9 January 2025, Preprint v1 · CC-BY 4.0

Qeios

PREPRINT

Research Article

TrafficLoc: Localizing Traffic Surveillance Cameras in 3D Scenes

Yan Xia[1,2], Yunxiang Lu[1], Rui Song[1], Oussema Dhaouadi[1,3], João F. Henriques[4], Daniel Cremers[1,2]

1. Technical University of Munich, Germany; 2. Munich Center for Machine Learning (MCML), Germany; 3. DeepScenario; 4. Visual Geometry Group, University of Oxford, United Kingdom

We tackle the problem of localizing the traffic surveillance cameras in cooperative perception. To overcome the lack of large-scale real-world intersection datasets, we introduce Carla Intersection, a new simulated dataset with 75 urban and rural intersections in Carla. Moreover, we introduce a novel neural network, TrafficLoc, localizing traffic cameras within a 3D reference map. TrafficLoc employs a coarse-to-fine matching pipeline. For image-point cloud feature fusion, we propose a novel Geometry-guided Attention Loss to address cross-modal viewpoint inconsistencies. During coarse matching, we propose an Inter-Intra Contrastive Learning to achieve precise alignment while preserving distinctiveness among local intra-features within image patch-point group pairs. Besides, we introduce Dense Training Alignment with a soft-argmax operator to consider additional features when regressing the final position. Extensive experiments show that our TrafficLoc improves the localization accuracy over the state-of-the-art Image-to-point cloud registration methods by a large margin (up to 86%) on Carla Intersection and generalizes well to real-world data. TrafficLoc also achieves new SOTA performance on KITTI and NuScenes datasets, demonstrating strong localization ability across both in-vehicle and traffic cameras. Our project page is publicly available at https://tum-luk.github.io/projects/trafficloc/.

Yan Xia and Yunxiang Lu equally contributed to this work.

Corresponding author: Daniel Cremers, cremers@tum.de

doi.org/10.32388/97AHJL

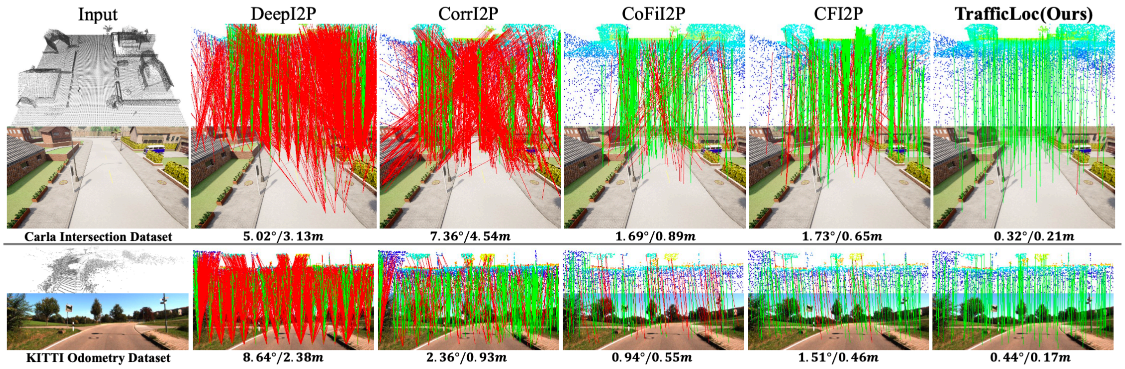


Figure 1. Localization accuracy on the proposed *Carla Intersection* and *KITTI* dataset. The point cloud is projected into a 2D view and shown above the image, with point colors indicating distance. The proposed TrafficLoc achieves better performance, with more correct (green) and fewer incorrect (red) point-to-pixel pairs. The first column presents the input point cloud and input image.

1. Introduction

Traffic surveillance cameras, as affordable and easy-to-install roadside sensors in cooperative perception, offer a broad, global perspective on traffic. Integrating this data with onboard sensors enhances situational awareness, supporting applications like early obstacle detection[1][2] and vehicle localization[3]. Localizing the 6-DoF pose (i.e. position and orientation) of each traffic camera within a 3D map is thus essential for cooperative perception.

Recent advances in 3D sensors have enhanced visual localization[4][5] and LiDAR-camera registration methods[6][7][8][9][10], as LiDAR-acquired point clouds provide accurate and detailed 3D information[11][12]. However, unlike these methods, traffic camera registration focuses on determining **fixed** camera poses within the point cloud scene. The main challenge of localizing the traffic cameras lies in several aspects: 1) Images and 3D reference point clouds are captured at different times and from different viewpoints, making it difficult to obtain the precise initial guess required for traditional registration methods[13]. 2) Directly projecting reference point clouds onto the images can lead to a 'bleeding problem'[7]. 3) Variable focal length cameras are generally used as traffic cameras for easy installation, causing the intrinsic parameters to be frequently changed during operation[14].

To date, only a few methods have been proposed for localizing traffic cameras in the 3D reference scene.[14] employs manual 2D-3D feature matching followed by optimization using distance transform.[3] uses panoramic images for point cloud reconstruction and then aligns traffic camera images with the resulting point cloud.[13] proposes to automate traffic image-to-point cloud registration by generating synthesized views from point clouds to reduce modality gaps. Although these methods achieve promising results, the requirement for manual intervention or panoramic/rendering image acquisition complicates their deployment. It is therefore important to develop the capability to perform traffic camera localization directly.

doi.org/10.32388/97AHJL

However, most of the existing datasets (*i.e.* KITTI[15] and NuScenes[16]) are limited to the in-vehicle cameras, lacking sufficient intersections and traffic cameras. To address this gap, we first introduce *Carla Intersection*, a new intersection dataset using the Carla simulator. *Carla Intersection* provides 75 intersections across 8 worlds, covering both urban and rural landscapes. Furthermore, we propose TrafficLoc, a novel neural network for localizing traffic cameras within a 3D global reference map. TrafficLoc follows a coarse-to-fine localization strategy, beginning with image patch-to-point group matching and refining localization through pixel-to-point matching.

We find that simply applying transformers for cross-modal feature fusion in current image-to-point cloud registration methods[10][9] struggle with limited geometric awareness and weak robustness to viewpoint variations. To address these challenges, we propose a novel Geometry-guided Feature Fusion (GFF) module, a Transformer-based architecture optimized by a novel Geometry-guided Attention Loss (GAL). GAL directs the model to focus on geometric-related regions during feature fusion, significantly improving performance in scenarios with large viewpoint changes. Another observation is that, when registering image patch and point group features in the coarse matching stage, the widely used contrastive learning in previous methods ignores one fact: The pixel features within the same image patch and the point features within the same 3D point group should be differentiated, even though they are inherently similar. To address this issue, we propose a novel Inter-Intra contrastive learning to preserve distinctiveness among local intra-features. Moreover, we find that only aligning features with sparsely paired image patch-point groups potentially neglects additional global features. Therefore, we introduce a dense training alignment strategy to back-propagate the calculated gradients to all spatial locations using a soft-argmax operator. These operations enable one-stage training to directly estimate accurate pixel-point correspondences.

To summarize, the main contributions of this work are:

- We set up a new simulated intersection dataset, *Carla Intersection*, to study the traffic camera localization problem in varying environments. *Carla Intersection* includes 75 intersections across 8 worlds, covering both urban and rural landscapes.
- We propose a novel neural network, *TrafficLoc*, following a coarse-to-fine localization pipeline.
- We propose a novel Geometry-guided Attention Loss to direct the model to focus on the geometric-related regions during cross-modality feature fusion.
- We propose a novel Inter-Intra contrastive learning and a dense matching alignment with a soft-argmax operator to achieve more precise image patch-point group alignment.
- We conduct extensive experiments on the proposed *Carla Intersection*, USTC intersection[13], KITTI[15] and NuScenes[16] datasets, showing the proposed *TrafficLoc* greatly improves over the state-of-the-art methods and generalizes well to real intersection data and in-vehicle camera localization task.

2. Related Work

The research of camera localization in 3D scenes has a long history starting in the early days of computer vision and robotics. We outline the typical solutions here, including visual localization and image-point cloud registration.

Visual localization. Visual localization aims to estimate the 6-DoF camera pose from a query image with respect to a 3D reference map. Structure-based methods[17][18][19][20][21][22][23][24] extract 2D local descriptors[25][26][27][28] from database images to build a 3D map via Structure-from-Motion (SfM), storing descriptors at each 3D point. Given a query image, they match its 2D descriptors with the 3D map to form 2D-3D correspondences, estimating the 6-DoF camera pose using a PnP solver[29] with RANSAC[30]. The image retrieval-based methods[31][32] are used to reduce search space for speed-up. Some Absolute Pose Regression (APR) methods[5][33][34][35] directly predict the 6 DoF camera pose from a single image but reply on uniformly sampled training images and cannot generalize to unknown scenarios. PCLoc[4] and FeatLoc[5] address viewpoint differences between query and database images by synthesizing new views using RGB-D images, enabling more accurate pose estimation. In this work, we aim to estimate the 6-DoF camera pose given a single image in unseen scenarios, especially for traffic surveillance cameras without offline calibrations in the intersections.

Image-to-Point cloud registration. To estimate the relative pose between an image and a point cloud, methods like 2D3D-MatchNet[36] and LCD[37] use deep networks to learn descriptors jointly from 2D image patches and 3D point cloud patches. 3DTNet[38] learns 3D local descriptors by integrating 2D and 3D local patches, treating 2D features as auxiliary information. Cattaneo *et al.*[39] establish a shared global feature space between 2D images and 3D point clouds using a teacher-student model. Recent VXP[40] improves the retrieval performance by enforcing local similarities in a self-supervised manner. DeepI2P[41] reformulates cross-matching as a classification task, identifying if a projected point lies within an image frustum. FreeReg[42] employs pretrained diffusion models and monocular depth estimators to unify image and point cloud features, enabling single-modality matching without training. EP2P-Loc[43] performs 2D patch classification for each 3D point in retrieved sub-maps, using positional encoding to determine precise 2D-3D correspondences. CorrI2P[8] directly matches dense per-pixel/per-point features in overlapping areas to establish I2P correspondences, while CoFiI2P[9] and CFI2P[10] employ a coarse-to-fine strategy, integrating high-level correspondences into low-level matching to filter mismatches. Recent VP2P[44] proposes an end-to-end Image-to-Point Cloud registration network with a differentiable PnP solver. However, all methods are limited to in-vehicle camera images. In this work, our approach generalizes well to various viewpoints, including car and traffic perspectives.

3. Problem Statement

Given a query RGB image $\mathbf{I} \in \mathbb{R}^{H \times W \times 3}$ with a resolution of $H \times W$ and a reference 3D point cloud $\mathbf{P} \in \mathbb{R}^{N \times 3}$, where N is the number of points, our goal is to estimate the 6-DoF relative transformation $\mathbf{T} = [\mathbf{R}|\mathbf{t}]$ between the image I and point cloud P, including rotation matrix $\mathbf{R} \in \mathbf{SO}(3)$ and translation vector $\mathbf{t} \in \mathbb{R}^3$, as well as the camera intrinsic matrix $\mathbf{K} \in \mathbb{R}^{3 \times 3}$.

According to the camera projection geometry, our localization problem can be formulated as follows:

$$\mathbf{T}', \mathbf{K}' = \underset{\mathbf{T},\mathbf{K}}{\arg\min} \sum_{(u_i, p_i) \in C^*} \|\mathbf{u}_i - \mathcal{F}(\mathbf{K}\mathbf{T}\mathbf{p}_i)\|, \tag{1}$$

where $\mathbf{u}_i$ is 2D pixel coordinates, $\mathbf{p}_i$ is 3D point coordinates, C^* is the set of ground-truth 2D-3D correspondences and $\|\cdot\|$ means the Euclidean distance. $\mathbf{p}_i$ is transformed into homogeneous coordinates implicitly when calculating $\mathbf{T}\mathbf{p}_i$. Function $\mathcal{F}(\cdot)$ is used for planar projection:

$$\mathcal{F}([x, y, z]^\top) = [u', v']^\top = [x/z, y/z]^\top. \tag{2}$$

4. Methodology

Fig. 2 shows our TrafficLoc architecture. Given a query image and a reference 3D point cloud, we first extract 2D patch features and 3D group features respectively, describied in Section 4.1. Next, we fuse the 2D patch and 3D group feature via a carefully designed GFF module, which will be explained in Section 4.2. Section 4.3 describes the coarse matching stage, establishing 2D patch-3D group correspondence. Furthermore, we match each 3D group center feature and the patch features generated from coarse matching in fine matching (Section. 4.1). Lastly, the RANSAC+EPnP module in Section. 4.5 exploits the point-pixel correspondences set to optimize the relative transformation.

