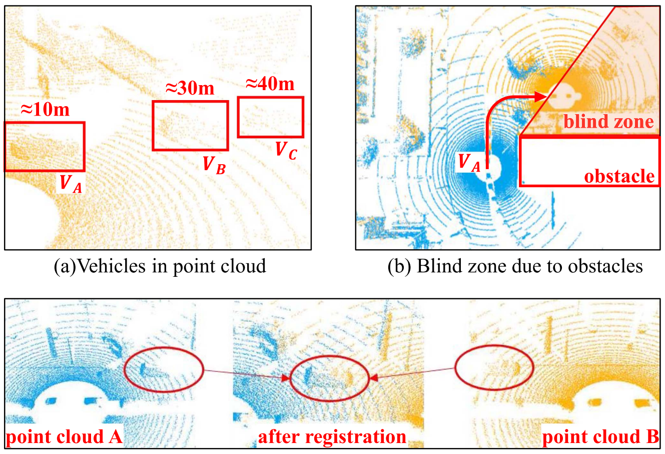
Enabling Long Range Point Cloud Registration in Vehicular Networks via Muti-Hop Relays

Zhenxi Wang, *Student Member, IEEE*, Hongzi Zhu, *Senior Member, IEEE*, Yunxiang Cai, Quan Liu, *Student Member, IEEE*, Shan Chang, *Member, IEEE*, Liang Zhang, *Student Member, IEEE*, and Minyi Guo, *Fellow, IEEE*

***Abstract*—Point cloud registration (PCR) can significantly extend the visual field and enhance the point density on distant objects, thereby improving driving safety. However, it is very challenging for vehicles to perform online registration between long-range point clouds. In this paper, we propose an online long-range PCR scheme in VANETs, called LoRaPCR, where vehicles achieve long-range registration through multi-hop short-range highly-accurate registrations. Given the NP-hardness of the problem, a heuristic algorithm is developed to determine best registration paths while leveraging the reuse of registration results to reduce computation costs. Moreover, we utilize an optimized dynamic programming algorithm to determine the transmission routes while minimizing the communication overhead. To the best of our knowledge, LoRaPCR is the first solution to achieve multi-vehicle point cloud long-range registration. Results of extensive experiments demonstrate that LoRaPCR can achieve high PCR accuracy with low relative translation and rotation errors of 0.55 meters and 1.43°, respectively, at a distance of over 100 meters, and reduce the computation overhead by more than 50% compared to the state-of-the-art method.**

***Index Terms*—Point cloud registration, cooperative sensing, multi-hop relay, VANETs.**



(a)Vehicles in point cloud (b) Blind zone due to obstacles

(c) Long-range PCR enhances vehicular perception capability

Fig. 1. Motivation of long-range PCR. (a) The resolution of a point cloud decreases with the distance to the LiDAR sensor. (b) The perception field of LiDAR is blocked by obstacles. (c) Point clouds collected on two vehicles with different viewpoints can be registered (aligned), enhancing the perception capability of both vehicles.

I. Introduction

Due to its precise panoramic view and nighttime adaptability, Light Detection and Ranging (LiDAR) sensors are installed on newly manufactured vehicles to perform tasks such as object detection [1], [2] and semantic segmentation [3], significantly improving driving safety. However, LiDAR is also accompanied by insufficient long-range resolution as illustrated in Fig. 1(a) and occlusion by obstacles as illustrated in Fig. 1(b). With vehicle-to-vehicle (V2V) communications in the Vehicular ad hoc networks (VANETs), sharing and registering point clouds between vehicles become possible [4], which enhances the perception capability of vehicles [5], [6]. For example, Fig. 1(c) demonstrates that a car, which is hard to recognize in each individual point cloud, can be identified after two of such point clouds are registered. We refer to the *online long-range point cloud registration (PCR) problem* as the problem that distant vehicles can accurately register their point clouds through one- or multi-hop V2V communication. Promisingly, if long-range point clouds of significant difference in terms of point cloud density and perspective can be well registered, it will greatly improve the performance of downstream tasks.

A feasible scheme for online long-range PCR problem in a vehicular scenario, however, has to meet four rigid requirements as follows: 1) the scheme should be able to deal with long-range registration (e.g., tens of meters) to facilitate downstream perception tasks; 2) due to the rapid movement of vehicles, the scheme should achieve online registration with low response time to ensure the timeliness of the registration results; 3) given the limited resources on a vehicle, such a scheme should be efficient in terms of computation and communication costs; 4) such a scheme should achieve high accuracy as the registered point clouds may be utilized in critical driving safety applications.

In the literature, most PCR work focuses on short-range point cloud alignment (e.g., within 10 meters), where they register point clouds with similar density and significant overlap. In

Manuscript received 29 March 2024; revised 20 July 2024; accepted 16 August 2024. Date of publication 21 August 2024; date of current version 5 November 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 61972081 and Grant 62432008, in part by the Natural Science Foundation of Shanghai under Grant 22ZR1400200, and in part by the Fundamental Research Funds for the Central Universities under Grant 2232023Y-01. Recommended for acceptance by M. Segata. *(Corresponding author: Hongzi Zhu.)*

Zhenxi Wang, Hongzi Zhu, Yunxiang Cai, Quan Liu, Liang Zhang, and Minyi Guo are with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: hongzi@sjtu.edu.cn).

Shan Chang is with the School of Computer Science and Technology, Donghua University, Shanghai 201620, China.

Digital Object Identifier 10.1109/TMC.2024.3446828

these methods, feature extractors aim to improve the quality of extracted representations, while outlier rejection methods [7], [8] attempt to distinguish erroneous responses. However, these methods fail to address long-range PCR challenges. While [6], [9] address the long-range problem by training a distance-insensitive feature extractor, these approaches lack optimization for communication overhead. EMP [5] enhances vehicles using point clouds from roadside infrastructure, but it does not consider utilizing multi-hop V2V communication to achieve long-range registration. As a result, to the best of our knowledge, there is no cost-efficient scheme which can accurately register long-distance point clouds in vehicular settings.

In our previous work presented at INFOCOM 2024, we propose *LoRaPCR*, an online long-range PCR scheme particularly designed for VANETs. The main idea of LoRaPCR is to utilize previous short-range PCRs of high accuracy to establish a superior multi-hop registration path for present long-range PCRs between each pair of vehicles. To this end, a base station (BS) or a roadside unit (RSU) first collects the information about the location, previous PCR records, and the current PCR requests of vehicles in its vicinity, and then makes the best multi-hop registration and transmission strategies, with the goal of guaranteeing a low accumulative PCR errors while optimizing the global computation and communication overheads. In this paper, we conduct a comprehensive analysis of the feasibility and error accumulation in multi-hop registration. We present experimental evidence that demonstrates LoRaPCR's compatibility with various pairwise registration methods and validate its effectiveness on real-world datasets.

Two main challenges are encountered. First, determining registration paths for the registration requests is challenging. Due to its NP-hardness and the enormous search space, obtaining an optimal solution in an online system is difficult. To address this challenge, we propose a heuristic algorithm based on the shortest path breadth-first search (BFS). More specifically, LoRaPCR prioritizes the simplest and most efficient one-hop requests. Then, when dealing with hard requests, it maximizes the reuse of previously used PCRs to reduce computational costs. For each hard request, we utilize a queue-optimized BFS algorithm to accelerate the search process and ensure the discovery of all feasible solutions, achieving the highest request satisfaction ratio.

Second, it is challenging to determine globally optimal transmission routes for registration requests. This issue is equivalent to the Graphical Steiner Minimal Tree (GSMT) problem [10], also known to be NP-hard. Although the scale of this problem is smaller than the previous one, existing solutions are far from meeting online requirements. To tackle this challenge, we propose a novel reduction algorithm based on dynamic programming. In this algorithm, we leverage the locality of the communication network to reduce the problem's scale. Specifically, we utilize the connectivity between target nodes to simplify the problem's scale, reducing it from the number of target nodes to the number of connected components. Moreover, we apply pruning techniques for two common special cases to further enhance the algorithm efficiency.

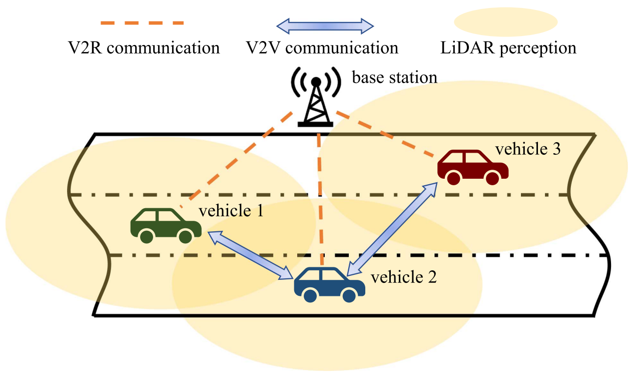


Fig. 2. Illustration of the system model.

We generate intensive vehicle traces in three typical driving scenarios (i.e., town, highway, and village) using the CARLA simulator [11] to validate the performance of LoRaPCR. We examine three different registration request preferences (i.e., near, far, and uniform) and observe an average request satisfaction ratio (RSR) of approximately 66% under the most challenging preference (i.e., far). In comparison, the state-of-the-art (SOTA) method achieved an RSR of only 4%. Extensive experiments are conducted and the results demonstrate that LoRaPCR significantly reduces the computation overhead by more than 50% compared to the SOTA method. To the best of our knowledge, LoRaPCR is the first PCR system in VANETs that achieves registration with point clouds separated over 100 meters, obtaining a high PCR accuracy with low relative translation and rotation errors of 0.55 meters and 1.43°, respectively.

