VILAM: Infrastructure-assisted 3D Visual Localization and Mapping for Autonomous Driving

Jiahe Cui, *Beihang University, The Chinese University of Hong Kong, and Tianmushan Laboratory;* Shuyao Shi and Yuze He, *The Chinese University of Hong Kong;* Jianwei Niu, *Beihang University;* Guoliang Xing, *The Chinese University of Hong Kong;* Zhenchao Ouyang, *Tianmushan Laboratory and International Innovation Institute of Beihang University*

https://www.usenix.org/conference/nsdi24/presentation/cui

This paper is included in the
Proceedings of the 21st USENIX Symposium on
Networked Systems Design and Implementation.

April 16–18, 2024 • Santa Clara, CA, USA

978-1-939133-39-7

VILAM: Infrastructure-assisted 3D Visual Localization and Mapping for Autonomous Driving

Jiahe Cui[123], Shuyao Shi[2], Yuze He[2], Jianwei Niu[*1], Guoliang Xing[2] and Zhenchao Ouyang[34]
[1]*Beihang University* [2]*The Chinese University of Hong Kong*
[3]*Tianmushan Laboratory* [4]*International Innovation Institute of Beihang University*

Abstract

Visual Simultaneous Localization and Mapping (SLAM) presents a promising avenue for fulfilling the essential perception and localization tasks in autonomous driving systems using cost-effective visual sensors. Nevertheless, existing visual SLAM frameworks often suffer from substantial cumulative errors and performance degradation in complicated driving scenarios. In this paper, we propose VILAM, a novel framework that leverages intelligent roadside infrastructures to realize high-precision and globally consistent localization and mapping on autonomous vehicles. The key idea of VILAM is to utilize the precise scene measurement from the infrastructure as global references to correct errors in the local map constructed by the vehicle. To overcome the unique deformation in the 3D local map to align it with the infrastructure measurement, VILAM proposes a novel elastic point cloud registration method that enables independent optimization of different parts of the local map. Moreover, VILAM adopts a lightweight factor graph construction and optimization to first correct the vehicle trajectory, and thus reconstruct the consistent global map efficiently. We implement the VILAM end-to-end on a real-world smart lamppost testbed in multiple road scenarios. Extensive experiment results show that VILAM can achieve decimeter-level localization and mapping accuracy with consumer-level onboard cameras and is robust under diverse road scenarios. A video demo of VILAM on our real-world testbed is available at https://youtu.be/lTlqDNipDVE.

1 Introduction

Visual SLAM utilizes video streams from cameras as input, reconstructs the 3D map of the unknown environment, and simultaneously determines the position and orientation of cameras with respect to their surroundings [5, 10, 13, 38]. It holds the potential to enable the critical perception and localization tasks required in autonomous driving systems [6, 44], especially in challenging environments characterized by the absence of traffic semantics, the lack of high-precision localization and prior driving maps, or where the road surroundings undergoes frequent structural changes. However, as an online localization and mapping paradigm, visual SLAM systems are vulnerable to cumulative drift caused by inherent sensor noises of commodity cameras as well as errors from the feature point extraction and matching algorithms [23]. A recent study [45] shows that current visual SLAM systems can yield up to $75m$ of cumulative drift after mapping $2.6km$ on real roads. A similar result is also shown in our real-world case study where a state-of-the-art visual SLAM algorithm [4] exhibits a drift of over $10m$ in map construction after a $400m$ drive on a campus road. Such drift not only leads to significant deviations in vehicle localization but also causes inconsistencies in the constructed 3D map, posing a substantial challenge to the reliability of autonomous driving systems.

To address these challenges, existing studies propose to employ loop-closures [4, 46] or integrate high-precision GNSS locations [5, 7] as global references for error correction. However, loop-closure methods require the presence of looped paths in vehicle trajectories, which is uncommon in autonomous driving situations. GNSS-based methods rely on centimeter-level accuracy GNSS-RTK equipment that can cost up to $4,000 per suite [1] and the availability of GPS signals. Such limitations present significant barriers for adoption in a variety of application scenarios, such as self-parking in underground garages or autonomous driving in urban canyons.

In this work, we exploit intelligent roadside infrastructure as a promising solution for enhancing vehicular SLAM. Intelligent roadside infrastructures, equipped with LiDARs and compute units, are increasingly available not only on public roads [8] but also in places like campuses [20] and parking lots [11, 35]. In particular, LiDARs are being progressively deployed on roadside infrastructures due to their decreasing prices in recent years [2]. Thanks to their stationary nature, infrastructures can obtain accurate localization and measurements of the environmental structure, which can serve as reliable references to correct the cumulative drift in vehic-

[*] Corresponding author.

ular SLAM. By opportunistically exploiting the references from infrastructure nodes along the road, vehicles can achieve high-performance visual SLAM over long-distance driving without looped paths or GNSS devices.

This paper proposes VILAM, the first infrastructure-assisted vehicular SLAM system that harnesses LiDAR measurements from distributed roadside infrastructure in real time. In designing VILAM, we address several key challenges. First, due to the cumulative drift in visual SLAM, the local map constructed by the vehicle suffers irregular deformation, making its alignment with global references challenging. We address this challenge by devising a novel elastic alignment approach, which optimizes each part of the local map independently to maximum overlap with global references. Second, roadside infrastructures may be installed sporadically on roadways. Therefore, merely aligning the local map with the global references is insufficient for obtaining a consistent global map. VILAM adopts a novel factor graph optimization method to infer the global map efficiently based on the local map and the alignment results. Third, the vast volume of raw LiDAR point clouds makes it challenging to share with passing vehicles. Meanwhile, dynamic objects such as vehicles and pedestrians can significantly deteriorate the scene measurements on the infrastructure. VILAM proposes a lightweight solution, which removes the low-occupancy segments of the accumulated point cloud based on the mobility of objects and reduces the redundancy and amount of 3D points to be shared while ensuring the accuracy of the static scene measurement. Lastly, VILAM does not require high-end sensors or localization devices on vehicles. This facilitates a new mapping paradigm for autonomous driving, especially beneficial in environments lacking pre-loaded HD maps.

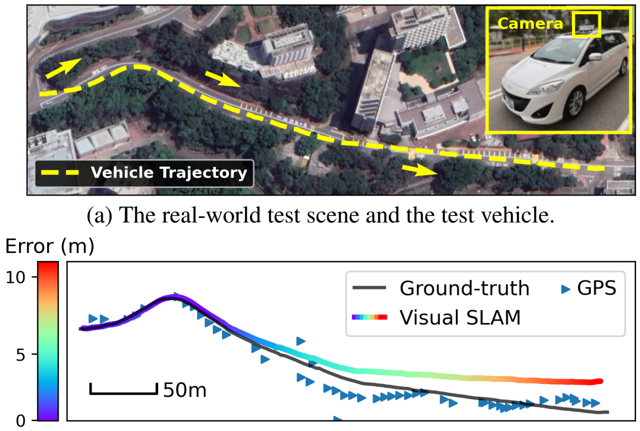
We have implemented VILAM end-to-end on a smart lamppost testbed, spanning multiple real-world road scenarios. We collect a new dataset that covers a total of $17.6km$ of driving trajectories with 44 infrastructure nodes in five typical road scenarios. Our extensive evaluation shows that VILAM exhibits a localization error within $0.5m$ and a mapping error within $0.7m$, which are less than 15% and 40% of the errors of state-of-the-art baselines, respectively, even when the coverage of roadside infrastructure is as low as 20%. Moreover, VILAM only transmits compact scene measurements at around 236 KB per frame, which reduces the data volume by about 90× and 17× compared with raw and down-sampled point clouds, respectively. Lastly, VILAM achieves an end-to-end system latency within $350ms$ for global map correction while updating the local map and vehicle localization at 15 Hz. A demo video of VILAM on our real-world testbed is available at https://youtu.be/lTlqDNipDVE.

2 Related Work

Visual SLAM. Visual SLAM is a specialized branch of SLAM that utilizes visual sensor data, typically from one or multiple cameras, to perform simultaneous mapping and localization. Existing visual SLAM methods [3, 7, 27, 49] build visual maps of 3D distinctive points by aggregating the extracted features from the input images. However, due to the noisy sensor measurements and the errors of feature matching, such methods suffer from cumulative errors over a period of exploration. Some works [4, 28, 38] utilize loop closure detection to eliminate such errors, which is not reliable in autonomous driving scenarios as vehicles rarely revisit the exact same locations in a short time. Some studies [17, 29, 48] construct globally consistent maps by merging the SLAM maps from multiple agents. However, they require the agents to frequently communicate with others and create overlapped maps actively. This requirement complicates the deployment of such methods in autonomous driving, where latency and computational efficiency are paramount. Other works [7, 25, 37] eliminate the cumulative errors by incorporating global constraints, such as global positioning from GNSS sensors. However, high-precision RTK-GNSS units are highly expensive, while the consumer-grade GNSS measurement is insufficient to effectively correct cumulative errors in SLAM. Furthermore, these methods tend to fail in environments where GNSS signals are not available, such as underground parking lots and urban canyons.

Camera-LiDAR Fusion. Previous research [32, 34, 42, 52] explore the fusion of camera images and LiDAR point clouds, with a focus on leveraging the strengths of these two sensors to enhance the perception abilities of autonomous vehicles across diverse scenarios. However, these fusion approaches assume that the camera and LiDAR are mounted together on the vehicle and are precisely calibrated. Therefore, they cannot cope with the unique fusion problem in infrastructure vehicle cooperative systems, where the LiDAR and cameras are typically separated, and the relative positions constantly change as the vehicle moves. Some studies [30, 39] can resolve the relative offset between the camera and LiDAR by registering camera images to LiDAR point clouds, thereby accomplishing sensor data fusion. However, the applicability of these methods is often challenged as their performance relies on the number of feature correspondences between the image and point cloud pair. Due to the significant difference in sensor perspectives between vehicles and infrastructure, there are generally fewer corresponding features.