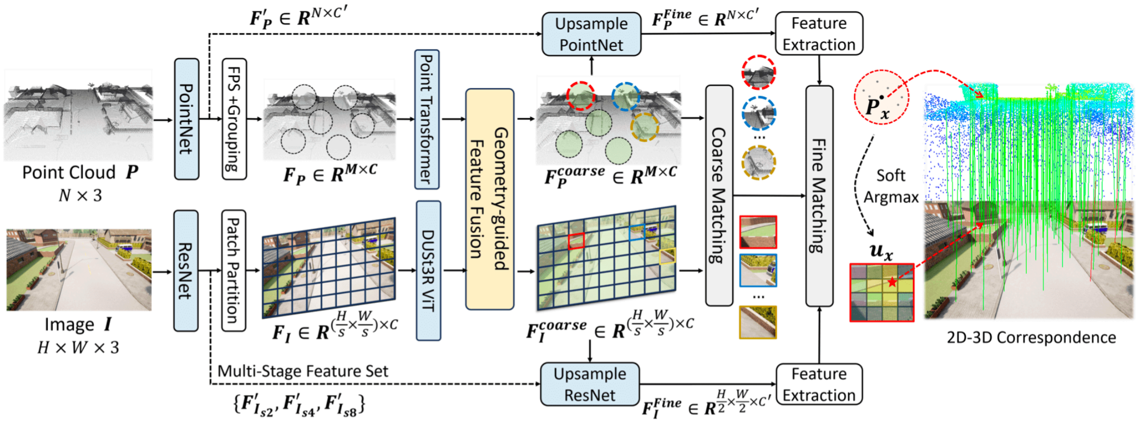

$F'_P \in R^{N\times C'}$
Upsample PointNet
$F_P^{Fine} \in R^{N\times C'}$
Feature Extraction
Point Cloud P
$N \times 3$
PointNet
FPS +Grouping
$F_P \in R^{M\times C}$
Point Transformer
Geometry-guided Feature Fusion
$F_P^{coarse} \in R^{M\times C}$
Coarse Matching
Fine Matching
P_x
Soft Argmax
u_x
2D-3D Correspondence
Image I
$H \times W \times 3$
ResNet
Patch Partition
$F_I \in R^{(\frac{H}{s}\times\frac{W}{s})\times C}$
DUSt3R ViT
$F_I^{coarse} \in R^{(\frac{H}{s}\times\frac{W}{s})\times C}$
Multi-Stage Feature Set
$\{F'_{I_{s2}}, F'_{I_{s4}}, F'_{I_{s8}}\}$
Upsample ResNet
$F_I^{Fine} \in R^{\frac{H}{2}\times\frac{W}{2}\times C'}$
Feature Extraction

Figure 2. Pipeline of our proposed TrafficLoc for relocalization. Taking a pair of 3D point cloud and a 2D image as input, TrafficLoc first performs feature extraction to obtain features in point group level and image patch level. The Geometry-guided Feature Fusion (GFF) module strengthens the feature and then match them based on similarity rule. Fine features are extracted based on the coarse matching results and fine matching is performed between the point group center and the extracted image window with a soft-argmax operation. The final generated 2D-3D correspondences are utilized to optimize the camera pose with RANSAC+EPnP[29][30] algorithm.

4.1. *Feature Extraction*

Following[43], we explore a dual branch to extract image and point cloud features using Transformer-based encoders.

2D patch descriptors. Following CFI2P[10], we first utilize ResNet-18[45] to extract multi-level features of image I and then use the feature at the coarsest resolution to generate 2D patch descriptors $\mathbf{F}_{\text{patch}} \in \mathbb{R}^{HW/s^2 \times C}$, where s is the resolution of non-overlapping patches and HW/s^2 is the number of image patches. To further enhance the spatial relationships within these image patches, we leverage the pre-trained ViT encoder in DUSt3R[46] to obtain 128-dim 2D patch descriptors owing to its strong capability in regressing 3D coordinates.

3D point group descriptors. We first utilize a standard PointNet[47] to extract point-wise features $\mathbf{F}'_{\mathbf{P}} \in \mathbb{R}^{N\times C'}$ for each point. Then, we conduct Farthest Point Sampling (FPS) to generate M super-points $\mathbf{Q} = \{\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_M\}$ and assign every point in $\mathbf{P}$ to its nearest center in $\mathbf{Q}$, formulating M point groups $G^P = \{G_1^P, G_2^P, \ldots, G_M^P\}$ and their associated

doi.org/10.32388/97AHJL

feature sets $G^{F'} = \{G_1^{F'}, G_2^{F'}, \ldots, G_M^{F'}\}$. Following CFI2P[10], we then use the Point Transformer[48] to enhance local geometry features and incorporate global contextual relationships for each point group.

4.2. Geometry-guided Feature Fusion

Although the Image-to-Point cloud registration approaches[9][10] achieved notable fusion success using a Transformer-based architecture, the inherent differences between different modalities and viewpoints are not well-explored. To address this issue, we introduce a novel Geometry-guided Feature Fusion (GFF) module to enhance the network's robustness to viewpoint variations during cross-modal feature fusion. Specifically, we design a Fusion Transformer architecture guided by a novel geometry-guided attention loss (GAL). Fig. 3 shows the detailed architecture of the GFF module.

Fusion Transformer. Similar to[10], we adopt a Transformer-based architecture with self-attention and cross-attention layers for cross-modal feature fusion. Given the image feature $\mathbf{F}_I$ and point cloud feature $\mathbf{F}_P$, we begin by adding sinusoidal positional embeddings to retain the spatial information within both modalities. In the self-attention module, a transformer encoder enhances features in each modality individually using standard scalar dot-product attention. The cross-attention layer is designed to fuse image and point cloud features by applying the attention mechanism across modalities. This design allows for the exchange of geometric and textural information between image and point cloud features, enabling a richer, modality-aware feature representation. More details are in Supp.

Geometry-guided Attention Loss (GAL). We find that directly applying Transformers for cross-modal feature fusion suffers from limited geometric awareness and weak robustness to viewpoint variations. We thus propose a novel geometry-guided attention loss that supervises cross-modal attention map during training based on geometric alignment, encouraging features to focus on their geometrically corresponding parts, as shown in Fig. 3 (right).

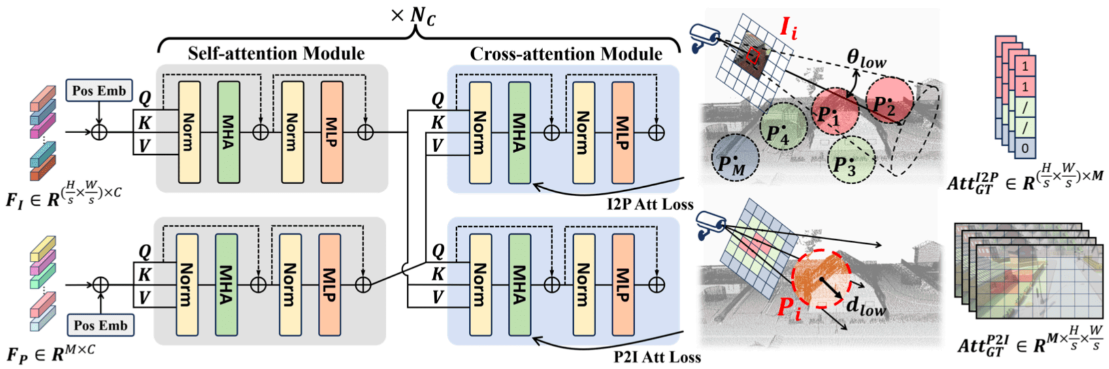


Figure 3. The pipeline of Geometry-guided Feature Fusion (GFF) module. GFF first use N_c layers of self and cross-attention module to enhance the feature across different modalities (left). The Geometry-guided Attention Loss is applied to the cross-attention map of the last fusion layer based on camera projection geometry (right).

Inspired by[49], we apply supervision to the cross-attention layer at the final stage of the Fusion Transformer, leveraging camera projection geometry as guidance. In **I2P** attention, we encourage the network to focus on relevant 3D point groups for each 2D patch feature I_i. This is achieved by penalizing attention values that have high values outside the target region and encouraging high attention within the desired area. We implement this through a Binary Cross Entropy (BCE) loss on the raw cross-attention map ATT_{I2P}:

$$L_{I2P}(i,j) = \mathrm{BCE}(\sigma(ATT_{I2P}(i,j)), 1_{I2P}(i,j)) \tag{3}$$
$$ATT_{I2P} = \mathbf{F}_I \mathbf{W}_Q (\mathbf{F}_P \mathbf{W}_K)^\top \in \mathbb{R}^{HW/s^2 \times M}$$

where σ is a sigmoid function, and $1_{I2P}(i,j)$ is a special indicator function defined as:

$$1_{I2P}(i,j) = \begin{cases} 1, & \text{if } \mathrm{Rad}(i,j) < \theta_{low} \\ 0, & \text{if } \mathrm{Rad}(i,j) > \theta_{up} \\ -1, & \text{otherwise} \end{cases} \tag{4}$$

The angular radius $\mathrm{Rad}(i,j)$ is given by $\mathrm{Rad}(i,j) = \arccos(\frac{\mathbf{OI}_i \cdot \mathbf{OP}_j}{\|\mathbf{OI}_i\| \|\mathbf{OP}_j\|})$. Here, $1_{I2P}(i,j)$ assigns a value of 1 when the angle between the camera ray OI_i (formed by the patch I_i and the camera center O) and the line to the 3D point center $\mathbf{P}_j$ is below a threshold θ_{low}. If this angle exceeds an upper threshold θ_{up}, the value is set to 0, indicating that the point is outside the target region. Points with angles between these thresholds are assigned a value of -1, allowing the network to flexibly learn the relationship between attention in these intermediate cases. Similarly, in **P2I** attention, we encourage each point group to focus on image patches within its target area of influence:

$$L_{P2I}(i,j) = \mathrm{BCE}(\sigma(ATT_{P2I}(i,j)), 1_{P2I}(i,j))$$

$$ATT_{P2I} = \mathbf{F}_P \mathbf{W}_Q (\mathbf{F}_I \mathbf{W}_K)^\top \in \mathbb{R}^{M \times HW/s^2}$$

$$1_{P2I}(i,j) = \begin{cases} 1, & \text{if } \mathrm{Dist}(P_i, OI_j) < \mathbf{d}_{low} \\ 0, & \text{if } \mathrm{Dist}(P_i, OI_j) > \mathbf{d}_{up} \\ -1, & \text{otherwise} \end{cases} \tag{5}$$

where Dist (P_i, OI_j) represents the distance from point center P_i to the camera ray OI_j (formed by the camera center O and the patch I_j) in the 3D space. Since all point groups share a similar receptive field during feature extraction, the distance threshold is set the same for each. This ensures point groups farther from the camera focus on a smaller target area within the image, aligning with the natural physical principle that closer objects appear larger while distant ones appear smaller.

The final geometry-guided attention loss is:

$$L_{Att} = \sum_{i=1}^{HW/s^2} \sum_{j=1}^{M} L_{I2P}(i,j) + L_{P2I}(j,i) \tag{6}$$

Note that the relative GT transformation matrix T_{GT} is required only during training. Experiments in Section 5.4 shows the effectiveness of GAL, leading to significant improvements in scenarios with large viewpoint changes.