We highlight the main contributions made in this work as follows: 1) A multi-hop PCR path selection algorithm based on a heuristic shortest path approach is proposed, effectively reducing the computational costs; 2) An online optimal transmission path selection algorithm is proposed, integrating locality-based reduction and the dynamic programming techniques; 3) extensive experiments are conducted and results demonstrate the efficacy of LoRaPCR, achieving SOTA performance for long-range PCRs. To the best of our knowledge, LoRaPCR is the first solution to achieve multi-vehicle point cloud long-range registration.

II. System Model and Problem Definition

A. System Model

In the long-range PCR problem, we consider the following two types of entities as shown in Fig. 2.

- *Vehicles:* are peers with equal capabilities as follows: 1) vehicles are capable of performing neural network computations to accomplish PCR tasks; 2) vehicles are capable of high-speed (e.g., more than 750Mb/s) V2V communication with a communication range of tens of meters, such as via millimeter-wave (mmWave). Additionally, vehicles can communicate with a BS or an RSU via longe-range communication links; 3) vehicles are equipped with LiDAR sensors that are capable of generating and storing point cloud data; 4) vehicles can obtain rough distance estimations of nearby vehicles but do not require precise localization. During online registration, we do not require any side channel information of vehicle position.

- *BS:* has the capability to utilize vehicle-to-roadside (V2R) communication for obtaining the information about the location of nearby vehicles, previous registration results and new PCR requests. The BS has basic computation and storage capabilities to make PCR strategies, which can be sent to vehicles for execution.

B. Problem Definition

We denote the sequence $\{X_1^{v_i}, X_2^{v_i}, \ldots, X_t^{v_i}\}$ as the time series of t point cloud frames of vehicle $v_i \in V$, where $X_k^{v_i} = \{p_n \in \mathbb{R}^3 | n = 1, 2, \ldots, N\}$ for $k \in [1, t]$ is the k-th point cloud frame of N points. At time t, vehicle v_i generates registration requests $\mathcal{R}_t^{v_i} \subseteq \{(v_i, v_j)|v_j \in I^{v_i}\}$, where I^{v_i} represents the interest region of v_i. A PCR request (v_i, v_j) is satisfied if there is a registration result (i.e., the transform matrix $M_{i,j}$ that aligns $X_t^{v_j} M_{i,j}^T$ with $X_t^{v_i}$) and the target point cloud data $X_t^{v_j}$ is transmitted to v_i. For registration request set $\mathcal{R}_t = \bigcup_{v_i \in V} \mathcal{R}_t^{v_i}$, the long-range multi-vehicle PCR problem is to find registration paths $\mathcal{P}_t$ and transmission paths $\mathcal{T}_t$ such that after they are executed, the registration requests are satisfied while the resulting computational and communication costs are minimized. The computational cost is determined by the number of registration tasks required, while the communication cost is determined by the number of transmissions needed along the transmission routes to satisfy each registration request.

III. Empirical Study

We first study the performance of underlying state-of-the-art PCR models (i.e., GCL [9]) in long-range setting.

A. Point Cloud Dataset Involving Multiple Vehicles

We consider the following two point cloud datasets for study:

- *KITTI:* We use the velodyne laser data set of the KITTI Vision Benchmark Suite, which consists of 11 sequences collected by the Velodyne HDL-64E (i.e. 64 channels, maximum measure range of 120m with a field of view (FOV) ranging from $-10°$ to $20°$).
- *CARLA trace:* Due to the lack of real-world multi-vehicle point cloud datasets, we utilize the CARLA simulator [11] to construct an intensive point cloud dataset involving 10 vehicles in three different driving scenarios. Each vehicle is equipped with a LiDAR sensor with the same parameter configuration as in KITTI. We establish a trajectory for these vehicles by means of a sequence of discrete path points. Initially, the vehicles are distributed within a confined area, and each vehicle's motion is regulated using CARLA's traffic manager. This process enables the generation of a multi-vehicle point cloud dataset, where overlapping relationships exist among point clouds from different vehicles. LiDAR sensors spin at 10 frames per second, capturing approximately 100k points per frame.

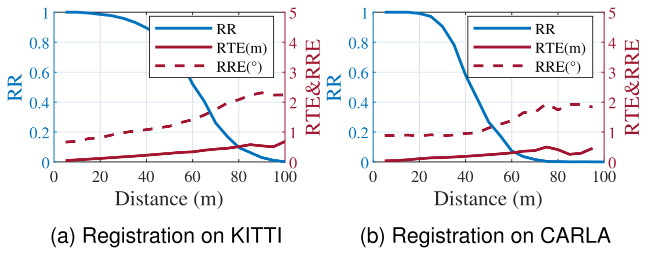


Fig. 3. Performance evaluation of GCL.

The vehicle fleet traverses every lane of each road on the map, and we record the point cloud data of each vehicle for each frame. Our dataset is available at https://lion.sjtu.edu.cn/project/projectDetail?id=41.

B. PCR Performance in Different Ranges

Following the standard protocols [12], we divide the KITTI dataset into training, validation, and test splits, and use the training and validation sets to train the GCL model. For the KITTI test set, we select point cloud pairs separated by distance range from 5 to 100 meters at an interval of 5 meters. Accordingly, for the multi-vehicle point cloud dataset, we organize point cloud pairs by pairing the target vehicle's point cloud with the hero vehicle's point cloud. Subsequently, we utilize the pre-trained GCL model to perform registration for these point cloud pairs on both datasets. We measure the registration performance in three metrics as follows.

- *Relative Translation Error (RTE):* The average magnitude of the difference between the estimated translation and the ground truth translation over all successful PCRs.
- *Relative Rotation Error (RRE):* The average degree of the difference between the estimated rotation and the ground truth rotation over all successful PCRs.
- *Registration Recall (RR):* A measure of the percentage of successful alignments within a specified threshold (i.e. RTE $<$2 meters and RRE $<$5°).

Observation: Fig. 3(a) plots the PCR results as functions of the distance between point clouds on the KITTI dataset. The registration recall decreases rapidly as the distance increases. High-quality PCR (RR$>$95%) can be achieved when the distance is less than 28 meters with low RTE and RRE of $<$0.1 meters and $<$1° (far below the required criterion of 2 meters and 5°), respectively. Similarly, as shown in Fig. 3(b), the same pattern stands for the multi-vehicle CARLA trace. Based on this, it is possible to achieve accurate long-range PCR through multi-hop short-range registrations. Theoretically, the success probability of multi-hop registration should be the product of the recall for each hop, and the upper bound of errors in RTE and RRE for multi-hop registration would be the accumulation of the RTE and RRE errors for each hop. An example of a two-hop registration is illustrated in Fig. 4(a).

C. Point Cloud Registration via Multi-Hop Relays

We further verify the feasibility of achieving long-range PCR with multi-hop short-range PCRs. For each frame of data, we set

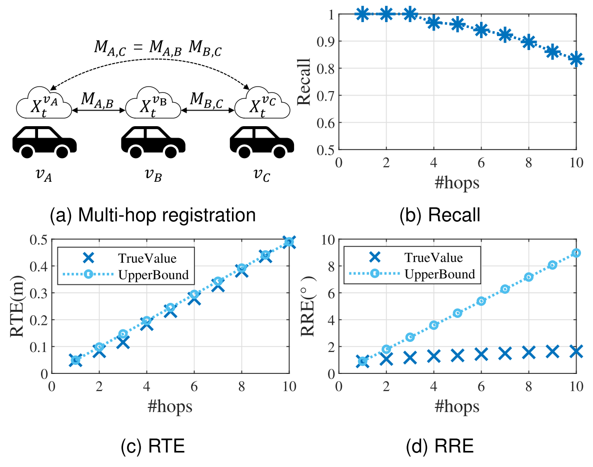
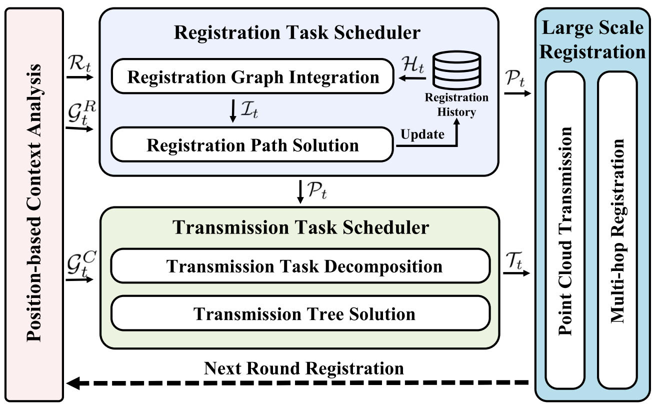


Fig. 4. Performance of multi-hop registration.



Position-based Context Analysis
Registration Task Scheduler
Registration Graph Integration
Registration History
Registration Path Solution
Update
Transmission Task Scheduler
Transmission Task Decomposition
Transmission Tree Solution
Next Round Registration
Large Scale Registration
Point Cloud Transmission
Multi-hop Registration

Fig. 5. The system architecture of LoRaPCR.

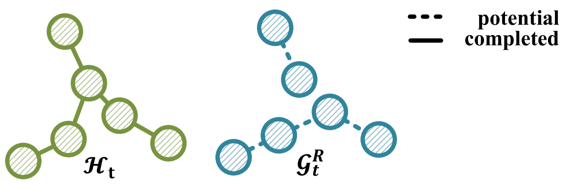


Fig. 6. Illustration of the integrated registration graph $\mathcal{I}_t$, which is integrated by the registration graph $\mathcal{G}_t^R$ and the registration history $\mathcal{H}_t$.