Infrastructure-assisted Localization and Mapping. The utilization of smart roadside infrastructure to assist autonomous vehicles is an emerging paradigm [33, 47]. Previous works have utilized wireless communication devices like WiFi [19] and UWB [15] on the infrastructures to achieve vehicle localization. However, roadside devices can only provide vehicles with approximate single-point location information, making it challenging to aid vehicles in map construction tasks. Some studies [22, 31] utilize pre-installed visual indicators, such as QR code markers, as references for vehicle localization. However, their effectiveness diminishes as the distance between



(a) The real-world test scene and the test vehicle.

(b) Trajectory estimated by visual SLAM.

Figure 1: A motivational case study. (a) An example scenario where the test vehicle drives through a long road. (b) The errors of the vehicle's trajectory estimated by visual SLAM without correction and the GPS localization samples.

the vehicle and the marker increases, making them unreliable for practical driving scenarios. Some studies [16, 18, 43] achieve vehicle localization by aligning the semantic correspondences observed collectively by both infrastructure and vehicles, thereby accomplishing perception fusion. These studies predominantly focus on localization within the perception range of a single infrastructure and the construction of local maps. Moreover, to ensure alignment accuracy, these studies usually require the equipping of high-precision LiDAR on both vehicle and infrastructure sides.

3 Background and Motivation

3.1 Preliminaries of SLAM Systems

SLAM technology constructs a 3D feature point map of an unknown environment while simultaneously determining the vehicle's localization within the map using onboard visual-inertial sensors, such as cameras and IMUs. Practical SLAM systems exhibit considerable diversity in their implementations, yet most systems typically consist of the following three modules:

Tracking. The tracking module goes through several key steps. First, it detects 2D feature points within the current image frame. Each point corresponds to a 3D map point $\mathbf{m}$ in the environment and has a feature descriptor $\mathbf{f}$. Then, these feature points are matched with those in a previous keyframe [50] according to the similarity of $\mathbf{f}$, establishing associations between the current frame and the keyframes. By combining information on the feature associations and IMU measurements, this tracking module computes the relative pose between the current frame and the keyframe. This process constantly updates the vehicle's localization w.r.t the starting position.

Local Mapping. This module continuously collects tracking results from each individual image frame within a local sliding window. It then leverages Bundle Adjustment [4] to construct a 3D feature point map and estimate the vehicle's trajectory within that window. The feature point map is represented as a set of 3D map points and their feature descriptors: $\mathcal{M} = \left\{(\mathbf{m}_j \in \mathbb{R}^3, \mathbf{f}_j) \middle| j < N\right\}$, where $\mathbf{m}_j$ is the 3D coordinates of the jth map point, and $\mathbf{f}_j$ is the corresponding feature descriptor. N denotes the total number of landmarks in the map. The estimated vehicle trajectory is a sequence of transformation matrices $\mathcal{T} = \{\mathbf{T}_0, \mathbf{T}_1, \cdots, \mathbf{T}_i, \cdots\}$. $\mathbf{T}_i = (\mathbf{R}_i, \mathbf{t}_i) \in \mathbb{SE}3$ is the estimation of the vehicle's pose at time i, where $\mathbf{R}_i$ and $\mathbf{t}_i$ denote the rotation and translation components of the pose, respectively. Note that $\mathcal{M}$ and $\mathcal{T}$ are generated in the local coordinate frame, whose origin is the starting point of the SLAM. In practical driving scenarios, $\mathcal{M}$ and $\mathcal{T}$ are typically approximated to the global coordinate frame using the vehicle's initial localization as the starting point.

Error Minimization. This module rectifies the accumulated errors in the estimated map and constructs a consistent global map. These errors mainly stem from factors like the sensor noises as well as feature mismatching during tracking and are integrated into the map frame-by-frame, eventually leading to an offset in the map. Most methods employ Loop Correction [23] to minimize such errors. When vehicles revisit a previously traversed area, the module identifies key features in the current frame, matches them with those stored in the map from previous visits, and then detects if a loop path exists. It then performs adjustments on the map points and the estimated trajectory to ensure consistency on the loop path, thereby mitigating accumulated errors. Some methods also utilize periodic global positional information, such as GNSS locations [5, 25, 37], to limit the growth of accumulated errors.

3.2 Limitations of SLAM

SLAM technologies can encounter unique challenges in autonomous driving scenarios. To illustrate these challenges, we present a motivating case study highlighting the limitations of SLAM in the context of autonomous driving.

We conducted a test drive using a vehicle along a campus road of around 450m, as depicted in Fig. 1(a). The test vehicle is equipped with a RealSense camera, a consumer-level GPS receiver, and a high-precision RTK unit. We employ ORB-SLAM3 [4], a state-of-the-art visual SLAM algorithm widely adopted in both research and industry, to process vehicle camera images. Fig. 1(b) presents a comparison between the vehicle trajectory estimated by ORB-SLAM3 and the ground truth trajectory recorded by the high-precision RTK unit. Initially (e.g., within the first 100m), the estimated trajectory closely aligns with the ground truth. However, as the vehicle continues to drive, a noticeable offset occurs. As discussed in Sec. 3.1, this offset is attributed to accumulated errors from the tracking module. In driving scenarios, due to the pres-

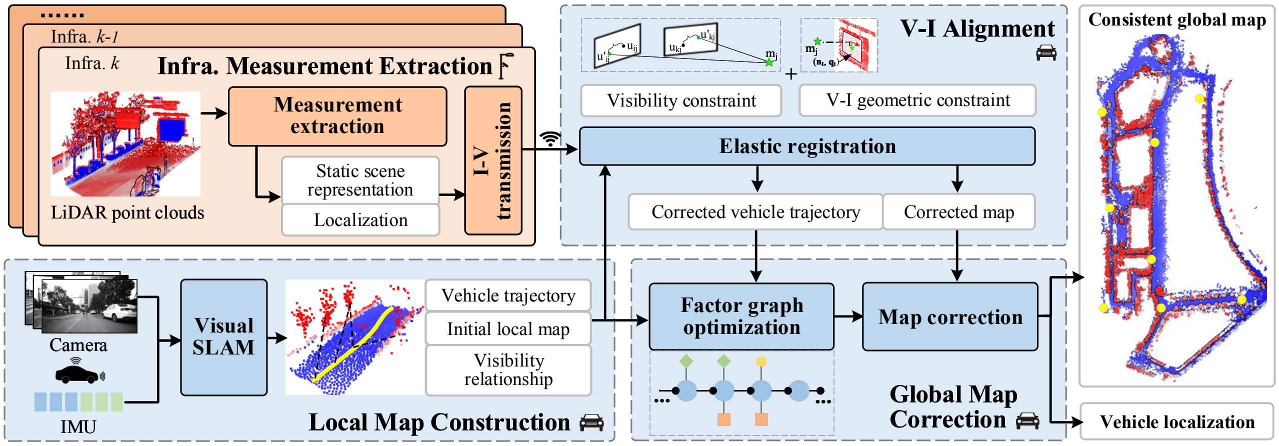


Figure 2: The architecture of VILAM. The orange and blue boxes denote infrastructure and vehicle operations, respectively.

ence of dynamic objects and the rapid perspective changes caused by high driving speed, incorrect feature associations between consecutive frames can lead to more severe pose estimation errors than in robotic applications. Lacking external supervision, these errors accumulate over time, leading to increasingly significant deviations in the vehicle's trajectory. For example, as shown in Fig. 1(b), when the trajectory length exceeds 400 m, the offset can exceed 10 m.

While Loop Correction can minimize the cumulative errors, it requires the vehicle to revisit previously traversed locations frequently to establish loop constraints. Such an approach might be practical in small-scale indoor localization scenarios. However, autonomous vehicles often travel long distances, and their trajectory planners aim to minimize redundant routes. Consequently, the commonly seen trajectory of the vehicles is as shown in Fig. 1(a). This characteristic reduces the frequency of loop closure opportunities compared to other SLAM applications like indoor robotics. Additionally, road and traffic conditions can change rapidly in autonomous driving scenarios. Conventional loop closure methods may struggle to handle dynamic objects and changing environments. For those methods that utilize global positioning to mitigate accumulated errors, Fig. 1(b) shows that consumer-level GPS cannot consistently provide accurate localization (with an average error exceeding 5 m). In certain scenarios (such as underground parking garages), vehicles can even fail to obtain GNSS signals for extended periods. Therefore, a significant gap remains between existing visual SLAM technologies and the vision of applying SLAM in autonomous driving.

4 System Design

4.1 System Overview

As discussed in Sec. 3.2, visual SLAM technology faces substantial challenges in driving scenarios due to significant mapping errors in complex traffic environments and the unreliability of error correction methods. This work addresses these challenges by utilizing intelligent roadside infrastructure to enhance vehicle SLAM. The stationary nature of the roadside infrastructure enables it to obtain accurate measurements of the environmental structure and global localization, which serve as reliable references to correct the accumulative errors in visual SLAM. By receiving and exploiting such references when continuously passing infrastructure nodes, vehicles can achieve high-performance SLAM over extensive distances in the absence of looped paths or precise global localization.

We propose VILAM, the first infrastructure-assisted vehicle SLAM system that harnesses LiDAR measurements from distributed roadside infrastructures as global references to enhance visual SLAM on the vehicle in real time. Fig. 2 shows the overview of VILAM. Specifically, VILAM consists of one module on infrastructure and three modules on the vehicle.