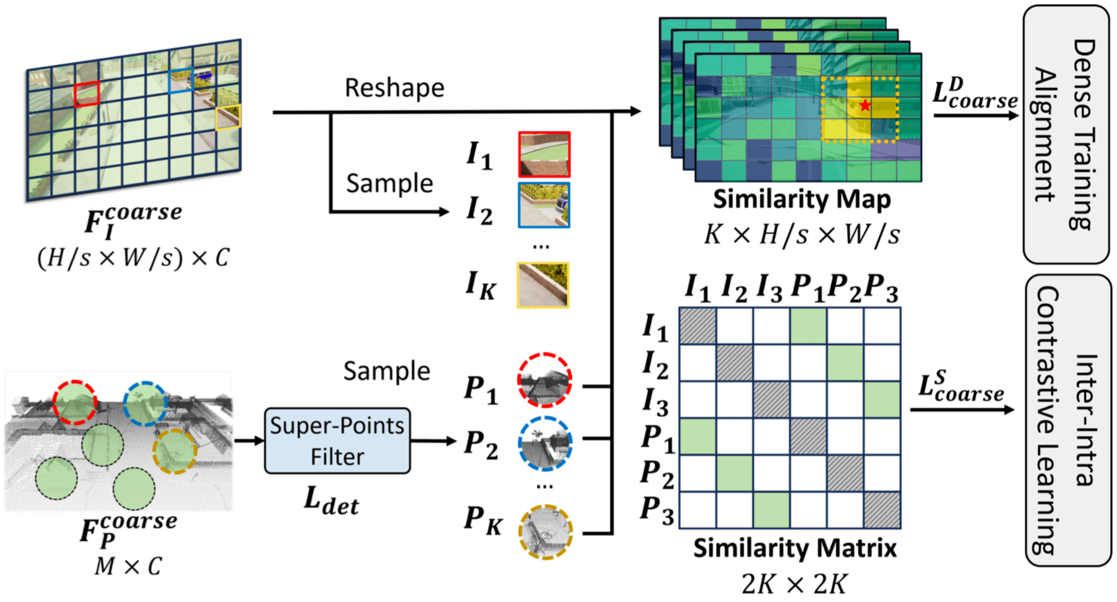


Figure 4. Coarse matching mechanism of TrafficLoc. The positive feature pairs are generated based on ground-truth transformation matrix. The coarse image feature $\mathbf{F}_I^{coarse}$ is reshaped to compute its similarity map with each coarse point feature.

4.3. Coarse Matching

Here we aim to match the point group with the image patch at the coarse level given the fused 2D patch features $\{\mathbf{F}_I^{coarse}\}_{i=1}^{N_p}$ and 3D point group features $\{\mathbf{F}_P^{coarse}\}_{i=1}^{N_g}$. Since a monocular camera can only capture a part of the 3D point cloud scene due to the limited field of view (FoV), we first apply a simple super-points filter with binary classification MLP head to predict super-points in or beyond the frustum. For each predicted in-frustum point group P_i, we estimate its corresponding coarse pixel u_i based on feature similarity to get the predicted coarse correspondence set $\hat{C}_{coarse} = \{(P_i, u_i)\}$, we use cosine similarity $s(\cdot,\cdot)$ to denote the similarity between two features:

$$s(P_i, I_j) = \frac{< \mathbf{F}_{P_i}^{coarse}, \mathbf{F}_{I_j}^{coarse} >}{\|\mathbf{F}_{P_i}^{coarse}\|\|\mathbf{F}_{I_j}^{coarse}\|} \tag{7}$$

Inter-Intra Contrastive Learning. To establish pixel-to-point correspondences, we find that the pixel features within the same image patch and the point features within the same 3D point group should be differentiated, even though they are inherently similar. Inspired by this, we propose a novel Inter-Intra contrastive learning objective to address the limitations of the widely used contrastive learning in[9][10]. The objective is not only to bring the features of 3D point groups and their corresponding image patches closer together, but also to ensure that the features within the same image patches and 3D point groups remain as differentiated as possible. This balance enhances more precise alignment while preserving distinctiveness among local intra-features. The loss details are in Sec. 4.6, and experimental results in Sec. 5.4 show its efficacy.

Dense Training Alignment. Moreover, we find that the Inter-Intra contrastive learning can only align features with sparsely paired patch-point groups, neglecting additional global features. Inspired by[50], we propose a dense training alignment strategy to back-propagate the calculated gradients to all spatial locations using a soft-argmax operator.

Specifically, for each sampled point group feature P_x, we compute its similarity map $S_x = \mathbf{F}_{P_i}^{coarse} \mathbf{F}_{I}^{coarse\,\mathrm{T}} \in \mathbb{R}^{1\times H/s \times W/s}$ with the target image feature $\mathbf{F}_{I}^{coarse}$, as shown in Figure 4. Then, we take soft-argmax over the similarity map to compute the predicted pixel position $\hat{u}_x = \mathrm{SoftArgmax}(S_x)$. We penalize the distance between the predicted position $\hat{u}_x$ and the target position u_x with a L2 norm loss (See Sec. 4.6 for details).

4.4. Fine Matching

After the coarse matching, we aim to generate point-to-pixel pairs in fine matching.

We first generate the image feature $\mathbf{F}_{I}^{Fine} \in \mathbb{R}^{H/2\times W/2\times C'}$ and the point feature $\mathbf{F}_{P}^{Fine} \in \mathbb{R}^{N\times C'}$ in fine-resolution via applying the two upsample networks, ResNet[45] and PointNet[47], for image and point cloud respectively. We then extract the local feature from the fine-resolution image patch, $\mathbf{F}_{I_x}^{Fine} \in \mathbb{R}^{w\times w\times C'}$, centered on the predicted coarse pixel coordinate $\hat{u}_x$. Additionally, we extract the corresponding feature $\mathbf{F}_{P_x}^{Fine} \in \mathbb{R}^{1\times C'}$ of the center point P_x within the point group. The fine matching process is defined as:

$$\begin{aligned} \hat{S}_x^{Fine} &= \mathbf{F}_{P_x}^{Fine} \mathbf{F}_{I_x}^{Fine\,\mathrm{T}} \in \mathbb{R}^{1\times w\times w} \\ \hat{u}_x^{Fine} &= \mathrm{SoftArgmax}(\hat{S}_x^{Fine}) \end{aligned} \tag{8}$$

where $\hat{S}_x^{Fine}$ is the fine similarity map between the point center and extracted image patch, and $\hat{u}_x^{Fine}$ is the final predicted 2D pixel corresponding to 3D point P_x.

4.5. Pose Estimation

Following[9], we use the RANSAC+EPnP[29][30] algorithm to filter out incorrect pixel-point pairs and estimate the relative pose of the camera based on the set of predicted pixel-point correspondence after the fine matching.

4.6. Loss Function

Our training loss includes three components:

In-frustum Detection Loss. We use a standard binary cross-entropy (BCE) loss L_{det} to supervise the in-frustum super-points classification.

Coarse Matching Loss. In coarse matching, we first sample K pairs from the ground-truth 2D-3D corresponding set $C_{\mathrm{coarse}}^{*} = \{(P_x, I_x) | I_x = \mathcal{F}(T_{GT} P_x)\}$, where T_{GT} is the ground-truth transformation matrix from point cloud coordinate system to image frustum coordinate system, and $\mathcal{F}$ denotes the mapping function that convert points from camera frustum to image patch position. We determine the negative pairs by checking whether the distance between the location of I_x and the projection of P_x on the image is larger than a threshold r or not. The process is the same within each individual modality.

The Inter-Intra contrastive learning loss is defined as:

$$L_{\text{coarse}}^{S} = \log[1 + \sum_{j} \exp^{\alpha_p (1 - s_p^j + m_p)} \sum_{k} \exp^{\alpha_n (s_n^k - m_n)}] \tag{9}$$

where α_p and α_n are the adaptive weighting factors for positive and negative pairs, respectively:

$$\begin{aligned} \alpha_p &= \gamma \cdot \max(0, 1 + m_p - s_p^j) \\ \alpha_n &= \gamma \cdot \max(0, s_n^k - m_n) \end{aligned} \tag{10}$$

in which γ is the scale factor and α_p and α_n are positive and negative margin for better similarity separation.

In addition, the loss in dense training alignment is defined as the L2 norm distance between the predicted position and the target position u_x:

$$L_{\text{coarse}}^{D} = \sum_{x}^{K} \|\hat{u}_x - u_x\|_2, \tag{11}$$

and we combine the Inter-Intra contrastive learning loss to form our final coarse matching loss $L_{\text{coarse}} = L_{\text{coarse}}^{S} + L_{\text{coarse}}^{D}$.

Fine Matching Loss. Since the fine image patch has a small region, we utilize Cross Entropy (CE) loss to sparsely supervise the fine matching process and apply dense L2 norm loss similar to L_{coarse}^{D}:

$$\begin{gathered} L_{fine}^{S} = -\frac{1}{K} \sum_{x=1}^{K} \sum_{c=1}^{w^2} S_{x,c}^{Fine} \log(softmax(\hat{S}_{x,c}^{Fine})) \\ L_{fine}^{D} = \sum_{x}^{K} \|\hat{u}_x^{Fine} - u_x^{Fine}\|_2 \end{gathered} \tag{12}$$

where $S_{x,c}^{Fine}$ equals to 1 when the 3D point P_x is corresponded with the c_{th} pixel in the flatten image patch, else 0. To avoid overfitting, we randomly shift the centered position of the extracted fine patch by up to $\pm w/2$ pixels, preventing the ground-truth pixel from always being at the center of the patch.

We combine the sparse and dense loss to form our final fine matching loss $L_{fine} = L_{fine}^{S} + L_{fine}^{D}$. Overall, our loss function is:

$$L = \lambda_1 L_{det} + \lambda_2 L_{coarse} + \lambda_3 L_{fine} \tag{13}$$

where $\lambda_1, \lambda_2, \lambda_3$ regulate the losses' contributions.

5. Experiments

5.1. Proposed Carla Intersection Dataset

The proposed *Carla Intersection Dataset* comprises 75 intersections across 8 worlds within the Carla[51] simulation environment, encompassing urban and rural landscapes. We use on-board LiDAR sensor to capture point cloud scans, which are then accumulated and downsampled to get the 3D point cloud of the intersection. For each intersection, we captured 768 training images and 288 testing images with known 6-DoF pose at a resolution of 1920x1080 pixel and a horizontal field of view (FOV) of 90°. In consideration of real-world traffic surveillance camera installations, our image collection spans heights from 6 to 8 meters, with camera pitch angles from 15 to 30 degrees. This setup reflects typical positioning to capture optimal traffic views under varied monitoring conditions. More details and visualization of our dataset are in the Supplementary Materials.

To access the model's generalization capability, we trained on 67 intersections from worlds $Town01$ to $Town07$, and tested on one unseen intersection from each of the 8 worlds. We divided the testing data into three sets:

- $\mathbf{Test}_{T1-T7}$ contains 7 unseen intersections from world $Town01$ to $Town07$, with 288 testing images each, matching the training pitch angles (15 or 30 degree).
- $\mathbf{Test}_{T1-T7hard}$ contains 7 same intersection scenes as in $\mathbf{Test}_{T1-T7}$, with images at pitch angles of 20 or 25 degrees, to evaluate robustness against viewpoint variations.
- $\mathbf{Test}_{T10}$ contains 1 unseen intersection scene from unseen world $Town10$. This split sets a high standard for evaluating the model's generalization ability, assessing its performance in unseen urban styles and intersections.

5.2. Experimental Setup

Datasets. We conduct experiments on the proposed Carla Intersection Dataset. In addition, we evaluate our TrafficLoc on one real USTC intersection dataset[13], and two in-vehicle camera benchmarks, KITTI Odometry[15] and Nuscenes[16]. For fair comparisons, we use the same training and evaluation pairs of image and point cloud data on KITTI and Nuscenes following previous image-point cloud registration methods[8][41].

Implementation Details. The training details and others are in the Supplementary Materials.

