LiDAR-based Cooperative Relative Localization

Jiqian Dong
Purdue University
dong282@purdue.edu

Qi Chen
Toyota Motor North America
qi.chen@toyota.com

Deyuan Qu
University of North Texas
deyuanqu@my.unt.edu

Hongsheng Lu
Toyota Motor North America
hongsheng.lu@toyota.com

Akila Ganlath
Toyota Motor North America
akila.ganlath@toyota.com

Qing Yang
University of North Texas
qing.yang@unt.edu

Sikai Chen
University of Wisconsin-Madison
sikai.chen@wisc.edu

Samuel Labi
Purdue University
labi@purdue.edu

***Abstract*—Vehicular cooperative perception aims to provide connected and automated vehicles (CAVs) with a longer and wider sensing range, making perception less susceptible to occlusions. However, this prospect is dimmed by the imperfection of onboard localization sensors such as Global Navigation Satellite Systems (GNSS), which can cause errors in aligning over-the-air perception data (from a remote vehicle) with a Host vehicle's (HV's) local observation. To mitigate this challenge, we propose a novel LiDAR-based relative localization framework based on the iterative closest point (ICP) algorithm. The framework seeks to estimate the correct transformation matrix between a pair of CAVs' coordinate systems, through exchanging and matching a limited yet carefully chosen set of point clouds and usage of a coarse 2D map. From the deployment perspective, this means our framework only consumes conservative bandwidth in data transmission and can run efficiently with limited resources. Extensive evaluations on both synthetic dataset (COMAP) and KITTI-360 show that our proposed framework achieves state-of-the-art (SOTA) performance in cooperative localization. Therefore, it can be integrated with any upper-stream data fusion algorithm and serves as a preprocessor for high-quality cooperative perception.**

***Index Terms*—Cooperative Localization, Error Calibration, Point Cloud Registration**

I. Introduction

Vehicle-to-Everything communications (V2X) enable connected and automated vehicles (CAVs) to "talk" to each other and operate cooperatively. As an immediate example, cooperative perception shares vehicles' local perception information (raw or processed) with surrounding peers to achieve a wider and longer perception field [1], [2]. There are many driving scenarios where cooperative perception can be beneficial. For example, as shown in the first row of Fig. 1 two CAVs are approaching an intersection from different streets. Cooperative perception information can help them "see" objects on other streets that may be occluded to their onboard sensors. In this setting, two cooperative CAVs are referred as Remote Vehicles (RVs) and Host Vehicles (HVs), respectively, with RV transmitting the over-the-air information to HV. Such over-the-air information offers extra lead time for both vehicles to make well-informed decisions, minimize collision risk, and ultimately increase navigation safety and efficiency.

Despite having great potential, the performance of cooperative perception hinges on the quality of sensor data fusion at recipient vehicles. Due to the imperfection of positioning sensors such as the Global Navigation Satellite System (GNSS), localization errors are inevitable, which can further cause data misalignment in coordinate transformation. As illustrated in the second column of Fig. 1, the HV (in blue) might fail to align RV's (in red) cooperative perception data as a consequence of the error in estimating RV's pose using GNSS. Under this circumstance, the RV's data is noisy and less useful.

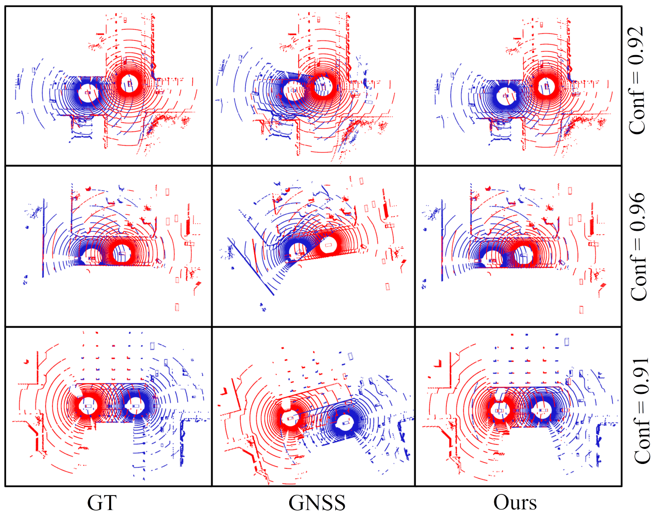


Fig. 1: Qualitative results of the proposed method. red: RV, blue: HV, Each row represents a different scene sampled from COMAP dataset. The columns represent the fused point clouds using ground truth (GT) data, raw GNSS measures (GNSS), and our method (Ours), respectively. The *Conf* indicates the confidence level output by our method.

To mitigate this problem, we propose a cooperative localization framework that runs as a preprocessor for all the downstream data fusion steps. It significantly reduces the relative localization error between CAVs using LiDAR point cloud registration. This framework achieves the state-of-the-art (SOTA) performance and can boost the data fusion as shown in the third column in Fig. 1. In particular, our framework sends only a small yet carefully chosen subset of RV's LiDAR point cloud to HV for point cloud registration, making it suitable for low-bandwidth V2X communication technologies (e.g., DSRC, LTE-V2X). These selected LiDAR points, called

keypoints, are combined with the 2D map data at HV to help the point cloud registration process find a high-quality transformation matrix even when few correspondences between HV's and RV's point clouds exist. Moreover, our algorithm features an additional "sanity-check" component that can estimate the quality of the computed transform matrix and only outputs accurate ones for downstream data fusion tasks.