On the infrastructure side, the *infrastructure measurement extraction* module (Sec. 4.3) periodically extracts a lightweight scene representation from the accumulated infrastructure LiDAR point clouds. By filtering out dynamic objects and merging redundant points, it obtains a high-quality yet compact measurement of the environmental structure. This refined measurement, along with the infrastructure's location, are utilized as *global references* and broadcast to all nearby vehicles.

On the vehicle side, the *local map construction* (Sec. 4.2) module employs existing visual SLAM modules based on the onboard sensor to keep constructing an initial 3D local map. To be compatible with diverse visual SLAM frameworks, it extracts three types of data that are common to most of the existing approaches: the feature point map, the historical vehicle localization (i.e., the vehicle trajectory), and the visibility relationship between them, for the following processing with the infrastructure measurement. Once the vehicle receives the global reference from the infrastructure, the *vehicle-infrastructure alignment* (Sec. 4.4) module aligns the local feature point map with the infrastructure measurement to correct the overlapped part of the local map and the latest vehicle trajectory. In particular, we devise a novel elastic alignment approach to address the deformation in local 3D maps that renders conventional rigid-oriented point cloud registration methods ineffective. Finally, the *global map correction* (Sec. 4.5) module leverages the corrected part of the vehicle trajectory to amend the historical vehicle trajectory and the entire feature point map. This module adopts a novel factor graph-based representation to encode only the lightweight vehicle trajectory and the alignment results. By optimizing the factor graph to iteratively correct the historical trajectory, it can infer a consistent global map efficiently based on the visibility relationship between the vehicle trajectory and the initial feature point map. To minimize compute overhead, the *local map construction* module runs continuously to update the local map and vehicle trajectory. Concurrently, the *vehicle-infrastructure alignment* and *global map correction* modules, triggered by global references from the infrastructure, operate in parallel. This parallel processing strategy guarantees efficient computation and timely updates for the map and localization.

4.2 Local Map Construction

We employ the tracking and the local mapping modules from visual SLAM frameworks (see Sec. 3.1) on the vehicle to continuously estimate the vehicle localization and construct an initial local map based on the camera image sequences. However, existing visual SLAM frameworks exhibit diverse designs due to varying sensor configurations and feature extraction methods [4, 38, 53], resulting in a multitude of map representations and data formats. To ensure robust compatibility across these visual SLAM frameworks, VILAM carefully extracts three data types: (i) The initial 3D feature point map $\mathcal{M}$. (ii) The estimated vehicle trajectory $\mathcal{T}$. (iii) The visibility relationship $\mathcal{C}$ between $\mathcal{M}$ and $\mathcal{T}$. This visibility relationship indicates that a feature point in $\mathcal{M}$ can be observed by the vehicle at different positions along $\mathcal{T}$, denoted by a set of tuples as

$$\mathcal{C} = \left\{(i,j,u,v)\middle|\mathbf{T}_i \in \mathcal{T}, \mathbf{m}_j \in \mathcal{M}, u,v \in \mathbb{Z}\right\} \qquad (1)$$

where the tuple (i,j,u,v) indicates that the map point $\mathbf{m}_j$ can be observed by vehicle at pose $\mathbf{T}_i$, and this observation corresponds to a 2D feature point (u,v) on the camera image captured at $\mathbf{T}_i$. These three types of data are common to most existing visual SLAM pipelines, depicting all the map point information in the initial local map and the spatial-temporal relationships between them.

4.3 Infrastructure Measurement Extraction

This module runs on each roadside infrastructure node to generate an accurate 3D measurement of the surrounding environment with precise localization, utilizing the data from the LiDAR on the infrastructure. This measurement is a set of points processed from LiDAR point clouds over a period of time. Thanks to the precise nature of LiDAR and the immovable location of the infrastructure, these points have global coordinates with errors within millimeters. Therefore, measurements from infrastructure distributed along the vehicle trajectory can serve as ideal global references to assist visual SLAM.

Measurement Extraction. LiDAR point clouds can be highly dynamic in traffic scenarios due to moving objects such as vehicles and pedestrians. Points belonging to moving objects may not have correspondences in the 3D local map constructed by the vehicle and thus cannot be used as global references for visual SLAM. Existing methods [14] for filtering moving objects, which typically use deep neural networks, can cause substantial computation overhead. Inspired by [24], we design a lightweight approach to eliminate the dynamic points, utilizing the accumulated point cloud over a period of time. Moreover, such accumulated point clouds have higher resolution to achieve a more accurate measurement of the environment structure. Specifically, we accumulate the LiDAR frames over a time period (e.g., 5 seconds) as a dense point cloud. We then voxelize this point cloud into voxel grids and calculate the occupancy of each grid (i.e., the number of points located in that grid). Static objects are present at consistent positions across all the LiDAR frames so the corresponding voxels have higher occupancy, and vice versa. Therefore, we filter out the points in voxels with low occupancy and merge the rest of the points to the center of each voxel, thus obtaining an accurate but extremely compact measurement of the static scene. Finally, we estimate the planar feature (i.e., the parameter of the tangent plane) for each point in the remaining point cloud and combine it as the feature of this point. Note that planar features not only are presented in the structures like roads and walls, but also can be extracted by differentiation from curved surfaces such as tree trunks. We denote the final point set from the kth infrastructure node as $\mathcal{P}_k$.

I-V Transmission. Given that $\mathcal{P}_k$ comprises only information regarding static scenes around the infrastructure, it can maintain consistency over a period of time. This allows for low-frequency measurement extraction on the infrastructure and a one-shot operation to transmit the measurement from the infrastructure to each vehicle. This approach substantially diminishes the computational and communication overhead on the infrastructure. Moreover, the measurement of an infrastructure node can be broadcast to all passing vehicles, enhancing the scalability of this infrastructure-assisted SLAM framework.

4.4 Vehicle-Infrastructure Alignment

When the vehicle passes an infrastructure node, there is usually a significant overlap between the field of views of the sensors on the vehicle and the infrastructure, which means that the LiDAR measurement from the infrastructure (i.e., $\mathcal{P}_k$)

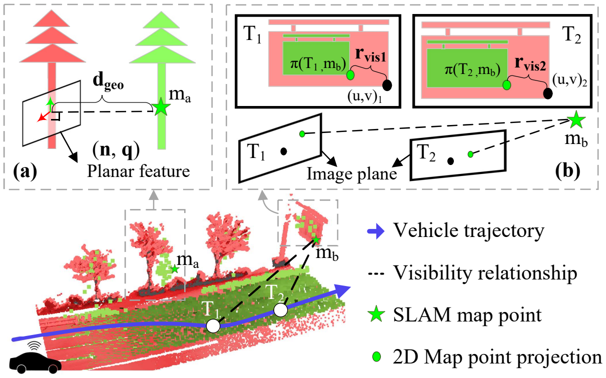


Figure 3: Illustration of (a) *geometry constraint* and (b) *visibility constraint* in the vehicle-infrastructure alignment. The red point cloud is the global reference from the infrastructure, and the green one is the constructed local map.

and the corresponding part of the vehicle's local map (i.e., $\mathcal{M}$) depict the same scene. Since $\mathcal{P}_k$ and $\mathcal{M}$ are both represented by 3D point sets with real-world coordinates, by aligning $\mathcal{M}$ to the global reference $\mathcal{P}_k$, the vehicle can correct the errors in the overlapped part of $\mathcal{M}$ and the corresponding part of vehicle trajectory $\mathcal{T}$. We denoted these local map and vehicle trajectory segments as $\tilde{\mathcal{M}}$ and $\tilde{\mathcal{T}}$, respectively.

Conventional point cloud alignment algorithms, also known as point cloud registration, are predicated on the rigid-body assumption of point clouds, making them ill-suited for this alignment task. These algorithms assume a rigid-body transformation relationship between $\tilde{\mathcal{M}}$ and $\mathcal{P}_k$, suggesting that the misalignment for each map point $\mathbf{m}_j$ in $\tilde{\mathcal{M}}$ is uniform. However, as discussed in Sec. 3.1, $\mathcal{M}$ is accumulated frame-by-frame during the visual SLAM process on the vehicle, with errors from both feature point extraction and vehicle pose estimation being integrated and compounded in $\mathcal{M}$. As a result, the misalignment distribution between each segment of $\tilde{\mathcal{M}}$ and the corresponding segment in $\mathcal{P}_k$ is not uniform.

To address this issue, we design an elastic point cloud registration approach. Our key idea is to optimize each part of $\tilde{\mathcal{M}}$ independently to achieve maximum overlap with $\mathcal{P}_k$ after alignment. This optimization is conducted based on two types of constraint: *geometry constraint* and *visibility constraint*. As illustrated in Fig. 3, the *geometric constraint* restricts the points in $\tilde{\mathcal{M}}$ to their closest planes in $\mathcal{P}_k$, while the *visibility constraint* further optimizes the position of each point in the plane using its coordinates from different viewpoints on the trajectory.

Geometry Constraint. We first associate each map point $\mathbf{m}_j$ with a plane in $\mathcal{P}_k$ by searching its nearest plane feature $(\mathbf{n}_j, \mathbf{q}_j)$ estimated during infrastructure measurement extraction (see Sec. 4.3), where $\mathbf{n}_j$ represents the plane's normal vector and $\mathbf{q}_j$ denotes the plane's center. The distance from $\mathbf{m}_j$ to the plane can be calculated by:

$$\mathbf{d_{geo}}(\mathbf{m}_j) = (\mathbf{m}_j - \mathbf{q}_j)^T \cdot \mathbf{n}_j \tag{2}$$

which denotes the misalignment between $\mathbf{m}_j$ and its corresponding object in $\mathcal{P}_k$. Therefore, we aim to minimize $\mathbf{d_{geo}}(\mathbf{m}_j)$ across the map segment $\tilde{\mathcal{M}}$ to constrain all the map points closer to $\mathcal{P}_k$.

