Geo-Localization With Transformer-Based 2D-3D Match Network

Laijian Li, Yukai Ma, Kai Tang, Xiangrui Zhao, Chao Chen, Jianxin Huang, Jianbiao Mei, and Yong Liu

***Abstract*—This letter presents a novel method for geographical localization by registering satellite maps with LiDAR point clouds. This method includes a Transformer-based 2D-3D matching network called D-GLSNet that directly matches the LiDAR point clouds and satellite images through end-to-end learning. Without the need for feature point detection, D-GLSNet provides accurate pixel-to-point association between the LiDAR point clouds and satellite images. And then, we can easily calculate the horizontal offset $(\Delta x, \Delta y)$ and angular deviation $\Delta\theta_{yaw}$ between them, thereby achieving accurate registration. To demonstrate our network's localization potential, we have designed a Geo-localization Node (GLN) that implements geographical localization and is plug-and-play in the SLAM system. Compared to GPS, GLN is less susceptible to external interference, such as building occlusion. In urban scenarios, our proposed D-GLSNet can output high-quality matching, enabling GLN to function stably and deliver more accurate localization results. Extensive experiments on the KITTI dataset show that our D-GLSNet method achieves a mean Relative Translation Error (RTE) of 1.43 m. Furthermore, our method outperforms state-of-the-art LiDAR-based geospatial localization methods when combined with odometry.**

***Index Terms*—Geo-localization, 2D-3D match, SLAM.**

I. Introduction

PEOPLE'S pursuit of accurate positioning has not diminished with the advent of GPS, but rather, they have more expectations for stable and accurate positioning over a long time. Accurate positioning over long periods is also the key to autonomous driving technology. While LiDAR odometry [1], [2] can provide highly accurate relative motion pose estimation, it is subject to inevitable cumulative drift. Vehicle localization technologies primarily rely on Global Navigation Satellite Systems (GNSS) and ground-based maps (e.g., using LiDAR point clouds) [3], [4] for estimating absolute, geo-referenced poses. However, in urban environments with tall buildings, known as "urban canyons," commercial GPS signals often become unavailable [5]. Therefore, prior maps have become a viable solution for the localization problem. They can compensate for the limitations of GPS signals and provide higher-precision 6-DoF pose estimation. Nevertheless, most prior map methods require pre-mapping the environment and performing localization relative to a geo-referenced point cloud database. Furthermore, the coverage of existing prior maps is sometimes more extensive, limiting their use in unknown environments. Thus, some studies [6], [7], [8], [9] have proposed using publicly available maps, such as GIS and satellite imagery, for robot localization, which have already covered most parts of the world. However, the biggest challenge lies in associating data of different types and perspectives.

The current localized feature associations include image feature matching, 3D point cloud matching, and matching between point clouds and images. Local feature matching between images without detectors [11], [12] have aroused general interest. Detector-free methods are robust against texture differences, lighting, and viewpoint changes. At the same time, direct registration methods [13], [14], [15] for point clouds have replaced traditional correspondence-based methods [16], [17], [18]. They use neural networks to estimate transformations in a Transformer-like manner. Previous geo-localization methods [6], [7], [8], [9] obtain a series of ground images or point clouds as input and output the vehicle's pose estimation relative to the geographic reference satellite image. [6], [7] rely on the semantic information of satellite maps for data association. [8], [9] directly aligns ground with aerial images via learned visual features. However, they all fail to achieve satisfactory levels of accuracy.

Inspired by the current Transformer-based detector-free local feature matching method [19], [20], we propose an end-to-end geometric structure-based LiDAR (3D) and satellite image (2D) registration method. Furthermore, we design an independent node GLN, relying only on satellite imagery, that can replace the role of GPS and correct the cumulative offset of all current LiDAR odometry. We will verify this through specific and detailed experiments about different SLAM systems coupled with GLN or GPS, respectively. Among these LiDAR methods, we choose A-LOAM[1] and LIO-SAM [2] for the subsequent experiments.

To summarize, the main contributions of this letter are as follows:

- We propose a novel point-level matching model for LiDAR (3D) and satellite images (2D), which can be directly used for registration to obtain their horizontal offset $(\Delta x, \Delta y)$ and angular offset $(\Delta\theta_{yaw})$.
- Based on the proposed D-GLSNet network, we develop a plug-and-play Geo-localization Node (GLN) with any

Manuscript received 10 April 2023; accepted 13 June 2023. Date of publication 29 June 2023; date of current version 5 July 2023. This letter was recommended for publication by Associate Editor B. Duncan and Editor P. Pounds upon evaluation of the reviewers' comments. This work was supported by the National Natural Science Foundation of China under Grant NSFC 62088101. *(Laijian Li and Yukai Ma contributed equally to this work.) (Laijian Li and Yukai Ma are co-first authors.) (Corresponding author: Yong Liu.)*

The authors are with the Institute of Cyber-Systems and Control, Zhejiang University, Hangzhou 310012, China (e-mail: lilaijian@zju.edu.cn; 12132060@zju.edu.cn; kaitang@zju.edu.cn; xiangruizhao@zju.edu.cn; chenchao1924@zju.edu.cn; 22032097@zju.edu.cn; jianbiaomei@zju.edu.cn; yongliu@iipc.zju.edu.cn).

Digital Object Identifier 10.1109/LRA.2023.3290526

[1][Online]. Available: https://github.com/hkust-aerial-robotics/a-loam.

LiDAR SLAM system. This GLN offers several advantages over the GPS, particularly regarding reduced susceptibility to external interferences like building occlusion.