From the technical perspective, the key novelty of the proposed method lies in our keypoints preprocessing module and an inlier augmentation mechanism in the inter-CAV point cloud registration step of our framework. It is worth noting that the point cloud registration step in our framework reuses the prior art, the Iterative Closest Point (ICP) algorithm. We duly recognize that although there are many more advanced alternative choices, the fundamental challenges to all of these solutions (including ICP) are that they all need a non-trivial overlap between the RV and HV's point clouds with a certain inlier-outlier ratio. Moreover, the existing approaches generally search correspondences using two full point clouds and may require the RV to send the entire 3D scan to HV. This is impractical in our scenario as the communication bandwidth among CAVs is limited and shared. Also, as RV and HV's relative location keeps varying over time, it is not guaranteed to always have enough "co-visible" regions (overlaps in the field of views) to find enough inliers. This requires any solution to self-adjust and rebalance the inliers and outliers. In this work, we employ the ICP algorithm as an example to demonstrate the architecture and flow of our framework, but the framework has the flexibility to be integrated with many other point cloud registration algorithms.

The main contributions of this work are summarised as follows:

- We propose a 3D LiDAR points registration framework which obtains the SOTA performance on relative pose estimation between cooperative CAVs.
- The proposed algorithm suppresses more than 80% of errors in relative pose estimation for CAVs, evaluated on both the simulated dataset (COMAP) and the realistic dataset created from KITTI-360.
- The proposed approach sends a small portion of the HV's point cloud (2.7% from COMAP and 15% from KITTI-360), significantly reducing requirement on communication bandwidth. It is also computationally efficient, making it suitable for edge devices such as vehicle onboard computing units.

The rest of the paper is organized as follows: Section II reviews the relative works for point cloud registration and vehicle localization; Section III introduces our overall matching framework; Section IV and V document the experiments settings and the corresponding results; and finally Section VI concludes the work.

II. Relative Works

A. Point Cloud Registration

Point cloud registration is the process of estimating the relative transformation between two sets of point clouds and aligning them into the same coordinate frame, which in essence solves the same problem as our LiDAR-based cooperative localization. There are two categories in point cloud registration methods, namely traditional approaches and learning-based approaches. Traditional approaches seek to find the transformation by either minimizing the point-wise distances e.g., ICP [3] and its many variants [4]–[6], or between correspondences found using hand-crafted features such as SHOT [7], Fast Point Feature Histogram (FPFH) [8], RoPS [9]. Generally, finding transformation through hand-crafted features is susceptible to outliers, therefore Random Sample Consensus (RANSAC) [10] is often applied for outlier filtering. In addition, there are other traditional approaches such as TEASER [11] that reformulate the point registration as optimization problems, which have demonstrated superior performance compared to RANSAC. More recently, many learning-based approaches leveraging deep neural networks to either generate matching features [12]–[15] or directly end-to-end learning the relative transformation were published [16], [17]. Despite with great prosperity, learning-based methods suffer from low interpretability and generally need large volumes of data for training compared to traditional approaches.

B. LiDAR-based Multi-Vehicle Localization

CAVs with LiDAR sensors can leverage perception-based vehicle localization, which generally has higher accuracy than GNSS in self-localization. Currently, cooperative localization can leverage the High-definition (HD) maps [18]–[20] or LiDAR odometry [21]–[23] for relative pose estimation and motion calibration. However, these methods need to store historical motion information and multiple LiDAR scans at different timestamps and can have a high requirement on the hardware. A more recent work using LiDAR point cloud registration for cooperative localization between two CAVs is [24]. In this work, the authors utilize maximum consensus [25] to find correspondence in the keypoints and calibrate the relative localization error.

C. Difference of Our Work

Compared to the previous works, ours differ in the following ways. First, the RV and HV can have limited "co-visible" regions, resulting in insufficient correspondence between point clouds for the aforementioned algorithms to function appropriately. Second, the communication bandwidth between RV and HV can be limited, which makes sharing the entire LiDAR point cloud infeasible. Thirdly, while it can be beneficial, we don't require CAVs to have an advanced positioning system based on LiDAR or camera. Therefore, from the deployment perspective, it has low requirements on the hardware and can be embedded into the existing CAV systems without additional perquisites.

III. Methodology

This section introduces the proposed cooperative localization framework. Fig. 2 shows its overall architecture which

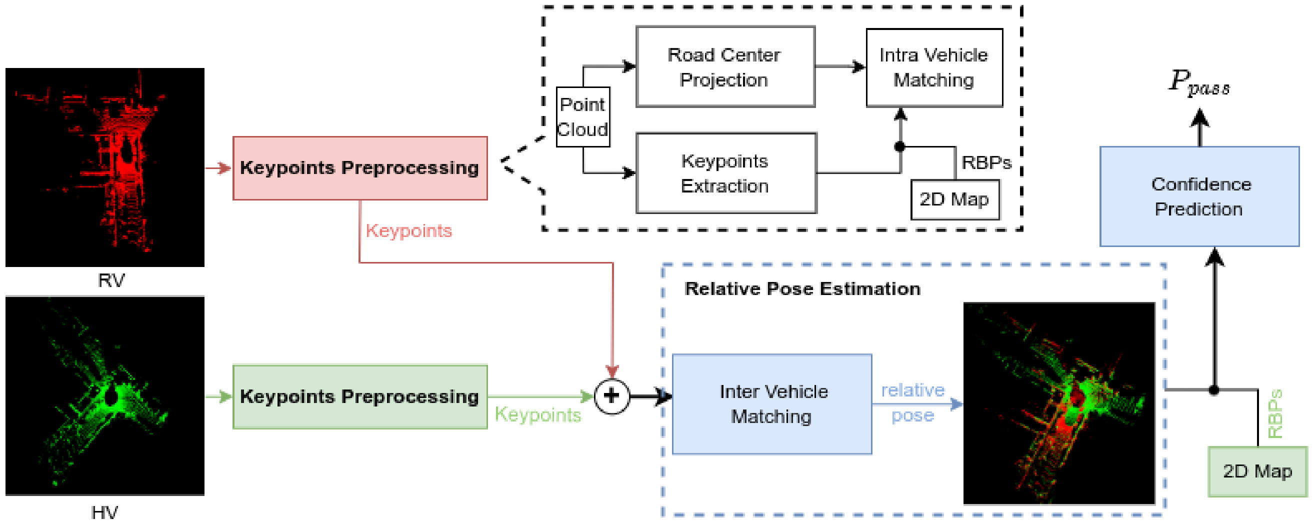


Fig. 2: Overall framework (red:RV, green:HV, blue:HV after fusion; RBPs: road boundary points)