Visibility Constraint. The geometry constraint can locate $\mathbf{m}_j$ to a plane but cannot determine its position within the plane. We propose to utilize the vehicle's visibility of $\mathbf{m}_j$ at different positions on $\tilde{\mathcal{T}}$ to further constrain its coordinate. As discussed in Sec. 3.1, a map point $\mathbf{m}_j$ can be observed by the vehicle from different locations on the trajectory, which is represented by the visibility relationship $\mathcal{C}$. We project $\mathbf{m}_j$ back to the images captured at these locations. If $\mathbf{m}_j$ is accurate, its projection point should be consistent with the corresponding 2D feature point. We calculate the distance between them by:

$$\mathbf{r_{vis}}(\mathbf{m}_j, \tilde{\mathcal{T}}) = \sum_i \left\| \pi\left((\mathbf{T}_i)^{-1}\mathbf{m}_j\right) - (u,v)^T \right\| \tag{3}$$

where $\pi(\cdot)$ denotes the projection function and $(i, j, u, v) \in \mathcal{C}$. Therefore, minimizing $\mathbf{r_{vis}}(\mathbf{m}_j, \tilde{\mathcal{T}})$ provides another constraint on the map points to ensure their consistency from different viewpoints on the vehicle trajectory.

By combining the two types of constraints across all map points, we formulate the point cloud alignment of $\tilde{\mathcal{M}}$ and $\tilde{\mathcal{T}}$ as an optimization problem:

$$\tilde{\mathcal{T}}^*, \tilde{\mathcal{M}}^* = \arg\min_{\tilde{\mathcal{T}}, \tilde{\mathcal{M}}} \sum_j \left\| \begin{bmatrix} \mathbf{d_{geo}} \\ \mathbf{r_{vis}} \end{bmatrix} \right\|_2^2 \tag{4}$$

where we leverage 2-norm to combine $\mathbf{d_{geo}}$ and $\mathbf{r_{vis}}$ as they can be approximated as the normal and radial distances between $\mathbf{m}_j$ and the groundtruth, since $\mathbf{d_{geo}}$ are usually parallel to the direction of the vehicle due to the depth estimation error, while the $\mathbf{r_{vis}}$ are within the image plane, which is usually perpendicular to the vehicle direction. Given the high dimension of the optimization variables (there can be thousands of map points in $\tilde{\mathcal{M}}$), we employ the sparse Levenberg-Marquardt algorithm [40] to solve this optimization problem.

4.5 Global Map Correction

After the alignment, we have the latest part of the initial local map and the vehicle trajectory corrected (i.e., $\tilde{\mathcal{M}}^*$ and $\tilde{\mathcal{T}}^*$) based on the global references from the infrastructure. Ideally, if the roadside infrastructure nodes are densely distributed with their field of view covering the entire road, an accurate global map can be simply obtained by stitching all the corrected map segments. However, given the complexity of real-world road environments and the varied density of roadside infrastructure distribution, it is challenging for roadside sensors to achieve comprehensive coverage of roadways. Therefore, we need to utilize $\tilde{\mathcal{M}}^*$ and $\tilde{\mathcal{T}}^*$ to correct the historical part of $\mathcal{M}$ and $\mathcal{T}$ and obtain a consistent global map. However, directly conducting the correction on $\mathcal{M}$ may incur

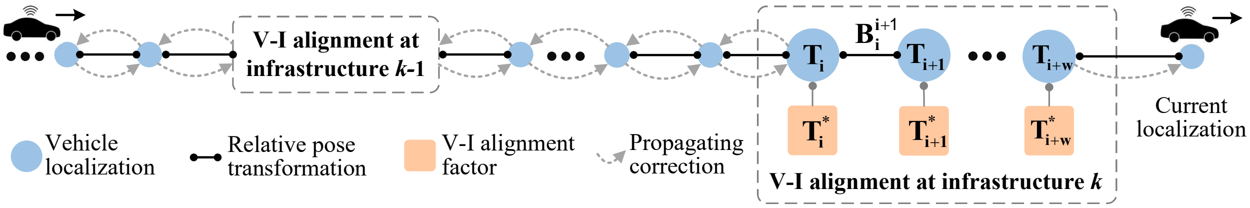


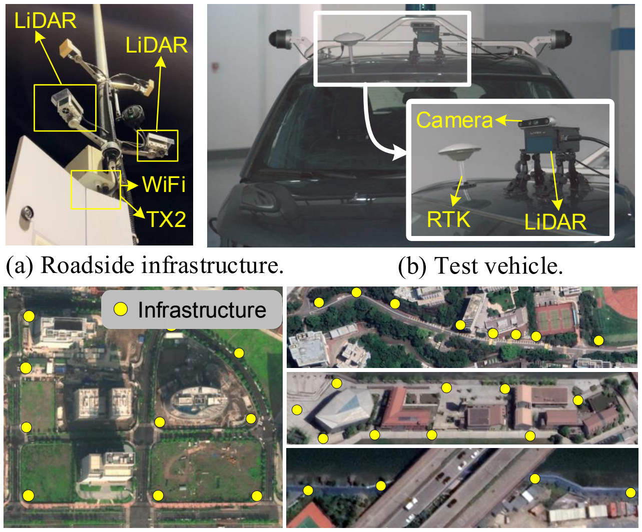
Figure 4: Illustration of the factor graph construction and optimization. The nodes and factors denote the poses along the original vehicle trajectory and the trajectory corrected with global references, respectively. By optimizing this factor graph, VILAM can efficiently correct historical and upcoming vehicle poses.

significant compute overhead since numerous map points in $\mathcal{M}$ have to be adjusted one by one based on $\tilde{\mathcal{M}}^*$. To address this challenge, we adopt a factor graphed-based representation to encode only the original vehicle trajectory $\mathcal{T}$ and its corrected part $\tilde{\mathcal{T}}^*$. By optimizing this lightweight factor graph, we can obtain the corrected historical trajectory and then infer the entire feature point map efficiently based on it.

Specifically, Fig. 4 shows the construction and optimization of the factor graph. The nodes in the graph denote the 3D poses along the original vehicle trajectory $\mathcal{T}$, and the binary edge $\mathbf{B}_i^{i+1}$ between two neighbor nodes represents the relative pose transformation between them. For the trajectory segment $\tilde{\mathcal{T}}$ corrected by the global references from the infrastructure, the corrected coordinate $\mathbf{T}_i^*$ acts as the factor of the node $\mathbf{T}_i$. We aim to optimize the value of nodes without factors based on all the binary edges and the factors. It can be described by the following optimization problem:

$$\mathcal{T}^* = \underset{\mathcal{T}}{\operatorname{argmin}} \sum_{\mathbf{T}_i \in \mathcal{T}} \left\| E\left(\mathbf{T}_{i+1}, \mathbf{T}_i \mathbf{B}_i^{i+1}\right) \right\|^2 + \sum_{\mathbf{T}_i^* \in \tilde{\mathcal{T}}^*} \left\| E\left(\mathbf{T}_i, \mathbf{T}_i^*\right) \right\|^2, \tag{5}$$

where $E(\cdot,\cdot)$ calculates the errors between two poses. This problem can be efficiently solved by the incremental factor graph optimization method [21], which iteratively corrects each node value under the constraints of its connected edges in a propagation manner. After such optimization processing over the entire graph, we obtain the entire corrected trajectory $\mathcal{T}^*$. Utilizing the visibility relationship $\mathcal{C}$ between the trajectory and the feature points in $\mathcal{M}$, we can infer the map points based on $\mathcal{T}^*$ and finally reconstruct a consistent global map.



(a) Roadside infrastructure. (b) Test vehicle.

(c) Test scenarios.

Figure 5: A real-world smart infrastructure testbed deployed in diverse scenarios for data collection and system evaluation.

5 Testbed and Dataset

Testbed. Fig. 5 (a) and (b) show the setups of the roadside infrastructure and the test vehicle in the real-world. Each roadside infrastructure is equipped with two Livox Horizon LiDARs installed at a height of 3.5m, covering both sides of the facility in a rear-facing configuration. It should be viable to apply VILAM's idea to 3D cameras on the infrastructure, and we have left this to future work. Additionally, it has an Nvidia Jetson TX2 computing module with Wi-Fi, capable of simple local information processing and communicating with the vehicle. The vehicle is equipped with a Realsense D455 camera (with built-in IMU). The camera is only used to collect raw images and IMU measurements, without depth information. The on-vehicle computing/communication unit is a laptop with an AMD Ryzen 2.90 GHz CPU and an NVIDIA RTX 2060 GPU. Additionally, the test vehicle is equipped with a Livox HAP LiDAR and an Asense ins570d RTK-GNSS receiver for ground truth collection. Specifically, we repeatedly scan the roads with LiDAR and merge the point clouds offline using an existing mapping system [41] to create the ground-truth map. The vehicle's ground-truth trajectory is obtained by aligning the point cloud frames captured by the LiDAR with the ground-truth map.