PCR tasks between the ego vehicle and a target vehicle ranging at a distance from 10 meters to 100 meters at an interval of 10 meters. For each registration task with a distance of $k \times 10$ meters, we utilize the pre-trained GCL model to perform k hops of registrations with a 10-meter distance. We solve the transform matrix for each registration hop and then multiply them to obtain the result of a multi-hop PCR.

The RTE and RRE of multi-hop registration should be computed as the addition and product of vectors for the RTE and RRE of each hop. In the worst-case, where translation and rotation errors accumulate in the same direction for each hop, the errors increase proportionally with the number of hops, which is referred to as the upper bound of RTE and RRE.

Fig. 4(b) illustrates the drop of RR as the number of hops increases. It can be seen that even with 5-hop registration, the registration recall still remains above 95%. On the other hand, Fig. 4(c) and (d) present the cumulative registration errors. Notably, the accumulated errors for a multi-hop registration remain within the empirical upper bound, increasing linearly with the number of hops. Specifically, RTE approximately follows a proportional relationship with the number of hops, while RRE exhibits a relatively smoother linear growth. Based on the above experiments, achieving long-range PCR through multi-hop registration is feasible. For a multi-hop registration path, we can estimate the registration success probability and the upper bound of registration errors by assessing the registration quality at each hop.

IV. Design of LoRaPCR

A. Overview

The core idea of LoRaPCR is to achieve long-range PCR through short-range PCR. Instead of rejecting long-range registration requests or obtaining an unreliable registration result, LoRaPCR searches for a high-quality registration path $X_t^{v_i} \leftrightarrow X_{t'}^{v_k} \leftrightarrow \cdots \leftrightarrow X_t^{v_j}$ to satisfy the long-range registration requests. The relay nodes in the registration path can be nearby point clouds of other vehicles ($X_t^{v_k}$) or even nearby point clouds from previous time instances ($X_{t'}^{v_k}$).

In each round of registration, each vehicle reports the neighbor information to the base station, enabling the base station to construct the registration graph $\mathcal{G}_t^R$ and communication graph $\mathcal{G}_t^C$. Then base station determines the registration paths and transmission routes $\mathcal{T}_t$ and decomposes them into individual registration tasks and transmission tasks for each vehicle. After every vehicle finishes its respective registration tasks and transmission tasks, the registration requests are satisfied, which means $X_t^{v_j}$ is transmitted to v_i, and v_i has $M_{i,j}$ to align $X_t^{v_j} M_{i,j}$ with $X_t^{v_i}$ for every $(v_i, v_j) \in \mathcal{R}_t$. As depicted in Fig. 5, LoRaPCR consists of two main components as follows:

Registration Task Scheduler: The registration task scheduler takes the registration requests and the registration graph as input, then combine the registration graph with the registration history and utilizes a shortest-path-based search algorithm to find registration paths for the registration requests. The registration task scheduler then sends the registration paths to the transmission task scheduler as well as vehicles.

Transmission Task Scheduler: The transmission task scheduler decomposes the registration paths into transmission tasks and searches for transmission routes for registration requests within the communication graph. Then the transmission task scheduler sends the transmission routes to vehicles.

B. Registration Task Scheduler

The registration task scheduler is used to determine registration paths $\mathcal{P}_t = \{Path(X_t^{v_i}, X_t^{v_j}) | (v_i, v_j) \in \mathcal{R}_t\}$ and it achieves this goal by three steps:

1) Registration Graph Integration: As shown in Fig. 6, in the registration graph $\mathcal{G}_t^R$ and registration history $\mathcal{H}_t$, each node denotes a point cloud $X_t^{v_i}$ while each edge $(X_t^{v_i}, X_{t'}^{v_j})$ denotes a potential registration (e.g., in $\mathcal{G}_t^R$) or a completed registration (e.g., in $\mathcal{H}_t$). We define the weight of edges as the computation cost, i.e., $w_{edge} = 1$ for a potential registration while $w_{edge} = 0$ for a completed registration. We can estimate the quality of registration using an empirical model and define it as the attribute of the corresponding edge. The registration task scheduler combines $\mathcal{G}_t^R$ and $\mathcal{H}_t$ into one integrated registration graph $\mathcal{I}_t$, and connect $\mathcal{G}_t^R$ and $\mathcal{H}_t$ by the temporal edges like $(X_t^{v_i}, X_{t'}^{v_i})$. The temporal edge $(X_t^{v_i}, X_{t'}^{v_i})$ means v_i can perform a registration computation between those two point clouds generated and stored by itself. The integrated registration graph represents all available registration edges to find registration paths for registration requests.

2) Registration Path Solution: After obtaining the integrated registration graph, the registration task scheduler determines the Minimum Request Satisfaction Graph(MRSG) for $\mathcal{R}_t$ on $\mathcal{I}_t$. The MRSG is defined as a subgraph with minimal computational cost, in which each request can identify a path if there exists a path in the original graph. The computational cost of MRSG equals the computational cost to implement $\mathcal{P}_t$, which can be represented as $\sum w_{edge}(edge \in \bigcup Path, Path \in \mathcal{P}_t)$. By solving the MRSG problem, the registration task scheduler can derive $\mathcal{P}_t$, which is the output of this module.

3) Registration Database Update: The registration task scheduler updates $\mathcal{H}_t \rightarrow \mathcal{H}_{t+1}$ according to $\mathcal{P}_t$ and the registration results returned by vehicles.

The algorithm to solve the MRSG problem determines the overall computation cost in VANETs and we have the following theorem:

Theorem 1: The MRSG problem is NP-hard.

Proof: To demonstrate the NP-hardness of the MRSG problem, we can establish a reduction from the Graphical Steiner Minimal Tree (GSMT) problem [10] with unit weight, which is known to be NP-hard. The formal definition of the GSMT problem with unit weight is as follows.

Consider an undirected connected graph $G = (V, E)$ with unit edge weights. Given a specific subset $Q \subseteq V$, representing a set of special vertices, GSMT is to determine the minimum cost tree of G such that there exists a path in the tree connecting every pair of special vertices in set Q.

To reduce the GSMT problem to the MRSG problem, we can set $\mathcal{R} = Q \times Q$ and $\mathcal{I} = G$. Obviously, in this case, MRSG is equivalent to GSMT, i.e., they have the same set of solutions. ■

To address the MRSG problem, one of the most straightforward approximate algorithms is to use the BFS method to find a shortest path for each request. BFS can achieve a high request satisfaction ratio by considering every possible path for each request, but it may not achieve the global minimum cost as it only considers local optimality. The pseudocode for the BFS-based algorithm is shown in Algorithm 1.

Considering the practical application requirements, we add the following two constraints to the registration paths:

- The registration recall of the registration path, which is the product of the registration recalls of each edge, must exceed a lower bound of 95%.
- The number of hops in the registration path is limited by an upper bound, which restricts the error range of the registration result obtained from the registration path.

Algorithm 1: BFS-Based Algorithm for MRSG Problem.

```
Input  : $\mathcal{R}_t$, $\mathcal{I}_t$
Output: $\mathcal{P}_t$
1  Initialize $\mathcal{P}_t$ as an empty set
2  for each $X_t^{v_i}$ in $\mathcal{I}_t$ do
3      $path\_lists[X_t^{v_i}]$ = ConstrainedSPFA($X_t^{v_i}$, $\mathcal{I}_t$)
4      for each $X_t^{v_j}$ such that $(X_t^{v_i}, X_t^{v_j}) \in \mathcal{R}_t$ do
5          $Path(X_t^{v_i}, X_t^{v_j})$ is minimum cost path in
             $path\_lists[X_t^{v_i}][X_t^{v_j}]$
6          Add $Path(X_t^{v_i}, X_t^{v_j})$ to $\mathcal{P}_t$
7  return $\mathcal{P}_t$
8  Function ConstrainedSPFA($X_{start}$, $\mathcal{I}_t$)
9      Initialize a list $path\_lists[X]$ for every node $X$ in $\mathcal{I}_t$
10     Initialize a queue and add $X_{start}$ to it
11     while the queue is not empty do
12         Take the front node $X_f$ out of the queue
13         for each neighbor $X_n$ of $X_f$ do
14             Try to add new path to $path\_lists[X_n]$
15             if quality of new path $\geq$ quality of any path in
                 $path\_lists[X_n]$ and cost of new path $<$ cost of that
                 path then
16                 Update the $path\_lists[X_n]$
17                 If $X_n$ is not in the queue, add it to the queue
18     return $path\_lists$
```

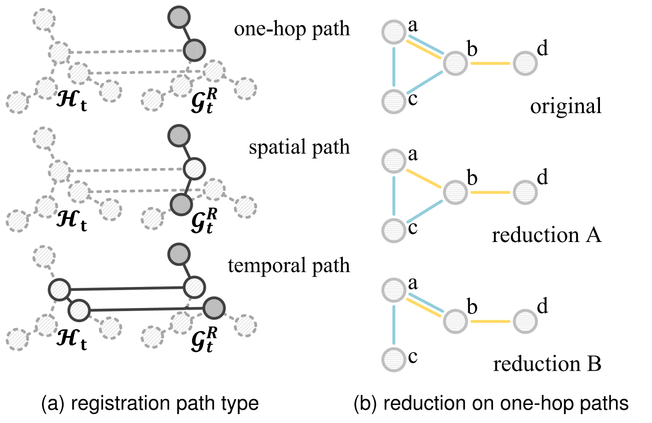


Fig. 7. Illustration of registration paths. (a) Three types of registration paths. (b) Effective reductions on one-hop paths can decrease the number of registrations.

Specifically, we employ the Shortest Path Faster Algorithm (SPFA), a queue-optimized BFS algorithm, to solve MRSG problem, given that the constraint conditions prevent infinite loops.