consists of three components: **Keypoints Preprocessing**, **Relative Pose Estimation**, and **Confidence Prediction**. For Keypoints Preprocessing, it can be divided into three submodules: Road Center Projection, Keypoints Extraction, and Intra Vehicle Matching. The working flow is as follows: the Keypoint Preprocessing module runs in parallel on both HV and RV independently to obtain representative and high-quality keypoints. Then the keypoints from RV are transmitted to HV. After that, the Relative Pose Estimation module deployed on HV computes the relative pose between the two CAVs. Finally, the Confidence Prediction module evaluates the results of upstream localization modules and removes low-confident outputs. More details on the design of each module can be found below.

A. *Keypoints Preprocessing*

1) Keypoints Extractor: As the name suggests, this module utilizes a deep learning based point classifier to extract the most representative points (keypoints) from the raw LiDAR point cloud. Inspired by the work in [24], we configure RV to select LiDAR points estimated to be reflected by road boundaries, buildings, fences, and walls as the keypoints. The reason is that these types of points well captured the geometry of the environment. This geometric information exists on both vehicles' map data and can be leveraged as correspondence in the point cloud registration process later on. We adopt a Voxelnet-based architecture similar to the feature extraction module in [26] to classify LiDAR points. As shown in Fig. 3, the network consists of two branches for obtaining point features: one for preserving raw point-based features while the other for generating voxel-based features. To reduce the computation load, Furthest Points Sampling (FPS) approach is applied to evenly sample a predefined number of sparse points from the raw 3D scans. To generate voxel-wise features, we adopt a 4-sparse-convolution-layer based sparse CNN backbone with a downsampling factor of 8. Then the

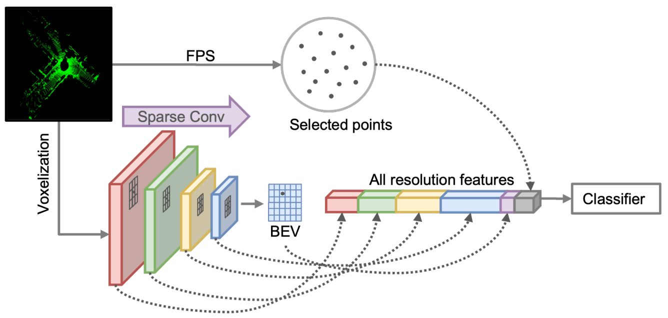


Fig. 3: Key point extractor architecture

output from the last sparse convolutional layer is compressed in height to generate an eye view (BEV). Based on the selected sparse points from the first branch, we further use the Voxel Set Abstraction (VSA) with the same parameters as in [26] to aggregate the voxel-based point features of different resolutions, the BEV features, and the raw point-wise features. Then, this combined feature map is fed into a classifier for label prediction.

2) Lane Center Projection: CAVs that rely on GNSS for positioning service can suffer large translation errors w.r.t the ground truth location. Such error impacts the relative localization performance by providing a noise-impaired initialization. To mitigate this error, we leverage 2D map data. More specifically, we project a vehicle's noisy GNSS location to the "closest" lane center point on the map, a technique widely used in navigation applications today. More specifically, the "closest" is defined in terms of both geometrical distance and heading direction, which is to guarantee that the vehicles are projected to the correct side of the road. By preprocessing the 2D map, we can obtain the lane center points (LCPs: $\boldsymbol{x} = [x, y]^T$) with their corresponding heading directions ($\boldsymbol{h}$), as shown in Fig. 4a: black points represent the LCPs, and

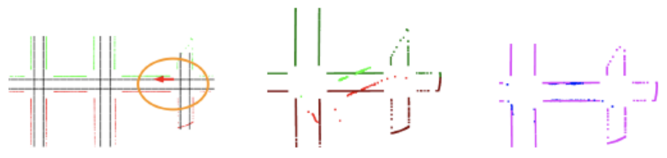

(a) 2D map information (b) before Intra Vehicle Matching (c) after Itra Vehicle Matching

Fig. 4: Illustration of an Exemplary 2D Map and Intra Vehicle Matching

the red arrow indicates one example point's heading direction. Then based on the raw pose estimation, we identify the closest LCP following Eqn. 1

$$[\boldsymbol{x_c}, \boldsymbol{h_c}] = \underset{\boldsymbol{x_c} \in \boldsymbol{x}, \boldsymbol{h_c} \in \boldsymbol{hs}}{\operatorname{argmin}} \ \omega_h |\boldsymbol{h_v} \quad \boldsymbol{h_c}| + \omega_t ||\boldsymbol{x_v} - \boldsymbol{x_c}||_2 \quad (1)$$

where $\boldsymbol{x_v}$ represents the vehicle location from raw GNSS. ω_h and ω_t are the weight for balancing heading difference and translation difference, respectively. Generally, in order to prevent projecting point clouds to the wrong side of the road, $\omega_h >> \omega_t$. Overall, in this stage, all the keypoints obtained from keypoints extractor are first projected to the global frame using raw vehicle poses (from GNSS). Then these points are moved onto the road with the translation $\boldsymbol{x_v} - \boldsymbol{x_c}$. For simplicity, this translation operation is denoted as $\boldsymbol{T_0^r}$ and $\boldsymbol{T_0^h}$ for the RV and HV, respectively.

