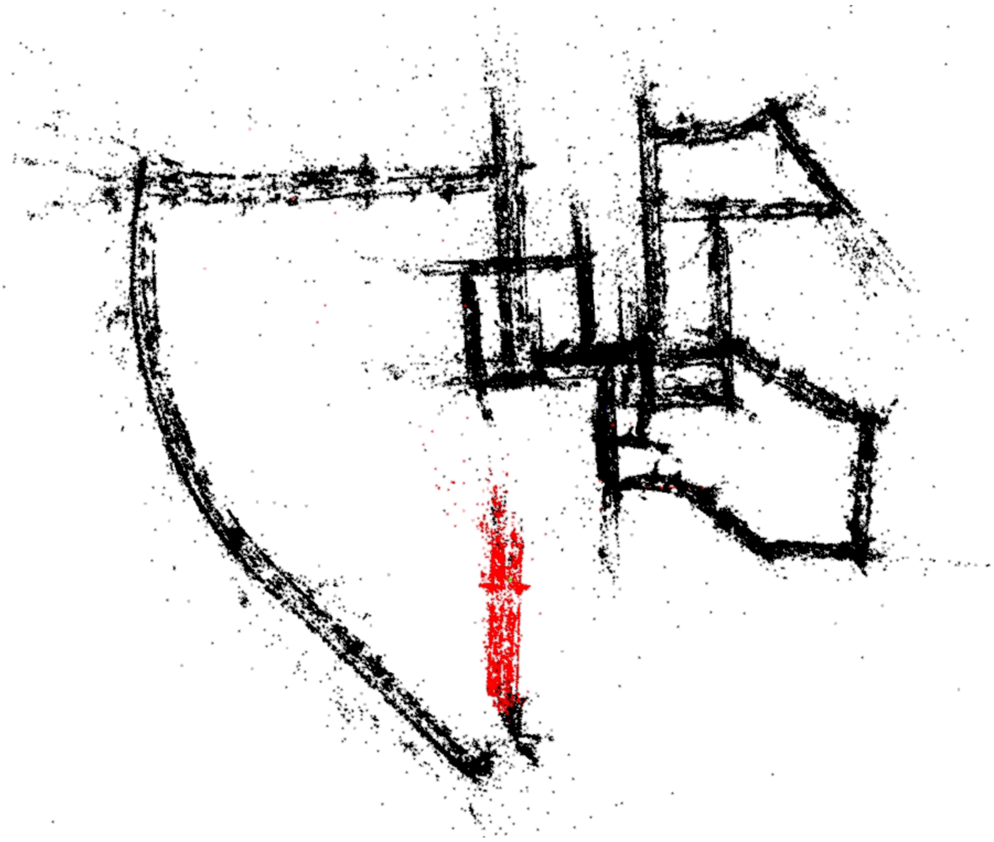


Figure 7. Feature point map of sequence 00 in KITTI odometry dataset [63] constructed by ORB-SLAM 3 [45]

2.3. LiDAR distortion

LiDAR is a highly potential sensor for environmental perception, especially for positioning and mapping. LiDAR directly provides precise distance measurements with 3D information and a long detection range. LiDAR is unaffected by visual feature limitations, including scenarios with low light conditions. This technology excels in generating precise maps and identifying obstacles within the environment. The LiDAR scanning procedure typically covers a 360° range by capturing data from various angles, resulting in what is referred to as a point cloud.

Nevertheless, LiDAR encounters a distortion issue when its movement coincides with the initiation of the scanning process, as depicted in Figure 8. If the map is used LiDAR for mapping, it is impact the map's accuracy [64]. Then, the localisation difficulty is increase when using the map basic on LiDAR data [65]. It is independent of the number of LiDAR [66].

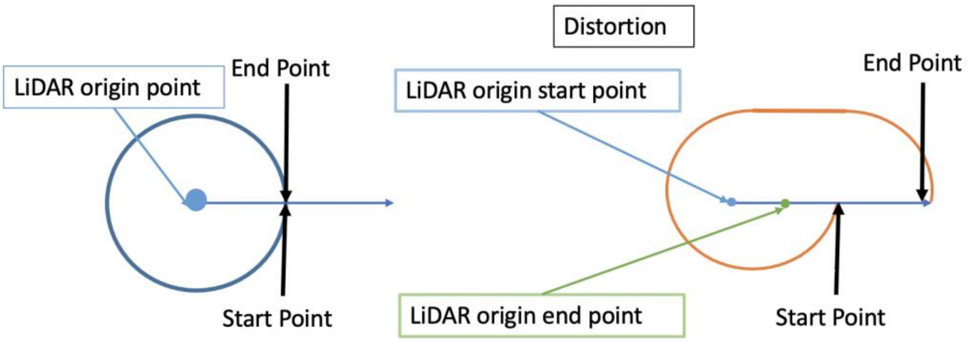


Figure 8. Illustration of the LiDAR distortion phenomenon

The distortion observed in LiDAR point clouds is commonly classified into two distinct categories which are ego-motion distortion and object-motion distortion. This research primarily concentrates on addressing ego-motion distortion.

Within the SLAM-based approaches, LOAM [64] and NDT-LOAM [67] stands as an exemplar, adeptly achieving efficient and precise scan matching for odometry and mapping. Another technique employs to correct the point cloud is the iterative closest point (ICP) and Normal Distribution Transform (NDT) methodology, which aligns consecutive scans to refine point positions [68]. However, this approach is susceptible to the influence of moving objects in the environment, such as vehicles [69]. To rectify this issue, certain correction methods leverage information from IMU or odometry measurements to adjust point positions. Nonetheless, it is crucial to note that IMU or odometry data introduce the challenge of cumulative error accumulation [70]. Therefore, Byun et al. [71] use GNSS/INS unit to provide highly accurate information about the vehicle position. To correct the distortion problem, which is shown in Figure 9, the corrected position can be calculated as Equ.2.

$$\mathbf{p}^{\mathrm{i}}_{\mathrm{t}+\Delta\mathrm{t}} = \mathbf{R}(\mathbf{p}^{\mathrm{i}}_{\mathrm{t}} - \mathbf{T}) \,, \tag{2}$$

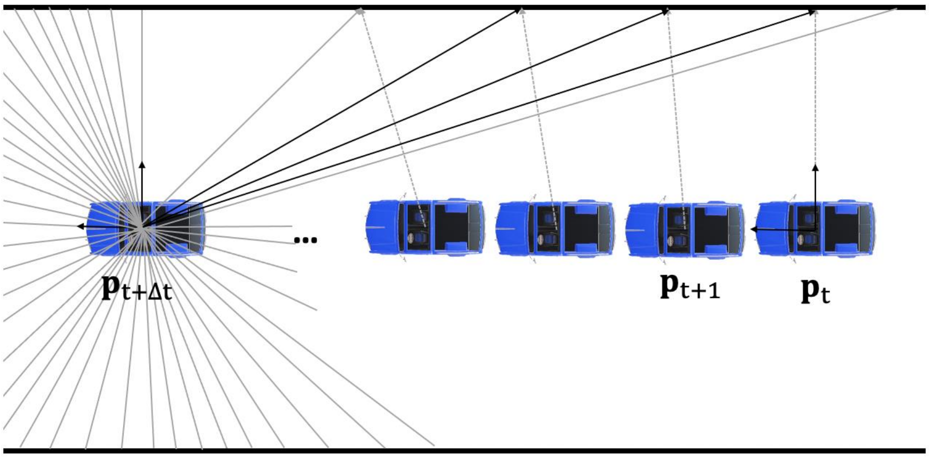
i : One revolution of LiDAR

$\mathbf{p}_{\mathrm{t}}$: Start position of the revolution

$\mathbf{p}_{\mathrm{t}+\Delta\mathrm{t}}$: End position of the revolution

$\mathbf{R}$: Rotation between $\mathbf{p}_{\mathrm{t}}$ and $\mathbf{p}_{\mathrm{t}+\Delta\mathrm{t}}$

$\mathbf{T}$: Translation between $\mathbf{p}_{\mathrm{t}}$ and $\mathbf{p}_{\mathrm{t}+\Delta\mathrm{t}}$



...
$\mathbf{p}_{\mathrm{t}+\Delta\mathrm{t}}$
$\mathbf{p}_{\mathrm{t}+1}$
$\mathbf{p}_{\mathrm{t}}$

Figure 9. One revolution of LiDAR with vehicle

2.4. Summary

To conclude, this paper proposed a method of visual SLAM which estimates the part of visual localization within a prior point cloud map. This method estimates the vehicle post by matching the features point map and point cloud map. The point cloud map is developed by the corrected point cloud which is corrected by the ground truth data of the GNSS/INS unit. The feature point map is developed by the monocular camera. For the scale issues of monocular cameras, this paper estimates the alignment by matching geometric between the features point map and point cloud map. The details of methodologies are introduced in Chapter 3.

3. METHODOLOGY

The proposed framework of this paper shows in Figure 10. This paper proposes a method that incorporates a dependable initialization process facilitated by a prior 3D point cloud map. It is an economical camera-based localization solution which inspiration from the methodology outlined by Caselitz et al [72]. This paper focuses on urban scenarios to unravel scientific challenges which aim to utilize a monocular camera for localization within a prior 3D point cloud map.

Initially, the approach involves utilizing visual features to reconstruct the local environment, employing sparse yet meaningful 3D feature points to form what is referred to as a local points map (LPM). The relative motion estimation provided by the visual odometers generates a reliable initial estimation concurrently. Subsequently, building upon the initial pose estimation and the generated local points map, the approach employs the ICP-based point cloud registration method [4] to align the LPM with the prior 3D point cloud map. This process effectively determines the system's pose within the map.