Real-world Dataset. We collect an extensive real-world dataset across five typical scenarios using our infrastructure and vehicle testbeds, as shown in Fig. 5(c). Table 1 details the data collected in each scenario. Each of these scenarios poses challenges for existing visual SLAM systems: (i) In the open roads and campus scenarios, there are few closed loops in the vehicle trajectories. (ii) The industrial park and underpasses have limited GNSS signal coverage ($\sim 60\%$), which is completely absent in the underground parking facil-

Table 1: Summary of the test scenarios, where "Traj." represents the vehicle trajectories, "GPS" represents the coverage of GNSS signals in the scenario, "Infra." represents infrastructure.

Scenario	Traj. length	Images	GPS	Infra. nodes
Open Road	6.6 km	45.1 k	~90%	13
Campus	1.3 km	15.4 k	~80%	8
Industrial park	5.9 km	40.2 k	~60%	9
Underpasses	0.4 km	3.6 k	~60%	4
Underground parking	3.4 km	33.0 k	0%	10

ity. The average infrastructure coverage across all scenarios is around 60%. We quantify the infrastructure coverage by calculating the proportion of the vehicle's ground-truth trajectory positions that fall within the infrastructure's point cloud. The inter-infrastructure node distance lies between $20m$ and $170m$. The vehicle's average speed is $5m/s$ (up to $10m/s$) due to the speed limits in most test scenarios. In summary, our dataset covers a total of $17.6km$ in driving trajectories, including $137.3k$ image frames captured by vehicles and $55.2k$ point cloud frames from 44 infrastructure nodes. This research has been granted IRB approval.

6 Evaluation

6.1 Evaluation Setup and Metrics

6.1.1 Experiment Setup

On the infrastructure side, the data rate of the LiDAR point cloud is 10 fps. We utilize a 5-second point cloud sequence for measurement extraction and set the voxel grid size to $0.5m$. On the vehicle side, for the *local map construction* task, we configure the vehicle camera to capture images at a data rate of 15 fps and IMU measurements at 200 fps. We set the image resolution to 768x480. For the *V-I alignment* task, we set the length of the local map segment to $30m$.

6.1.2 Evaluation Metrics

APE, ARE and RTE. We employ Average Positioning Error (APE) and Average Rotation Error (ARE) [36] to evaluate the absolute accuracy of the trajectory estimated by the SLAM algorithm. Specifically, APE quantifies the translational discrepancies between the estimated trajectory and the ground truth trajectory at each frame: $APE = \frac{1}{n}\sum_i^n \left\| E_t\left(trans(\mathbf{T_i^{est}}), trans(\mathbf{T_i^{gt}})\right) \right\|$, where n denotes the frame number. Similarly, ARE measures the rotational discrepancies between the estimated and the ground truth trajectory: $ARE = \frac{1}{n}\sum_i^n \left\| E_r\left(rot(\mathbf{T_i^{est}}), rot(\mathbf{T_i^{gt}})\right) \right\|$. Furthermore, we leverage the Relative Trajectory Error (RTE) from the KITTI Benchmark [12] to compute the degree of trajectory drift over time.

Chamfer Distance. We adopt the Chamfer Distance (CD) to assess the difference between the point cloud map X constructed by the SLAM algorithm and the ground truth map Y. CD is computed by summing the squared distances between nearest neighbor correspondences of two point clouds: $CD(X,Y) = \sum_{x\in X} \min_{y\in Y} ||x-y||_2^2 + \sum_{y\in Y} \min_{x\in X} ||x-y||_2^2$.

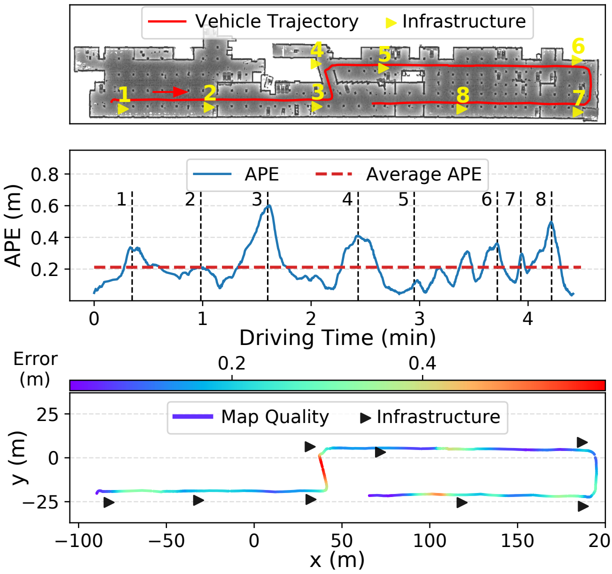


Figure 6: An end-to-end evaluation in the underground parking scenario. Above: the vehicle trajectory and infrastructure locations in the test scene. Middle: the error of the estimated vehicle localization during the driving trace. Bottom: the final map quality along the vehicle trajectory.

6.1.3 Baselines.

We implement three state-of-the-art visual SLAM algorithms as baselines. OpenVINS [13] is a conventional local mapping framework without error minimization. ORB-SLAM3 [4] and GVINS [5] employ loop-closures and GNSS locations, respectively, as global constraints to correct the cumulative drift. VILAM can incorporate all these algorithms within the *local map construction* module. However, due to the diverse GNSS coverage across different test scenarios, we adopt ORB-SLAM3 in VILAM's implementation for evaluation unless otherwise noted.

6.2 End-to-end System Evaluation

We evaluate the end-to-end system performance of VILAM in real-world scenarios described in Table 1. Fig. 6 illustrates the performance of VILAM in the Underground Parking scenario. A video clip of the complete localization and mapping result of the test vehicle is available at https://youtu.be/lTlqDNipDVE. In the upper subfigure, the grey background represents the floorplan of the parking lot extracted from the ground truth map, the red line depicts the driving trajectory, and the yellow dots represent the locations of the infrastructure nodes. The node is numbered in the

Table 2: Trajectory accuracy comparison of VILAM with existing SLAM methods across various scenarios. The unit of the APE metric is meters.

Method	Open roads		Campus		Industrial park		Underpasses		UDG parking		**All Scenarios**	
	APE	RTE	APE	RTE	APE	RTE	APE	RTE	APE	RTE	APE	RTE
OpenVINS [13]	8.170	3.54%	10.406	9.92%	4.772	1.48%	11.30	6.71%	6.144	3.37%	8.158	5.01%
OpenVINS + VILAM	**0.355**	**1.46%**	**0.493**	**3.93%**	**0.380**	**0.60%**	**0.347**	**3.04%**	**0.294**	**0.71%**	**0.373**	**1.94%**
ORB-SLAM3 [4]	5.745	2.91%	2.646	2.45%	2.556	1.60%	3.568	3.85%	3.278	2.06%	3.558	2.58%
ORB-SLAM3 + VILAM	**0.351**	**1.87%**	**0.427**	**1.75%**	**0.296**	**1.16%**	**0.359**	**2.63%**	**0.219**	**0.76%**	**0.330**	**1.63%**
GVINS [5]	2.456	3.37%	1.710	3.52%	3.505	2.41%	4.219	3.69%	4.059	2.78%	3.190	3.15%
GVINS + VILAM	**0.403**	**2.30%**	**0.354**	**2.38%**	**0.325**	**0.65%**	**0.340**	**3.13%**	**0.249**	**0.42%**	**0.335**	**1.77%**

order the vehicle passes by. The middle subfigure shows the real-time localization accuracy (APE) of VILAM. The numbered vertical lines represent the moment the vehicle encounters an infrastructure node. We observe that, in the areas between infrastructures, there is a slight upward trend in the localization error. The error is corrected once the vehicle passes an infrastructure and optimizes its local map through *V-I alignment*. The lower subfigure shows the trajectory estimated by VILAM and the accuracy of the constructed map. We color-code the vehicle trajectory using the map-construction error (Chamfer Distance) of the corresponding region. It can be seen that, with *global map correction*, the error distribution of the map maintains consistent uniformity in both infrastructure-covered and uncovered regions. We further evaluate the impact of the coverage of infrastructure deployment on VILAM's performance in Sec. 6.6.

6.3 Performance of VILAM

6.3.1 Trajectory Estimation

As mentioned in Sec. 4, VILAM tracks the vehicle by associating image frames to derive the vehicle trajectory. Therefore, the accuracy of the estimated trajectory indicates VILAM's continuous real-time localization performance. Table 2 presents a comparison of trajectory accuracy between the baselines and VILAM implementations based on them across five testing scenarios. The results show that all VILAM implementations outperform the corresponding baseline algorithms. Specifically, OpenVINS exhibits substantial errors (i.e., over 8.1 m on average) in all scenarios. ORB-SLAM3 and GVINS enhance accuracy by introducing global constraints but still have significant errors, especially in scenarios that lack loop paths (e.g., open roads and underpasses) and GNSS coverage (e.g., underground parking), respectively. In contrast, VILAM achieves an APE within 0.4 m and an RTE within 2% on average across all scenarios while demonstrating robustness to the algorithms used in the *local map construction* module.

Fig. 7 shows a further evaluation of the benefits of the VILAM framework exploiting global references from roadside infrastructure. "Baseline" and "VILAM " refer to the original baseline algorithms and the VILAM implementations using these algorithms in the *local map construction* module, respectively. "w/ Landmark" and "w/ Infra." indicates the modified baseline algorithms. "w/ Landmark" correct SLAM by utilizing the detected infrastructure positions from vehicle-side images as reference landmarks. "w/ Infra." correct the SLAM results by directly integrating the point clouds from the infrastructure through a conventional point cloud registration method [26]. We can find that simply utilizing information from the infrastructure can improve the performance of the baseline algorithms. However, such improvements are minor for ORB-SLAM3 and GVINS, as they utilize loop detection and GNSS to correct some errors, respectively. In contrast, VILAM significantly outperforms "w/ Infra." and "w/ Landmark" across all three basic algorithm implementations, achieving over an 80% reduction in APE. This is because VILAM's elastic alignment approach overcomes the deformation in the local map, thereby significantly improving alignment accuracy. Moreover, the *global map correction* module of VILAM utilizes the alignment results to optimize both the historical map and the real-time localization results. We delve further into the performance of these two modules in Section 6.5.