3) Intra Vehicle Matching: Apart from positional errors, heading errors can also significantly impair initialization quality for the downstream point cloud registration algorithm, which can even lead the process to diverge. Therefore, we design an intra vehicle matching module for keypoints refinement so as to create a better initialization for point cloud registration between RV and HV. Here, "intra" means this stage is conducted within each individual CAV so as to match its keypoints with its own map data. More specifically, for HV (the process is identical for RV), we match its "road boundary points" (RBPs) (i.e. Keypoints from previous step and labeled as "road boundary“) with the corresponding RBPs from the map as shown in Fig. 4b and 4c. This stage requires the map RBPs to have directional information indicating the "left" and "right" w.r.t the projected LCP's heading direction ($\boldsymbol{h_c}$) as plotted in Fig. 4a, where red and green points indicating left RBPs and right RBPs, respectively. The RBPs from vehicles obtained from keypoints extractor are also labeled with "left" and "right" based on the sign of points' y coordinates in the local framework.

With these ingredients, the intra-vehicle matching stage seeks to register RBPs using the "class-based ICP" algorithm whose key idea is to match the closest points from different clouds yet with the same class. The outputs of the intra-vehicle matching stage contain: a) two transformation matrices ($\boldsymbol{T_1^r}$ and $\boldsymbol{T_1^h}$) that can align the RBP keypoints from both RV and HV to the road in the map, respectively; b) 2 matching "supports" s_1^r and s_1^h, (the proportion of points that can be matched, range from 0-1) indicating the matching performance for RV and HV. These support values are saved for the later confidence prediction module. The effect of this stage, as shown in Fig. 4c (blue points are from the vehicle, pink points are from the map), indicates the CAV RBPs are having better alignment with the map RBPs with a much smaller heading error. After this stage of calibration, the keypoints from HV can be transmitted to RV for relative pose estimation.

B. Relative Pose Estimation

So far, the Keypoints Preprocessing module has provided a good initialization for inter-vehicle localization. However, there may still exist the problem of a low inlier-outlier ratio, which can further lead to poor registration performance. In theory, for two sets of point clouds to be registered successfully, there needs to exist sufficient correct point-wise correspondence. This is the underlying assumption for all the algorithms surveyed in Sec. II. Such an assumption may not always hold for driving scenarios. Considering a scenario when an HV and an RV drive from perpendicular streets to the same intersection (first row in Fig. 1). The correspondences between their keypoints only exist in the "co-visible" region that can be commonly seen by both CAVs, which may only take a small portion of all the keypoints. Therefore, the ratio between number of points with correspondence (inliers) to the rest pairs of points (outliers), called inlier-outlier ratio, is tiny. This can cause failure to any point cloud registration algorithms. Moreover, there exits no way for HV and RV to know this ratio ahead of time and stop the algorithm accordingly. So in our design, we specifically design an inlier augmentation mechanism so that our algorithm can accommodate a wide range of inlier-outlier ratios.

We propose to improve the inlier-outlier ratio by the usage of the aforementioned 2D map. More specially, after getting RV's keypoints, HV can recompute the map RBPs that RV has matched in its previous intra-vehicle matching stage. This can be achieved simply by sampling on HV's 2D map for map points that are around or nearest to RV's shared RBP keypoints. HV adds these regenerated artificial RV map RBP to its keypoints set as if they were scanned by HV's LiDAR. Then the augmented HV keypoints are registered with the received RV keypoints. Note that the artificial RV map RBPs regenerated by HV are inliers to the received RV keypoints and can serve as anchors to prevent point cloud registration algorithms from completely diverging when facing low inlier-outlier ratio scenarios. In addition, this augmentation can also balance the effect of intra-vehicle matching and inter-vehicle matching, preventing the latter stage from completely wash away the effects of the former stage. After augmenting, we again leverage the class-based ICP algorithm to compute the transformation matrix between HV and RV. The outputs from this stage are a transformation matrix $\boldsymbol{T_2}$ and its corresponding support value s_2.

C. Confidence Predictor

As the system starts with a deep learning-based keypoints extraction module, there may exist errors and uncertainties, which can cause failure in the subsequent matching stages. In order to be deployed in real CAVs, an additional alarming function is needed to judge whether the matching result is good enough for cooperative sensing. Therefore, we further design this module to output a confidence score for the entire matching framework. We first conduct another round of class-based ICP to match the fused RBPs (containing both HV and RV's RBPs) with the map RBPs, which will output another transformation matrix $\boldsymbol{T_3}$ and its corresponding matching support s_3. Then we leverage the data-driven approach and train a logistic regression model to predict a binary flag ("pass" or "fail") indicating whether to accept the results from the matching algorithm. The model takes all the pre-computed four "support" values from all the matching stages (s_1^h, s_1^r, s_2, and s_3) as input and output the probability of "pass". The labels for training the model can be customized according to the application context (e.g., the translation and heading errors of relative localization below certain thresholds). We report the model performance on different criteria on different in the later V-B section. In addition, $\boldsymbol{T_3}$ can be used to further refine the HV's pose if the predicted confidence is high enough.