Evaluation Metrics. Following previous works[9][8][41], we evaluate the localization performance with relative rotation error (RRE), relative translation error (RTE) and registration recall (RR). RRE and RTE are defined as:

$$\mathrm{RRE} = \sum_{i=1}^{3} |\mathbf{r}(i)|, \quad \mathrm{RTE} = \|\mathbf{t}_{gt} - \mathbf{t}_{pred}\| \tag{14}$$

where $\mathbf{r}$ is the Euler angle vector of $\mathbf{R}_{gt}^{-1}\mathbf{R}_{pred}$, $\mathbf{R}_{gt}$ and $\mathbf{t}_{gt}$ are the ground-truth rotation and translation matrix, $\mathbf{R}_{pred}$ and $\mathbf{t}_{pred}$ represent the estimated rotation and translation matrix. RR denotes the fraction of successful registrations among the test dataset. A registration is considered as successful when the RRE is smaller than τ_r and the RTE is smaller than τ_t, i.e. $10^\circ/5m$.

5.3. Evaluation Results

We first evaluate TrafficLoc on our *Carla Intersection Dataset* and compare with other baseline methods. Table 1summarizes the results. Our method outperforms all baseline methods by a large margin in all three test splits, which indicates that TrafficLoc is robust to viwepoint changes and has great generalization ability on *unseen* traffic scenarios. Specifically, the RRE reduces **85%**, **66%**, **86%** and the RTE reduces **82%**, **78%**, **64%** compared to the previous state-of-the-art CoFiI2P[9] in three test splits respectively. Even in entirely unseen city-style environments and unseen scenes ($\mathbf{Test}_{T10}$), our model maintains robust localization capability, while other baseline methods fail. Additionally, our model achieves high accuracy while maintaining efficient inference time. When intrinsic parameters K are unknown, we use intrinsics predicted by DUSt3R[46] to initialize the intrinsics for RANSAC+EPnP[29][30], which slightly decreases performance and inference speed but still yields strong results. The result of VP2P[44] on *Carla Intersection Dataset* and its RR on Nuscenes are unavailable since the training code is not provided.

	$\mathbf{Test}_{T1-T7}$		$\mathbf{Test}_{T1-T7hard}$		$\mathbf{Test}_{T10}$		**KITTI Odometry**			**Nuscenes**			**Runtime**
	RRE(°)	**RTE** (m)	**RRE(°)**	**RTE** (m)	**RRE(°)**	**RTE** (m)	**RRE(°)**	**RTE** (m)	**RR(%)**	**RRE(°)**	**RTE** (m)	**RR(%)**	t**(sec)**
DeepI2P-Cls[41]	9.02	6.31	9.10	6.12	18.30	11.46	5.88	1.13	80.18	7.37	2.22	62.67	**0.23**
DeepI2P-2D[41]	20.31	7.01	33.43	7.93	49.93	17.09	3.87	1.42	74.50	3.02	1.95	92.53	8.78
DeepI2P-3D[41]	17.97	7.29	34.86	9.49	43.11	17.41	6.08	1.21	38.34	7.06	1.73	18.82	18.62
CorrI2P[8]	20.08	12.63	27.95	13.97	32.23	14.45	2.72	0.90	92.19	2.31	1.7	93.87	1.33
VP2P[44]	/	/	/	/	/	/	2.39	0.59	95.07	2.15	0.89	/	0.76
CFI2P[10]	4.46	1.92	8.56	2.48	10.81	7.15	1.38	0.54	99.44	1.47	1.09	99.23	0.33
CoFiI2P[9]	4.24	2.82	7.87	5.34	17.78	7.43	1.14	0.29	**100.00**	1.48	0.87	98.67	0.64
Ours with K_{Pred}	2.04	1.72	4.56	2.19	4.61	4.80	/	/	/	/	/	/	1.74
Ours with K_{GT}	**0.66**	**0.51**	**2.64**	**1.13**	**2.53**	**2.69**	**0.87**	**0.19**	**100.00**	**1.38**	**0.78**	**99.45**	0.85

Table 1. Quantitative localization results on the proposed *Carla Intersection*, KITTI[15] and *Nuscenes*[16] datasets. We report median RRE and median RTE for *Carla Intersection* and mean RRE and mean RTE for KITTI and *Nuscenes* following previous Image-to-Point cloud registration methods[9][8][41]. Our model achieves the best performance on all datasets, especially on unseen scenarios.

Besides *Carla Intersection Dataset*, we evaluate our TrafficLoc on KITTI and Nuscenes benchmarks. TrafficLoc achieves the best performance on both datasets, achieving **34%** RTE improvement compared to the previous state-of-the-art CoFiI2P[9] and $< 1^\circ$ RRE first-time on KITTI, indicating its strong ability also on in-vehicle view cases.

To test the generalizability of our TrafficLoc, we evaluate it on real-world intersection from the USTC dataset[13], using a model trained on the *Carla Intersection*. Note that the test intersection is totally unseen and the traffic camera is uncalibrated. Figure 5 shows the qualitative localization result. Since ground-truth data is unavailable, we project the point cloud onto the image plane with predicted transformation matrix $\mathbf{T} = [\mathbf{R}|\mathbf{t}]$ and intrinsic parameters K. The projection image shows a clear overlap with the input image, validating the accuracy of our localization.

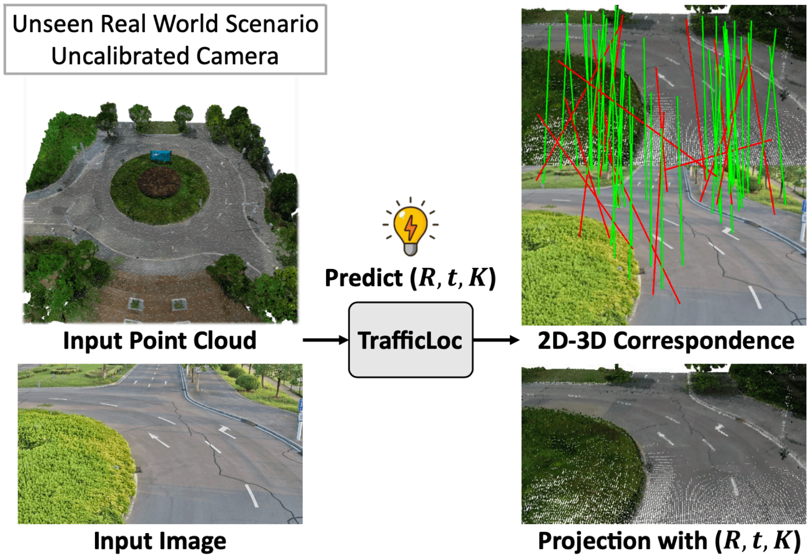


Figure 5. Localization performance of our TrafficLoc on the USTC intersection dataset[13]. Note that the model is trained on the Carla Intersection dataset.

5.4. *Ablation Study*

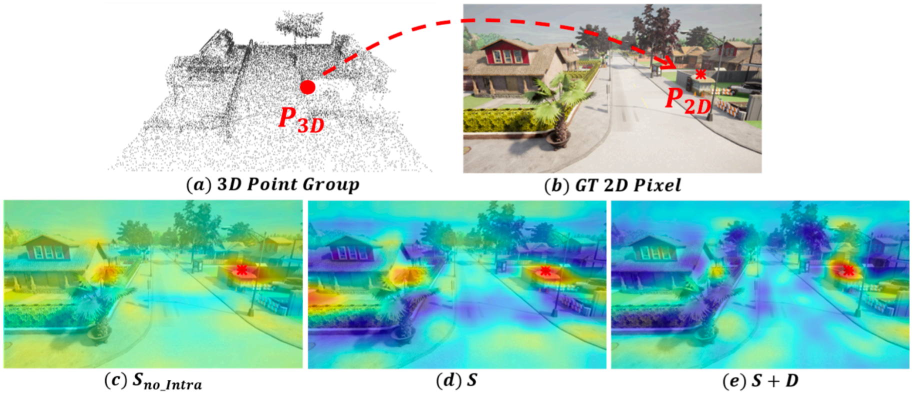
We evaluate the effectiveness of different proposed components in our TrafficLoc here.

Matching mechanism. Table 2 presents the quantitative results of different matching loss functions across all three test splits of the *Carla Intersection Dataset*. When only using normal contrastive loss S_{no_intra} in coarse matching, the model exhibits relatively high error across all test splits, particularly on dataset $Test_{T10}$ with an unseen world style. Both the Inter-Intra contrastive learning loss L^{S}_{coarse} and dense training alignment L^{D}_{coarse} significantly improve the performance, and the fine matching loss further enhances the model's ability to handle seen world styles. The model equipped with the Geometry-guided Attention Loss L_{att} outperforms its counterpart without GAL on all metrics (see the last two rows), showing a particularly notable improvement of **20.4%** in the RTE metric on dataset $\mathbf{Test}_{T1-T7hard}$, which highlights the robustness of GAL to viewpoint variations.

	CM	FM	L_{Att}	**Test**$_{T1-T7}$		**Test**$_{T1-T7hard}$		**Test**$_{T10}$	
				RRE(°)	**RTE(m)**	**RRE(°)**	**RTE(m)**	**RRE()**	**RTE(m)**
Baseline	S_{no_intra}	/	/	1.53	0.82	3.79	1.98	7.35	7.47
	S	/	/	1.27	0.74	3.72	1.87	4.09	3.26
	S	/	✓	0.95	0.64	3.12	1.46	3.03	2.86
	S+D	/	/	1.01	0.62	3.23	1.65	2.99	2.80
	S+D	✓	/	0.84	0.62	3.17	1.42	2.98	2.83
Ours	S+D	✓	✓	**0.66**	**0.51**	**2.64**	**1.13**	**2.53**	**2.69**

Table 2. Ablation Study on loss function and model design. We report median RRE and median RTE results on all three test splits of *Carla Intersection Dataset*. CM denotes Coarse Matching and FM denotes Fine Matching. S means using Inter-Intra contrastive learning loss L^S_{coarse} and D means using dense training alignment loss L^D_{coarse}. S_{no_intra} means using normal contrastive loss instead of L^S_{coarse}. L_{att} represents applying the Geometry-guided Attention Loss (GAL).

As displayed in Figure 6, when using normal contrastive loss without intra mechanism, the similarity map exhibits generally high values throughout. After incorporating the Inter-Intra contrastive learning loss, the distinction within the similarity map increases, indicating a more pronounced distributional difference among features within the same modality. However, due to the sparse supervision during training, multiple peaks (red regions) remain. With the addition of Dense Loss, which provides global supervision across the entire image during training, the similarity map displays a single peak region, demonstrating strong robustness in matching.



(a) **3D Point Group**
(b) ***GT* 2*D Pixel***
(c) S_{no_Intra}
(d) S
(e) $S + D$

Figure 6. Visualization result of using different loss function. (a), (b) denote the point group center P_{3D} and its corresponding pixel P_{2D}. (c), (d), (e) show the similarity map between point group and image feature. Blue means low similarity and red means high.

Attention map visualization. Figure 7 displays the cross-attention map between two modalities. With the use of L_{att}, the P2I attention map of point group P_{3D} tends to concentrate more on the image region where the point group is projected, while the I2P attention map for patch I_{2D} assigns greater weights to the area traversed by the camera ray of this patch. Both observations highlight the geometry-awareness of the proposed geometry-guided attention loss.