One way to optimize the BFS algorithm is by utilizing the overlap between different paths to reuse registration edges as much as possible. A feasible approach is to record the edges used in the determined paths and to reuse them as much as possible. In this approach, the order in which the paths for different requests are determined becomes crucial to achieve efficient path reuse. As shown in Fig. 7(a), the registration paths can be categorized into three types: 1) the one-hop path, which indicates a single hop registration 2) the spatial path, which involves using extra nodes at the same time instance as a relay. 3) the temporal path, which involves using extra nodes from a previous time instance as a relay. Among the three types of paths, one-hop paths are the most efficient and intuitive. If a request has an one-hop path

Algorithm 2: OHRF Algorithm for MRSG Problem.

```
Input  : $\mathcal{R}_t$, $\mathcal{I}_t$
Output: $\mathcal{P}_t$
Initialize $\mathcal{P}_t$ as an empty set
Initialize $hard\_requests$ as an empty list
for each $(X_t^{v_i}, X_t^{v_j})$ in $\mathcal{R}_t$ do
    if $(X_t^{v_i}, X_t^{v_j})$ is one-hop reachable in $\mathcal{I}_t$ then
        $Path(X_t^{v_i}, X_t^{v_j}) = (X_t^{v_i}, X_t^{v_j})$
        Add $Path(X_t^{v_i}, X_t^{v_j})$ to $\mathcal{P}_t$
        Set cost of edge $(X_t^{v_i}, X_t^{v_j})$ in $\mathcal{I}_t$ to 0
    else
        Add $(X_t^{v_i}, X_t^{v_j})$ to $hard\_requests$
for each $X_t^{v_i}$ in $\mathcal{I}_t$ do
    $path\_lists[X_t^{v_i}]$ = ConstrainedSPFA($X_t^{v_i}$, $\mathcal{I}_t$)
    for each $(X_t^{v_i}, X_t^{v_j})$ in $hard\_requests$ do
        $Path(X_t^{v_i}, X_t^{v_j})$ is minimum cost path in
          $path\_lists[X_t^{v_i}][X_t^{v_j}]$
        Add $Path(X_t^{v_i}, X_t^{v_j})$ to $\mathcal{P}_t$
return $\mathcal{P}_t$
```

available, then there is high probability that the registration path for this request in the global optimal solution will indeed be an one-hop path. Hence, we propose a novel algorithm called One-Hop Request First (OHRF). The core idea of the OHRF algorithm is to reuse the one-hop paths when making decisions for other requests. More specifically, we first check whether each registration request can be satisfied by a one-hop path. If so, we directly use the one-hop path as the solution. Then, we create a duplicate graph, denoted as $\mathcal{I'}_t$, and set the weight of all edges used in the one-hop solutions to 0in $\mathcal{I'}_t$. For the other requests, referred to as hard requests, we employ the BFS in $\mathcal{I'}_t$ to search for the shortest path as their solutions. By adopting this approach, we effectively reuse one hop paths when searching for the shortest path for hard requests, thereby saving computational costs. The pseudocode for the OHRF algorithm is shown in Algorithm 2.

In addition, we discover that there is also a substantial amount of redundancy among one-hop paths. As depicted in Fig. 7(b), the blue edges represent paths corresponding to one-hop requests, while the orange edges represent paths corresponding to hard requests. In original case, the hard request reuses one edge from the one-hop request, resulting in a global cost of 4. Under the premise of ensuring that requests are satisfied, we can perform reduction on one hop paths. Reduction A removes some edges while still ensuring the satisfaction of all requests. Unfortunately, this approach may not necessarily reduce the global computation cost, as the removed edges could be reintroduced in the paths of hard requests.

Based on the above observations, we propose a novel reduction method. The core idea is to record edges used by hard requests and constructs the minimum spanning tree to reduce one-hop paths while preserving the recorded edges. It can be seen that in Reduction B the global computation cost is 3. The pseudocode for the OHRF-based algorithm with reduce is shown in Algorithm 3. Due to the complexity primarily arising from running SPFA V times, the time complexity for Algorithms 1, 2, and 3 is $O(V^2E)$.

Algorithm 3: OHRF-Based Algorithm for MRSG Problem With Reduction.

```
Input  : $\mathcal{R}_t$, $\mathcal{I}_t$
Output: $\mathcal{P}_t$
Initialize $\mathcal{P}_t$ as an empty set
Initialize $hard\_requests$ as an empty list
Initialize $one\_hop\_paths$ as an empty list
for each $(X_t^{v_i}, X_t^{v_j})$ in $\mathcal{R}_t$ do
    if $(X_t^{v_i}, X_t^{v_j})$ is one-hop reachable in $\mathcal{I}_t$ then
        $Path(X_t^{v_i}, X_t^{v_j}) = (X_t^{v_i}, X_t^{v_j})$
        Add $Path(X_t^{v_i}, X_t^{v_j})$ to $one\_hop\_paths$
        Set cost of edge $(X_t^{v_i}, X_t^{v_j})$ in $\mathcal{I}_t$ to 0
    else
        Add $(X_t^{v_i}, X_t^{v_j})$ to $hard\_requests$
for each $X_t^{v_i}$ in $\mathcal{I}_t$ do
    $path\_lists[X_t^{v_i}]$ = ConstrainedSPFA($X_t^{v_i}$, $\mathcal{I}_t$)
    for each $(X_t^{v_i}, X_t^{v_j})$ in $hard\_requests$ do
        $Path(X_t^{v_i}, X_t^{v_j})$ is minimum cost path in
          $path\_lists[X_t^{v_i}][X_t^{v_j}]$
        Mark the reused path in $one\_hop\_paths$
        Add $Path(X_t^{v_i}, X_t^{v_j})$ to $\mathcal{P}_t$
Construct a graph, $G_{hop}$, using $one\_hop\_paths$
Compute the MST, $T_{hop}$, of $G_{hop}$ while trying to retain marked
  paths
for $Path(X_t^{v_i}, X_t^{v_j})$ in $one\_hop\_paths$ do
    if $Path(X_t^{v_i}, X_t^{v_j})$ is in the MST $T_{hop}$ then
        Add $Path(X_t^{v_i}, X_t^{v_j})$ to $\mathcal{P}_t$
    else
        Set cost of edge $(X_t^{v_i}, X_t^{v_j})$ in $\mathcal{I}_t$ to 0
        $path\_lists[X_t^{v_i}]$ = ConstrainedSPFA($X_t^{v_i}$, $\mathcal{I}_t$)
        $Path(X_t^{v_i}, X_t^{v_j})$ is minimum cost path in
          $path\_lists[X_t^{v_i}][X_t^{v_j}]$
        Add $Path(X_t^{v_i}, X_t^{v_j})$ to $\mathcal{P}_t$
return $\mathcal{P}_t$
```

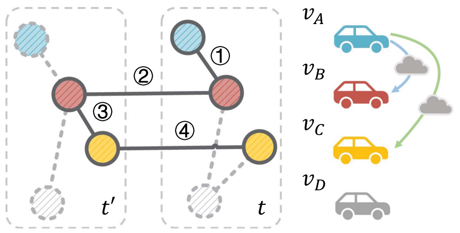


Fig. 8. The procedure of decomposing registration path into transmission tasks.

C. Transmission Task Scheduler

The transmission task scheduler is used to determine transmission paths $\mathcal{T}_t$ and it achieves this goal by two steps:

1) Transmission Task Decomposition: The transmission task scheduler takes $\mathcal{P}_t$ as input, and for every $Path \in \mathcal{P}_t$, it can be decomposed into several transmission tasks. As shown in Fig. 8, $Path(X_t^{v_C}, X_t^{v_A})$ consists of 4 edges. For edge ①, there is a transmission task of v_A transmitting $X_t^{v_A}$ to v_B. For edge ② and ④, there is no transmission task since the point cloud pair belongs to the same vehicle. For edge ③, there is no transmission task as the registration has already been completed at time t'. The completed registration result can be directly obtained from the

Registration Network Database. Besides, there is another transmission task of transmitting $X_t^{v_A}$ to v_C to satisfy the registration request $(X_t^{v_C}, X_t^{v_A})$. Then we can partition the transmission tasks based on the content of the data being transmitted. Each partition consists of a list of transmission tasks of transmitting $X_t^{v_{src}}$ from the source vehicle v_{src} to the target vehicle v_{tgt}.

2) Transmission Tree Solution: After partitioning the transmission tasks, the transmission task scheduler processes each partition sequentially, as they do not interfere with each other. For each partition, the transmission task scheduler needs to determine an optimal transmission route for each transmission task, aiming to minimize the overall communication overhead. This problem is referred to as the Minimum Transmission Graph(MTG) problem and we have the following theorem.

Theorem 2: The MTG problem is NP-hard.

Proof: The MTG problem can be formulated as finding the minimum-weight subgraph in an undirected graph with unit edge weights, such that there exists a path between the source node and all target nodes.

By proving the equivalence between the MTG problem and the Graphical Steiner Minimal Tree (GSMT) problem, we can establish that the MTG problem is NP-hard. Let's assume that the solution to the MTG problem is a subgraph denoted as X.

According to the definition of the MTG problem, the source vertex v_{src} must be connected to every target vertex v_{tgt}, and any two target vertices v_{tgt_1} and v_{tgt_2} can be connected through a path $v_{tgt_1} \leftrightarrow v_{src} \leftrightarrow v_{tgt_2}$. Therefore, subgraph X must be a connected acyclic graph, i.e., a tree, and it must contain both the source vertex v_{src} and all the target vertices v_{tgt}. This proves the equivalence between the MTG problem and the GSMT problem. ■

Based on the above proof, we can indeed transform the MTG problem into an equivalent GSMT problem to solve it. For the GSMT problem, there exists a dynamic programming algorithm [13] with a time complexity of $O(n^c * 2^m)$, where n represents the number of nodes, and m denotes the number of special nodes in the graph. While we may not be able to reduce the computational complexity, we can improve the algorithm's efficiency by reducing the problem size. For our specific application scenario, we have the following observation: Since the target nodes are all within the interest region of the source node, the distances between the target nodes are not likely to be too long. Consequently, there is a high probability that the target nodes are directly connected in the communication graph. In order to leverage this locality of the communication graph, we present the following theorem:

Theorem 3: In the unit-weighted GSMT problem (G, Q), if there exists an edge connecting two nodes in Q, then there exists a GSMT (G, Q) that includes this edge.