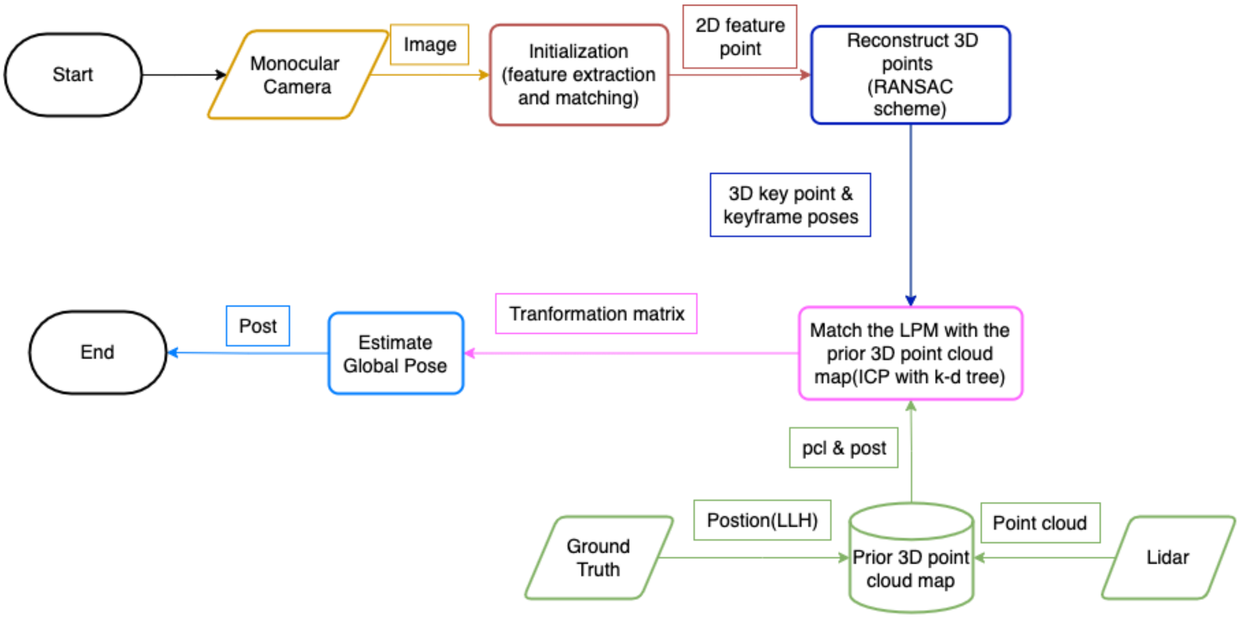

Start
Monocular Camera
Image
Initialization (feature extraction and matching)
2D feature point
Reconstruct 3D points (RANSAC scheme)
3D key point & keyframe poses
Match the LPM with the prior 3D point cloud map(ICP with k-d tree)
Tranformation matrix
Estimate Global Pose
Post
End
pcl & post
Ground Truth
Postion(LLH)
Prior 3D point cloud map
Point cloud
Lidar

Figure 10. System flow chart

3.1. Prior 3D point cloud map

LiDAR (3D point cloud data) and ground truth information (GNSS position data) are used to generate the 3D prior point cloud map. The prior point cloud combined with several point cloud messages accompanied by a camera pose. However, in the LiDAR sensor scanning procedure, the LiDAR data grapples with a distortion issue. Conventionally, LiDAR scans one revolution (360°) from points situated around its centre. When a vehicle equipped with a LiDAR sensor is stationary, the coordinates of the LiDAR's origin remain constant. As a result, the initiation and termination points of LiDAR scanning align. If the vehicle is in motion while the LiDAR sensor is operating, distortion occurs due to the extended scanning period resulting from the motion displacement of the LiDAR's origin. Figure 11 elucidates the reason for LiDAR distortion.

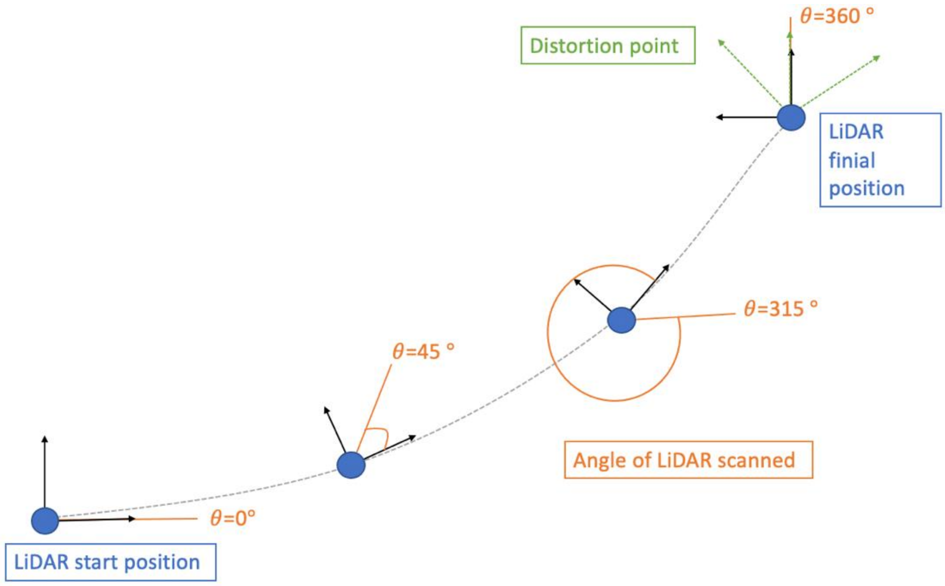


Figure 11. LiDAR distortions arising from vehicle motion

The ground truth data from GNSS can be used to correct the distortion issue of point cloud data from LiDAR. The ground truth data includes Latitude, Longitude, and Height (LLH) positions, along with quaternions. LLH values establish the Local East-North-Up (ENU) coordinate system for localized processing, while quaternions facilitate the determination of rotations between various coordinate systems.

$$\mathbf{D} = \left\{t_{D_n}, \mathbf{p}_{\mathbf{D_n}}, \boldsymbol{q}_{\boldsymbol{D_n}}, n = 1, \ldots, k\right\}, \tag{3}$$

$$\mathbf{G} = \begin{Bmatrix} t_{G_1} & \cdots & t_{G_m} \\ \mathbf{p}_{\mathbf{G_1}} & \cdots & \mathbf{p}_{\mathbf{G_m}} \\ \mathbf{q}_{\mathbf{G_1}} & \cdots & \mathbf{q}_{\mathbf{G_m}} \end{Bmatrix}, \tag{4}$$

$$\mathbf{P} = \left\{t_{P_n}, \mathbf{p}_{\mathbf{P_n}}, \mathbf{q}_{\mathbf{P_n}}, n = 1, \ldots, k\right\}, \tag{5}$$

D : A set of point cloud data afflicted by distortion within a single frame

G : A set of ground truth data (Time, LLH position and quaternions)

P : A set of corrected point cloud data in one frame

t : Message time

p : Coordinated in the ENU coordinate system (LLH position)

q : Quaternion

Interpolation is used to determine the precise position of each point cloud data. For the ENU position, linear interpolation is used. For the quaternion, spherical linear interpolation is used.

$$\frac{t_{D_n}-t_{G_i}}{t_{G_j}-t_{G_i}} = \frac{\mathbf{p}_{\mathbf{P_n}}-\mathbf{p}_{\mathbf{G_i}}}{\mathbf{p}_{\mathbf{G_j}}-\mathbf{p}_{\mathbf{G_i}}}\ , \tag{6}$$

$$\mathbf{q}_{\mathbf{P_n}}=\frac{\sin{(1-r)\theta}}{\sin{(\theta)}}\mathbf{q}_{\mathbf{G_i}} + \frac{\sin{(r\theta)}}{\sin{(\theta)}}\mathbf{q}_{\mathbf{G_j}}, \tag{7}$$

i,*j* : The near ground truth message time with t_{D_n}

r : Interpolation coefficient

θ : Angle between q_{G_i} and q_{G_j}

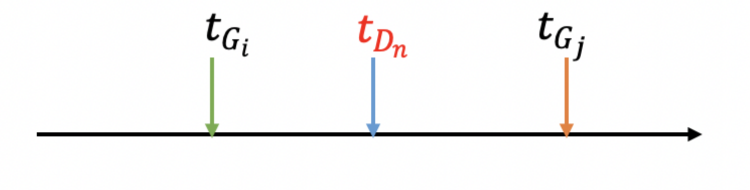


Figure 12. Sequence of selected data points

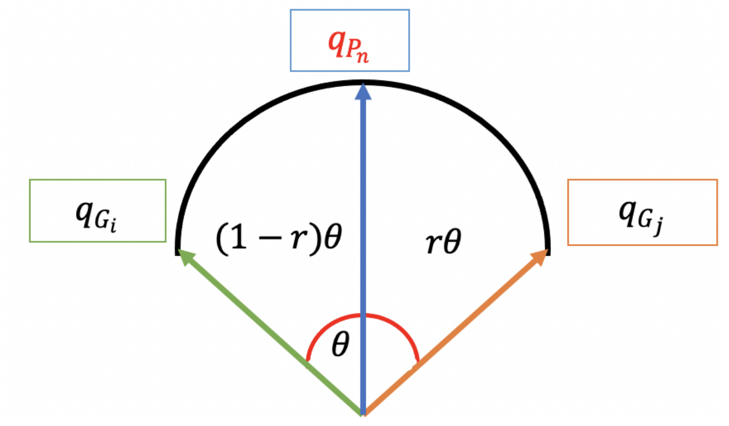


Figure 13. Spherical linear interpolation

3.2. Feature Matching

This procedure employs the ORB algorithm, which integrates the Oriented Features from the Accelerated and Segments Test (FAST) algorithm and the Rotated Binary Robust Independent Elementary Features (BRIEF) algorithm [73] . The oriented FAST algorithm is employed for the task of feature extraction.

To extract the feature points, the FAST algorithm compares pixel intensities with those of the pixel's surroundings. However, this approach lacks orientation and multi-scale consideration. To address this, the ORB algorithm incorporates an image pyramid along with a Harris corner detector, enabling the detection of key points across various scales. To ascertain orientation, the algorithm assumes that corner intensity is displaced from the centre, and orientation is deduced via image moment calculations [73].

To enhance the original BRIEF algorithm [74] , the ORB algorithm proposed the Rotated BRIEF algorithm which accounts for feature point rotation. It is a crucial factor in image matching. The descriptors for each key point detected through the oriented FAST algorithm calculates by the binary feature vectors. For smoothing image patches, a 7x7 Gaussian kernel is employed, and the feature vector ($\mathbf{f_n(B)}$) for each patch is defined by n binary tests.

$$\tau(B;x,y) = \begin{cases} 1, \; B(x)_l < \; B(y)_l \\ 0, \; B(x)_l \geq \; B(y)_l \end{cases}, \tag{8}$$

$$\mathbf{f}_{\boldsymbol{n}}(\boldsymbol{B}) = \sum_{1<i<n} 2^{i-1}\tau(B;\mathrm{x}_i, y_i), \tag{9}$$

τ : Binary test
n : Vector length
$B(x)_l$: Intensity of pixel x in patch B

Nevertheless, the BRIEF algorithm does not account for the orientation of the feature point. To overcome this constraint, the Rotated BRIEF algorithm was developed. This enhanced version integrates orientation information by incorporating sine and cosine values that are multiplied with a set of feature points, which are then rotated based on the key point's orientation.

$$\mathbf{S} = \begin{pmatrix} u_1 & \dots & u_n \\ v_1 & \dots & v_n \end{pmatrix}, \tag{10}$$

$$\mathbf{R}_{\boldsymbol{\theta}} = \begin{bmatrix} cos\,\theta & -sin\,\theta \\ sin\,\theta & cos\,\theta \end{bmatrix}, \tag{11}$$

$$\mathbf{S}_{\boldsymbol{\theta}} = \mathbf{R}_{\boldsymbol{\theta}}\mathbf{S}, \tag{12}$$

$$\mathbf{g}_{\boldsymbol{n}}(\boldsymbol{p},\boldsymbol{\theta}) := \mathbf{f}_{\boldsymbol{n}}(\boldsymbol{p})|(\mathrm{x}_i, y_i) \in \mathbf{S}_{\boldsymbol{\theta}}, \tag{13}$$

θ : Key point orientation from oriented FAST algorithm
S : Feature set of n binary tests at location (u_i, v_i)

R_{θ} : Corresponding rotation matrix
$\boldsymbol{S_{\theta}}$: A set of feature points rotated according to the key point orientation

The descriptors of key points in two consecutive images are matched using the Brute-Force Matcher with Hamming distance metric when the feature descriptors are computed. This metric estimates the closest distance between descriptors. A cross-check is implemented to enhance accuracy.

$$D(b_1, b_2) = b_1 \oplus b_2, \tag{14}$$

D : Hamming distance
b1, b2 : Feature descriptors of two different image

3.3. Local Points Map Reconstruction

This module proceeds to convert the feature points from the feature-matching process into corresponding 3D map points and keyframe poses when the matched 2D feature points are acquired. This collection of localized data is referred to as a local point map, and it serves as the foundation for solving the local Bundle Adjustment problem.