The above matching steps are summarized in the Alg. 1

Algorithm 1: Relative Localization Algorithm

```
Data: Map RBPs, LCPs, RV keypoints $\boldsymbol{K^r}$, HV keypoints $\boldsymbol{K^h}$
/* In the local frame of HV and RV */
1  $\boldsymbol{T_0^r}$ = LaneCenterProject($\boldsymbol{K^r}$, LCPs);
2  $\boldsymbol{T_0^h}$ = LaneCenterProject($\boldsymbol{K^h}$, LCPs);
3  $\boldsymbol{\hat{K}^r} = \boldsymbol{T_0^r K^r}$, $\boldsymbol{\hat{K}^h} = \boldsymbol{T_0^h K^h}$;
4  $\boldsymbol{T_1^r}, s_1^r$ = ClassBasedICP($\boldsymbol{\hat{K}^r}$, RBPs) ;
5  $\boldsymbol{T_1^h}, s_1^h$ = ClassBasedICP($\boldsymbol{\hat{K}^h}$, RBPs) ;
6  $\boldsymbol{\hat{K}^r} = \boldsymbol{T_1^r K^r}$, $\boldsymbol{\hat{K}^h} = \boldsymbol{T_1^h K^h}$;
7  Transmit $\boldsymbol{\hat{K}^r}$ and $s_1^r$ to HV ;
/* In global frame */
8  $\boldsymbol{\hat{K}^h}$ = InlierAugment($\boldsymbol{\hat{K}^h}, \boldsymbol{\hat{K}^r}$, MPBs);
9  $\boldsymbol{T_2}, s_2$ = ClassBasedICP($\boldsymbol{\hat{K}^r}$, $\boldsymbol{K^h}$) ;
10 $\boldsymbol{\hat{K}^r} = \boldsymbol{T_2 \hat{K}^r}$;
11 $\boldsymbol{K_{fused}}$ = Concatenate($\boldsymbol{\hat{K}^h}, \boldsymbol{\hat{K}^r}$)
12 $\boldsymbol{T_3}, s_3$ = ClassBasedICP($\boldsymbol{K_{fused}}$ , MPBs)
13 Conf = LogisticRegression($s_1^r, s_1^h, s_2, s_3$)
```

IV. Experiments

A. Datasets

1) Synthetic Dataset COMAP: COMAP [27] is a synthetic dataset that is co-simulated by CARLA [28] and SUMO [29]. It is suitable for the cooperative perception-related task since each frame contains multiple cooperative CAVs with varying sizes of "co-visible" regions. The dataset contains 7788 frames of samples, with each frame containing point clouds, accurate vehicle poses, and object detection bounding boxes for all cooperative CAVs. In addition, the "virtual" sensors implemented in CARLA can provide high-quality and realistic measures with ground truth labels for the point clouds, which can facilitate the training of our keypoint extractor neural network. Hence, in this paper, we utilize this synthetic dataset collected for empirical studies, following the concept of multiple pioneering works such as [30], [31].

2) Realitic Dataset from KITTI-360: To further test the proposed algorithm in more realistic situations, we modify the KITTI-360 dataset [32] to create cooperative perception scenarios. More specifically, since the original KITTI-360 dataset only contains frames for one single autonomous vehicle (AV), we fuse two frames of the same AV at different timestamps and treat them as 2 cooperative CAVs. To guarantee the existence of "co-visible" regions between 2 CAVs, the two frames selected are constrained in time difference.

3) Error Injection: The raw data in both COMAP and KITTI-360 only contain the accurate vehicle poses for both RV and HV. However, the ultimate goal of our proposed algorithm is to reduce the error in the relative pose for cooperative perception, which requires an "erroneous" dataset for evaluation. In this work, we manually inject noises into the vehicle poses by randomly adding translations in x- and y- direction and rotation in the yaw (heading) from zero mean normal distributions: $\alpha\mathcal{N}(\boldsymbol{0}, \boldsymbol{\sigma})$. Where α represents the error scale, which can reflect the uncertainty in the GNSS system for each individual CAV. In the subsequent experiments, we set the standard deviation of the translation error as $\boldsymbol{\sigma_x} = [1, 1]^T(m)$ and the standard deviation of heading error as $\sigma_\psi = 2°$, then vary the α value from 1 to 8.

B. Baseline Methods

To validate the performance of the proposed matching algorithm, three baseline approaches with the same keypoints configuration are adopted in this study:

1) Maximum consensus algorithm: As introduced in [25] and utilized in [24] maximum consensus is a greedy approach that searches over a grid of potential transformation matrices and picks the one that can produce the maximum number of matches. By definition, it is not an iterative approach and can have a large search space when GNSS error is non-trivial.

2) Vanilla ICP algorithm: This baseline utilizes a vanilla class-based ICP algorithm to match the keypoints extracted from raw 3d scans. The reason for only applying ICP on the keypoints is that it has the same transmission requirement as the proposed hierarchical matching algorithm.

3) Raw GNSS: This is the "do nothing" baseline that purely relies on the raw GNSS measurements to compute the relative pose between RV and HV.

C. Experiment Setup

1) Training keypoint extractor: To train our keypoint extractor model, we use the cross entropy loss with 5 categories and a set of class balance weights. Overall 25 epochs are trained on the COMAP training set (70% of 7788 frames) with the Adam optimizer and an initial learning rate of 0.001. As for the other training parameters, we utilize the same configuration as in [24]. After training, the model is sufficient to output keypoints for downstream matching.

2) Prerequisites on map: As mentioned in the methodology section, our proposed matching framework requires a "coarse" 2D map with some information including road boundary points (RBPs) and lane center points (LCPs) shown in Fig. 4a. Road boundary points can be obtained either through 2D satellite maps such as Google Maps or 3D LiDAR scans from multiple AV trips. In our work, we adopt the latter approach and extract the envelope of all the road points detected by AV in multiple trips. More specifically, we first aggregate all the road points from all CAVs and all frames, then utilize Polylidar 3D [33] tool to extract the boundary polygon as the road boundary points. Once the road boundary points are obtained, the rest information as in Fig. 4a can be acquired via geometry.