Proof: Let's assume that G contains an edge (q_1, q_2), where $q_1, q_2 \in Q$. Let T be a GSMT of Q in G. If $(q_1, q_2) \notin T$, without loss of generality, let's assume that there exists an edge $(x, y) \neq (q_1, q_2)$ on the unique path between q_1 and q_2. So we can remove the edge (x, y) and add the edge (q_1, q_2) to form a new GSMT T' which includes the edge (q_1, q_2). ■

Furthermore, we can extend the aforementioned theorem to a set of edges:

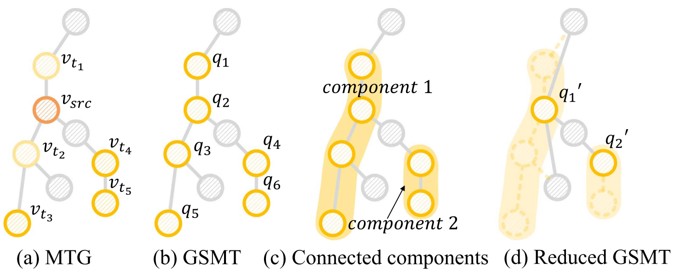


Fig. 9. The procedure of reducing MTG problem.

Algorithm 4: Optimized Dynamic Programming Algorithm With Reduction for GSMT Problem.

```
   Input  : Undirected graph $G = (V, E)$, special node set $Q \subseteq V$
   Output: GSMT $T$
1  Construct subgraph $QG = (Q, QE)$, where
     $QE = \{(q_1, q_2) \mid q_1, q_2 \in Q\}$
2  Find connected components of $QG$, store MSTs of each connected
     component
3  Merge nodes within each connected component of $QG$ to form new
     graph $G'$, $Q'$
4  if number of connected components in $QG$ = 1 then
5      return the MST of the single connected component
6  else if number of connected components in $QG$ = 2 then
7      $T_{\text{MST1}} \leftarrow$ MST of the first connected component in $QG$
8      $T_{\text{MST2}} \leftarrow$ MST of the second connected component in $QG$
9      $u, v \leftarrow$ special nodes representing the two connected
         components
10     $T_{\text{SP}} \leftarrow$ ShortestPath($G'$, $u$, $v$)
11     $T \leftarrow$ Merge $T_{\text{MST1}}$, $T_{\text{MST2}}$, and $T_{\text{SP}}$
12 else
13     $T_{\text{DP}} \leftarrow$ DP($Q'$, $G'$)
14     $T_{\text{MST}} \leftarrow$ Merge MSTs of all connected components in $QG$
15     $T \leftarrow$ Merge $T_{\text{DP}}$ and $T_{\text{MST}}$
16 return $T$
```

Theorem 4: In the unit-weighted GSMT problem (G, Q), if there exists a set of edges $E_Q \subseteq \{(q_1, q_2) | q_1, q_2\} \in Q$ and the edges in E_Q do not form any cycle, then there exists a GSMT (G, Q) that includes every edge in E_Q.

Proof: When $|E_Q| = 1$, according to Theorem 3, the proposition holds true. Assume that the proposition holds true when $|E_Q| = k$. Now, let's consider the case when $|E_Q| = k + 1$:

Choose one edge $(q_1, q_2) \in E_Q$, there exists a GSMT $(G, Q) = T$ that includes every edge in $E_Q \setminus (q_1, q_2)$. If $(q_1, q_2) \notin T$, without loss of generality, let's assume that there exists an edge $(x, y) \notin E_Q$ on the unique path between q_1 and q_2. So we can remove the edge (x, y) and add the edge (q_1, q_2) to form a new GSMT T' which includes every edge in E_Q.

By induction, we have shown that the proposition holds for all cases, and thus, the statement is proven to be true. ■

By leveraging Theorem 4, we can simplify the GSMT problem by organizing the special nodes into several connected components, which we refer to as "Connected Components Reduction". As shown in Fig. 9(a), there are 5 target vehicles for v_{src}: 2 one-hop transmission task to v_{t_1}, v_{t_2} and 3 transmission task to $v_{t_3}, v_{t_4}, v_{t_5}$. First, we can transform the above problem into a GSMT problem, as illustrated in Fig. 9(b). Furthermore, we can organize the special nodes into two connected components in Fig. 9(c). The reduced GSMT problem is depicted in Fig. 9(d). Consequently, the algorithm's complexity reduces from $O(n^c \cdot 2^m)$ to $O(n^c \cdot 2^{m'})$, where m' represents the number of connected components. In conclusion, through the Connected Components Reduction, we achieve an exponential speed improvement. The pseudocode for the Optimized Dynamic Programming Algorithm with Reduction is shown in Algorithm 4.

To further optimize the algorithm's performance, we apply pruning techniques for the cases where $m' = 1$ and $m' = 2$. The former is equivalent to finding the minimum spanning tree, while the latter corresponds to computing the shortest path between two connected components.

V. Evaluation

A. Methodology

We conduct extensive simulations using the CARLA simulator to evaluate the performance of LoRaPCR concerning various vehicle spatial distributions and different registration request scenarios. Specifically, we select three different maps in the CARLA simulator due to their distinct road characteristics, which may affect the spatial distribution of vehicles. We set the number of vehicles in the scene to be from 10 to 100 at an interval of 10 to test LoRaPCR's performance under different vehicle densities.

To generate vehicle registration requests, each vehicle selects up to $\text{REQ}_{\max}$ vehicles from the interest region (i.e. within a range of 100 meters) based on their preferences. We test the cases where $\text{REQ}_{\max}$ is set to 5, 10, 15, and 20, and we define three different preference as follows: 1) uniform: refers to evenly assign the probability of being selected to all vehicles. 2) near: refers to assign a weight of $e^{(100-d)/20}$ to vehicles d meters away from the interest region and then normalized to obtain the probability. 3) far: refers to assign a weight of $e^{d/20}$ to vehicles d meters away from the interest region and then normalized to obtain the probability.

We collect point clouds from vehicle-mounted LiDARs at various distances in the CARLA simulator. A pre-trained GCL model is used for registration to fit the quality of single-hop point cloud registration (i.e. recall, RTE and RRE) across different distances and to analyze the error accumulation in multi-hop point cloud registration. For each PCR task, we estimate the registration success probability and error based on the distance between point cloud pairs using the registration quality estimation model fitted in the aforementioned process.

We define four metrics as follows:

- *Request Satisfaction Rate (RSR):* refers to the ratio of the number of requests for which registration both path and transmission route are successfully found to the total number of requests.
- *Registration Cost Per Request (RCPR):* refers to the average number of registrations performed for each registration request, indicating the average computational overhead.
- *Communication Cost Per Request (CCPR):* refers to the average number of point cloud transmissions performed for each registration request, indicating the average communication overhead.
- *RTE & RRE:* refers to the average estimated upper bound of RTE and RRE for the registration results of each satisfiable registration request.

We compare our method with two candidate methods as follows:

- *Traditional direct registration methods (DRCT):* refers to the approach of determining whether each request can be directly registered in one hop. If feasible, the registration and transmission are executed; otherwise, the request is rejected directly. Any pairwise registration method can be considered as a DRCT method. In subsequent experiments, we assume the DRCT method is based on GCL by default.
- *BFS-based registration methods (LoRaBFS):* refers to using BFS to find a shortest path for each request without considering the reuse of duplicate edges, which is a straightforward method for implementing multi-hop registration. Due to the lack of consideration for multi-vehicle registration problem in existing PCR methods, we implement this baseline to adapt pairwise PCR methods (assumed to be GCL by default) to our system model.

TABLE I
Ablation Study

Preference	Algorithm	RSR	RCPR	CCPR	RTE(m)	RRE($^\circ$)
Far	LoRaBFS	0.66	0.72	1.99	0.53	1.41
	LoRaOHRF	0.66	0.71	**1.99**	**0.52**	**1.40**
	LoRaPCR	**0.66**	**0.49**	2.01	0.53	1.42
Uniform	LoRaBFS	0.81	0.68	1.36	0.38	1.29
	LoRaOHRF	**0.81**	0.56	**1.36**	**0.36**	**1.28**
	LoRaPCR	0.80	**0.35**	1.39	0.39	1.32
Near	LoRaBFS	0.91	0.81	**1.27**	0.25	1.17
	LoRaOHRF	**0.91**	0.67	1.27	**0.23**	**1.17**
	LoRaPCR	0.90	**0.33**	1.45	0.25	1.23

B. Parameter Configuration

1) Ablation Study: First, we evaluate the effectiveness of OHRF strategy and reduction on one-hop paths. We select the Town scenario in the CARLA simulator with $vehiclenumber = 80$ and $\text{REQ}_{\max} = 10$. The maximum number of hops for registration paths is set to 7. We test the performance of LoRaBFS, LoRaOHRF, and LoRaPCR under three different registration request preferences.