The initial step involves calculating the fundamental matrix (F) and the homography matrix (H) using an eight-point algorithm. In order to enhance the stability and precision of the solution, the module normalizes the coordinates of the input point set.

$$\mathbf{p_1}^{\mathbf{T}}\mathbf{F}\mathbf{p_2} = 0, \tag{15}$$

$\mathbf{p_1}, \mathbf{p_2}$: A pair of matching feature points in two Frames
F : Fundamental matrix

Given n sets of linear equations provided by n pairs of matching points, the following equations is obtained:

$$\mathbf{Wf} = \begin{bmatrix} u_1{}'u_1 & u_1{}'v_1 & u_1{}' & v_1{}'u_1 & v_1{}'v_1 & v_1{}' & u_1 & v_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_n{}'u_n & u_n{}'v_n & u_n{}' & v_n{}'u_n & v_n{}'v_n & v_n{}' & u_n & v_n & 1 \end{bmatrix} \mathbf{f} = 0 \qquad (16)$$

W : Matrix of n pairs of matching points (N x 9)

As the **W** matrix has the least squares problem, it recovers rotation and translation by singular value decomposition (SVD) to solve the least squares problem.

The least squares solution is obtained:

$$\begin{cases} \min\limits_{\mathbf{f}} ||\mathbf{Wf}||^2 \\ \mathrm{s.t.}\ ||\mathbf{f}|| = 1 \end{cases} \qquad (17)$$

Singular value decomposition ($\mathbf{W} = \mathbf{UDV}^{\boldsymbol{T}}$) is performed on **W** matrix .The entries of **F** matrix are the components of the column of **V** corresponding to the least singular vector. As the fundamental matrix has the constraint of rank 2, the **F** matrix must be singular.

The least squares problem is obtained:

$$\begin{cases} \min\limits_{\mathbf{f}} ||\mathbf{F} - \overline{\boldsymbol{F}}||^2 \\ \mathrm{s.t.}\, \mathrm{def}(\mathbf{F}) = 0 \end{cases} \qquad (18)$$

Therefore, the fundamental matrix is obtained:

$$\mathbf{F} = \mathbf{H}'^{\mathbf{T}}\bar{\mathbf{F}}\mathbf{H} \qquad (19)$$

$\mathbf{H}$: Transformation matrix of $\mathbf{p_1}$ and $\mathbf{p_2}$.

$\mathbf{H}'$: Transformation matrix of $\overline{\mathbf{p_1}}$ and $\overline{\mathbf{p_2}}$.

Subsequently, the module evaluates the results of random sample consensus (RANSAC) using the concept of reprojection error. It assumes that reprojecting points from the current frame onto the reference frame generates a straight line (l_2).

$$l_2 = \boldsymbol{F}^{-1}\begin{bmatrix} u_2 \\ v_2 \\ 1 \end{bmatrix}, \qquad (20)$$

$\begin{bmatrix} u_2 \\ v_2 \\ 1 \end{bmatrix}$: Coordinate of the current frame

However, it exists a projection error in which the point did not lie precisely on this line. The projection error estimate by the cumulatively RANSAC score in conjunction with the current matrix.

Simultaneously, the module computes the homography matrix (H) following a similar procedure as the fundamental matrix (F). Then, the score ratio between these two matrices is computed to determine which model to select.

The chosen matrix can recover the rotation ($\mathbf{R}$) and translation matrix ($\mathbf{t}$) of camera motion which is the essential matrix. The essential matrix ($\mathbf{E}$) is obtained:

$$\mathbf{E} = [\mathbf{R}|\mathbf{t}] = \begin{cases} \mathbf{R_1{=}UWV^T} \\ \mathbf{R_2{=}UW^TV^T} \\ \mathbf{t_1{=}ER^T} \\ \mathbf{t_2{=}{-}ER^T} \end{cases} \qquad (21)$$

U : Gene coefficient vectors from SVD

W : Rotation matrix obtained by rotating 90° along the Z axis

$\mathbf{V^T}$: Expression level vectors from SVD

Finally, the module employs a triangulation algorithm with SVD to calculate the 3D point based on the recovered essential matrix. The 3D coordinates calculate by the SVD of A matrix. The 3D point is obtained:

$$Ax = \begin{bmatrix} v_1 P_{12} - P_{11} \\ P_{10} - u_1 P_{11} \\ v_2 P_{22} - P_{21} \\ P_{20} - u_2 P_{21} \end{bmatrix} x = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{22}$$

u_1, v_1, u_2, v_2: Feature point of the first frame and second frame, respectively

P_1, P_2: Projection matrix of the first frame and second frame, respectively

3.4. Localization in Prior Map

3.4.1. Correspondences

In this process, the variables are described using both 3D Euclidean spaces and the Lie group SE(3).

$$\mathbf{K} = \{\mathbf{k}_i \in \mathbb{R}^3, i = 1, \dots, n\}, \tag{23}$$

$$\mathbf{P} = \{\mathbf{p}_j \in \mathbb{R}^3, j = 1, \dots, m\}, \tag{24}$$

$$\mathbf{T} = \{\mathbf{T}_i \in \mathrm{SE}(3), i = 1, \dots, t\}, \tag{25}$$

$$\mathbf{F} = \{\mathbf{F}_i \in \mathrm{SE}(3), i = 1, \dots, f\}, \tag{26}$$

K : A set of reconstruction 3D points
P : A set of 3D points in the prior 3D point cloud map
T : A set of transformations between **K** and **P**
F : A set of keyframe poses

This approach uses the ICP algorithm in conjunction with KD-tree for conducting nearest neighbour searches to establish correspondences between the reconstructed local points and the points within the 3D prior point cloud map. This approach employs the ICP algorithm in conjunction with KD-tree for conducting nearest neighbour searches. The process involves utilizing both the local points map and the prior point cloud map to identify corresponding points. These correspondences are iteratively refined through the estimation of transformations between the two point clouds.

The prior point cloud map is constructed into a KD-tree structure, which is subsequently employed to locate the nearest neighbour of each point within a predefined search radius. This approach facilitates the establishment of accurate point correspondences.

$$\mathbf{K}_j = argmin\left|\left|\mathbf{P}_{\,k} - \ \mathbf{K}_i\right|\right|, for\ i = 1, \ldots, M, k = 1, \ldots, N, \tag{27}$$

$\mathbf{K}_j$: Closest neighbour point of $\mathbf{P}_{\,k}$ in K
N : Number of points in P
M : Number of points in K

The set of correspondences of prior point cloud map and reconstructed 3D point cloud consists of pairs (i, j), where each pair represents the correspondence between point i in the prior point cloud map and point j in the reconstructed 3D point cloud.

$$\mathbf{C} = \{ (i,j) | \, i \in \mathbf{R}, j \in \mathbf{P}\}, \tag{28}$$

3.4.2. Alignment

To estimate the alignment, this process calculates by a set of correspondences. The alignment process involves estimating the alignment between the locally reconstructed point map and the prior 3D point cloud map This paper solves a non-linear least squares minimization problem with the assistance of the g2o optimization framework [11] to determine the transformation between the local point clouds and the prior point clouds.

Firstly, this paper estimates the transformation between the two 3D point clouds by g2o. This paper uses Sim3 transformation combines both rigid-body transformation and scaling which shows in equation 29. Also, it uses exponential map parametrization which maps a point from a Lie algebra to an element of the Lie group [75] which is used to parametrize rotations and other transformations to ensure they remain within the valid parameter space [11]. It shows in equation 31.

$$\mathbf{T} = [s\mathbf{R} \mid \mathbf{t}] \in sim(3), \mathbf{R} \in \mathrm{SO}(3), \mathbf{t} \in \mathbb{R}^3, s \in \mathbb{R}, \tag{29}$$

T : Transformation matrix (4x4)
r : Rotation matrix (3x3)
t : Translation vector (3x1)
s : Scaling factor (scalar)

The element of sim(3) is obtained:

$$sim(3) \in \begin{bmatrix} u \\ \omega \\ \lambda \end{bmatrix} \in \mathbb{R}^7, s = e^{\sigma} \tag{30}$$

The exponential map of Sim (3) (Lie group) and sim(3) (Lie algebra) is obtained:

$$exp_{Sim(3)} \begin{bmatrix} u \\ \omega \\ \lambda \end{bmatrix} = \ [s\mathbf{R} \mid \mathbf{t}] = \left[e^{\sigma} exp_{So(3)} \mid \mathbf{W}\lambda\right], \tag{31}$$

$$\mathbf{W} = \frac{1-\exp(-\lambda)}{\lambda}\mathbf{I} + (\alpha \cdot (\beta - \gamma) + \gamma)\ \mathbf{w}_{\boldsymbol{x}} + (\alpha \cdot (\rho - \upsilon) + \upsilon)\mathbf{w}_{\boldsymbol{x}}{}^{\mathbf{2}}, \tag{32}$$

$$\alpha = \frac{\lambda^2}{\lambda^2 + (\omega^T\omega)}, \tag{33}$$

$$\beta = \frac{\exp(-\lambda) - 1 + \lambda}{\lambda^2}, \tag{34}$$

$$\gamma = \frac{1-\cos\left(\omega^T\omega\right)}{(\omega^T\omega)^2} - \lambda\left(\frac{1-\left(\frac{\sin\left(\omega^T\omega\right)}{\omega^T\omega}\right)}{(\omega^T\omega)^2}\right), \tag{35}$$

$$\rho = \frac{1-\lambda+0.5\lambda^2-\exp(-\lambda)}{\lambda^2}, \tag{36}$$

$$\upsilon = \frac{1-\left(\frac{\sin\left(\omega^T\omega\right)}{\omega^T\omega}\right)}{\omega^T\omega^2} - \ \lambda\left(\frac{0.5-\frac{1-\cos\left(\omega^T\omega\right)}{(\omega^T\omega)^2}}{(\omega^T\omega)^2}\right), \tag{37}$$

W : Different power series of translation component

$\mathbf{w}_{\boldsymbol{x}}$: Skew-symmetric matrix and the axis of rotation w

Second, this paper optimization the process to optimal values of T that aiming to minimize the aggregate of squared errors between the transformed points within the prior 3D point cloud map and their corresponding counterparts within the reconstructed local point map. The cost function of the optimization problem can be written as,

$$minimize\ \sum w_i * \|\mathbf{T} * \mathbf{P}_i - \mathbf{R}_i\|^2, \tag{38}$$

w_i : Weight assigned to each correspondence

Then, it utilizes Levenberg-Marquardt algorithm [12] iteratively update the variables based on the constraints and the associated cost functions to estimate the transformation matrix (T) until the sum of squared errors reaches its minimum. In each iteration, the algorithm calculates the Jacobian matrix (J) of the objective function in relation to the parameters of T, utilizing this information to determine the necessary update to the transformation matrix T:

$$\mathbf{T}_k = \mathbf{T}_{k-1} + (\mathbf{J}^T\mathbf{J} + \lambda\mathbf{I})^{-1}\mathbf{J}^T\mathbf{r}, for\ k = 1 \dots . N\ , \tag{39}$$

$\mathbf{J}$: Jacobian matrix of the error function with respect to T
$\mathbf{r}$: Residual vector
λ : Damping parameter
$\mathbf{I}$: Identity matrix
N : Number of iteratively

By minimizing the cost function, the optimal values for the relative transformation between the two point clouds are deduced. Subsequently, amalgamating all the similarity transformations during the iterative process generates the estimation of the reference frame's pose within the map. Finally, the global pose of the vehicle is estimated.