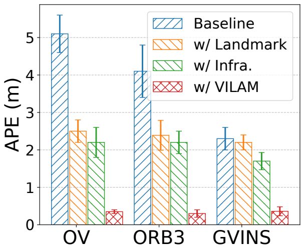


Figure 7: Performance improvement of VILAM using different SLAM methods for *local map construction*.

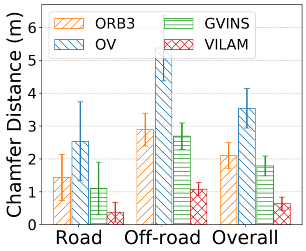


Figure 8: The Chamfer Distance between the constructed map and the ground truth map.

6.3.2 Map Quality Evaluation

In this section, we evaluate the quality of the map constructed by VILAM. Like mainstream SLAM methods, VILAM builds a 3D feature point map, primarily used for vehicle relocalization by reusing the map. Therefore, we mainly focus on the accuracy of the reconstructed map points' coordinates and the overall consistency of the map.

Fig. 8 compares the CD between the estimated map of

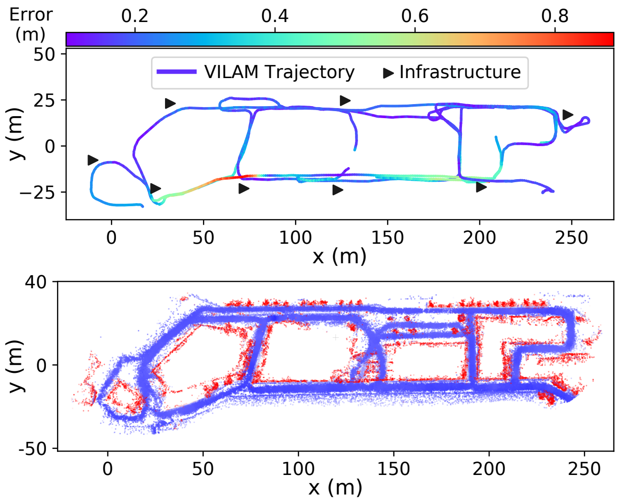


Figure 9: Visualization of the trajectory and the 3D feature point map estimated by VILAM. Above: the infrastructure locations and the estimated vehicle trajectory by VILAM. Bellow: the bird-eye view of the 3D map built by VILAM, where colors(blue to red) indicate different point heights.

VILAM / baseline SLAM and the ground-truth map. The higher the CD, the greater the error of the estimated map. As autonomous driving perception tasks primarily focus on road information and relocalization tasks mainly concern off-road three-dimensional structure and texture features, we divide the map points into two main categories: road and Off-road structure. Consistent with the trajectory evaluation results, baseline methods incorporating global constraints such as loop-closing or GNSS (ORB-SLAM and GVINS) exhibit an improvement over approaches relying solely on Local Mapping. However, the overall map accuracy of these methods still falls short when compared to the ground truth. VILAM addresses this limitation through its *global map correction* module. By effectively integrating corrections from multiple infrastructure-based VI-Alignment with the factor graph, VILAM enhances global consistency and improves map accuracy. We visualize the point cloud map constructed by VILAM in Fig. 9. Due to the high accuracy and consistency of the map, the structures of buildings around the road are clearly discernible.

6.4 System Overhead

System Latency. In Fig. 10, we present the end-to-end latency distribution of VILAM throughout a driving trace. Here, "Local Map" represents *local map construction*, and "Global Map" represents *global map correction* . The infrastructure-related tasks, namely *I-V Transmission*, *V-I alignment*, and *global map correction*, are only triggered when the vehicle encounters an infrastructure. VILAM operates with minimal computational overhead during intervals without infrastructure coverage, as only the *local map construction* task remains active. A detailed timeline as the vehicle passes through one of the infrastructures is illustrated in Fig. 11. Thanks to VILAM's concurrent design, *local map construction* is not blacked by the infrastructure-related tasks. This ensures it outputs continuous vehicle localization based on previously corrected maps, maintaining real-time system performance. It is critical for downstream tasks that require real-time localization, such as trajectory planning. The average latency for infrastructure-related tasks is about 0.35 s, ensuring a near real-time correction of both the map and localization after the vehicle passes through an infrastructure.

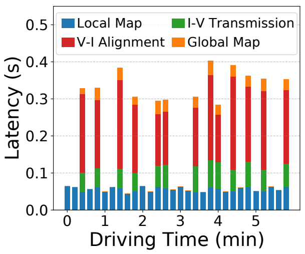


Figure 10: End-to-end latency of VLIAM on the vehicle side.

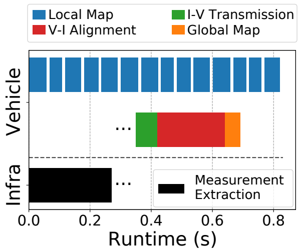


Figure 11: The detailed runtime of each module.

Table 3: The average size and transmission time of the shared data measured on an 802.11ac network. PC denotes "Point Cloud", and SSR denotes "Static Scene Representation."

Data type	Trans mode	Sync	Size	Time
Raw PC	Continuous	Required	21.1 MB	3.93s
Downsampled PC	Continuous	Required	4.1 MB	0.78s
VILAM SSR	**Single-shot**	–	**236.3 KB**	**0.04s**

I-V Transmission. We further evaluated the data transmission overhead associated with the vehicle's acquisition of point clouds from infrastructure. The results are presented in Table. 3. When directly streaming a raw point cloud sequence (about 100 frames) to the vehicle, the data transfer volume exceeds 20 MB, and it is also burdened by a considerable transmission latency. After performing voxel downsampling on each point cloud frame, the number of points is reduced, yet the final data volume still surpasses 4 MB. This can be attributed to the fact that points in each frame contain redundant measurements of static structures within the scene. As discussed in Sec. 4.3, VILAM addresses this by merging the points of static structures across multiple frames, effectively eliminating redundancy. Consequently, the final transmitted Static Scene Representation requires a mere 236 KB of data.

Influence of Infrastructure Measurement Precision. In Sec. 4.3, the voxel grid size influences the size and precision of the extracted representation, which in turn affects the latency of I-V transmission and the subsequent performance of the *V-I alignment*. As illustrated in Fig. 12, when the voxel size is increased, more redundant points occupying the same spatial location are merged, leading to a consistent reduc-

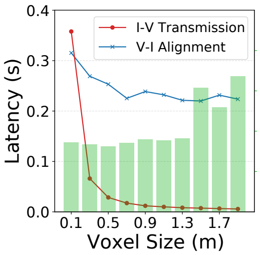


Figure 12: The impact of voxel size on the system overhead and the performance of VILAM.

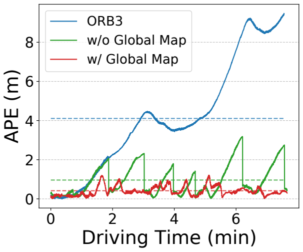


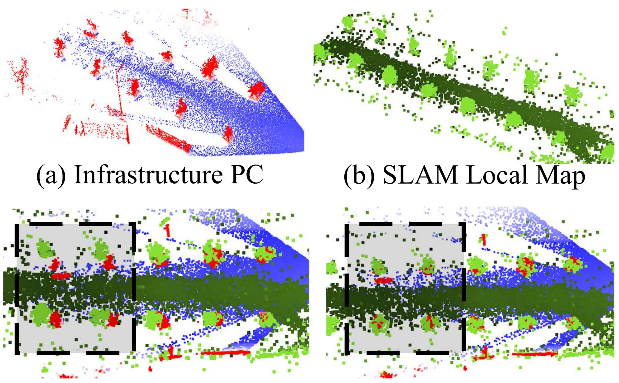
Figure 13: The influence of the *global map correction* module on the performance of VILAM.

tion in the size of the representation and thereby significantly decreasing the transmission latency. Meanwhile, the latency of *V-I alignment* only reduces at first. This reduction is due to the decreased point number after the voxel grid partition, resulting in faster neighboring searches during alignment. However, when the voxel size exceeds about $0.7m$, this latency reduction plateaus because the reduced precision of the representation at larger voxel sizes means that the joint optimization process requires more time to converge. The green bars in Fig. 12 depict the influence of voxel size on the accuracy of the *V-I alignment*. There is a slight degradation in the accuracy with the voxel size increases. In our practical implementation, we set the voxel size to $0.5m$, which strikes a balance by ensuring lower latency without compromising accuracy.

6.5 Micro Benchmarks

Performance of V-I Alignment. The result of the *V-I alignment* directly influences the accuracy of the global constraints that VILAM derives from the infrastructure. In this section, we provide a detailed evaluation of the performance of *V-I alignment*. We compare the following two registration approaches with the proposed *V-I alignment* method. 1) image-to-point (I2P): The continuous frame images acquired by the vehicle near an infrastructure are sequentially registered with the infrastructure point cloud. 2) Point-to-point (P2P): Align the vehicle's local map with the infrastructure point cloud using 3D point cloud registration algorithms.