Table I lists the performance of LoRaBFS, LoRaOHRF and LoRaPCR under three different registration request preferences. It can be seen that LoRaOHRF reduces RCPR to a certain extent, while improving CCPR, RTE, and RRE slightly. This is likely because determining the one-hop request paths first helps in finding better registration paths and transmission routes for hard requests. Furthermore, reduction on one-hop paths further decreases RCPR while slightly reducing the other metrics, which still remain within an acceptable range. Among the three preferences, "far" is the most representative of long-range registration scenarios and is also the most challenging. Therefore, we use this preference as the default in subsequent experiments.

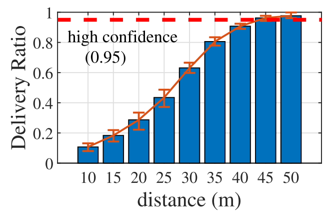


Fig. 10. Delivery ratio.

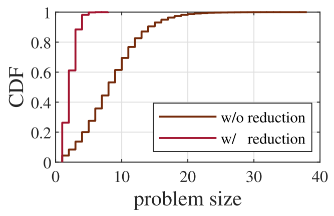


Fig. 11. The CDF of m.

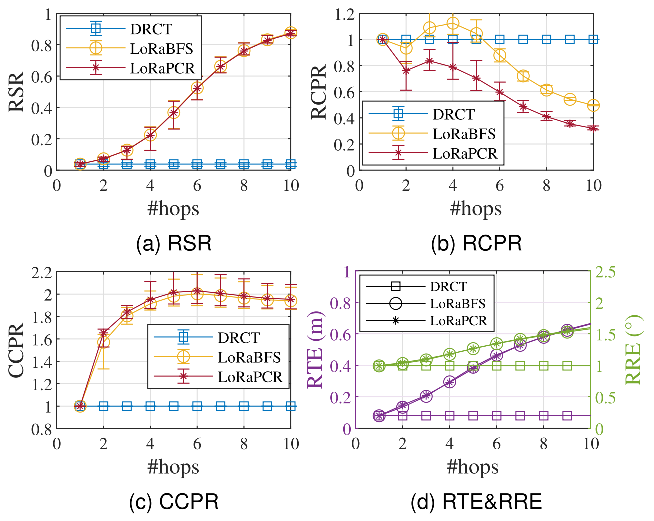


Fig. 12. Performance of DRCT, LoRaBFS and LoRaPCR with different maximum number of hops.

2) Impact of Communication Range: In the experiments, we assume that vehicles could engage in direct high-speed V2V communication with other vehicles within their communication range to support point cloud transmission tasks. The communication range of vehicles can influence the completion of point cloud transmission tasks. We define the delivery ratio as the ratio of the number of satisfied registration requests to the total number of registration requests that have feasible registration paths. We measure the delivery ratio of registration requests under different communication ranges, and the results are shown in Fig. 10. When the communication range equals the interest region radius(i.e. 100 meters), all point cloud transmission tasks can be completed through one-hop transmission. As the communication range decreases, some point cloud transmission tasks may require multi-hop transmission or may become unreachable, leading to a decrease in the delivery ratio. It can be observed that the delivery ratio continuously increases as the communication distance expands. When the communication distance reaches 45 m, the delivery ratio reaches 95%. Therefore, in the subsequent experiments, we set the communication distance to 45 m.

3) Impact of the Connected Component Reduction: To evaluate the effectiveness of the Connected Component Reduction in the Transmission Task Scheduler, we conduct experiments under the same conditions as in the ablation study and record the cumulative distribution of the problem size. As shown in Fig. 11, the Connected Component Reduction effectively reduces the problem size, and in most of the cases (i.e. $>80\%$), the problem size (the value of m) does not exceed 5, which is acceptable for the DP algorithm with a time complexity of $O(n^c 2^m)$. In most of the cases, the reduced algorithm can complete the computation within 1ms, achieving a speedup of more than 1000 times compared to the non-reduced version.

4) Impact of Maximum Number of Hops: We control the error of registration paths by limiting the maximum number of hops. We vary the maximum number of hops from 1 to 10 while keeping the remaining conditions consistent with the ablation study. We test the performance of DRCT, LoRaBFS and LoRaPCR under far preference.

Fig. 12 shows that LoRaPCR demonstrates significant advantages in RCPR, while performing comparably to LoRaBFS in other metrics. Although DRCT demonstrates good performance in CCPR and RTE & RRE, its low RSR shows that it falls short of meeting the desired requirements. The results of the above experiments also validate our theoretical findings. When $\text{HOP}_{\max} = 1$, both LoRaBFS and LoRaPCR degenerate into the DRCT algorithm. When $\text{HOP}_{\max} = 2$, the system uses neighboring point cloud registration results for multi-hop registration, but it does not use previous time step results (which would require at least 3 hops). As a result, edge reuse starts to occur, leading to a decrease in RCPR. However, multi-hop registration paths introduce additional one-hop transmission tasks, resulting in an increase in CCPR. As $\text{HOP}_{\max}$ gradually increases, the system relaxes the requirements on the quality of registration paths, leading to an increase in RSR as more requests can be satisfied. Although more requests are fulfilled, the number of registration tasks in the system reaches saturation, meaning the total registration number and the additional transmission tasks do not increase. Consequently, RCPR increases with the growth of RSR, while CCPR remains stable (the impact of additional transmission tasks gradually becomes negligible in the overall transmission task).

From Fig. 12(d), we can observe that when $\text{HOP}_{\max}$ is less than 8, the average RTE is below 0.6 m, and the average RRE is below 1.5°. Therefore, we set $\text{HOP}_{\max} = 7$ in the following experiments.

C. Impact of Vehicle Density

To demonstrate the impact of different vehicle densities on the performance of LoRaPCR, we vary the number of vehicles from 10 to 100 at intervals of 10 and test the performance of

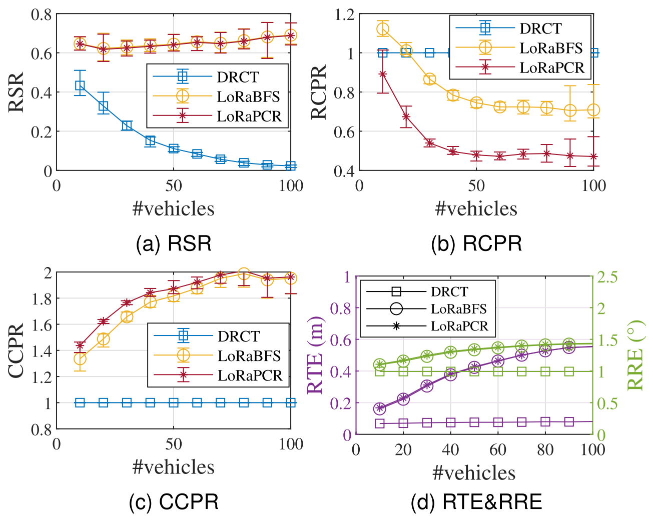


Fig. 13. Performance of DRCT, LoRaBFS and LoRaPCR with different vehicle density.

LoRaPCR in the Town scenario with $\text{REQ}_{\max} = 10$ under three different registration request preferences. All other parameters remain consistent with those used in the ablation study.

As shown in Fig. 13, when the vehicle density is low, the advantages of multi-hop registration are not very prominent. However, as the vehicle density increases, the RCPR of multi-hop registration exhibits a significant improvement. It should be noted that when the vehicle density is very low, the number of vehicles within the interest region of a vehicle may not reach the threshold of $\text{REQ}_{\max}$, invalidating the registration request preference and leading to relatively easy requests. As the vehicle density increases, the far preference begins to take effect, making the requests more challenging, resulting in an increase in CCPR, RTE and RRE. Meanwhile, RSR also shows a slight increase because of the increase of potential relay vehicles, but due to the aforementioned factors, the improvement is relatively small. Throughout these conditions, LoRaPCR consistently maintains a considerable advantage in RCPR, while its growth in CCPR remains within an acceptable range.

D. *Impact of Different Models*

LoRaPCR is a multi-hop point cloud registration system that is compatible with any point cloud pairing registration model, enhancing its long-range registration capabilities. In addition to the GCL model, we also fit single-hop and multi-hop error models for the FCGF model to simulate the performance of LoRaPCR based on FCGF. We evaluate the performance of LoRaPCR based on the GCL and FCGF models separately, across varying radii of interest regions, while ensuring that all other conditions remain consistent with those outlined in the ablation study.

As depicted in Fig. 14, LoRaPCR notably boosts long-range registration performance. When the RSR surpasses 80%, LoRaPCR extends the interest region radius of FCGF from under 20 meters to close to 40 meters, and that of CBGF from around 30 meters to 80 meters. Furthermore, LoRaPCR significantly lowers computational expenses, with the added communication costs, RTE, and RRE being tolerable.

E. *Impact of Scenarios*

The distinct characteristics of different scenarios' roads can impact the spatial distribution of vehicles. We test the performance of LoRaPCR in three different scenarios while keeping the other conditions consistent with the ablation study. Specifically, the town scenario features complex road networks with dense vehicle traffic, the village scenario has many curved roads and sparse traffic, and the highway scenario displays vehicles distributed in a linear pattern. The results in Fig. 15 show that multi-hop registration methods (LoRaBFS and LoRaPCR) can maintain a high RSR across various scenarios. LoRaPCR exhibits a significant advantage in RCPR compared to LoRaBFS, and the increased overhead in CCPR remains within an acceptable range. Both methods have similar registration errors.

F. *Impact of Maximum Number of Requests*

The maximum number of requests for single vehicle affects the overall number of requests in the VANETs. We test the performance of LoRaPCR under different values (5, 10, 15, 20) of $\text{REQ}_{\max}$, while keeping the other conditions consistent with the ablation study. Fig. 16 shows that LoRaPCR maintains a significant advantage in RCPR compared to the other two methods while the disadvantage of CCPR remains within an acceptable range. As $\text{REQ}_{\max}$ gradually decreases, the far preference becomes more effective, making the registration requests more challenging. Despite this, LoRaPCR still maintains a significant advantage in RCPR and performs roughly on par with LoRaBFS in other metrics.