3) Matching radius: There is only one hyper-parameter in each class-based ICP step, which is the searching radius for inlier matching points. In other words, for each keypoint in the RV, the class-based ICP finds the nearest neighbor in the HV's keypoints set. If the distance of this pair of two points is below the radius threshold r, they are considered a "inlier". Then, the overall transformation matrix is computed through all the inliers. Intuitively, this radius should increase as the initialization error increases. Based on our experiments, the algorithm yields the best performance if setting radius r as the function of error scale: $r = 3\alpha + 2$ for intra vehicle matching stage and a fixed radius as 3m for inter vehicle matching and confidence regression stages.

4) Evaluation Metrics: The output from our proposed framework is a "relative" pose between the RV and HV CAVs, which is a transformation matrix that describes the pose of RV CAV from HV CAV's perspective. To evaluate the efficacy of the proposed framework, following the previous work [15], [34], we report the error between the calibrated relative pose with the ground truth pose in terms of relative translation error Δ (m) and relative heading angle error θ (degree).

V. Results

A. Quantitative Matching Results

1) On COMAP dataset: The experiments on COMAP dataset are conducted by varying the error scale factor from 1-8. The mean values of translation errors Δ and heading errors θ are documented in Tab. I. The columns in Tab. I represent relative pose errors for the following experiments: the raw GNSS measure (**GNSS**), maximum consensus algorithm (**max con**), vanilla class-based ICP (**ICP**, no intra vehicle matching stage), and our proposed approach (**ours**).

2) On KITTI-360 dataset: The same experiments as on the COMAP dataset are conducted on the selected KITTI-360 scenarios. To prevent repetition, we report the same statistics with error scales $[1, 3, 5, 8]$ in Tab. II indicating the small, medium, and large error cases.

From these results, the proposed matching approach attains the highest accuracy in cooperative localization with the lowest errors for both translation and orientation than other baselines. Also, as the error scale varies from 1-8, our approach output consistently better results with an error reduction rate higher than 70%. In contrast, baselines including vanilla class-based ICP and maximum consensus algorithms can show only decent performance in the low error cases. As the error scale increase, their performance decreases drastically, and the maximum consensus can even amplify the errors in high error scale cases.

As an ablation study, we remove the inlier-augmentation operation as introduced in III-B while keeping the rest modules of the framework unchanged. The result is in the **ours-aug** columns in Tab. I, which indicates the proposed inlier-augmentation mechanism is effective.

B. Results on Confidence Prediction

As mentioned before, the confidence prediction model can be trained with customized labels. In this work, we test the following criteria for generating positive samples (pass):

- Criterion I: After the two stage calibration, $\Delta_2 < 1.5m$ & $\theta_2 < 3°$.
- Criterion II: After the two stage calibration, $\Delta_2 < 3m$ & $\theta_2 < 5°$.
- Criterion III: After calibration, the relative localization error is smaller than using raw GNSS measurements: $\Delta_2 < \Delta_{GNSS}$ & $\theta_2 < \theta_{GNSS}$.

In Tab III, we report the proportion of positive samples over all the data (the proportion that satisfies the criteria in all the frames), the precision, recall, F1 score, and Area under ROC curve (AUC) for the regression model. In real-world applications, one can also customize the confidence threshold to adjust precision and recall for "pass" and "fail".

C. Transmission Bandwidth Analysis

As a main contribution of this work is to achieve accurate cooperative localization by only transmitting keypoints, in this section, we compare the number of selected keypoints versus the number of total points in the original frame for both COMAP and KITTI-360 dataset. The mean, standard deviation (Std.), and selection rate (Rate) for all the frames are shown in Tab. IV. For the KITTI dataset, we use all the keypoints (reflecting the static objects including walls, fences, buildings, and road boundaries) in the frame without FPS. Therefore, the number of keypoints is much greater than COMAP. This is also one of the underlying reasons that our proposed framework has higher performance on KITTI-360 than COMAP as reflected in Tab. I and Tab. II

D. Visualization of Matching Results

1) COMAP: The visualization of fused point clouds from three random cases with high matching confidence is shown in Fig. 1. Specifically, the "confidences" in the figure are trained with criterion I: $\Delta_2 < 1.5m$ & $\theta_2 < 3°$. These results indicate the efficacy of the proposed approach in that both point clouds and detected vehicle bounding boxes inside the co-visible regions can be registered.

2) KITTI-360: Fig. 5 plots the fused point clouds (with highlighted regions) in one representative scenario under different error scales. As the error scale increases, the remaining errors in translation and heading increase. When the error scale reaches 8, the matching fails to output a high-quality

TABLE I: Relative pose errors on COMAP

error	Mean error in translation Δ (m)						Mean error in heading θ (degree)					
scale	**GNSS**	**max con**	**ICP**	**ours-aug**	**ours**	**reduction rate**	**GNSS**	**max con**	**ICP**	**ours-aug**	**ours**	**reduction rate**
1	2.29	1.60	1.77	0.50	**0.31**	0.82	2.75	0.89	1.20	0.37	**0.24**	0.75
2	4.58	4.95	2.33	0.58	**0.37**	0.89	5.60	1.94	1.30	0.39	**0.25**	0.92
3	6.87	9.96	3.00	0.87	**0.72**	0.89	8.48	6.01	1.64	0.59	**0.55**	0.90
4	9.17	14.20	4.57	1.13	**0.94**	0.89	11.32	17.20	4.58	0.81	**0.68**	0.90
5	11.46	16.73	6.75	1.98	**1.55**	0.86	14.16	28.32	8.06	2.10	**1.37**	0.89
6	13.75	16.65	10.62	2.74	**2.09**	0.85	17.00	29.61	12.94	2.77	**2.12**	0.89
7	16.04	15.20	13.06	3.74	**2.82**	0.82	19.84	36.62	15.93	3.84	**3.21**	0.86
8	18.33	15.00	14.39	5.61	**4.16**	0.74	22.67	30.60	18.70	8.06	**6.00**	0.81