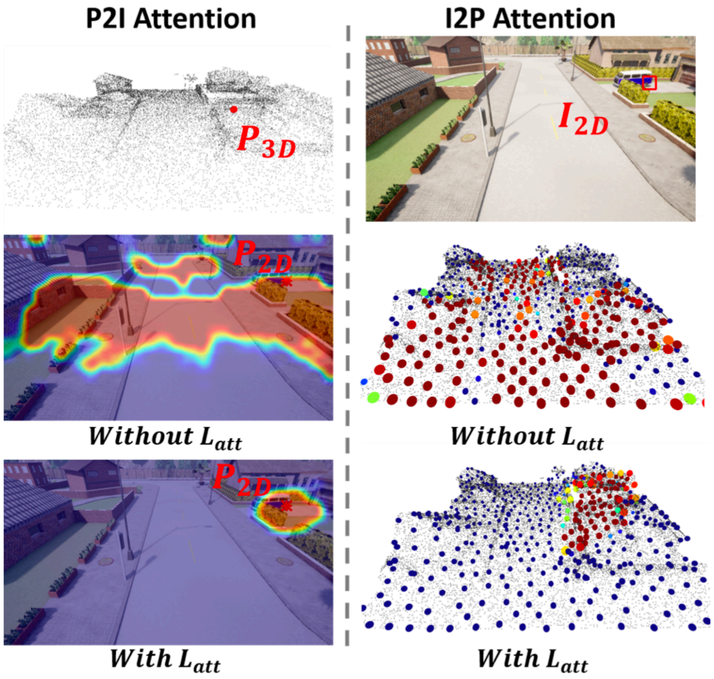


Figure 7. Visualization result of P2I and I2P attention map when using Geometry-guided Attention Loss L_{att} or not. Red color indicates high attention value and blue means low value.

Geometry-guided Attention Loss. The ablation results for the Geometry-guided Attention Loss (GAL) are summarized in Table 3. We conducted experiments on the *Carla Intersection Dataset* with GAL using different threshold parameters and applying GAL across different layers of the Geometry-guided Feature Fusion (GFF) module.

	$\theta_{low}(^\circ)$	$\theta_{up}(^\circ)$	$d_{low}(m)$	$d_{up}(m)$	Layer	$\mathbf{Test}_{T1-T7}$		$\mathbf{Test}_{T1-T7hard}$		$\mathbf{Test}_{T10}$	
						RRE(°)	RTE (m)	RRE(°)	RTE (m)	RRE(°)	RTE (m)
Baseline	/	/	/	/	/	0.84	0.62	3.17	1.42	2.98	2.83
	10	10	3	5	Last	1.24	0.80	3.49	1.53	6.07	7.45
	10	20	3	3	Last	1.27	0.83	3.05	1.46	3.55	2.95
	20	30	3	5	Last	0.91	0.59	2.71	1.27	3.05	2.78
	10	20	5	7	Last	0.85	0.55	2.63	1.15	3.08	2.75
	10	20	3	5	First	1.00	0.59	2.68	1.14	4.30	3.23
	10	20	3	5	All	1.02	0.62	3.01	1.19	3.45	3.33
Ours	10	20	3	5	Last	**0.66**	**0.51**	**2.64**	**1.13**	**2.53**	**2.69**

Table 3. Ablation Study on Geometry-guided Attention Loss (GAL). θ_{low} and θ_{up} denote the angular threshold for I2P attention, while d_{low} and d_{up} represent the distance threshold for **P2I** attention. "Layer" specifies the fusion layer within the Geometry-guided Feature Fusion (GFF) module where GAL is applied.

When the lower and upper threshold are set to the same value (see the second and third row), the model performs worse than not applying GAL, which highlights the importance of defining a tolerant region that enables the network to flexibly learn attention relationships for intermediate cases between the lower and upper thresholds. With thresholds θ_{low}, θ_{up}, d_{low} and d_{up} set to 10°, 20°, $3\ m$ and $5\ m$, our model consistently outperforms the baseline without GAL across all metrics. Moreover, we observed that applying GAL to either the first layer or all layers of the GFF module yields worse localization results compared to applying it only to the last layer. This is mostly because such configurations constrain the network's ability to capture global features during the early stages (or initial layers) of multimodal feature fusion.

Feature extraction backbone. Table 4 illustrates the results under different image and point cloud feature extraction backbone. Our model performs best when using DUSt3R[46] and Point Transformer[48] as backbones, benefiting from DUSt3R's strong generalization ability. Even with a frozen DUSt3R, the model achieves comparable performance. In contrast, when using ResNet[45] or PiMAE[52], the model's performance declines due to the lack of attentive feature aggregation during the feature extraction stage. When utilizing PiMAE, we load the pretrained weights of its point encoder.

	Img Enc	PC Enc	RRE(°)	RTE(m)
Baseline	ResNet[45]	PiMAE[52]	1.25	0.87
	ResNet[45]	PT[48]	0.85	0.58
	DUSt3R[46]	PiMAE[52]	1.03	0.75
	DUSt3R*[46]	PT[48]	0.77	0.59
Ours	DUSt3R[46]	PT[48]	**0.66**	**0.51**

Table 4. Ablation Study on Model Backbone. We report median RRE and median RTE resuls on test split $\mathbf{Test}_{T1-T7}$. DUSt3R* means using frozen DUSt3R backbone during training.

Effectiveness of DTA and GAL. To verify the effectiveness of the Dense Training Alignment (DTA) and Geometry-guided Attention Loss (GAL), we conducted further experiments on the KITTI Odometry dataset[15] using the existing state-of-the-art network CoFiI2P[9] as the base model. Since CoFiI2P also adopts a coarse-to-fine matching approach with a similar transformer-based feature fusion module, but only employs a standard contrastive circle loss, DTA and GAL can be integrated into the network straightforwardly. The experimental results shown in Table 5 demonstrate that CoFiI2P achieves improved performance on both RRE and RTE metrics when equipped with DTA or GAL. Notably, with both components applied, CoFiI2P achieves improvements of **25.4%** and **24.1%** in RRE and RTE, respectively.

Base Model	DTA	GAL	RRE(°)	RTE(m)	RR(%)
CoFiI2P	×	×	1.14	0.29	**100.00**
CoFiI2P	✓	×	0.94	0.24	**100.00**
CoFiI2P	×	✓	1.01	0.27	**100.00**
CoFiI2P	✓	✓	**0.85**	**0.22**	**100.00**

Table 5. Experimental results on KITTI Odometry dataset[15] based on current SOTA model CoFiI2P[9]. "DTA" and "GAL" means whether we add **Dense Training Alignment** mechanism and **Geometry-guided Attention Loss** L_{att} into CoFiI2P during the training, respectively. We report the mean RRE, mean RTE, and RR metrics for comparison.

Localization with unknown intrinsic parameters. Ablation results of localization with predicted intrinsic parameters are shown in Table 6. In the absence of ground-truth intrinsic parameters during inference, we leverage DUSt3R[46] to predict the focal length of the images. The camera is assumed to follow a simple pinhole camera model, with the principle point fixed at the center of the image. When using predicted intrinsic parameters instead of ground-truth focal length, the localization accuracy shows a significant decline. However, enabling focal length refinement during RANSAC + EPnP[29][30] yields notable improvement on $\mathbf{Test}_{T1-T7}$, while maintaining similar performance on other two test splits. This suggests that refining predicted focal length during pose estimation is more effective when the correspondences are of higher quality.

	GT	**Refine**	$\mathbf{Test}_{T1-T7}$		$\mathbf{Test}_{T1-T7hard}$		$\mathbf{Test}_{T10}$	
	Focal	**Focal**	**RRE(°)**	**RTE(m)**	**RRE(°)**	**RTE(m)**	**RRE(°)**	**RTE(m)**
Ours	×	×	2.04	1.72	4.56	2.19	4.61	4.80
	×	✓	0.95	0.80	3.74	2.36	3.88	5.06
	✓	×	**0.66**	**0.51**	**2.64**	**1.13**	**2.53**	**2.69**

Table 6. Ablation study on localization with intrinsic parameters predicted by DUSt3R[46]. We report the median RRE and median RTE across all three test splits of the *Carla Intersection Dataset*. “GT Focal” refers to using the ground-truth focal length during inference, and “Refine Focal” enables focal length optimization as part of the RANSAC + EPnP[29][30] process.

Fusion transformer block number. Table 7 shows the experimental results of using different numbers of feature fusion layers N_C in Geometry-guided Feature Fusion (GFF) module. Our model achieves the best performance when utilizing a four-layer structure.

	N_c	**RRE(°)**	**RTE(m)**
Baseline	2	0.96	0.55
	6	0.73	0.59
	8	0.88	0.58
Ours	4	**0.66**	**0.51**

Table 7. Ablation study on the number of feature fusion layers N_c in Geometry-guided Feature Fusion (GFF) module. We report median RRE and median RPE on test split $\mathbf{Test}_{T1-T7}$ of *Carla Intersection Dataset*.

Input point cloud size. We conducted ablation studies to investigate the effect of input point cloud size on the representation learning process. The number of coarse point groups was fixed to $M = 512$, as these groups were generated using Farthest Point Sampling (FPS), ensuring uniform sampling across the point cloud. As shown in Table 8, the localization accuracy decreases with lower point cloud densities, as overly sparse point cloud lose local critical structural details. On the other hand, higher-density point clouds place a heavy computational burden. To balance computational efficiency and accuracy, we selected an input size of 20,480 points.

Point Number	RRE($^\circ$)	RTE(m)	FLOPs
5120	0.86	0.68	126.73G
10240	0.81	0.62	146.38G
20480	0.66	**0.51**	185.73G
40960	**0.59**	0.52	264.35G

Table 8. Ablation study on the input point cloud size. We report median RRE and median RPE on test split $\mathbf{Test}_{T1-T7}$ of *Carla Intersection Dataset*. The FLOPs is calculated during the inference process.

6. Conclusion

In this work we focus on the under-explored problem of traffic camera localization, which is an important capability for fully-integrated spatial awareness among city-scale camera networks and vehicles. Such large-scale sensor fusion has the potential to enable more robustness, going beyond the limitations of a single vehicle's point of view. We proposed a novel method, TrafficLoc, which we show to be effective. To facilitate training and evaluation we propose the novel Carla Intersection dataset, focusing on the case of intersections, which is a common placement for traffic cameras and a focal point for traffic safety. We hope that this dataset will facilitate more research into integrated camera networks for robust, cooperative perception.

Appendix A. Overview

In this supplementary material, we provide a detailed explanations of our TrafficLoc and the proposed *Carla Intersection Dataset*. In Sec. B, we outline the data collection process and provide visualizations of our *Carla Intersection Dataset*. Sec. C describes the elements of the Fusion Transformer in the GFF module, followed by Sec. D with implementation details of our network architecture and training procedure. Finally, Sec. E offers additional visualizations of our localization results across different datasets.

Appendix B. Carla Intersection Dataset

Our proposed *Carla Intersection Dataset* consists of 75 intersections across 8 worlds ($Town01$ to $Town07$ and $Town10$) within the Carla [51] simulation environment, encompassing both urban and rural landscapes. $Town01$ to $Town07$ include multiple intersections for training and testing, while $Town10$ contains only one intersection for testing.

Specifically, we utilize the first Intersection scenario from each world (e.g. $Town01$ $Int1$, $Town02$ $Int1$, ..., $Town07$ $Int1$, $Town10$ $Int1$) for testing, with all remaining intersections reserved for training.

Images. For each intersection, we captured 768 training images and 288 testing images with known ground-truth 6-DoF pose at a resolution of 1920x1080 pixel and a horizontal field of view (FOV) of 90°, equals to a focal length of 960. To generate these images, we sampled camera positions in a grid-like pattern with different heights at the center of each intersection. For each position, we captured images at 8 yaw angles (spaced at 45° intervals) and 2 pitch angles. Figure 8 shows the sampled poses for example intersections.