As shown in Table. 4, CorrI2P [39] exhibits large overall error. This is mainly attributed to the significant perspective difference between the vehicle's camera and the infrastructure's LiDAR. Consequently, few feature correspondences are matched between the vehicle camera images and the infrastructure point cloud, leading to lower registration accuracy and poor robustness. P2P approaches, on the other hand, utilize the entire local point cloud map from the vehicle side, efficiently improving the number of corresponding features and enhancing the completeness of the perception perspective compared to the I2P method. However, as described in Sec. 4.4, the local map exhibits deformations due to cumulative errors. Rigid body-based registration methods, such as MAC [51] and Fast-GICP [26], struggle to accurately align most regions of the local map to the infrastructure measurement. Taking Fast-GICP as an example, as depicted in Fig. 14 (c), only the starting region of the local map matches well with the infrastructure point cloud. Yet, there is a significant deviation towards the end of the local map (highlighted by the grey rectangle).


(a) Infrastructure PC (b) SLAM Local Map

(c) Fast-GICP Result (d) V-I Alignment Result

Figure 14: Visualization of the registration between the vehicle local map and the infrastructure measurements.

Table 4: Performance of the proposed elastic alignment method and conventional registration algorithms. "VILAM VIA" denotes the V-I Alignemnt module of VILAM, "I2P" represents image-to-point cloud registration algorithm, and "P2P" represents point cloud-to-point cloud registration algorithm.

Method	Type	APE (m)	ARE (°)	Time (s)
CorrI2P [39]	I2P	1.74 ± 0.65	10.31 ± 2.29	2.98
MAC [51]	P2P	1.57 ± 0.27	8.71 ± 2.36	0.65
Fast-GICP [26]		1.36 ± 0.34	9.07 ± 2.19	**0.13**
S-ICP [9]		1.07 ± 0.39	7.39 ± 1.85	0.61
VILAM		**0.31** ± **0.08**	**2.29** ± **0.52**	0.24

S-ICP [9] introduces a scale factor into the rigid-body registration model, which helps alleviate errors caused by the rigid-body assumption and achieve lower APE. The proposed *V-I alignment* further divides the local map into fine-grained fragments, allowing each region of the local map to align better with the infrastructure measurement, as shown in Fig. 14 (d). This significantly reduces APE and ARE, with only a slight increase in calculation delay. Furthermore, the concurrent design of VILAM ensures that it does not impact the real-time performance of the overall system, making this latency negligible.

Performance of Global Map Correction. In this section, we evaluate the influence of the *global map correction* module on mapping consistency. We perform ORB-SLAM3, VILAM without *global map correction*, and the full VILAM setup on the data trace and assessed the resultant trajectory accuracy.

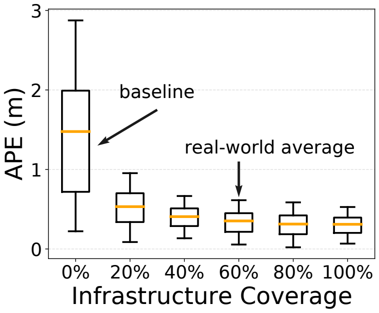


Figure 15: Performance of VILAM under different infrastructure density.

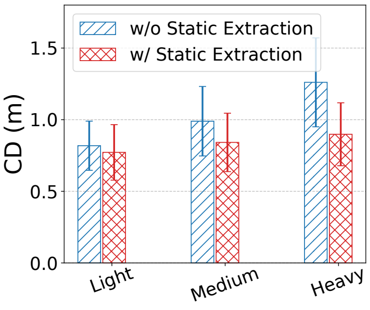


Figure 16: Performance of VILAM under different traffic conditions.

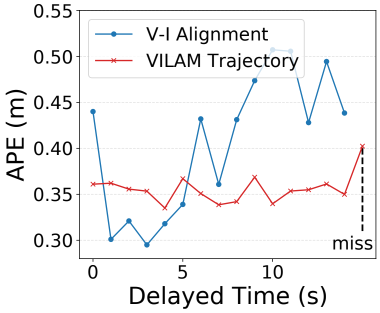


Figure 17: Performance of VILAM under different I-V Transmission latency.

The result is presented in Fig. 13. As the vehicle's driving time increases, the trajectory error of ORB-SLAM3 demonstrates a continuous growth trend due to the absence of global constraints. In the case of VILAM without *global map correction*, only the trajectories near the infrastructure are corrected by *V-I alignment*. The trajectory errors persistently escalate for the gap areas between the infrastructures, resulting in abrupt error shifts at the subsequent infrastructure points (e.g., the APE at the 2nd minute in Fig. 13). Although the APE without *global map correction* is substantially reduced compared to the baseline SLAM methods, such inconsistencies seriously degrade the map quality. Upon integrating the *global map correction* module, the overall APE remains within a low range.

6.6 Robustness Analysis

Infrastructure Coverage. As observed in Sec. 6.2, the accuracy of VILAM can be affected by the coverage of infrastructure, i.e., the proportion of the area that infrastructure LiDARs can perceive. We evaluate VILAM under varying infrastructure coverage in an underground parking garage. Specifically, we place mobile poles equipped with LiDARs at different intervals to simulate varied infrastructure coverage. The results are depicted in Fig. 15. Although the error escalates as the coverage diminishes, VILAM significantly outperforms the baseline even with extremely low infrastructure coverage (i.e., the average APE is kept around $0.5m$ at 20% infrastructure coverage). This demonstrates VILAM's robustness to diverse infrastructure setups in the real world.

Environmental Dynamics. We compare the performance of VILAM with and without static scene extraction under different traffic conditions. The results in Fig. 16 show that the error of VILAM without static scene extraction increases significantly under heavier traffic, while the CD of VILAM remains under $1m$ across all the traffic conditions. This evaluation confirms VILAM's robustness to environmental dynamics.

Delayed I-V Transmission. During the transmission of the extracted Static Scene Representation from the infrastructure to the vehicle, connectivity issues or transmission errors may delay VILAM from timely accessing the infrastructure's information for map correction. To evaluate the influence of the delayed time on VILAM's performance, we manually set latency at the infrastructure side to simulate transmission delays. The result is shown in Fig. 17. The delay time is measured from the moment when the vehicle is closest to the infrastructure. Interestingly, for *V-I alignment*, moderate transmission delays can actually enhance alignment accuracy. This benefit arises as the vehicle constructs a more extensive local map during the delayed time, leading to a larger overlap with the infrastructure point cloud. However, as the delay time further increases, the cumulative error of the local map intensifies, leading to a reduction in *V-I alignment* accuracy. Meanwhile, the overall trajectory precision of VILAM is relatively impervious to delayed times. A noticeable trajectory error increase occurs only when the vehicle entirely misses the current infrastructure's measurement. The *global map correction V-I alignment* can leverage this information to maintain a consistent trajectory as long as the vehicle eventually receives the infrastructure measurement and undergoes *V-I alignment*.

7 Conclusion

In this paper, we present VILAM, the first system that leverages distributed roadside infrastructures to accomplish high-precision and globally consistent localization and mapping for autonomous vehicles. We implement VILAM end-to-end and evaluate its performance across various challenging driving scenarios. The experiment results show that VILAM effectively enhances the performance of existing SLAM methods in terms of localization accuracy, map consistency, and system robustness.

Acknowledgements. We would like to thank our shepherd Dongsu Han and the anonymous reviewers for their valuable feedback. This work was supported in part by National Key R&D Program of China under Grant No. 2023YFB4503700, Research Grants Council (RGC) of Hong Kong under General Research Fund under Grant 14222222, the Centre for Perceptual and Interactive Intelligence (CPII) under Grant EW01610 (RP4-3), NSFC under Grant No. LY23F020026, 62303036, 62372027, 62372028 and U23B2025.

References

[1] Pwrpak7-e1, [2023] url=https://novatel.com/products/gnss-inertial-navigation-systems/combined-systems/pwrpak7d-e1, journal=Novatel.

[2] Lidar market size, [2023] url=https://www.mordorintelligence.com/industry-reports/global-lidar-market, journal=LiDAR Market Size, Overview.

[3] Fawad Ahmad, Hang Qiu, Ray Eells, Fan Bai, and Ramesh Govindan. CarMap: Fast 3d feature map updates for automobiles. In *17th USENIX Symposium on Networked Systems Design and Implementation (NSDI 20)*, pages 1063–1081, Santa Clara, CA, February 2020. USENIX Association.

[4] Carlos Campos, Richard Elvira, Juan J Gómez Rodríguez, José MM Montiel, and Juan D Tardós. Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam. *IEEE Transactions on Robotics*, 37(6):1874–1890, 2021.

[5] Shaozu Cao, Xiuyuan Lu, and Shaojie Shen. Gvins: Tightly coupled gnss–visual–inertial fusion for smooth and consistent state estimation. *IEEE Transactions on Robotics*, 38(4):2004–2021, 2022.

[6] Jun Cheng, Liyan Zhang, Qihong Chen, Xinrong Hu, and Jingcao Cai. A review of visual slam methods for autonomous driving vehicles. *Engineering Applications of Artificial Intelligence*, 114:104992, 2022.

[7] Kai-Wei Chiang, Guang-Je Tsai, Hone-Jay Chu, and Naser El-Sheimy. Performance enhancement of ins/gnss/refreshed-slam integration for acceptable lane-level navigation accuracy. *IEEE Transactions on Vehicular Technology*, 69(3):2463–2476, 2020.

[8] Christian Creß, Zhenshan Bing, and Alois C Knoll. Intelligent transportation systems using external infrastructure: A literature survey. *arXiv preprint arXiv:2112.05615*, 2021.

[9] Shaoyi Du, Nanning Zheng, Shihui Ying, Qubo You, and Yang Wu. An extension of the icp algorithm considering scale factor. In *2007 IEEE International Conference on Image Processing*, volume 5, pages V–193. IEEE, 2007.