4. EXPERIMENT

The experimentation encompasses two distinct datasets to assess the system's performance across varying environments. Throughout this evaluation, we compare the effectiveness of the following pipelines:

1) Assessing the precision of our proposed approach by using the 3D local points from ORB-SLAM method.

ORB-SLAM [45]: Employing the ORB approach to extract features and descriptors followed by the calculation of 3D local points.

2) Employing diverse datasets to assess our proposed method of evaluation involved comparing the estimated camera trajectory and the ground truth trajectory against the existing map.

KITTI odometry dataset [63]: Presents outdoor data from rural regions captured by a vehicle.

UrbanNav dataset [60]: Offers outdoor data from urban settings captured via a vehicle.

4.1. Experiment Setup

4.1.1. KITTI odometry dataset

This paper uses the KITTI odometry dataset which data was collected around Karlsruhe, Germany to evaluate the performance of the proposed method. In this study, the construction of the 3D point cloud map was achieved utilizing the OXTS RT3003 inertial and GPS navigation system (100 Hz), the Velodyne HDL-64E rotating 3D laser scanner, and the PointGray Flea2 grayscale cameras. These components facilitated the acquisition of ground truth, 3D LiDAR data, and camera images, respectively. The complete experimental setup is depicted in Figure 14.

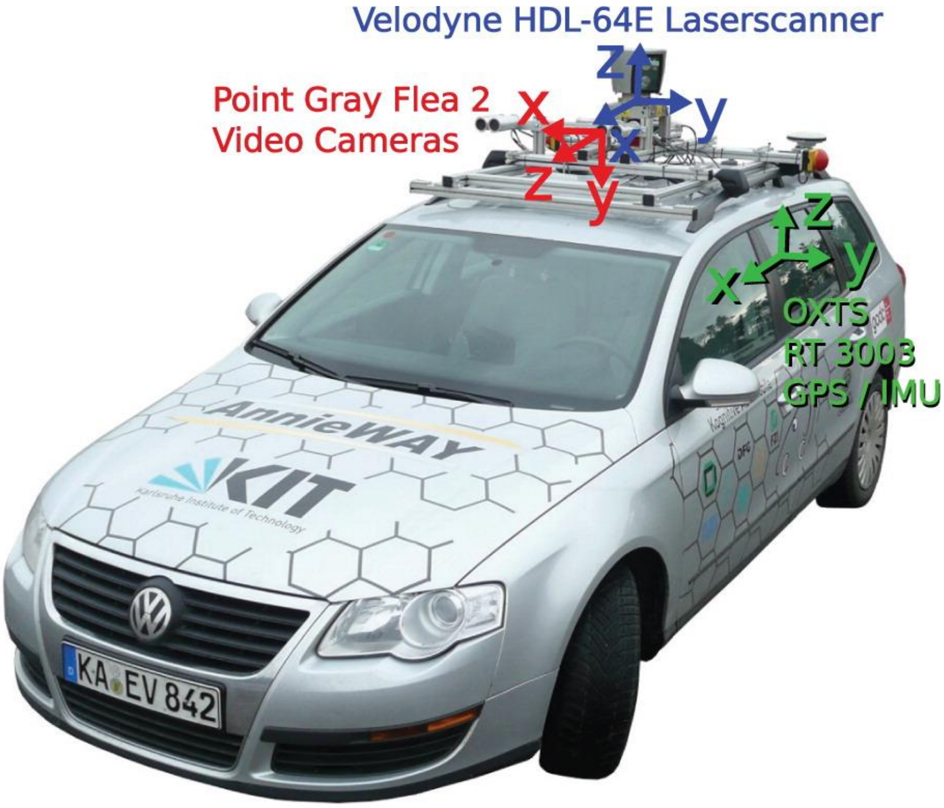


Figure 14. Experiment setup of KITTI odometry dataset [63]

4.1.2. UrbanNav dataset

The performance evaluation of the proposed method was conducted using the UrbanNav dataset, which encompasses data collected within Hong Kong, representative of a typical urban canyon environment. The construction of the 3D point cloud map in this study was facilitated by employing the NovAtel SPAN-CPT + Inertial Explorer (IE) (1 Hz), HDL 32E Velodyne (10 Hz), and ZED2 Stereo (15

Hz) systems. These systems provided ground truth, 3D LiDAR data, and camera images, respectively. During the data collection process, raw GPS measurements were gathered using a commercial-grade u-blox F9P GNSS receiver (1 Hz). A comprehensive depiction of the experimental setup is illustrated in Figure 15.

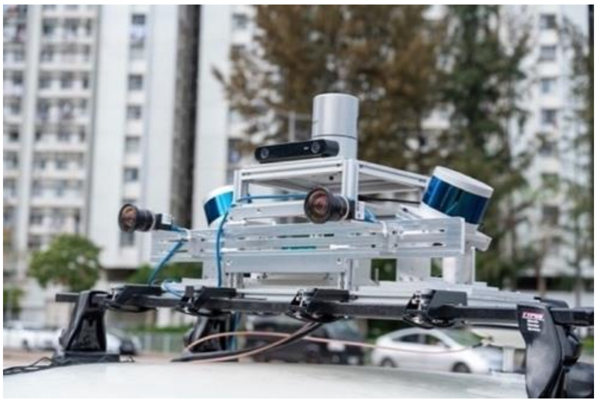


Figure 15. Experiment setup of UrbanNav dataset

Image data for this study was acquired utilizing the ZED2 Stereo camera, which was affixed to the vehicle. The calibration of the cameras was executed through the MATLAB method. To ensure coherence, all the gathered data underwent collection and synchronization via a robot operating system (ROS) [76].

4.2. Experimental Evaluation

4.2.1. Prior 3D point cloud map

The approach employed in this paper involves utilizing the prior 3D point cloud map for alignment with the 3D feature points. It's important to note that the LiDAR data introduces a distortion issue. This phenomenon is illustrated in Figure 16, which illustrates the prior 3D point cloud map exhibiting distortion, as observed in the UrbanNav dataset.

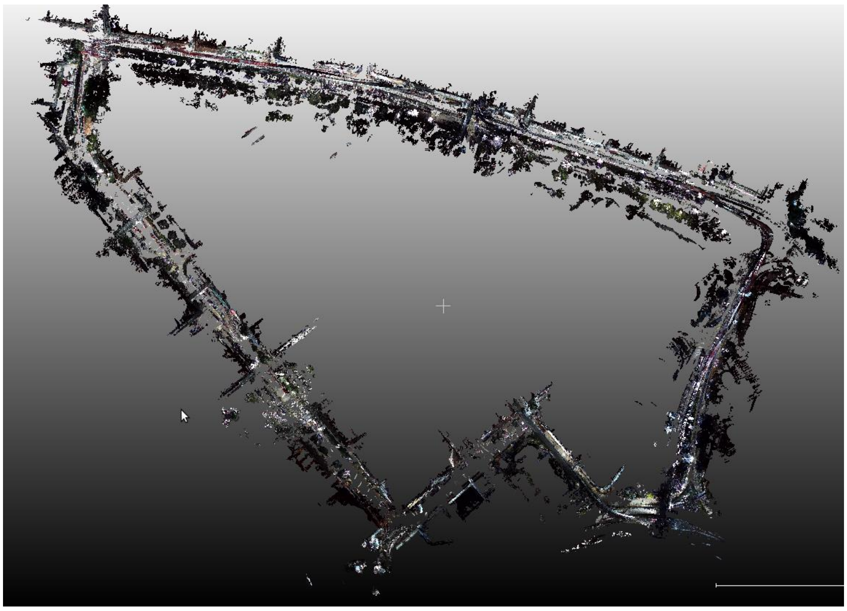


Figure 16. Illustrates the prior 3D point cloud map exhibiting distortion, as observed in the UrbanNav dataset

Figure 17 and Figure 18 shows the point cloud when the vehicle is at the starting location. The original point cloud is displayed as white. The corrected point cloud is displayed as brown. It shows that those points are matched with each other due to without vehicle's angular or longitudinal movements. Figure 18 shows the point cloud with the timestamp. The start point of the LiDAR scan is displayed as blue. The last point of the LiDAR scan is displayed as red.

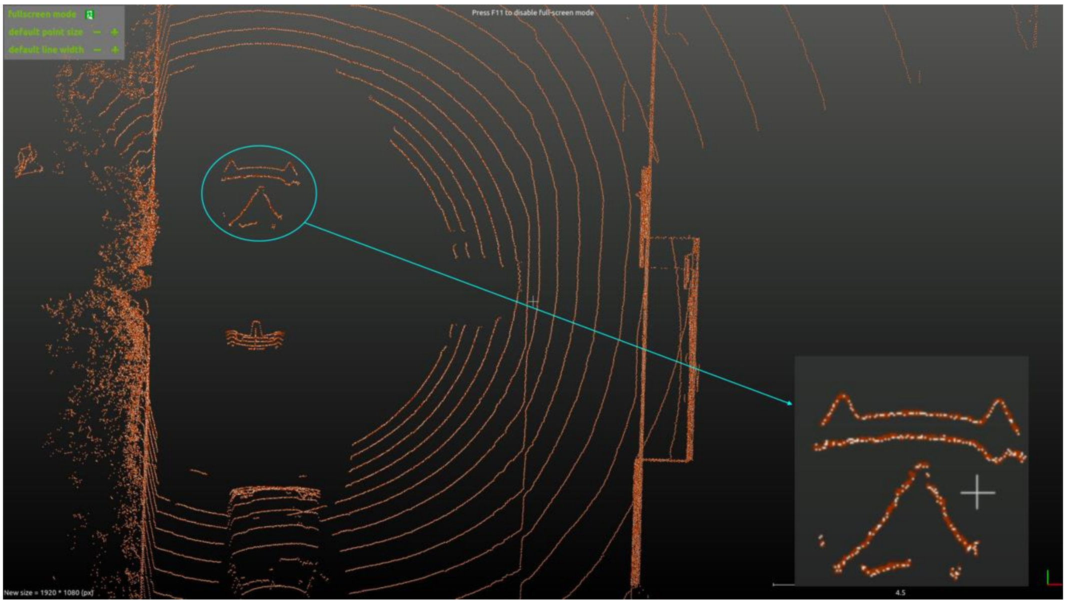

fullscreen mode
default point size
default line width
Press F11 to disable full-screen mode
New size = 1920 * 1080 (px)
4.5

Figure 17. Overall view: Original point cloud (white) and corrected point cloud (brown).