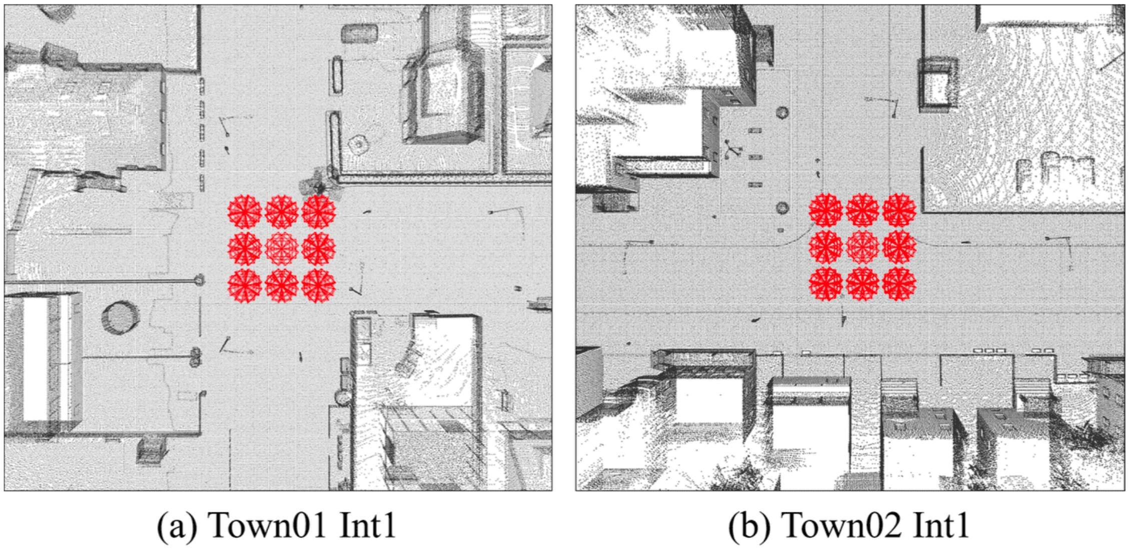


Figure 8. Sampled testing image poses of (a) Town01 Intersection1 and (b) Town02 Intersection1.

Table 9 summarizes the image data collection details for our *Carla Intersection Dataset*. All training images were captured with downward pitch angles of 15° and 30° at heights of $6\,m$, $7\,m$, and $8\,m$. Testing images in the test splits $\mathbf{Test}_{T1-T7}$ and $\mathbf{Test}_{T10}$ share the same pitch angles as the training images, but were captured at heights of $6.5\,m$ and $7.5\,m$. Additionally, for the test split $\mathbf{Test}_{T1-T7hard}$, we captured 288 additional testing images for each intersection using the same positions as in $\mathbf{Test}_{T1-T7}$, but with different pitch angles of 20° and 25°, at heights of $6.5\,m$ and $7.5\,m$. These data capture settings closely reflect the real-world traffic surveillance camera installations following HIKVISION[53], ensuring typical positioning to provide optimal traffic views under varied monitoring conditions. The differences between three distinct test splits also allow us to evaluate the model's generalization ability across unseen intersections and unseen world styles. Note that all testing intersections were not seen during the training.

	Training	$\textbf{Test}_{T1-T7}$	$\textbf{Test}_{T1-T7hard}$	$\textbf{Test}_{T10}$
worlds	$Town$01-07	$Town$01-07	$Town$01-07	$Town$10
# intersections	67	7	7	1
# images per scene	768	288	288	288
height (m)	6 / 7 / 8	6.5 / 7.5	6.5 / 7.5	6.5 / 7.5
pitch (°)	15 / 30	15 / 30	20 / 25	15 / 30
seen intersection	–	×	×	×
seen world	–	✓	✓	×

Table 9. Image data collection details of the proposed *Carla Intersection Dataset*. "# intersections" means the number of intersection scenes in each split dataset and "# images per scene" means the number of images in each intersection scene. "Seen intersection" and "seen world" represent whether the testing intersections are seen and whether the testing intersections are from the seen world during the training process, respectively.

Point Clouds. To capture the point cloud of each intersection, we utilize a simulated LiDAR sensor in the Carla[51] environment, which emulates a rotating LiDAR using ray-casting. The LiDAR operates at a rotation frequency of 10 frames per second (FPS), with a vertical field of view (FOV) ranging from 10° (upper) to -30° (lower). The sensor generates 224,000 points per second across all lasers. Other parameters of the simulated LiDAR follow the default configuration in Carla. As shown in Figure 9, the LiDAR scans were captured in an on-board manner. Then, we accumulated all scans into a single point cloud and downsampled it with a resolution of $0.2\ m$. Finally, the point cloud for each intersection was cropped to a region measuring $100\ m \times 100\ m \times 50\ m$, focusing on the area of interest for our study.

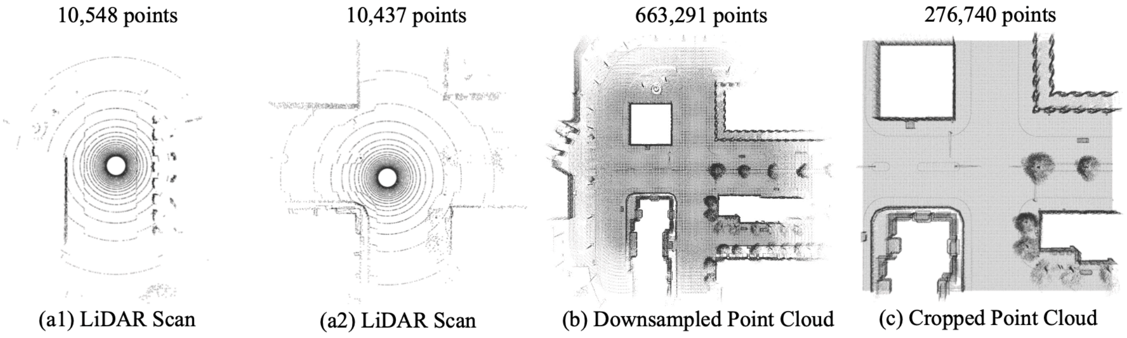


Figure 9. Point cloud capturing example from $Town10Int1$. (a1) and (a2) depict the LiDAR scan from a single frame. (b) shows the aggregated and downsampled point cloud. (c) presents the final cropped point cloud with dimensions of $100\ m \times 100\ m \times 100\ m$.

doi.org/10.32388/97AHJL

During the data capturing process, we disabled dynamic weather variations and set the weather condition in Carla simulation environment to the default weather parameters of world $Town10$. Some examples of our *Carla Intersection Dataset* are shown in Figure 11. Our data collection codes and datasets will be publicly available upon acceptance.

Appendix C. Geometry-guided Feature Fusion

Our Geometry-guided Feature Fusion (GFF) module comprises of N_c transformer-based fusion blocks, each consisting of a self-attention layer followed by a cross-attention layer.

Given the image feature $\mathbf{F}_I$ and point cloud feature $\mathbf{F}_P$, both enriched with positional embeddings, the self-attention layer enhances features within each modality individually using standard multi-head scalar dot-product attention:

$$\dot{\mathbf{F}} = \mathbf{Q} + \mathrm{MHA}(\mathbf{Q}, \mathbf{K}, \mathbf{V}), \tag{15}$$

where $\mathbf{Q} = \mathbf{K} = \mathbf{V} = \mathbf{F} \in \mathbb{R}^{N_t \times C}$ denotes the *Query*, *Key* and *Value* matrices, and $\mathbf{F}$ represents either $\mathbf{F}_I$ or $\mathbf{F}_P$ depending on the modality. Within the MHA layer, the attention operation is conducted by projecting $\mathbf{Q}, \mathbf{K}$ and $\mathbf{V}$ using h heads:

$$\begin{aligned} \mathrm{MHA}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) &= [\mathrm{head}_1, \ldots, \mathrm{head}_h]\mathbf{W}^O \qquad (16) \\ \mathrm{head}_i &= \mathrm{Attention}(\mathbf{Q}\mathbf{W}_i^Q, \mathbf{K}\mathbf{W}_i^K, \mathbf{V}\mathbf{W}_i^V) \end{aligned}$$

where $\mathbf{W}_i^{Q,K,V,O}$ denote the learnable parameters of linear projection matrices and the $\mathrm{Attention}$ operation is defined as:

$$\mathrm{Attention}(q, k, v) = \mathrm{Softmax}\left(\frac{q \cdot k^\top}{\sqrt{d_k}}\right) v, \tag{17}$$

where d_k is the dimension of latent feature.

The cross-attention layer fuses image and point cloud features by applying the attention mechanism across modalities, following the same formulation as Equation 15. However, the *Query*, *Key* and *Value* matrices differs based on the direction of attention. Specifically, for **I2P** (Image-to-Point Cloud) attention, we use $\mathbf{Q} = \mathbf{F}_I$ and $\mathbf{K} = \mathbf{V} = \mathbf{F}_P$, while for **P2I** (Point Cloud-to-Image) attention, we set $\mathbf{Q} = \mathbf{F}_P$ and $\mathbf{K} = \mathbf{V} = \mathbf{F}_I$.

Layer Normalization is applied to ensure stable training. For our GFF module, we set $N_c = 4$ and $h = 4$. Both the input channel C and the latent dimension d_k are set to 256.

Appendix D. Implementation Details

In *Carla Intersection Dataset*, each intersection point cloud represents a region of $100m \times 100m \times 50m$ and contains over 200,000 points. Following[54], as a preprocessing step, we first divide each intersection point cloud into several $50m \times 50m \times 50m$ voxels with a stride of $25m$. For each voxel V_i, we assign an associated set of images $\{I_i\}$ based on the overlap ratio between the image frustum and the voxel. Specifically, a voxel V_i is associated with an image I_i if more than 30% projected points lie within the image plane. During each training epoch, we uniformly sample B images for each voxel from its associated image set, resulting in $B \cdot N_v$ training image-point cloud pairs, where N_v denotes the total number of voxels.

The input images are resized to 288 $\times$ 512, and the input point cloud size is 20480 points. We utilize a pre-trained Vision Transformer from DUSt3R_ViT Large[46] to extract the image feature. For coarse matching, we use a resolution of 1/16 of

doi.org/10.32388/97AHJL

the input resolution for both the image and point cloud ($s = 16, M = 512$), with a coarse feature channel size of $C = 256$. For fine matching, we adopt a resolution of $(H/2 \times W/2 \times C')$ for fine image feature and $(N \times C')$ for fine point feature, where H, W and N equal to input dimensions and the fine feature channel size is set to $C' = 64$. As part of data augmentation, we apply random center cropping to the input images before resizing operation to simulate images captured by different focal lengths. The input point cloud is first normalized into a unit cube, followed by random rotations around the z-axis (up to 360°) and random shifts along the xy-plane (up to $0.1\ m$).

The whole network is trained for 25 epochs with a batch size of 8 using the Adam optimizer[55]. The initial learning rate is set to 0.0005 and is multiplied by 0.5 after every 5 epochs. For the joint loss function, we set $\lambda_1 = \lambda_2 = \lambda_3 = 1$ and add the Geometry-guided Attention Loss L_{att}. The safe radius r, positive margin m_p, negative margin m_p and scale factor γ in loss function are set to 1, 0.2, 1.8 and 10, respectively. For the Geometry-guided Attention Loss (GAL), the angular thresholds θ_{low} and θ_{up} are set to 10° and 20°, while the distance thresholds d_{low} and d_{up} are set to 3m and 5m, respectively. The training is conducted on a single NVIDIA RTX 6000 GPU and takes approximately 40 hours.

During inference, we utilize the super-point filter to select reliable in-frustum point groups from the fused coarse point features $\mathbf{F}_P^{coarse}$, using a confidence threshold of 0.9. In the **coarse matching stage**, we compute the coarse similarity map between each point group and the image. Following[50], a **window soft-argmax** operation is employed on similarity map to estimate the corresponding coarse pixel position. This involves first identifying the target center with an argmax operation, followed by a soft-argmax within a predefined window (window size set to 5). In the **fine matching stage**, with the predicted coarse pixel position, we first extract a fine local patch feature of size $w \times w$ from the fine image feature and select the fine point feature of each point group center, and then compute the fine similarity map between each point group center and the extracted local patch. Since the extracted local fine image patch has a relative small size ($w = 8$), a soft-argmax operation is applied over the **entire** fine similarity map to determine the final corresponding 2D pixel for each 3D point group center. Finally, we estimate the camera pose using RANSAC + EPnP[29][30] based on the predicted 2D-3D correspondences. For cases where one single image is associated with multiple point clouds, an additional RANSAC + EPnP step is performed using all inliers from each image-point cloud pair to compute the final camera pose.