[10] Richard Elvira, Juan D Tardós, and Jose MM Montiel. Orbslam-atlas: a robust and accurate multi-map system. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6253–6259. IEEE, 2019.

[11] Abrar Fahim, Mehedi Hasan, and Muhtasim Alam Chowdhury. Smart parking systems: comprehensive review based on various aspects. *Heliyon*, 7(5), 2021.

[12] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361, 2012.

[13] Patrick Geneva, Kevin Eckenhoff, Woosik Lee, Yulin Yang, and Guoquan Huang. Openvins: A research platform for visual-inertial estimation. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4666–4672. IEEE, 2020.

[14] Yulan Guo, Hanyun Wang, Qingyong Hu, Hao Liu, Li Liu, and Mohammed Bennamoun. Deep learning for 3d point clouds: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 43(12):4338–4364, 2020.

[15] O Hassan, I Adly, and KA Shehata. Vehicle localization system based on ir-uwb for v2i applications. In *2013 8th International Conference on Computer Engineering & Systems (ICCES)*, pages 133–137. IEEE, 2013.

[16] Yuze He, Li Ma, Zhehao Jiang, Yi Tang, and Guoliang Xing. Vi-eye: semantic-based 3d point cloud registration for infrastructure-assisted autonomous driving. In *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*, pages 573–586, 2021.

[17] F Hidayat, BR Trilaksono, and H Hindersah. Distributed multi robot simultaneous localization and mapping with consensus particle filtering. In *Journal of physics: conference series*, volume 801, page 012003. IOP Publishing, 2017.

[18] Md Hossain, Ibrahim Elshafiey, Abdulhameed Al-Sanie, et al. Cooperative vehicle positioning with multi-sensor data fusion and vehicular communications. *Wireless Networks*, 25(3):1403–1413, 2019.

[19] Mohamed Ibrahim, Ali Rostami, Bo Yu, Hansi Liu, Minitha Jawahar, Viet Nguyen, Marco Gruteser, Fan Bai, and Richard Howard. Wi-go: accurate and scalable vehicle positioning using wifi fine timing measurement. In *Proceedings of the 18th International Conference on Mobile Systems, Applications, and Services*, pages 312–324, 2020.

[20] Zhehao Jiang, Neiwen Ling, Xuan Huang, Shuyao Shi, Chenhao Wu, Xiaoguang Zhao, Zhenyu Yan, and Guoliang Xing. Coedge: A cooperative edge system for distributed real-time deep learning tasks. In *Proceedings of the 22nd International Conference on Information Processing in Sensor Networks*, pages 53–66, 2023.

[21] Michael Kaess, Hordur Johannsson, Richard Roberts, Viorela Ila, John J Leonard, and Frank Dellaert. isam2:

Incremental smoothing and mapping using the bayes tree. *The International Journal of Robotics Research*, 31(2):216–235, 2012.

[22] Jan Kallwies, Bianca Forkel, and Hans-Joachim Wuensche. Determining and improving the localization accuracy of apriltag detection. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8288–8294. IEEE, 2020.

[23] Iman Abaspur Kazerouni, Luke Fitzgerald, Gerard Dooly, and Daniel Toal. A survey of state-of-the-art on visual slam. *Expert Systems with Applications*, 205:117734, 2022.

[24] Giseop Kim and Ayoung Kim. Remove, then revert: Static point cloud map construction using multiresolution range images. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10758–10765. IEEE, 2020.

[25] Dániel Kiss-Illés, Cristina Barrado, and Esther Salamí. Gps-slam: an augmentation of the orb-slam algorithm. *Sensors*, 19(22):4973, 2019.

[26] Kenji Koide, Masashi Yokozuka, Shuji Oishi, and Atsuhiko Banno. Voxelized gicp for fast and accurate 3d point cloud registration. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11054–11059. IEEE, 2021.

[27] Juichung Kuo, Manasi Muglikar, Zichao Zhang, and Davide Scaramuzza. Redesigning slam for arbitrary multi-camera systems. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2116–2122. IEEE, 2020.

[28] Mathieu Labbé and François Michaud. Rtab-map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation. *Journal of Field Robotics*, 36(2):416–446, 2019.

[29] Pierre-Yves Lajoie, Benjamin Ramtoula, Yun Chang, Luca Carlone, and Giovanni Beltrame. Door-slam: Distributed, online, and outlier resilient slam for robotic teams. *IEEE Robotics and Automation Letters*, 5(2):1656–1663, 2020.

[30] Jiaxin Li and Gim Hee Lee. Deepi2p: Image-to-point cloud registration via deep classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15960–15969, 2021.

[31] Zheqi Li and Jidong Huang. Study on the use of qr codes as landmarks for indoor positioning: Preliminary results. In *2018 IEEE/ION position, location and navigation symposium (PLANS)*, pages 1270–1276. IEEE, 2018.

[32] Jiarong Lin, Chunran Zheng, Wei Xu, and Fu Zhang. R^2 live: A robust, real-time, lidar-inertial-visual tightly-coupled state estimator and mapping. *IEEE Robotics and Automation Letters*, 6(4):7469–7476, 2021.

[33] Zongwei Liu, Hao Jiang, Hong Tan, and Fuquan Zhao. An overview of the latest progress and core challenge of autonomous vehicle technologies. In *MATEC Web of Conferences*, volume 308, page 06002. EDP Sciences, 2020.

[34] Zhenchao Ouyang, Jiahe Cui, Xiaoyun Dong, Yanqi Li, and Jianwei Niu. Saccadefork: A lightweight multi-sensor fusion-based target detector. *Information Fusion*, 77:172–183, 2022.

[35] T Perković, P Šolić, H Zargariasl, D Čoko, and Joel JPC Rodrigues. Smart parking sensors: State of the art and performance evaluation. *Journal of Cleaner Production*, 262:121181, 2020.

[36] David Prokhorov, Dmitry Zhukov, Olga Barinova, Konushin Anton, and Anna Vorontsova. Measuring robustness of visual slam. In *2019 16th International Conference on Machine Vision Applications (MVA)*, pages 1–6. IEEE, 2019.

[37] Tong Qin, Shaozu Cao, Jie Pan, and Shaojie Shen. A general optimization-based framework for global pose estimation with multiple sensors. *arXiv preprint arXiv:1901.03642*, 2019.

[38] Tong Qin, Peiliang Li, and Shaojie Shen. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics*, 34(4):1004–1020, 2018.

[39] Siyu Ren, Yiming Zeng, Junhui Hou, and Xiaodong Chen. Corri2p: Deep image-to-point cloud registration via dense correspondence. *IEEE Transactions on Circuits and Systems for Video Technology*, 2022.

[40] Jose Jesus De Rubio. Stability analysis of the modified levenberg-marquardt algorithm for the artificial neural network training. *IEEE Transactions on Neural Networks and Learning Systems*, PP(99):1–15, 2021.

[41] Tixiao Shan, Brendan Englot, Drew Meyers, Wei Wang, Carlo Ratti, and Daniela Rus. Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5135–5142. IEEE, 2020.

[42] Tixiao Shan, Brendan Englot, Carlo Ratti, and Daniela Rus. Lvi-sam: Tightly-coupled lidar-visual-inertial odometry via smoothing and mapping. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5692–5698. IEEE, 2021.

[43] Shuyao Shi, Jiahe Cui, Zhehao Jiang, Zhenyu Yan, Guoliang Xing, Jianwei Niu, and Zhenchao Ouyang. Vips: Real-time perception fusion for infrastructure-assisted autonomous driving. In *Proceedings of the 28th Annual International Conference on Mobile Computing And Networking*, pages 133–146, 2022.

[44] Ashutosh Singandhupe and Hung Manh La. A review of slam techniques and security in autonomous driving. In *2019 third IEEE international conference on robotic computing (IRC)*, pages 602–607. IEEE, 2019.

[45] Lukas von Stumberg and Daniel Cremers. Dm-vio: Delayed marginalization visual-inertial odometry. *IEEE Robotics and Automation Letters*, 7(2):1408–1415, 2022.

[46] Konstantinos A Tsintotas, Loukas Bampis, and Antonios Gasteratos. The revisiting problem in simultaneous localization and mapping: A survey on visual loop closure detection. *IEEE Transactions on Intelligent Transportation Systems*, 23(11):19929–19953, 2022.

[47] Jessica Van Brummelen, Marie O'Brien, Dominique Gruyer, and Homayoun Najjaran. Autonomous vehicle perception: The technology of today and tomorrow. *Transportation research part C: emerging technologies*, 89:384–406, 2018.

[48] Srihaarika Vijjappu. Distributed decentralised visual slam for multi-agent systems, 2020.

[49] Jingao Xu, Hao Cao, Zheng Yang, Longfei Shangguan, Jialin Zhang, Xiaowu He, and Yunhao Liu. SwarmMap: Scaling up real-time collaborative visual SLAM at the edge. In *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*, pages 977–993, Renton, WA, April 2022. USENIX Association.

[50] Georges Younes, Daniel Asmar, Elie Shammas, and John Zelek. Keyframe-based monocular slam: design, survey, and future directions. *Robotics and Autonomous Systems*, 98:67–88, 2017.

[51] Xiyu Zhang, Jiaqi Yang, Shikun Zhang, and Yanning Zhang. 3d registration with maximal cliques. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17745–17754, 2023.

[52] Shibo Zhao, Hengrui Zhang, Peng Wang, Lucas Nogueira, and Sebastian Scherer. Super odometry: Imu-centric lidar-visual-inertial estimator for challenging environments. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8729–8736. IEEE, 2021.

[53] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R Oswald, and Marc Pollefeys. Nice-slam: Neural implicit scalable encoding for slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12786–12796, 2022.