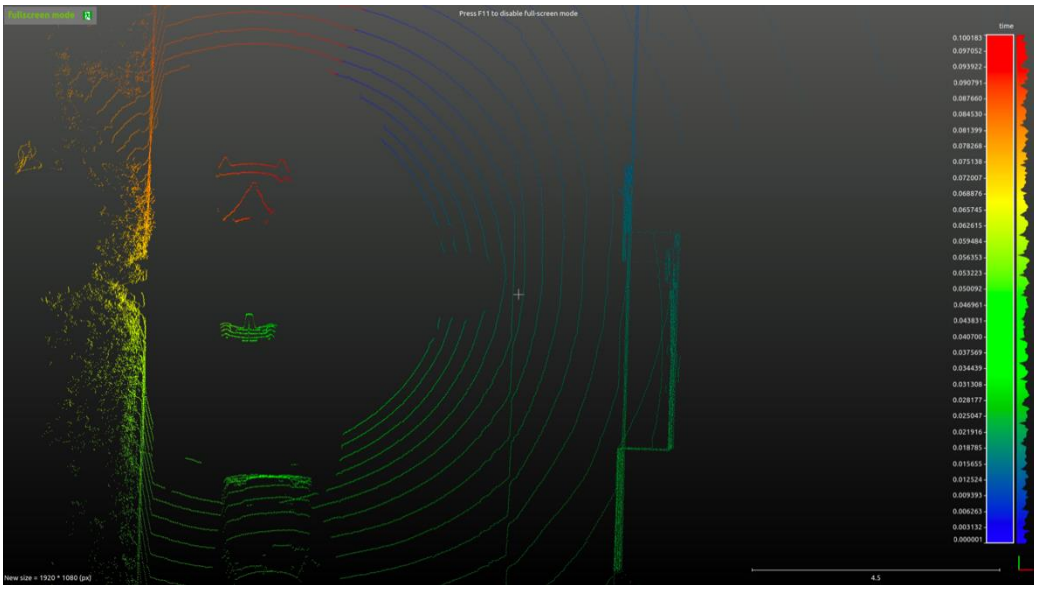
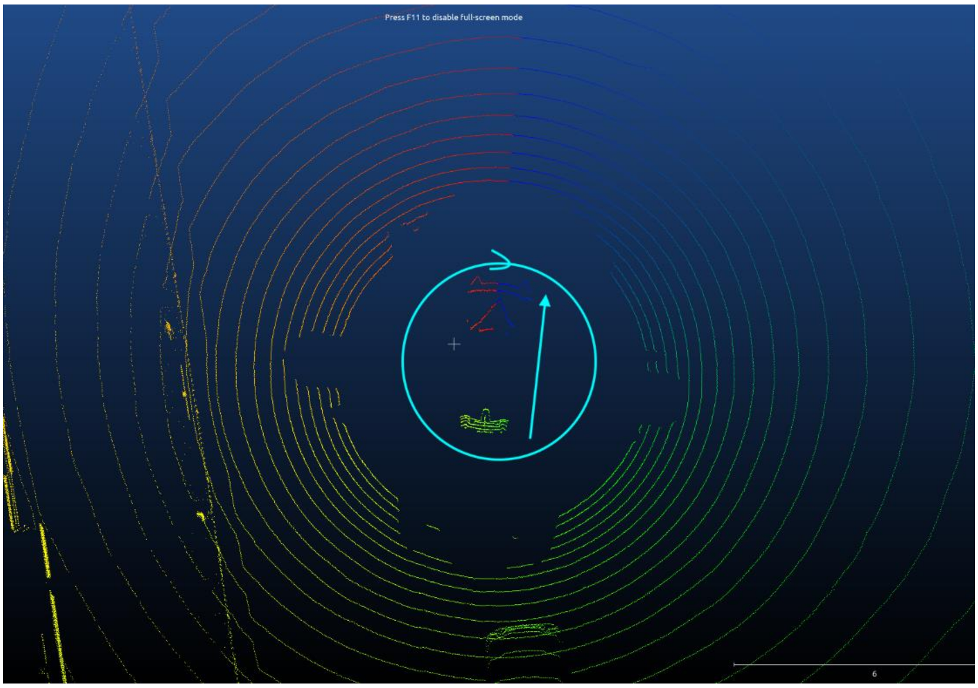


Figure 18. Overall view with timestamp: Start point of the LiDAR scan in one revolution (blue) and the last point of the LiDAR scan in one revolution (red).

Figure 19 and Figure 20 shows the vehicle is turning which has the movement of angular and longitudinal. Figure 19 shows the point cloud with the timestamp. It shows the start point and the last point of the LiDAR scan in one revolution match.

Actually, the start point and the last point of the LiDAR scan in one revolution did not match as the vehicle is turning. In Figure 20, it divided the point cloud map including the original and corrected point cloud into 4 sections (Q1 to Q4). In Q1, it shows that the original and corrected point cloud match. However, the original and corrected point cloud significant differences from Q2 to Q4. The original point cloud in Q2 and Q3 seems to be closer than the corrected point cloud as the vehicle driving direction. For the same reason, the original point cloud in Q4 seems to be further away, than the corrected point cloud. The circles around the vehicle show that the difference between the original and corrected point cloud is increasing. Figure 21 shows the close-up view of Q3 and Q4 of the scenario in Figure 20. It shows the difference between the original corrected point cloud with the same direction of vehicle speed.



Press F11 to disable full-screen mode

Figure 19. The original point cloud with the timestamp: Start point of the LiDAR scan in one revolution (blue) and the last point of the LiDAR scan in one revolution (red)

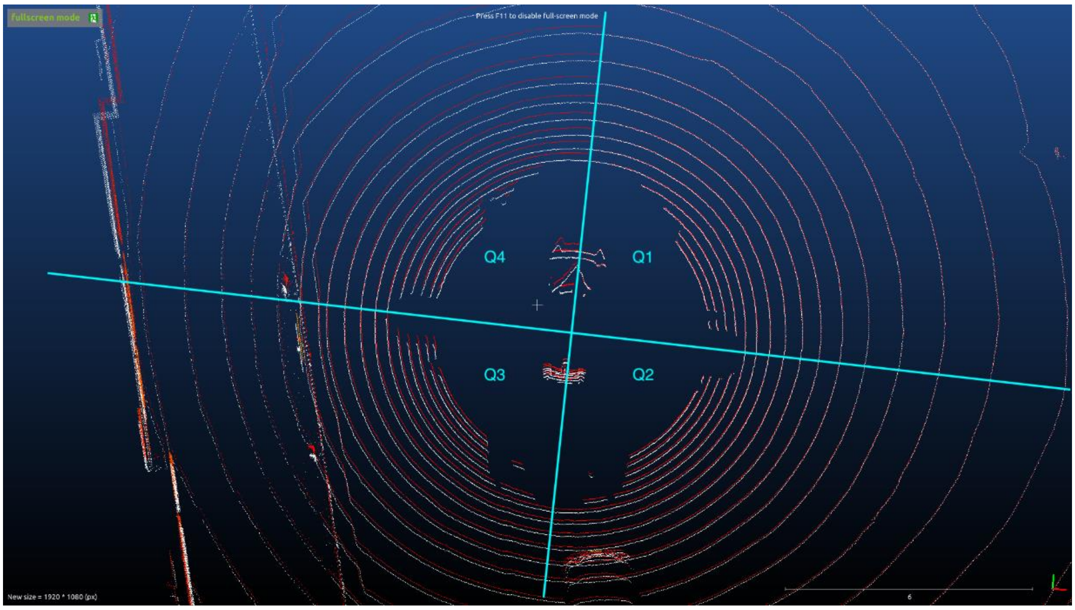

Fullscreen mode
Press F11 to disable full-screen mode
Q4
Q1
Q3
Q2
New size = 1920 * 1080 (px)
6

Figure 20. The original and point cloud corrected: Original point cloud (white) and corrected point cloud (red).

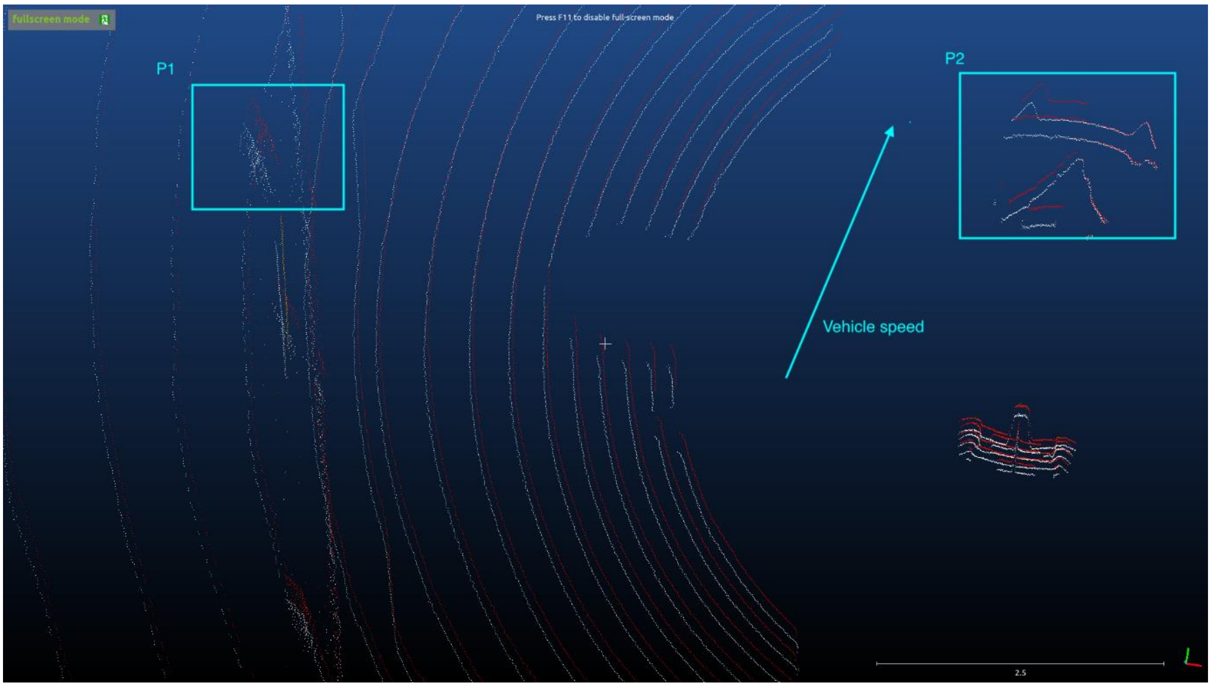


Figure 21. Close-up view of Figure 20.

4.2.2. Initialization

Camera data serves as the input source of images for this method. During the initialization phase, the ORB algorithm is employed to extract and describe the identified feature points within the input data. Subsequently, the feature points detected in each image are utilized for matching with the preceding image. The outcomes of feature point matching in two consecutive images are demonstrated in Figure 22.