For experiments on the KITTI Odometry[15] and Nuscenes[16] datasets, we ensure a fair comparison by adopting the same procedures as in previous works[41][8][9] to generate image-point cloud pairs.

In the KITTI Odometry dataset[15], there are 11 sequences with ground-truth camera calibration parameters. Sequences 0-8 are used for training, while sequences 9-10 are reserved for testing. Each image-point cloud pair was selected from the same data frame, meaning the data was captured simultaneously using a 2D camera and a 3D LiDAR with fixed relative positions. During training, the image resolution was set to 160×512 pixels, and the number of points was fixed at 20480. The model was trained with a batch size of 8 until convergence. The initial learning rate is set to 0.001 and is multiplied by 0.5 after every 5 epochs.

For the NuScenes dataset[16], we utilized the official SDK to extract image-point cloud pairs, with the point clouds being accumulated from the nearby frames. The dataset includes 1000 scenes, of which 850 scenes were used for training and

150 for testing, following the official data split. The image resolution was set to 160×320 pixels, and the number of points was fixed at 20480.

Appendix E. More Visualization Results

In this section, we present more localization results. Figure 10 and Figure 12 compare the localization performance of TrafficLoc with other baseline methods on the KITTI Odometry dataset[15] and all three test splits of the *Carla Intersection Dataset*. Our TrafficLoc predicts a higher number of correct point-to-pixel correspondences, and the point cloud projected with the predicted pose exhibits greater overlap with the image, demonstrating superior alignment.

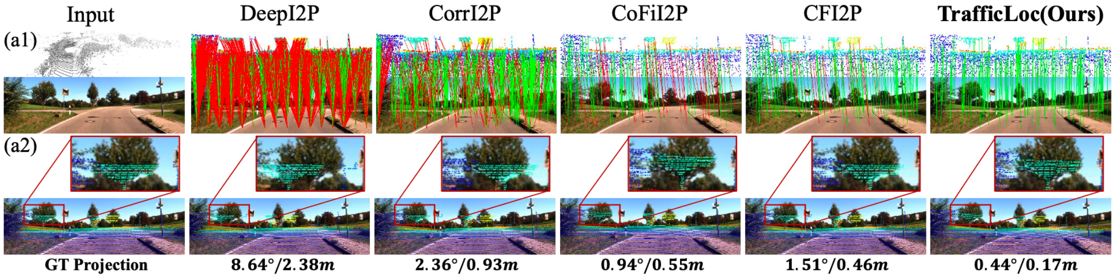
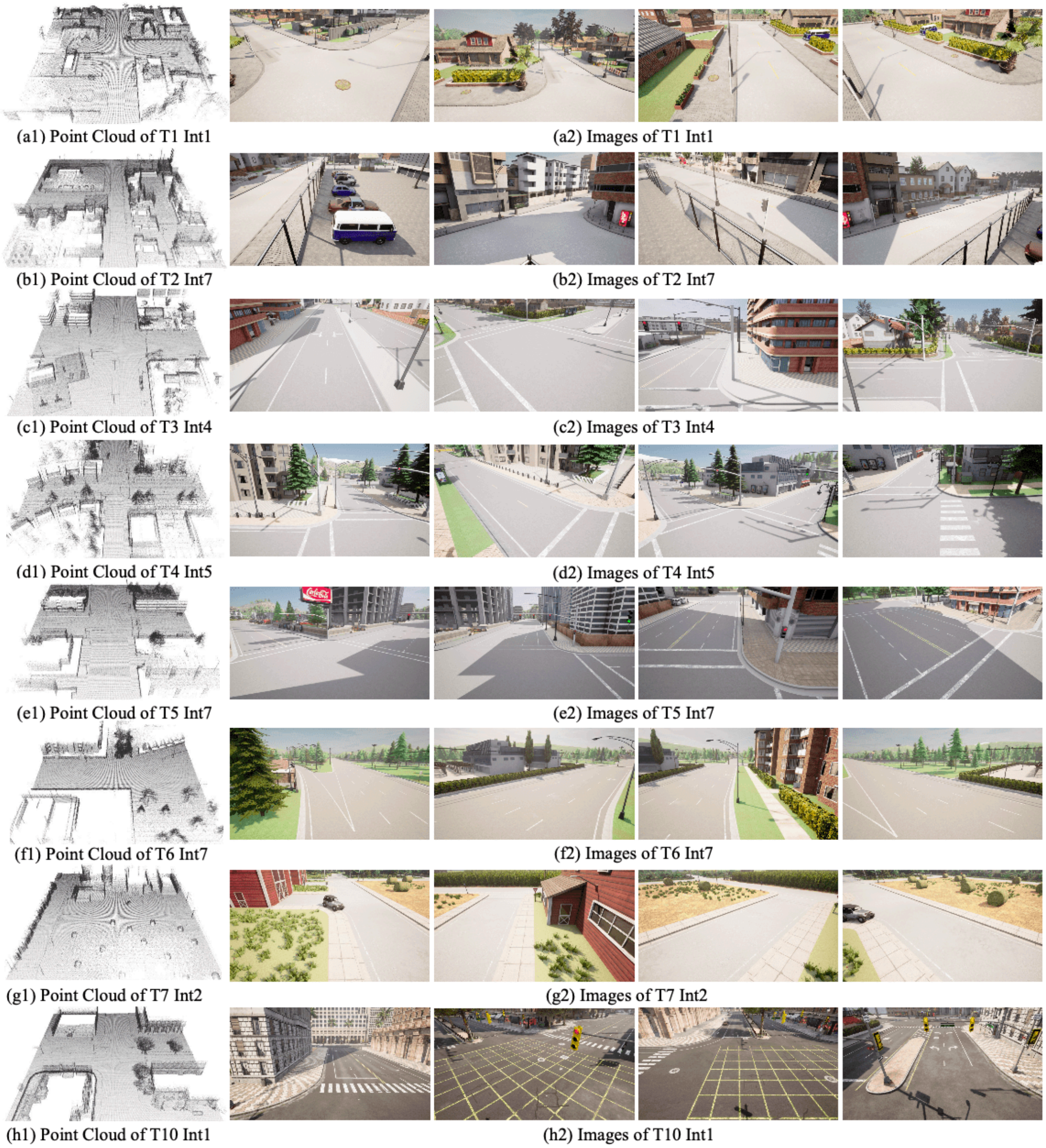


Figure 10. Qualitative results of our TrafficLoc and other baseline methods on the KITTI Odometry dataset[15]. (a1) shows predicted correspondences and (a2) visualizes the point cloud projected onto the image plane. The first column provides the input point cloud, the input image and the ground-truth projection for reference.



(a1) Point Cloud of T1 Int1

(a2) Images of T1 Int1

(b1) Point Cloud of T2 Int7

(b2) Images of T2 Int7

(c1) Point Cloud of T3 Int4

(c2) Images of T3 Int4

(d1) Point Cloud of T4 Int5

(d2) Images of T4 Int5

(e1) Point Cloud of T5 Int7

(e2) Images of T5 Int7

(f1) Point Cloud of T6 Int7

(f2) Images of T6 Int7

(g1) Point Cloud of T7 Int2

(g2) Images of T7 Int2

(h1) Point Cloud of T10 Int1

(h2) Images of T10 Int1

Figure 11. Example point clouds and images data of our *Carla Intersection Dataset*. T1 means $Town01$ and Int1 means $Intersection1$. Since all instances of the $Intersection1$ scenario across different worlds are included in the test set, we focus on showcasing their testing images (e.g. T1 Int1 and T10 Int1). For other intersections, we present the training images instead.

doi.org/10.32388/97AHJL

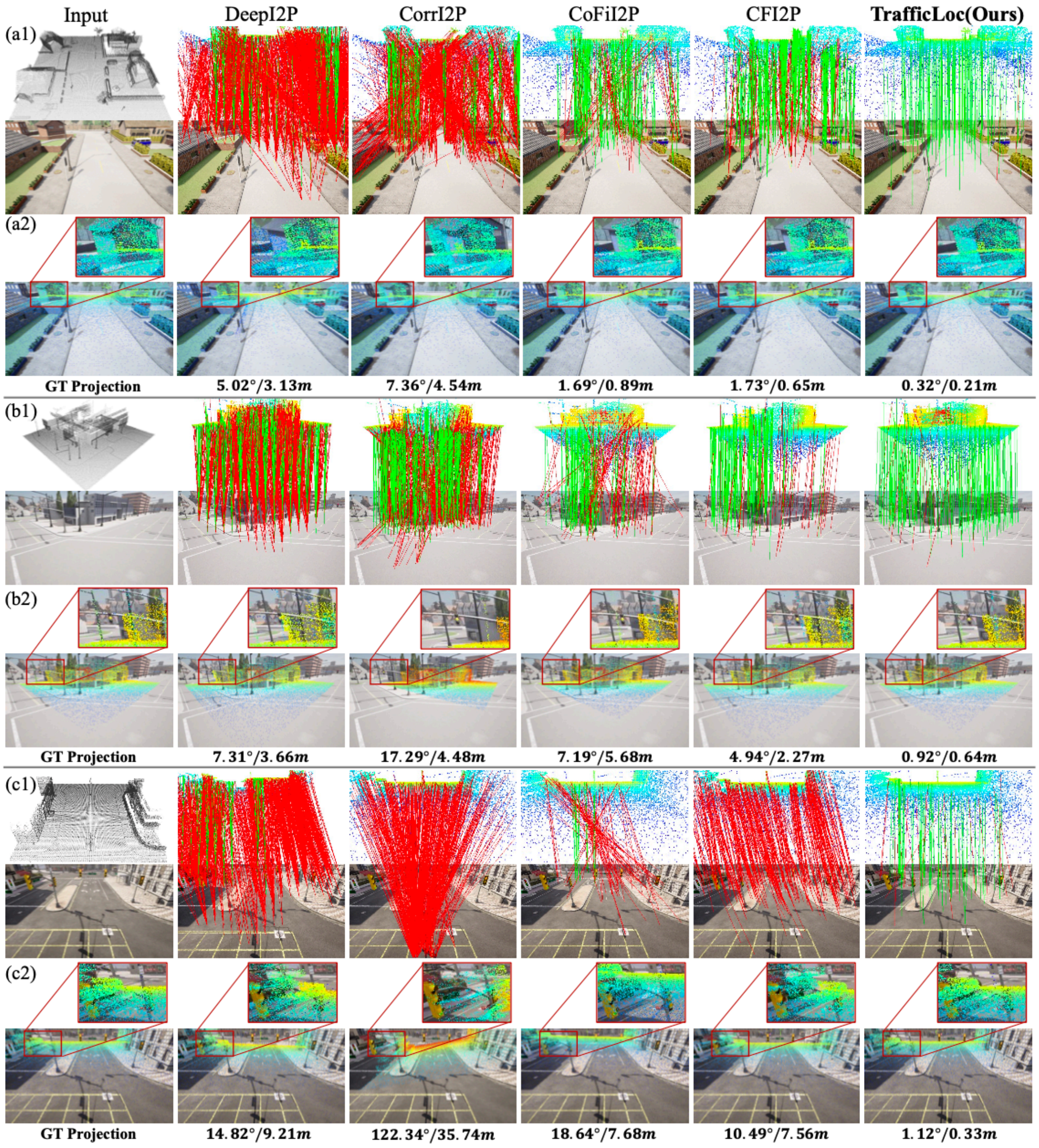


Figure 12. Qualitative results of our TrafficLoc and other baseline methods on the *Carla Intersection Dataset*. The point cloud is projected onto a 2D view and displayed above the image, with point colors indicating distance. The proposed TrafficLoc achieves superior performance, with more correct (green) and fewer incorrect (red) point-to-pixel pairs. (a1) shows predicted correspondences on $\mathbf{Test}_{T1-T7}$ and (a2) visualizes the point cloud projected onto the image plane. Similarly, (b1) and (b2) show results on $\mathbf{Test}_{T1-T7hard}$, (c1) and (c2) show results on $\mathbf{Test}_{T10}$. The first column provides the input point cloud, the input image and the ground-truth projection for reference.