VI. Real-World Data Showcase

We validated the performance of LoRaPCR on the MARS dataset [14] to demonstrate its feasibility in real-world scenarios. The MARS dataset is a real-world dataset collected by a fleet of autonomous vehicles (2 to 3 cars) driving within a certain geographical area. The MultiAgent subset facilitates collaborative driving with multiple vehicles simultaneously present at the same location. We conduct experiments using the three-vehicle scenarios from this subset, as shown in Fig. 17(a), varying the radii of interest regions while keeping other parameters consistent with those in the ablation study.

As shown in Fig. 17(b), the RSR gradually decreases as the radius of the interest region increases. LoRaPCR demonstrates a significant advantage in RSR, maintaining a high satisfaction rate even under long-range registration conditions. It should be noted that due to the high cost of point cloud data collection and annotation, there is a lack of real-world datasets with a large number of vehicles and data frames. Therefore, we can only perform a preliminary feasibility test of LoRaPCR on the MARS dataset. In the brief segments with three vehicles, the available registration relays for LoRaPCR are quite limited. Despite this, LoRaPCR still demonstrates significant advantages.

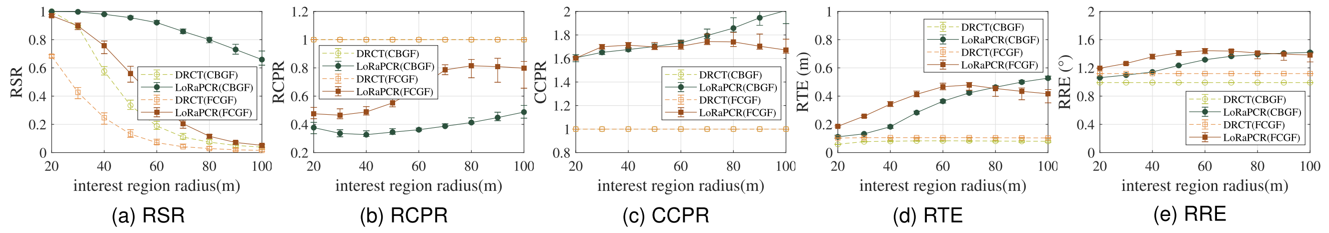


Fig. 14. Performance of LoRaPCR based on different models in different interest region.

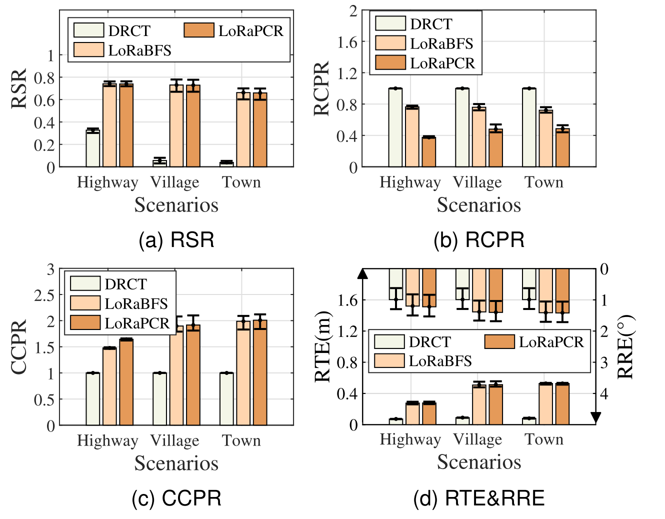


Fig. 15. Performance of DRCT, LoRaBFS and LoRaPCR in different scenarios.

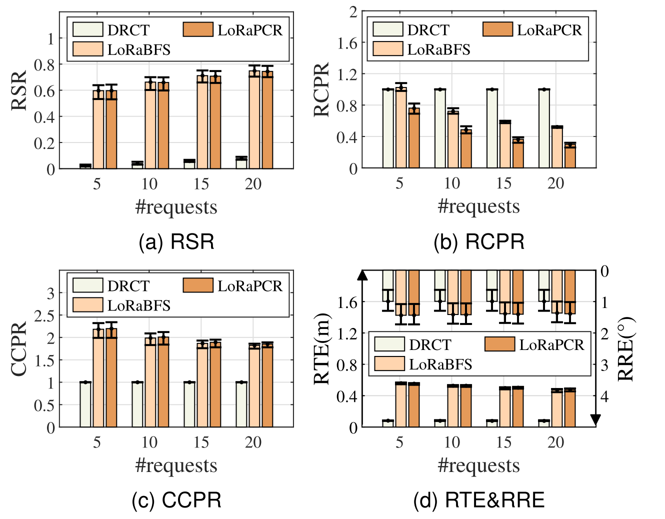


Fig. 16. Performance of DRCT, LoRaBFS and LoRaPCR in different $\mathrm{REQ}_{\max}$.

VII. Discussion

A. Low Vehicle Density

When the vehicle density is low, it is possible that there may be no relay vehicle between two distant vehicles, making it hard to satisfy the registration request. In such cases, LoRaPCR attempts to utilize temporal point cloud data as a relay. If in a previous time instance the two vehicles are close enough to perform PCR or there is a relay vehicle in between, the current registration request can still be satisfied by multi-hop registration. In practical scenarios, if vehicles are very sparse and the distance between them is large, we can consider it relatively safe, and there may not be a strong demand for PCR.

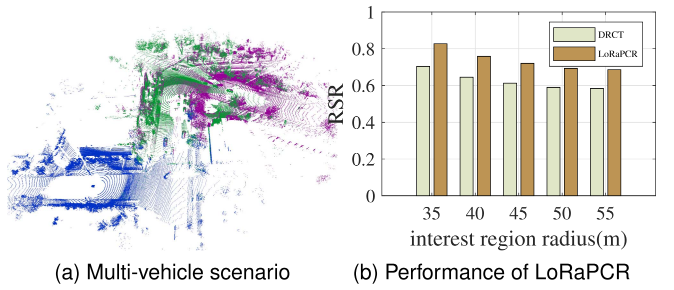


Fig. 17. Performance evaluation of LoRaPCR on MARS.

B. Individual Communication Load Optimization

LoRaPCR is aiming to minimize the global communication load while overlooking the individual communication load of each vehicle. If a particular vehicle serves as a central node in the communication network, its communication load may become excessively high. To address this issue, we can introduce an algorithm to adjust the communication routes. In the MTG problem, the optimal transmission tree can have multiple solutions, which means that if a node's communication load is high, we can choose an alternative optimal transmission tree to avoid selecting that node while ensuring a constant global communication load remains unchanged. If there is no other viable route for a specific transmission task, we can only use a "best-effort" strategy to satisfy the registration request to the best of the systems' communication capacity.

C. Surge in Computational Tasks

LoRaPCR allows for the reuse of previous registration results to save on computational resources, which may lead to fluctuations in computational load. For instance, if many pairwise registrations occur at a particular moment, vehicles might reuse these registration results for some time thereafter, resulting in minimal computation. However, when the particular moment's registration results cannot be reused due to issues such as registration accuracy, the system may experience a surge in computational tasks, causing load fluctuations. Introducing a degree of randomness in the selection of registration paths can mitigate this issue, preventing vehicles from always opting to reuse previous results instead of conducting new registrations.

VIII. Related Work

We classify the existing relevant work into two categories: indoor PCR and outdoor PCR.

Indoor PCR: Indoor PCR can be further categorized based on the matched features into two groups: patch-based features and fully convolutional features. 3DMatch [15] extracts local features by employing 3D convolutions on local areas. PPF-Net [16] utilizes robust point-pair features extracted by PointNet for registration. PerfectMatch [17] adopts a voxelized smoothed density value (SDV) representation to obtain robust features. DIP [18] develops an effective method to register point clouds without initial alignment using distinctive 3D local deep descriptors. The latest advancements in SpinNet [19] and BUFFER [20] integrate cylindrical features that are equivalent under SO(2) rotations with backbones that are entirely convolutional. However, all these methods are local patch-based, which incurs significant computational overhead and cannot accomplish online registration of large-scale outdoor point clouds.

Outdoor PCR: Outdoor PCR can be further divided into two categories: pairwise V2V registration and multi-vehicle registration with infrastructure.

Pairwise V2V registration aims to optimize registration methods to improve registration performance. FCGF [12] introduces metric learning and utilizes sparse convolutions to accelerate computation. D3Feat [21] proposes a keypoint selection strategy to overcome inherent density variations. Predator [22] uses an overlap attention module to address the low overlap problem in PCR. APR [6] leverages an autoencoder to reconstruct a denser aggregated point cloud, allowing the encoder to extract features with rich local geometry information, thereby enhancing PCR accuracy. However, the aforementioned methods face difficulties in achieving PCR with distance exceeding 30 m, which fails to meet the demands of autonomous driving. Due to the high cost of annotating point cloud data, several unsupervised point cloud registration methods are developed. Liu, Jingbin [23] proposes an improved method for extracting point cloud keypoints based on rotation compensation and a convolutional end-to-end unsupervised point cloud registration network. However, this method is only utilized for registering consecutive frames of point clouds, which typically do not exceed distances of 1 meter. EYOC [24] trains feature extractors through near-range point cloud registration, gradually extending to longer distances. Unfortunately, EYOC fails to achieve registration distances exceeding 50 meters.