TABLE II: Relative pose errors on KITTI-360

error	Mean error in translation Δ (m)					Mean error in heading θ (degree)				
scale	**GNSS**	**max con**	**ICP**	**ours**	**reduction rate**	**GNSS**	**max con**	**ICP**	**ours**	**reduction rate**
1	2.60	2.41	1.01	**0.34**	0.81	4.50	3.85	2.37	**0.60**	0.81
3	6.83	7.62	1.12	**0.62**	0.86	13.81	11.97	2.62	**0.95**	0.91
5	11.68	6.62	1.55	**0.82**	0.88	22.15	20.75	3.52	**1.12**	0.92
8	18.22	22.33	4.45	**1.88**	0.86	35.08	32.21	11.42	**2.40**	0.90

TABLE III: Confidence prediction results

criteria	proportion	precision	recall	F1	AUC
I	0.80	0.95	0.83	0.88	0.91
II	0.83	0.97	0.85	0.91	0.94
III	0.94	0.96	0.73	0.83	0.76

TABLE IV: # of keypoints v.s. # total points

Dataset	# Keypoints		# All points		Rate
	Mean	**Std.**	**Mean**	**Std.**	
COMAP	497	71	18102	422	0.027
KITTI-360	17995	8736	116710	5416	0.154

localization, as shown in the last column of Fig. 5, where RV's point clouds are flipped. In this case, the confidence prediction is 0.32. The system will output a "fail" flag indicating the localization results are unreliable.

VI. Conclusion

In this paper, we present a LiDAR-based matching framework to achieve accurate relative localization, which provides a solid foundation for cooperative perception. The framework sequentially applies class-based ICP to refine the relative pose between the RV and HV. According to the results from extensive empirical experiments conducted on both the simulated dataset COMAP and the realistic dataset KITTI-360, our framework can significantly reduce the errors from GNSS and outperform multiple baselines by a great margin. Furthermore, it contains a confidence prediction module that can automatically judge the reliability of the matching results. In addition, the proposed framework only has low requirements on self-localization solutions such as GNSS sensors and has the benefit of saving transmission bandwidth and computation resources. Therefore, it can be directly deployed as the preprocessor for any downstream cooperative sensing applications. Moving forward, cooperative localization can be further enhanced by jointly considering multiple sensors other than LiDAR. In our future work, we will explore the matching algorithm for simultaneously considering LiDAR and cameras, and also consider the temporal fusion when historical information is available. Furthermore, the algorithm can be tested in real-world experiments for robustness and efficiency.

References

[1] Q. Chen, S. Tang, Q. Yang, and S. Fu, "Cooper: Cooperative perception for connected autonomous vehicles based on 3d point clouds," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, pp. 514–524.

[2] Q. Chen, X. Ma, S. Tang, J. Guo, Q. Yang, and S. Fu, "F-cooper: Feature based cooperative perception for autonomous vehicle edge computing system using 3d point clouds," in *2019 ACM/IEEE Symposium on Edge Computing (SEC)*, p. 88–100.

[3] P. Besl and N. D. McKay, "A method for registration of 3-d shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992.

[4] S. Rusinkiewicz and M. Levoy, "Efficient variants of the icp algorithm," in *Proceedings third international conference on 3-D digital imaging and modeling*. IEEE, 2001, pp. 145–152.

[5] C. Olsson, F. Kahl, and M. Oskarsson, "Branch-and-bound methods for euclidean registration problems," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 5, pp. 783–794, 2008.

[6] J. Yang, H. Li, and Y. Jia, "Go-icp: Solving 3d registration efficiently and globally optimally," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 1457–1464.

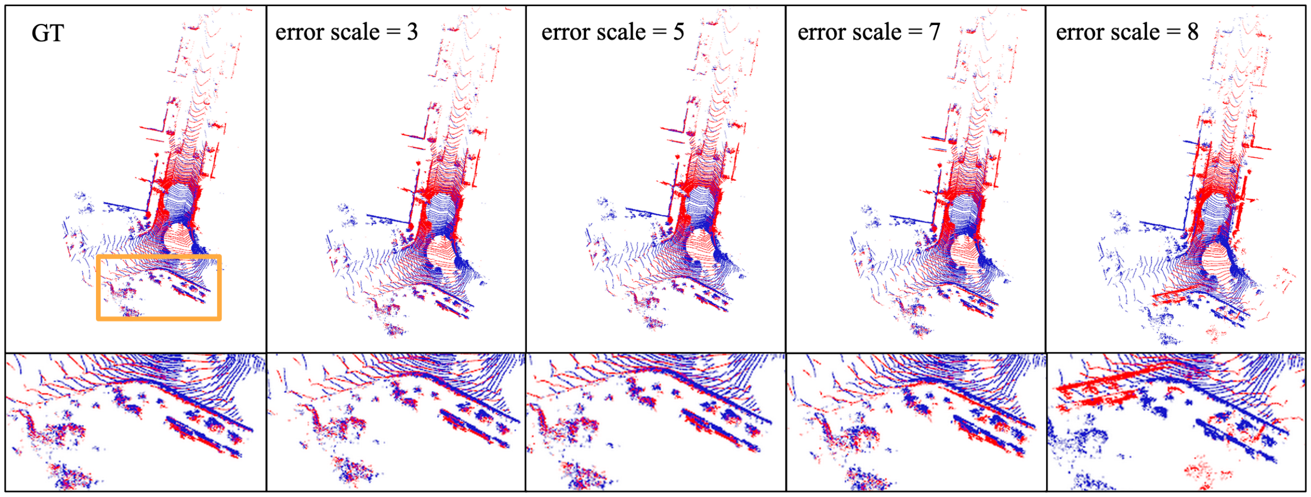
[7] F. Tombari, S. Salti, and L. Di Stefano, "Unique signatures of histograms for local surface description," in *European conference on computer vision*. Springer, 2010, pp. 356–369.