References

1. ^Ravi Kiran B, Roldao L, Irastorza B, Verastegui R, Suss S, Yogamani S, Talpaert V, Lepoutre A, Trehard G. Real-time dynamic object detection for autonomous driving using prior 3d-maps. In: Proceedings of the European Conference on Computer Vision (ECCV) Workshops. 2018. p. 0–0.
2. ^Xia Y, Wu Q, Li W, Chan AB, Stilla U (2023). "A lightweight and detector-free 3d single object tracker on point clouds". IEEE Transactions on Intelligent Transportation Systems. 24 (5): 5543–5554.
3. a, bVuong K, Tamburo R, Narasimhan SG. Toward planet-wide traffic camera calibration. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. 2024. p. 8553-8562.
4. a, bHyeon J, Kim J, Doh N. "Pose correction for highly accurate visual localization in large-scale indoor spaces." In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). 2021 Oct; p. 15974-15983.
5. a, b, cBach TB, Dinh TT, Lee JH. "FeatLoc: Absolute Pose Regressor for Indoor 2D Sparse Features with Simplistic View Synthesizing." ISPRS Journal of Photogrammetry and Remote Sensing. 189: 50-62, 2022. doi:10.1016/j.isprsjprs.2022.04.021. Link to article.
6. ^Yuan C, Liu X, Hong X, Zhang F (2021). "Pixel-level extrinsic self calibration of high resolution lidar and camera in targetless environments". IEEE Robotics and Automation Letters. 6 (4): 7517–7524.
7. a, bKoide K, Oishi S, Yokozuka M, Banno A (2023). "General, single-shot, target-less, and automatic LiDAR-camera extrinsic calibration toolbox". arXiv preprint arXiv:2302.05094.
8. a, b, c, d, e, f, gRen S, Zeng Y, Hou J, Chen X (2022). "CorrI2P: Deep image-to-point cloud registration via dense correspondence". IEEE Transactions on Circuits and Systems for Video Technology. 33 (3): 1198–1208.
9. a, b, c, d, e, f, g, h, i, j, k, l, m, nKang S, Liao Y, Li J, Liang F, Li Y, Zou X, Li F, Chen X, Dong Z, Yang B (2023). "CoFiI2P: Coarse-to-fine correspondences for image-to-point cloud registration". arXiv preprint arXiv:2309.14660. 2023.
10. a, b, c, d, e, f, g, h, iYao G, Xuan Y, Chen Y, Pan Y (2023). "CFI2P: Coarse-to-Fine Cross-Modal Correspondence Learning for Image-to-Point Cloud Registration". arXiv preprint arXiv:2307.07142.
11. ^Xia Y, Xu Y, Li S, Wang R, Du J, Cremers D, Stilla U (2021). "Soe-net: A self-attention and orientation encoding network for point cloud based place recognition". In: Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition. 2021:11348–11357.
12. ^Xia Y, Gladkova M, Wang R, Li Q, Stilla U, Henriques JF, Cremers D. "Casspr: Cross attention single scan place recognition." In: Proceedings of the IEEE/CVF International Conference on Computer Vision. 2023. p. 8461–8472.
13. a, b, c, d, e, fSheng Y, Zhang L, Li X, Duan Y, Zhang Y, Zhang Y, Ji J (2024). "Rendering-Enhanced Automatic Image-to-Point Cloud Registration for Roadside Scenes". arXiv preprint arXiv:2404.05164.
14. a, bJing X, Han F, Ding X, Wang Y, Xiong R. Intrinsic and extrinsic calibration of roadside lidar and camera. In: 2022 China Automation Congress (CAC). IEEE; 2022. p. 2367-2372.
15. a, b, c, d, e, f, g, h, i, jGeiger A, Lenz P, Urtasun R. (2012). "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite". In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).

16. a, b, c, d, e, f*Caesar H, Bankiti V, Lang AH, Vora S, Liong VE, Xu Q, Krishnan A, Pan Y, Baldan G, Beijbom O. "nuScenes: A multimodal dataset for autonomous driving." In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2020 May. doi:10.1109/cvpr42600.2020.01164.*

17. ^*Agarwal S, Furukawa Y, Snavely N, Simon I, Curless B, Seitz SM, Szeliski R (2011). "Building Rome in a Day". Communications of the ACM.* **54** *(10): 105–112.*

18. ^*Heinly J, Schonberger JL, Dunn E, Frahm JM (2015). "Reconstructing the World in Six Days." In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).*

19. ^*Ozyesil O, Voroninski V, Basri R, Singer A (2017). "A Survey of Structure from Motion". arXiv preprint arXiv:1701.08493.*

20. ^*Sarlin PE, Cadena C, Siegwart R, Dymczyk M (2019). "From Coarse to Fine: Robust Hierarchical Localization at Large Scale". Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 12716--12725.*

21. ^*Sattler T, Leibe B, Kobbelt L (2011). "Fast Image-Based Localization using Direct 2D-to-3D Matching". In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). IEEE. pp. 667–674.*

22. ^*Sattler T, Leibe B, Kobbelt L (2016). "Efficient & effective prioritized matching for large-scale image-based localization". IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI).* **39** *(9): 1744–1756.*

23. ^*Snavely N, Seitz SM, Szeliski R (2008). "Modeling the World from Internet Photo Collections". International Journal of Computer Vision (IJCV).* **80** *(2): 189--210.*

24. ^*Taira H, Okutomi M, Sattler T, Cimpoi M, Pollefeys M, Sivic J, Pajdla T, Torii A (2018). "InLoc: Indoor Visual Localization with Dense Matching and View Synthesis". In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 7199–7209.*

25. ^*DeTone D, Malisiewicz T, Rabinovich A. Superpoint: Self-supervised interest point detection and description. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). 2018. p. 224–236.*

26. ^*Dusmanu M, Rocco I, Pajdla T, Pollefeys M, Sivic J, Torii A, Sattler T. "D2-Net: A Trainable CNN for Joint Detection and Description of Local Features." In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2019.*

27. ^*Lowe DG (2004). "Distinctive Image Features from Scale-Invariant Keypoints". International Journal of Computer Vision (IJCV).* **60** *(2): 91–110.*

28. ^*Revaud J, Weinzaepfel P, de Souza CR, Humenberger M (2019). "R2D2: Repeatable and Reliable Detector and Descriptor". In: Advances in Neural Information Processing Systems (NeurIPS), 2019.*

29. a, b, c, d, e, f, g*Lepetit V, Moreno-Noguer F, Fua P (2009). "EPnP: An Accurate O(n) Solution to the PnP Problem". International Journal of Computer Vision (IJCV).* **81** *(2): 155–166.*

30. a, b, c, d, e, f, g*Fischler MA, Bolles RC (1981). "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography". Commun. ACM.* **24** *(6): 381–395. doi:10.1145/358669.358692.*

31. ^*Arandjelovic R, Gronat P, Torii A, Pajdla T, Sivic J. "NetVLAD: CNN architecture for weakly supervised place recognition." In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2016:5297-5307.*

32. ^*Humenberger M, Cabon Y, Guerin N, Morat J, Revaud J, Rerole P, Pion N, de Souza C, Leroy V, Csurka G (2020). "Robust Image Retrieval-based Visual Localization using Kapture". arXiv preprint arXiv: 2007.13867.*

33. ^*Brahmbhatt S, Gu J, Kim K, Hays J, Kautz J (2018). "Geometry-Aware Learning of Maps for Camera Localization". Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2018: 2616–2625.*

34. ^*Kendall A, Cipolla R (2017). "Geometric loss functions for camera pose regression with deep learning". In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5974–5983.*

35. ^*Sattler T, Zhou Q, Pollefeys M, Leal-Taixe L (2019). "Understanding the Limitations of CNN-based Absolute Camera Pose Regression". Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3302–3312.*

36. ^*Feng M, Hu S, Ang MH, Lee GH (2019). "2D3D-MatchNet: Learning to Match Keypoints Across 2D Image and 3D Point Cloud". In: International Conference on Robotics and Automation (ICRA). IEEE. pp. 4790--4796.*

37. ^*Pham QH, Uy MA, Hua BS, Nguyen DT, Roig G, Yeung SK. "LCD: Learned Cross-Domain Descriptors for 2D-3D Matching." In: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI). 2020; 34(07): 11856–11864.*

38. ^*Xing X, Cai Y, Lu T, Cai S, Yang Y, Wen D (2018). "3DTNet: Learning Local Features Using 2D and 3D Cues". In: International Conference on 3D Vision (3DV). pp. 435-443.*

39. ^*Cattaneo D, Vaghi M, Fontana S, Ballardini AL, Sorrenti DG. Global visual localization in LiDAR-maps through shared 2D-3D embedding space. In: International Conference on Robotics and Automation (ICRA); 2020. p. 4365-4371.*

40. ^*Li YJ, Gladkova M, Xia Y, Wang R, Cremers D. "VXP: Voxel-Cross-Pixel Large-scale Image-LiDAR Place Recognition." In: 2025 International Conference on 3D Vision (3DV); 2025.*

41. a, b, c, d, e, f, g, h*Li J, Lee GH. DeepI2P: Image-to-Point Cloud Registration via Deep Classification. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2021. p. 15960–15969.*

42. ^*Wang H, Liu Y, Wang B, Sun Y, Dong Z, Wang W, Yang B (2023). "FreeReg: Image-to-point cloud registration leveraging pretrained diffusion models and monocular depth estimators". arXiv preprint arXiv:2310.03420. arXiv:2310.03420.*

43. a, b*Kim M, Koo J, Kim G (2023). "Ep2p-loc: End-to-end 3d point to 2d pixel localization for large-scale visual localization". Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 21527–21537.*

44. a, b, c*Zhou J, Ma B, Zhang W, Fang Y, Liu YS, Han Z (2024). "Differentiable registration of images and lidar point clouds with voxelpoint-to-pixel matching". Advances in Neural Information Processing Systems. 36.*

45. a, b, c, d, e*He K, Zhang X, Ren S, Sun J (2016). "Deep residual learning for image recognition". In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778.*

46. a, b, c, d, e, f, g, h, i*Wang S, Leroy V, Cabon Y, Chidlovskii B, Revaud J (2024). "Dust3r: Geometric 3d vision made easy". In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 20697–20709.*

47. a, b*Qi CR, Su H, Mo K, Guibas LJ (2017). "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation." In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 652–660.*

48. a, b, c, d, e*Zhao H, Jiang L, Jia J, Torr PHS, Koltun V (2021). "Point transformer". In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 16259–16268.*

49. ^*Bhalgat Y, Henriques JF, Zisserman A. "A light touch approach to teaching transformers multi-view geometry." In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2023. p. 4958-4969.*

50. a, b*Zhang J, Herrmann C, Hur J, Chen E, Jampani V, Sun D, Yang MH. "Telling left from right: Identifying geometry-aware semantic correspondence." In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2024. p.*

3076-3085.

51. a, b, c*Dosovitskiy A, Ros G, Codevilla F, López A, Koltun V (2017). "CARLA: An Open Urban Driving Simulator". Conference on Robot Learning. Oct 2017.*
52. a, b, c*Chen A, Zhang K, Zhang R, Wang Z, Lu Y, Guo Y, Zhang S (2023). "Pimae: Point cloud and image interactive masked autoencoders for 3d object detection". In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2023: 5291–5301.*
53. ^*HIKVISION (2016--2024). ANPR product functions and problem troubleshooting. Available from: https://www.securitywholesalers.com.au/files/ANPRINSTALLATION1.pdf.*
54. ^*Tang S, Tang S, Tagliasacchi A, Tan P, Furukawa Y (2023). "Neumap: Neural coordinate mapping by auto-transdecoder for camera localization". In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2023: 929–939.*
55. ^*Kingma DP, Ba JL. "Adam: A method for stochastic optimization." In: International Conference on Learning Representations; 2015. Sourced from Microsoft Academic - https://academic.microsoft.com/paper/2964121744.*

Declarations

Funding: No specific funding was received for this work.

Potential competing interests: No potential competing interests to declare.