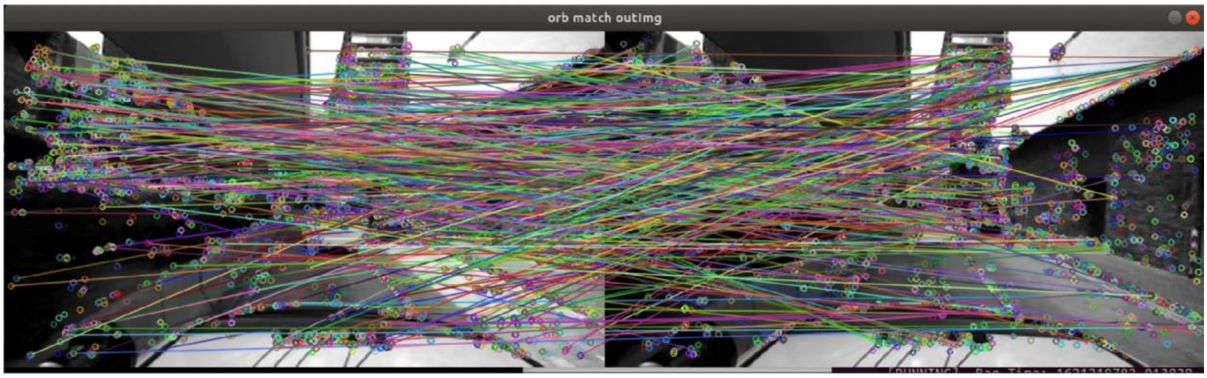


Figure 22. Feature point matching in two consecutive images

4.2.3. Reconstruct

During the reconstruction phase, pairs of feature points are employed to reconstruct 3D points within each frame. These reconstructed feature points are subsequently aggregated to create 3D map points and keyframe poses, resulting in the formation of a local point map. An instance of the local point map in a single frame is depicted in Figure 23, while the amalgamation of multiple frames is illustrated in Figure 24 using RViz [77].

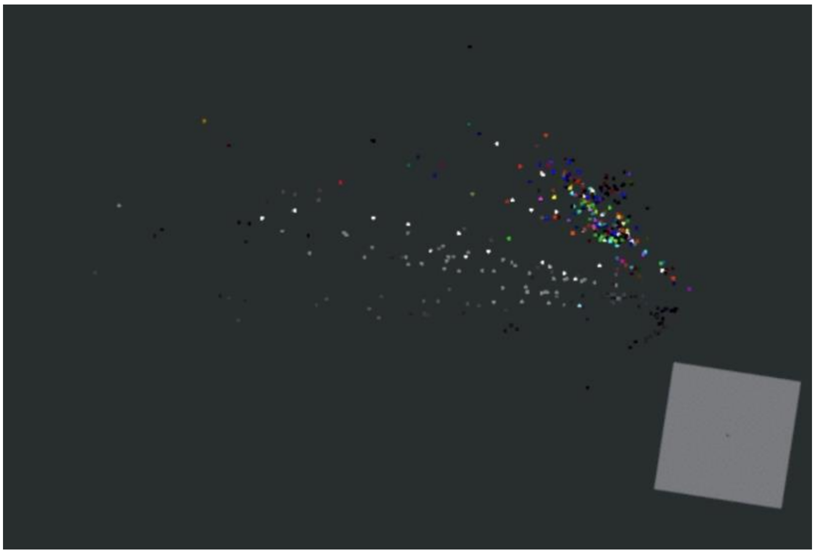


Figure 23 Illustration of a local point map within a single frame displayed in RViz

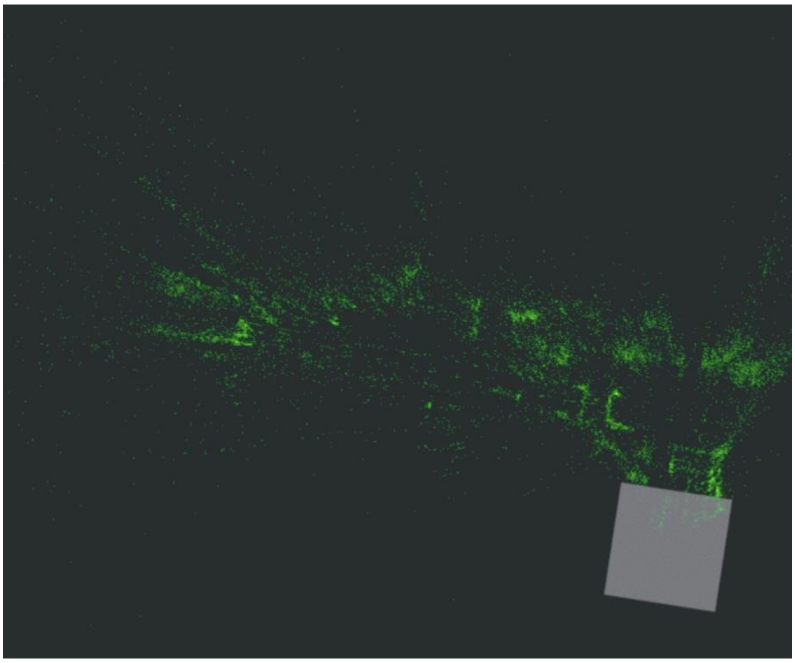


Figure 24. Illustration of the combination of multiple frames displayed in RViz

4.2.4. Point Cloud Registration

The evaluation of the proposed method encompassed two distinct datasets featuring different environments. In the first assessment, the KITTI odometry dataset was utilized to gauge the accuracy of our approach. Secondly, the UrbanNav dataset was employed for evaluation under urban conditions, involving a comparison between the estimated camera trajectory and the ground truth trajectory with the prior point cloud map.

4.2.4.1. Evaluate the accuracy

The evaluation of our proposed method's accuracy was conducted using the KITTI odometry dataset. This dataset offers a comprehensive set of LiDAR data, stereo images, and ground truth data. For our evaluation, we exclusively utilized the imagery from 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 25 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 scale issue is severe gradually and the drift happened. Figure 26 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.

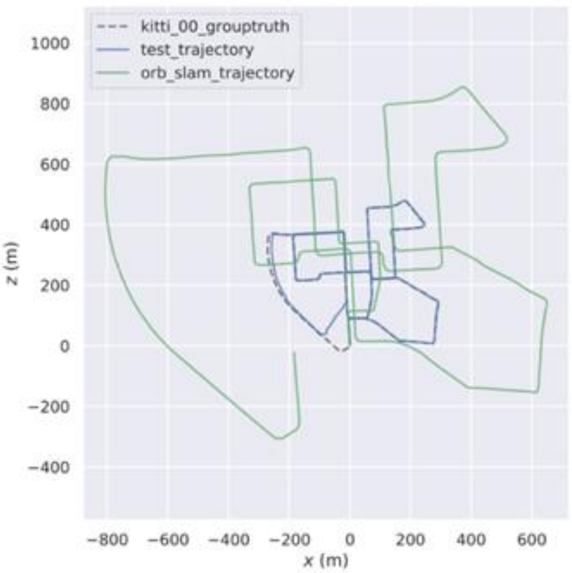


Figure 25. The trajectory is derived from the KITTI ground truth data, the ORB-SLAM and our proposed method

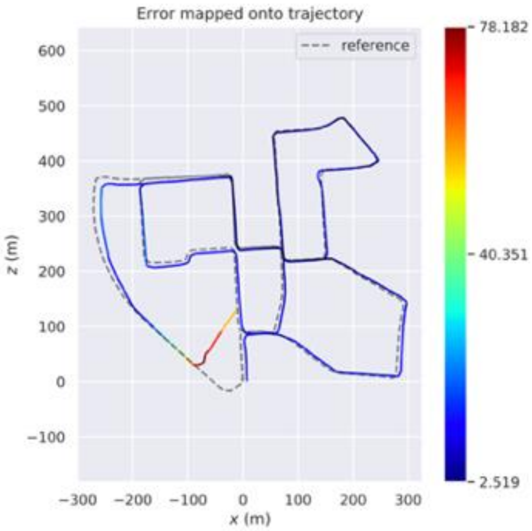


Figure 26. The trajectory error is calculated by comparing the KITTI ground truth data with our method

The experiments were repeated 6 times to mitigate randomness. Table 1 presents the absolute trajectory error (ATE) for each dataset. The outcomes encompass the trajectory error between our method and the KITTI ground truth trajectory, as well as between the ORB-SLAM trajectory and the KITTI ground truth trajectory. The results highlight the superior performance of our method compared to the visual-only approach.

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 comprise the average ATE across 6 runs

4.2.4.2. Evaluate in Urban conditions

We conducted our experimentation using data obtained by our UrbanNav dataset team. This data was gathered within the vicinity of Kowloon Tong, Hong Kong. The ground truth data was computed based on the information from the NovAtel SPAN-CPT + IE system, operating at 1 Hz. For our evaluation, we solely employed the imagery captured by the left camera of the ZED 2 camera.

The ground truth data serves as our reference for the camera's accurate pose. This allows us to compute the 6-DoF (Degrees of Freedom) camera position error. In this specific experiment, we selected a straight road within the urban environment for assessment. The outcomes include trajectories from ORB-SLAM data and our method, which are compared against the trajectory derived from the ground truth data, as visualized in Figure 27. Additionally, Figure 28 illustrates the trajectory error specifically in the translation aspect.

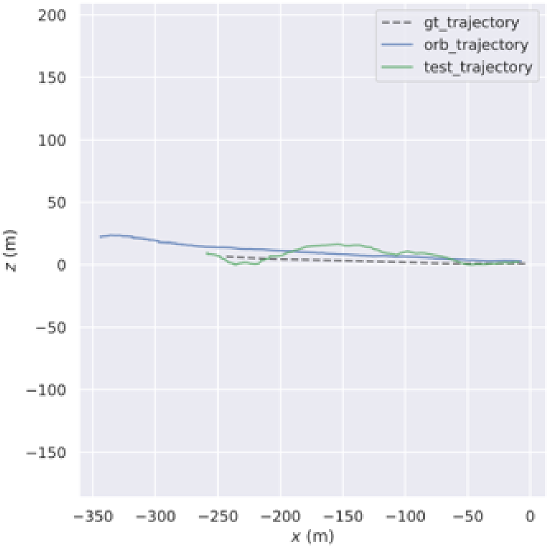


Figure 27. The trajectory outcomes of both ORB-SLAM and our method are compared against the ground truth data. These multiple trajectories are depicted in a line chart using the XZ axis.

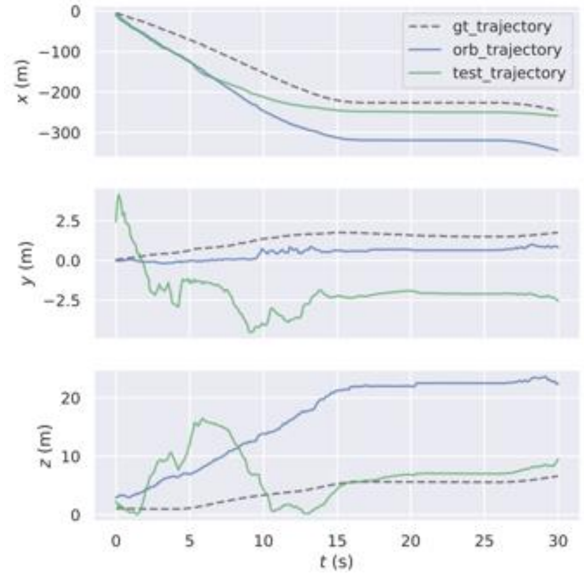


Figure 28. The trajectory results of both ORB-SLAM and our method are contrasted with the ground truth data. These multiple trajectories are visualized in a line chart, with the X-axis denoting time, the Y-axis representing position, and the Z-axis indicating time.