Multi-vehicle registration with infrastructure methods enhances registration quality by constructing registration systems in VANETs. VI-Eye [25] leverages regular geometries in driving scenes to extract saliency points and efficiently achieve registration, enabling real-time PCR. However, the perception range of vehicles is still limited. VIPS [26] recognizes objects and utilizes their lean representations to construct a graph for matching, effectively extending the vehicle's perception range. However, the application scope of VIPS is limited due to the fact that vehicles do not share raw point cloud data. Both of the above methods require point cloud overlap between vehicles and infrastructure, which limits the coverage of infrastructure and leads to high deployment costs. EMP [5] utilizes Voronoi diagrams [27] and bandwidth considerations to partition regions for point cloud fusion. It can significantly enhance the perception range of vehicles. However, this system relies on infrastructure as an intermediary for point cloud transmission and depends on high-precision GPS to solve position relationships.

IX. Conclusion

In this work, we have developed an online long-range multi-vehicle PCR system called LoRaPCR. In LoRaPCR, vehicles can achieve long-range registration through multi-hop short-range high-quality registration. By collecting vehicle position information and registration history, the base station can assign efficient registration and transmission strategies for every registration request, reducing computation and communication overhead in VANETs. To the best of our knowledge, LoRaPCR is the first solution to achieve multi-vehicle point cloud long-range registration. We have implemented a prototype of LoRaPCR and conducted extensive experiments. The results demonstrate the effectiveness of LoRaPCR.

References

[1] H. Zhu et al., "VPFNet: Improving 3D object detection with virtual point based LiDAR and stereo data fusion," *IEEE Trans. Multimedia*, vol. 25, pp. 5291–5304, 2023.

[2] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander, "Joint 3D proposal generation and object detection from view aggregation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 1–8.

[3] C. Xu et al., "SqueezeSegV3: Spatially-adaptive convolution for efficient point-cloud segmentation," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 1–19.

[4] J. Shen et al., "mmV2V: Combating one-hop multicasting in millimeter-wave vehicular networks," in *Proc. IEEE Int. Conf. Distrib. Comput. Syst.*, 2022, pp. 735–742.

[5] X. Zhang et al., "EMP: Edge-assisted multi-vehicle perception," in *Proc. Annu. Int. Conf. Mobile Comput. Netw.*, 2021, pp. 545–558.

[6] Q. Liu, Y. Zhou, H. Zhu, S. Chang, and M. Guo, "APR: Online distant point cloud registration through aggregated point cloud reconstruction," in *Proc. Int. Joint Conf. Artif. Intell.*, 2023, pp. 1204–1212.

[7] X. Bai et al., "PointDSC: Robust point cloud registration using deep spatial consistency," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 15 859–15 869.

[8] G. D. Pais, S. Ramalingam, V. M. Govindu, J. C. Nascimento, R. Chellappa, and P. Miraldo, "3DRegNet: A deep neural network for 3D point registration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 7193–7203.

[9] Q. Liu, H. Zhu, Y. Zhou, H. Li, S. Chang, and M. Guo, "Density-invariant features for distant point cloud registration," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2023, pp. 18169–18179.

[10] F. K. Hwang and D. S. Richards, "Steiner tree problems," *Networks*, vol. 22, no. 1, pp. 55–89, 1992.

[11] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proc. Annu. Conf. Robot Learn.*, 2017, pp. 1–16.

[12] C. Choy, J. Park, and V. Koltun, "Fully convolutional geometric features," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 8958–8966.

[13] P. K. Agarwal and M. Sharir, "Efficient algorithms for geometric optimization," *ACM Comput. Surv.*, vol. 30, no. 4, pp. 412–458, 1998.

[14] Y. Li et al., "Multiagent multitraversal multimodal self-driving: Open MARS dataset," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 22 041–22 051.

[15] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser, "3DMatch: Learning local geometric descriptors from RGB-D reconstructions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 1802–1811.
[16] H. Deng, T. Birdal, and S. Ilic, "PPFNet: Global context aware local features for robust 3D point matching," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 195–205.
[17] Z. Gojcic, C. Zhou, J. D. Wegner, and A. Wieser, "The perfect match: 3D point cloud matching with smoothed densities," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 5545–5554.
[18] F. Poiesi and D. Boscaini, "Distinctive 3D local deep descriptors," in *Proc. Int. Conf. Pattern Recognit.*, 2020, pp. 5720–5727.
[19] S. Ao, Q. Hu, B. Yang, A. Markham, and Y. Guo, "SpinNet: Learning a general surface descriptor for 3D point cloud registration," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 11 753–11 762.
[20] S. Ao, Q. Hu, H. Wang, K. Xu, and Y. Guo, "BUFFER: Balancing accuracy, efficiency, and generalizability in point cloud registration," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 1255–1264.
[21] X. Bai, Z. Luo, L. Zhou, H. Fu, L. Quan, and C.-L. Tai, "D3Feat: Joint learning of dense detection and description of 3D local features," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 6358–6366.
[22] S. Huang, Z. Gojcic, M. Usvyatsov, A. Wieser, and K. Schindler, "PREDATOR: Registration of 3D point clouds with low overlap," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 4265–4274.
[23] J. Liu, X. Lv, X. Gong, Y. Liang, and J. Hyyppä, "An unsupervised learning network for large-scale LiDAR point clouds registration," *IEEE Trans. Veh. Technol*, early access, Jun. 21, 2024, doi: 10.1109/TVT.2024.3417415.
[24] Q. Liu, H. Zhu, Z. Wang, Y. Zhou, S. Chang, and M. Guo, "Extend your own correspondences: Unsupervised distant point cloud registration by progressive distance extension," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 20 816–20 826.
[25] Y. He, L. Ma, Z. Jiang, Y. Tang, and G. Xing, "VI-Eye: Semantic-based 3D point cloud registration for infrastructure-assisted autonomous driving," in *Proc. Annu. Int. Conf. Mobile Comput. Netw.*, 2021, pp. 573–586.
[26] S. Shi et al., "VIPS: Real-time perception fusion for infrastructure-assisted autonomous driving," in *Proc. Annu. Int. Conf. Mobile Comput. Netw.*, 2022, pp. 133–146.
[27] F. Aurenhammer, "Voronoi Diagrams—A survey of a fundamental geometric data structure," *ACM Comput. Surv.*, vol. 23, no. 3, pp. 345–405, 1991.

Zhenxi Wang (Student Member, IEEE) received the bachelor's degree in computer science from Shanghai Jiao Tong University, Shanghai, China, in 2022. He is currently working toward the master degree with Computer Science Department, Shanghai Jiao Tong University. His research interest mainly lies in point cloud processing and cooperative sensing.

Hongzi Zhu (Senior Member, IEEE) received the PhD degree in computer science from Shanghai Jiao Tong University, in 2009. He was a post-doctoral fellow with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, and the Department of Electrical and Computer Engineering, University of Waterloo, in 2009 and 2010, respectively. He is a professor with the Department of Computer Science and Engineering, Shanghai Jiao Tong University. His research interests include mobile sensing and computing, and Internet of Things. He received the Best Paper Award from IEEE Globecom 2016. He was a leading guest editor of the *IEEE Network Magazine*. He is an associate editor of *IEEE Transactions on Vehicular Technology* and *IEEE Internet of Things Journal*. He is a senior member of the IEEE Computer Society, IEEE Communication Society, and IEEE Vehicular Technology Society. For more information, please visit http://lion.sjtu.edu.cn.

Yunxiang Cai received the PhD degree from the Department of Computer Science and Engineering, Shanghai Jiao Tong University (SJTU), in 2023. He is currently working as an algorithm engineer with Huawei Technologies Company Ltd. His research interests include vehicular ad hoc networks, AI-empowered IoTs, and mobile computing.

Quan Liu (Student Member, IEEE) received the bachelor's degree in computer science from Shanghai Jiao Tong University, Shanghai, China, in 2021. He is currently working toward the master degree with Computer Science Department, Shanghai Jiao Tong University. His research interest mainly lies in 3D computer vision and point cloud processing.

Shan Chang (Member, IEEE) received the PhD degree in computer software and theory from Xian Jiaotong University, in 2013. From 2009 to 2010, she was a visiting scholar with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology. She was also a visiting scholar with the BBCR Research Lab, University of Waterloo, from 2010 to 2011. She is now a professor with the Department of Computer Science and Technology, Donghua University, Shanghai. Her research interests include security and privacy in mobile networks and sensor networks.

Liang Zhang (Student Member, IEEE) received the BE degree in computer science and technology from Northeastern University in China, in 2018. She is currently working toward the PhD degree with the Department of Computer Science and Engineering, Shanghai Jiao Tong University. Her research interests include stream processing and resource scheduling in the cloud or edge computing environment. For more information, please visit https://zl-cs.github.io/.

Minyi Guo (Fellow, IEEE) received the BSc and ME degrees in computer science from Nanjing University, China, and the PhD degree in computer science from the University of Tsukuba, Japan. Additionally, he is the Department of Computer Science and Engineering director of the Embedded and Pervasive Computing Center, Shanghai Jiao Tong University. Before joining SJTU, he had been a professor with the School of Computer Science and Engineering, University of Aizu, Japan. He received the national science fund for distinguished young scholars from NSFC in 2007. His present research interests include parallel/distributed computing, compiler optimizations, Big Data, and cloud computing. He has more than 400 publications in major journals and international conferences in these areas. He received 7 best/highlight paper awards from international conferences including ALSPOS 2017 and ICCD 2018. He is now editor-in-chief of *IEEE Transactions on Sustainable Computing*, and on the editorial board of *IEEE Transactions on Parallel and Distributed Systems*, *IEEE Transactions on Cloud Computing*, and *Journal of Parallel and Distributed Computing*. He is the Zhiyuan chair professor, and an ACM distinguished member.