[8] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *2009 IEEE International Conference on Robotics and Automation*, 2009, pp. 3212–3217.

[9] Y. Guo, F. Sohel, M. Bennamoun, M. Lu, and J. Wan, "Rotational projection statistics for 3d local surface description and object recognition," *International journal of computer vision*, vol. 105, no. 1, pp. 63–86, 2013.

[10] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[11] H. Yang, J. Shi, and L. Carlone, "Teaser: Fast and certifiable point cloud registration," *IEEE Transactions on Robotics*, vol. 37, no. 2, pp. 314–333, 2020.



	error scale = 3	error scale = 5	error scale = 7	error scale = 8
GNSS	Δ = 1.89m, θ = 16.71°	Δ = 3.64m, θ = 28.31°	Δ = 5.77m, θ = 39.64°	Δ = 6.97m, θ = 45.31°
Ours	Δ = 0.07m, θ = 0.08°, Conf = 0.97	Δ = 0.07m, θ = 0.11°, Conf = 0.97	Δ = 0.18m, θ = 0.45°, Conf = 0.97	Δ = 2.58m, θ = 59.03°, Conf = 0.32

Fig. 5: Fused point clouds on KITTI-360 dataset, first row: GNSS, second row: ours; red: RV, blue: HV

[12] H. Deng, T. Birdal, and S. Ilic, "Ppfnet: Global context aware local features for robust 3d point matching," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 195–205.

[13] Z. J. Yew and G. H. Lee, "3dfeat-net: Weakly supervised local 3d features for point cloud registration," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 607–623.

[14] H. Deng, T. Birdal, and S. Ilic, "3d local features for direct pairwise registration," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3244–3253.

[15] E. Arnold, S. Mozaffari, and M. Dianati, "Fast and robust registration of partially overlapping point clouds," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1502–1509, 2021.

[16] V. Sarode, X. Li, H. Goforth, Y. Aoki, R. A. Srivatsan, S. Lucey, and H. Choset, "Pcrnet: Point cloud registration network using pointnet encoding," *arXiv preprint arXiv:1908.07906*, 2019.

[17] W. Lu, G. Wan, Y. Zhou, X. Fu, P. Yuan, and S. Song, "Deepvcp: An end-to-end deep neural network for point cloud registration," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 12–21.

[18] F. Ghallabi, G. El-Haj-Shhade, M.-A. Mittet, and F. Nashashibi, "Lidar-based road signs detection for vehicle localization in an hd map," in *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2019, pp. 1484–1490.

[19] A. Poulose, M. Baek, and D. S. Han, "Point cloud map generation and localization for autonomous vehicles using 3d lidar scans," in *2022 27th Asia Pacific Conference on Communications (APCC)*, 2022, pp. 336–341.

[20] X. Lin, F. Wang, B. Yang, and W. Zhang, "Autonomous vehicle localization with prior visual point cloud map constraints in gnss-challenged environments," *Remote Sensing*, vol. 13, no. 3, 2021. [Online]. Available: https://www.mdpi.com/2072-4292/13/3/506

[21] J. Zhang and S. Singh, "Loam: Lidar odometry and mapping in real-time." in *Robotics: Science and Systems*, vol. 2, no. 9. Berkeley, CA, 2014, pp. 1–9.

[22] ——, "Visual-lidar odometry and mapping: low-drift, robust, and fast," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 2174–2181.

[23] X. Zheng and J. Zhu, "Efficient lidar odometry for autonomous driving," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 8458–8465, 2021.

[24] Y. Yuan, H. Cheng, and M. Sester, "Keypoints-based deep feature fusion for cooperative vehicle detection of autonomous driving," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 3054–3061, 2022.

[25] T.-J. Chin and D. Suter, "The maximum consensus problem: recent algorithmic advances," *Synthesis Lectures on Computer Vision*, vol. 7, no. 2, pp. 1–194, 2017.

[26] S. Shi, C. Guo, L. Jiang, Z. Wang, J. Shi, X. Wang, and H. Li, "Pv-rcnn: Point-voxel feature set abstraction for 3d object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 10 529–10 538.

[27] Y. Yuan and M. Sester, "Comap: A synthetic dataset for collective multi-agent perception of autonomous driving," *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 43, pp. 255–263, 2021.

[28] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Conference on robot learning*. PMLR, 2017, pp. 1–16.

[29] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wießner, "Microscopic traffic simulation using sumo," in *2018 21st international conference on intelligent transportation systems (ITSC)*. IEEE, 2018, pp. 2575–2582.

[30] T.-H. Wang, S. Manivasagam, M. Liang, B. Yang, W. Zeng, and R. Urtasun, "V2vnet: Vehicle-to-vehicle communication for joint perception and prediction," in *European Conference on Computer Vision*. Springer, 2020, pp. 605–621.

[31] E. E. Marvasti, A. Raftari, A. E. Marvasti, Y. P. Fallah, R. Guo, and H. Lu, "Feature sharing and integration for cooperative cognition and perception with volumetric sensors," 2020. [Online]. Available: https://arxiv.org/abs/2011.08317

[32] Y. Liao, J. Xie, and A. Geiger, "KITTI-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d," *arXiv preprint arXiv:2109.13410*, 2021.

[33] J. Castagno and E. Atkins, "Polylidar3d-fast polygon extraction from 3d data," *Sensors*, vol. 20, no. 17, 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/17/4819

[34] C. Choy, W. Dong, and V. Koltun, "Deep global registration," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 2514–2523.