On the X-axis, both methods exhibit similar patterns, with our method closely aligning with the ground truth data. However, on the Y-axis and Z-axis, our method showcases more drift than the ORB-SLAM trajectory, particularly within the initial 15 seconds, attributed to the complexity of scenes during that period. In Figure 29, the absolute trajectory error (ATE) of both our trajectory and the ORB-SLAM trajectory is displayed, while Figure 30 presents a box plot of the absolute trajectory error. Notably, although our method initially displays more pronounced drift during the initial 6 seconds, it subsequently rectifies the camera pose, leading to an improved interquartile range compared to the ORB-SLAM method. This suggests that our method is effective in mitigating accumulated errors over time.

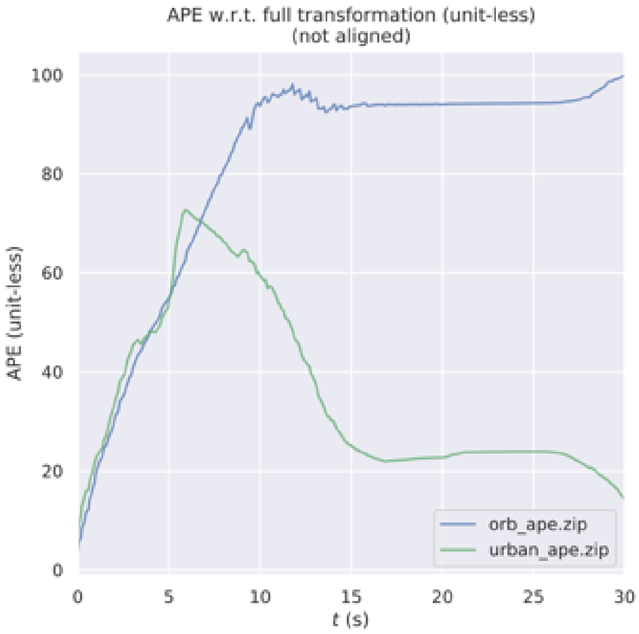


Figure 29. A line chart depicting the Absolute Trajectory Error (ATE) of both our trajectory and the ORB-SLAM trajectory

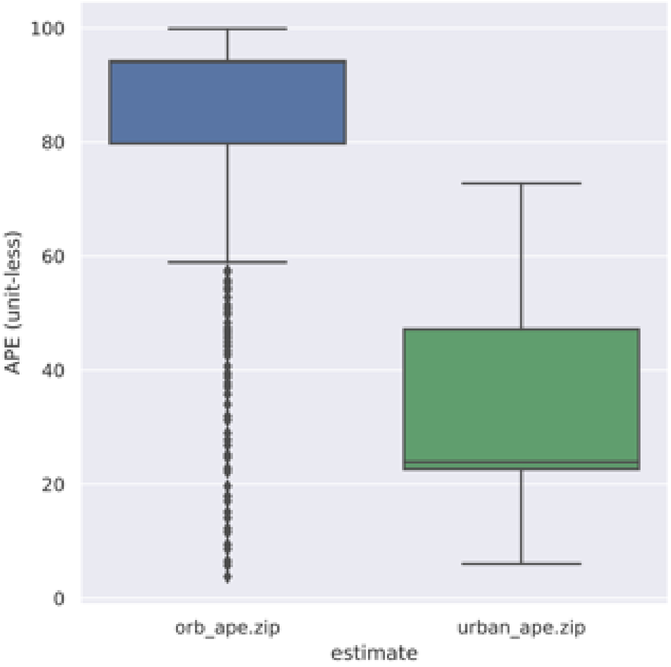


Figure 30. A box plot illustrating the Absolute Trajectory Error (ATE) of both our trajectory and the ORB-SLAM trajectory.

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

[1] F. Gustafsson *et al.*, “Particle filters for positioning, navigation, and tracking,” *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 425–437, Feb. 2002, doi: 10.1109/78.978396.

[2] Y. Gu, L.-T. Hsu, and S. Kamijo, “Passive Sensor Integration for Vehicle Self-Localization in Urban Traffic Environment,” *Sensors (Basel)*, vol. 15, no. 12, pp. 30199–30220, Dec. 2015, doi: 10.3390/s151229795.

[3] W. Wen *et al.*, “UrbanLoco: A Full Sensor Suite Dataset for Mapping and Localization in Urban Scenes,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, May 2020, pp. 2310–2316. doi: 10.1109/ICRA40945.2020.9196526.

[4] A. V. Segal, D. Haehnel, and S. Thrun, “Generalized-ICP,” *Robotics: science and systems*, vol. 2, no. 4, p. 435, 2009.

[5] P. Biber and W. Strasser, “The normal distributions transform: a new approach to laser scan matching,” in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*, Oct. 2003, pp. 2743–2748 vol.3. doi: 10.1109/IROS.2003.1249285.

[6] Q. Zou, Q. Sun, L. Chen, B. Nie, and Q. Li, “A Comparative Analysis of LiDAR SLAM-Based Indoor Navigation for Autonomous Vehicles,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 6907–6921, Jul. 2022, doi: 10.1109/TITS.2021.3063477.

[7] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, “A Survey of Autonomous Driving: Common Practices and Emerging Technologies,” *IEEE Access*, vol. 8, pp. 58443–58469, 2020, doi: 10.1109/ACCESS.2020.2983149.

[8] N. Akai, L. Y. Morales, E. Takeuchi, Y. Yoshihara, and Y. Ninomiya, “Robust localization using 3D NDT scan matching with experimentally determined uncertainty and road marker matching,” in *2017 IEEE Intelligent Vehicles Symposium (IV)*, Jun. 2017, pp. 1356–1363. doi: 10.1109/IVS.2017.7995900.

[9] Sae International, “Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles,” *SAE international*, vol. 4970, no. 724, pp. 1–5, 2018.

[10] T. Taketomi, H. Uchiyama, and S. Ikeda, “Visual SLAM algorithms: A survey from 2010 to 2016,” *IPSJ Transactions on Computer Vision and Applications*, vol. 9, no. 1, pp. 1–11, 2017.

[11] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, “G2o: A general framework for graph optimization,” in *2011 IEEE International Conference on Robotics and Automation*, May 2011, pp. 3607–3613. doi: 10.1109/ICRA.2011.5979949.

[12] J. J. Moré, “The Levenberg-Marquardt algorithm: Implementation and theory,” in *Numerical Analysis*, G. A. Watson, Ed., in Lecture Notes in Mathematics. Berlin, Heidelberg: Springer, 1978, pp. 105–116. doi: 10.1007/BFb0067700.

[13] J. J. Leonard and H. F. Durrant-Whyte, “Simultaneous map building and localization for an autonomous mobile robot,” in *Proceedings IROS ’91:IEEE/RSJ International Workshop on Intelligent Robots and Systems ’91*, Nov. 1991, pp. 1442–1447 vol.3. doi: 10.1109/IROS.1991.174711.

[14] C. Debeunne and D. Vivet, “A Review of Visual-LiDAR Fusion based Simultaneous Localization and Mapping,” *Sensors*, vol. 20, no. 7, Art. no. 7, Jan. 2020, doi: 10.3390/s20072068.

[15] Y. He and S. Chen, “Advances in sensing and processing methods for three-dimensional robot vision,” *International Journal of Advanced Robotic Systems*, vol. 15, no. 2, p. 1729881418760623, 2018.

[16] C. Cadena *et al.*, “Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age,” *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309–1332, Dec. 2016, doi: 10.1109/TRO.2016.2624754.

[17] J. Zhang and S. Singh, “Low-drift and real-time lidar odometry and mapping,” *Auton Robot*, vol. 41, no. 2, pp. 401–416, Feb. 2017, doi: 10.1007/s10514-016-9548-2.

[18] H. Strasdat, J. M. Montiel, and A. J. Davison, “Visual SLAM: why filter?,” *Image and Vision Computing*, vol. 30, no. 2, pp. 65–77, 2012.

[19] A. Chiuso, P. Favaro, H. Jin, and S. Soatto, “Structure from motion causally integrated over time,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 24, no. 4, pp. 523–535, 2002.

[20] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, “MonoSLAM: Real-Time Single Camera SLAM,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1052–1067, Jun. 2007, doi: 10.1109/TPAMI.2007.1049.

[21] P. Geneva, K. Eckenhoff, W. Lee, Y. Yang, and G. Huang, “Openvins: A research platform for visual-inertial estimation,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2020, pp. 4666–4672.

[22] E. Eade and T. Drummond, “Monocular SLAM as a graph of coalesced observations,” in *2007 IEEE 11th International Conference on Computer Vision*, IEEE, 2007, pp. 1–8.

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

[24] H. Strasdat, J. M. M. Montiel, and A. J. Davison, “Real-time monocular SLAM: Why filter?,” in *2010 IEEE International Conference on Robotics and Automation*, May 2010, pp. 2657–2664. doi: 10.1109/ROBOT.2010.5509636.

[25] J. Engel, T. Schöps, and D. Cremers, “LSD-SLAM: Large-Scale Direct Monocular SLAM,” in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T.

Tuytelaars, Eds., in Lecture Notes in Computer Science. Cham: Springer International Publishing, 2014, pp. 834–849. doi: 10.1007/978-3-319-10605-2_54.

[26] S. B. Knorr and D. Kurz, "Leveraging the User's Face for Absolute Scale Estimation in Handheld Monocular SLAM," in *2016 IEEE International Symposium on Mixed and Augmented Reality (ISMAR),* Sep. 2016, pp. 11–17. doi: 10.1109/ISMAR.2016.20.

[27] S. Song, M. Chandraker, and C. C. Guest, "Parallel, real-time monocular visual odometry," in *2013 IEEE International Conference on Robotics and Automation,* May 2013, pp. 4698–4705. doi: 10.1109/ICRA.2013.6631246.

[28] J. Gräter, T. Schwarze, and M. Lauer, "Robust scale estimation for monocular visual odometry using structure from motion and vanishing points," in *2015 IEEE Intelligent Vehicles Symposium (IV),* Jun. 2015, pp. 475–480. doi: 10.1109/IVS.2015.7225730.

[29] D. Zhou, Y. Dai, and H. Li, "Reliable scale estimation and correction for monocular Visual Odometry," in *2016 IEEE Intelligent Vehicles Symposium (IV),* Jun. 2016, pp. 490–495. doi: 10.1109/IVS.2016.7535431.

[30] H. Strasdat, J. Montiel, and A. J. Davison, "Scale drift-aware large scale monocular SLAM," *Robotics: Science and Systems VI,* vol. 2, no. 3, p. 7, 2010.

[31] T. Sattler, B. Leibe, and L. Kobbelt, "Fast image-based localization using direct 2D-to-3D matching," in *2011 International Conference on Computer Vision,* Nov. 2011, pp. 667–674. doi: 10.1109/ICCV.2011.6126302.

[32] P. Moulon, P. Monasse, and R. Marlet, "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, 2013, pp. 3248–3255. Accessed: Feb. 28, 2022. [Online]. Available: https://www.cv-foundation.org/openaccess/content_iccv_2013/html/Moulon_Global_Fusion_of_2013_ICCV_paper.html

[33] A. Torii, R. Arandjelović, J. Sivic, M. Okutomi, and T. Pajdla, "24/7 place recognition by view synthesis," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* Jun. 2015, pp. 1808–1817. doi: 10.1109/CVPR.2015.7298790.

[34] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision,* vol. 60, no. 2, pp. 91–110, Nov. 2004, doi: 10.1023/B:VISI.0000029664.99615.94.

[35] M. Dusmanu *et al.*, "D2-Net: A Trainable CNN for Joint Description and Detection of Local Features," presented at the Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 8092–8101. Accessed: Feb. 28, 2022. [Online]. Available: https://openaccess.thecvf.com/content_CVPR_2019/html/Dusmanu_D2-Net_A_Trainable_CNN_for_Joint_Description_and_Detection_of_CVPR_2019_paper.html

[36] J. Revaud *et al.*, "R2D2: repeatable and reliable detector and descriptor," *arXiv preprint arXiv:1906.06195*, 2019.

[37] P.-E. Sarlin, D. DeTone, T. Malisiewicz, and A. Rabinovich, "SuperGlue: Learning Feature Matching With Graph Neural Networks," presented at the Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 4938–4947. Accessed: Feb. 28, 2022. [Online]. Available: https://openaccess.thecvf.com/content_CVPR_2020/html/Sarlin_SuperGlue_Learning_Feature_Matching_With_Graph_Neural_Networks_CVPR_2020_paper.html

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

1491. Accessed: Feb. 28, 2022. [Online]. Available: https://openaccess.thecvf.com/content_cvpr_2017/html/Schonberger_Comparative_Evaluation_of_CVPR_2017_paper.html

[39] G. Csurka, C. R. Dance, and M. Humenberger, "From handcrafted to deep local features," *arXiv:1807.10254 [cs]*, Jun. 2019, Accessed: Feb. 28, 2022. [Online]. Available: http://arxiv.org/abs/1807.10254

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

[41] R. W. Wolcott and R. M. Eustice, "Visual localization within LIDAR maps for automated urban driving," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sep. 2014, pp. 176–183. doi: 10.1109/IROS.2014.6942558.

[42] G. Pascoe, W. Maddern, and P. Newman, "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, 2015, pp. 9–16. Accessed: Feb. 28, 2022. [Online]. Available: https://www.cv-foundation.org/openaccess/content_iccv_2015_workshops/w8/html/Pascoe_Direct_Visual_Localisation_ICCV_2015_paper.html

[43] M. Muglikar, Z. Zhang, and D. Scaramuzza, "Voxel Map for Visual SLAM," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, May 2020, pp. 4181–4187. doi: 10.1109/ICRA40945.2020.9197357.

[44] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *2011 IEEE International Conference on Robotics and Automation*, May 2011, pp. 1–4. doi: 10.1109/ICRA.2011.5980567.

[45] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial and Multi-Map SLAM," *IEEE Trans. Robot.*, vol. 37, no. 6, pp. 1874–1890, Dec. 2021, doi: 10.1109/TRO.2021.3075644.

[46] T. VanCourt, Y. Gu, and M. C. Herbordt, "Three-dimensional template correlation: object recognition in 3D voxel data," in *Seventh International Workshop on Computer Architecture for Machine Perception (CAMP'05)*, IEEE, 2005, pp. 153–158.

[47] R. A. Newcombe *et al.*, "Kinectfusion: Real-time dense surface mapping and tracking," in *2011 10th IEEE international symposium on mixed and augmented reality*, Ieee, 2011, pp. 127–136.

[48] A. Dai, M. Nießner, M. Zollhöfer, S. Izadi, and C. Theobalt, "BundleFusion: Real-Time Globally Consistent 3D Reconstruction Using On-the-Fly Surface Reintegration," *ACM Trans. Graph.*, vol. 36, no. 4, p. 76a:1, Jul. 2017, doi: 10.1145/3072959.3054739.

[49] G. Grisetti, C. Stachniss, and W. Burgard, "Improving Grid-based SLAM with Rao-Blackwellized Particle Filters by Adaptive Proposals and Selective Resampling," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, Apr. 2005, pp. 2432–2437. doi: 10.1109/ROBOT.2005.1570477.

[50] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM: A factored solution to the simultaneous localization and mapping problem," *Aaai/iaai*, vol. 593598, 2002.

[51] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: an efficient probabilistic 3D mapping framework based on octrees," *Auton Robot*, vol. 34, no. 3, pp. 189–206, Apr. 2013, doi: 10.1007/s10514-012-9321-0.

[52] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto, "Voxblox: Incremental 3D Euclidean Signed Distance Fields for on-board MAV planning," in *2017 IEEE/RSJ*

International Conference on Intelligent Robots and Systems (IROS), Sep. 2017, pp. 1366–1373. doi: 10.1109/IROS.2017.8202315.

[53] B. Lau, C. Sprunk, and W. Burgard, “Improved updating of Euclidean distance maps and Voronoi diagrams,” in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2010, pp. 281–286. doi: 10.1109/IROS.2010.5650794.

[54] F. Moosmann and C. Stiller, “Velodyne slam,” in *2011 ieee intelligent vehicles symposium (iv)*, IEEE, 2011, pp. 393–398.

[55] D. Chekhlov, M. Pupilli, W. Mayol-Cuevas, and A. Calway, “Real-time and robust monocular SLAM using predictive multi-resolution descriptors,” in *Advances in Visual Computing: Second International Symposium, ISVC 2006 Lake Tahoe, NV, USA, November 6-8, 2006. Proceedings, Part II 2*, Springer, 2006, pp. 276–285.

[56] G. Grisetti, C. Stachniss, and W. Burgard, “Improved techniques for grid mapping with rao-blackwellized particle filters,” *IEEE transactions on Robotics*, vol. 23, no. 1, pp. 34–46, 2007.

[57] J. J. Leonard and H. J. S. Feder, “A computationally efficient method for large-scale concurrent mapping and localization,” in *Robotics Research: The Ninth International Symposium*, Springer, 2000, pp. 169–176.

[58] S. Thrun, Y. Liu, D. Koller, A. Y. Ng, Z. Ghahramani, and H. Durrant-Whyte, “Simultaneous localization and mapping with sparse extended information filters,” *The international journal of robotics research*, vol. 23, no. 7–8, pp. 693–716, 2004.

[59] D. Holz and S. Behnke, “Sancta simplicitas - on the efficiency and achievable results of SLAM using ICP-based incremental registration,” in *2010 IEEE International Conference on Robotics and Automation*, May 2010, pp. 1380–1387. doi: 10.1109/ROBOT.2010.5509918.

[60] L.-T. Hsu *et al.*, “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), Sep. 2021, pp. 226–256. doi: 10.33012/2021.17895.

[61] G. Klein and D. Murray, “Parallel tracking and mapping for small AR workspaces,” in *2007 6th IEEE and ACM international symposium on mixed and augmented reality*, IEEE, 2007, pp. 225–234.

[62] T. Qin, P. Li, and S. Shen, “VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator,” *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, Aug. 2018, doi: 10.1109/TRO.2018.2853729.

[63] A. Geiger, P. Lenz, and R. Urtasun, “Are we ready for autonomous driving? the kitti vision benchmark suite,” in *2012 IEEE conference on computer vision and pattern recognition*, IEEE, 2012, pp. 3354–3361.

[64] J. Zhang and Sanjiv Singh, “LOAM: Lidar Odometry and Mapping in Real-time,” Robotics: Science and Systems, Jul. 2014, pp. 1–9. Accessed: Apr. 14, 2023. [Online]. Available: https://www.roboticsproceedings.org/rss10/p07.pdf

[65] R. W. Wolcott and R. M. Eustice, “Robust LIDAR localization using multiresolution Gaussian mixture maps for autonomous driving,” *The International Journal of Robotics Research*, vol. 36, no. 3, pp. 292–319, Mar. 2017, doi: 10.1177/0278364917696568.

[66] I. Baldwin and P. Newman, “Laser-only road-vehicle localization with dual 2D push-broom LIDARS and 3D priors,” in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2012, pp. 2490–2497. doi: 10.1109/IROS.2012.6385677.

[67] S. Chen *et al.*, “NDT-LOAM: A Real-time Lidar odometry and mapping with weighted NDT and LFA,” *IEEE Sensors Journal*, vol. 22, no. 4, pp. 3660–3671, 2021.

[68] S. Schneider, M. Himmelsbach, T. Luettel, and H.-J. Wuensche, “Fusing vision and LIDAR - Synchronization, correction and occlusion reasoning,” in *2010 IEEE Intelligent Vehicles Symposium*, Jun. 2010, pp. 388–393. doi: 10.1109/IVS.2010.5548079.

[69] S. Hong, H. Ko, and J. Kim, “VICP: Velocity updating iterative closest point algorithm,” in *2010 IEEE International Conference on Robotics and Automation*, May 2010, pp. 1893–1898. doi: 10.1109/ROBOT.2010.5509312.

[70] M. BROSSARD and S. BONNABEL, “Learning Wheel Odometry and IMU Errors for Localization,” in *2019 International Conference on Robotics and Automation (ICRA)*, May 2019, pp. 291–297. doi: 10.1109/ICRA.2019.8794237.

[71] J. Byun, K. Na, B. Seo, and M. Roh, “Drivable Road Detection with 3D Point Clouds Based on the MRF for Intelligent Vehicle,” in *Field and Service Robotics: Results of the 9th International Conference*, L. Mejias, P. Corke, and J. Roberts, Eds., in Springer Tracts in Advanced Robotics. Cham: Springer International Publishing, 2015, pp. 49–60. doi: 10.1007/978-3-319-07488-7_4.

[72] T. Caselitz, B. Steder, M. Ruhnke, and W. Burgard, “Monocular camera localization in 3D LiDAR maps,” in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2016, pp. 1926–1931. doi: 10.1109/IROS.2016.7759304.

[73] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “ORB: An efficient alternative to SIFT or SURF,” in *2011 International conference on computer vision*, Ieee, 2011, pp. 2564–2571.

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

[75] A. W. Knapp and A. W. Knapp, *Lie groups beyond an introduction*, vol. 140. Springer, 1996.

[76] M. Quigley *et al.*, “ROS: an open-source Robot Operating System,” *ICRA workshop on open source software*, vol. 3, no. 3.2, p. 5, May 2009.

[77] H. R. Kam, S.-H. Lee, T. Park, and C.-H. Kim, “RViz: a toolkit for real domain data visualization,” *Telecommun Syst*, vol. 60, no. 2, pp. 337–345, Oct. 2015, doi: 10.1007/s11235-015-0034-5.