- We conduct extensive experiments on the KITTI dataset and use different odometry combined with GLN. Results show that our method outperforms the state-of-the-art LiDAR-based geo-localization methods and demonstrates the feasibility of replacing GPS with GLN.
- The code for the D-GLSNet and the dataset of satellite images aligned to KITTI can be downloaded from https://github.com/yzdad/D-GLSNet.git.

The rest of this article is organized as follows: Section II summarizes the related works in recent years. Our system is introduced in Section III. Section IV includes details about experimental settings and results on several datasets. Ultimately, we conclude a brief overview of our system and a future outlook in Section V.

II. Related Work

A. 2D-3D Matching

Matching between 2D images and 3D point clouds presents a unique challenge compared to matching within a single domain. The substantial differences in data types between the two make it a heterogeneous data-matching problem. Currently, there needs to be more research focused on addressing this issue. Some approaches employ a fusion of conventional hand-crafted techniques and machine learning. Li et al. [21] use hand-crafted 3D object-level embeddings as learning targets and learn corresponding image embeddings through CNN networks to complete the learning of object-level cross-domain descriptions. Similarly, 3DTNet [22] uses 2D and 3D local patches as input, and feature extraction of the 3D local pudding is assisted by features of 2D picture blocks, which improves the discriminable effort of 3D features. However, it can only be used for 3D matching.

Other methods use a pure learning approach, LCD [23] proposes a novel approach to learn cross-domain descriptors for 2D and 3D local patch matching, which employs a dual self-encoder neural network that maps 2D and 3D inputs to a shared latent space representation, respectively. Similar to LCD, 2D-3DMatchNet [24] proposes a 3D point cloud to 2D image feature extraction network, which uses traditional corner point extraction to get pixel-to-point pair matching, but is still essentially 2D and 3D local pudding pair matching. P2-Net [25] extracts each point descriptor, detects key point locations in a single forward pass, and then generates 2D point to 3D point matches from the extracted descriptions. Similarly, our work also generates pixel-to-point matches.

B. Cross-View Localization

Cameras and LiDAR are commonly used for ego-motion estimating, which has led to the development of two cross-view localization methods in remote sensing. One such method is image-based geo-localization, which involves querying ground images from an aerial image database [26]. To overcome the significant viewpoint differences between the two types of images, Kim et al. [8] used a dual-branch network to extract potential vectors of ground images and aerial image databases, using feature distance to measure the similarity between them. Again, [27], [28] adopted a dual-branch network

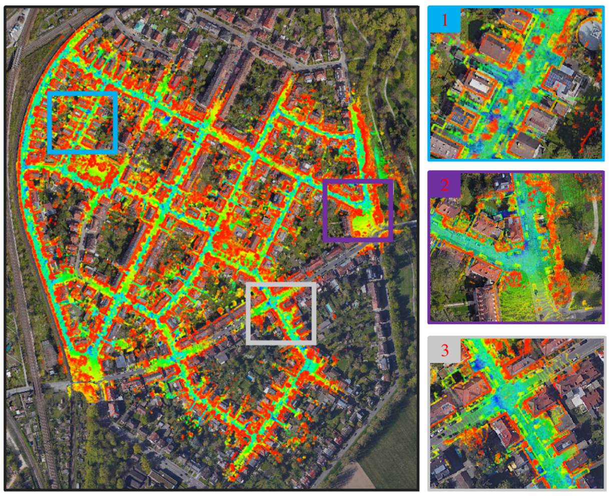

Fig. 1. Results of geo-localization and mapping using A-LOAM + GLN on the KITTI 00 dataset [10]. Our proposed geo-localization method accurately calculates the transformation of point clouds in 3-Dof of freedom on a satellite map. We have modularized the localization method so that any LiDAR-based odometry can use it. The right-hand side of this image shows some details.

structure and employed different methods to minimize the differences between the image domains. Other image-based geo-localization methods that utilize semantic information [7], [29] typically involve semantic segmentation of ground and satellite images, followed by comparison based on projection relationships. However, the accuracy of this approach depends on the labels used and the precision of the network.

Cross-view matching using LiDAR presents several challenges, including differences in point cloud density, distribution, and occlusions. However, the absolute scale provided by LiDAR and satellite images enables higher accuracy in cross-view matching. Data association in LiDAR can be complex due to differences in data modality, and some methods rely on manual or semantic techniques to perform data association. For example, Yan et al. [30] constructed a match-ready descriptor for top-down view maps from OpenStreetMap [cite] that enables efficient 3D LiDAR-based localization. Miller et al. [6] segmented LiDAR and Google Maps separately, computed Truncated Distance Fields (TDFs) for aerial semantic maps, and compared them with LiDAR data class-wise. AGCV-LOAM [31] uses a neural network to process LiDAR grid maps and satellite image patches, outputting attitude correction values that are added to the factor map for pose optimization.

Further, hybrid methods input images and LiDAR, such as those proposed by [9], [32], using LiDAR and image continuous data streams to improve perception and tracking. Nevertheless, they use LiDAR as an aid to provide depth or additional information. In contrast to previous work, our method directly learns the correspondence between LiDAR points and satellite images end-to-end using only LiDAR data. This method makes full use of the information of the LiDAR point cloud rather than just using semantics or depth.

III. Methodology

Inspired by LoFTR [19], we propose a two-stage matching model D-GLSNet for generating relatively dense correspondences between an image $I \in \mathbb{R}^{H\times W}$ and a point cloud

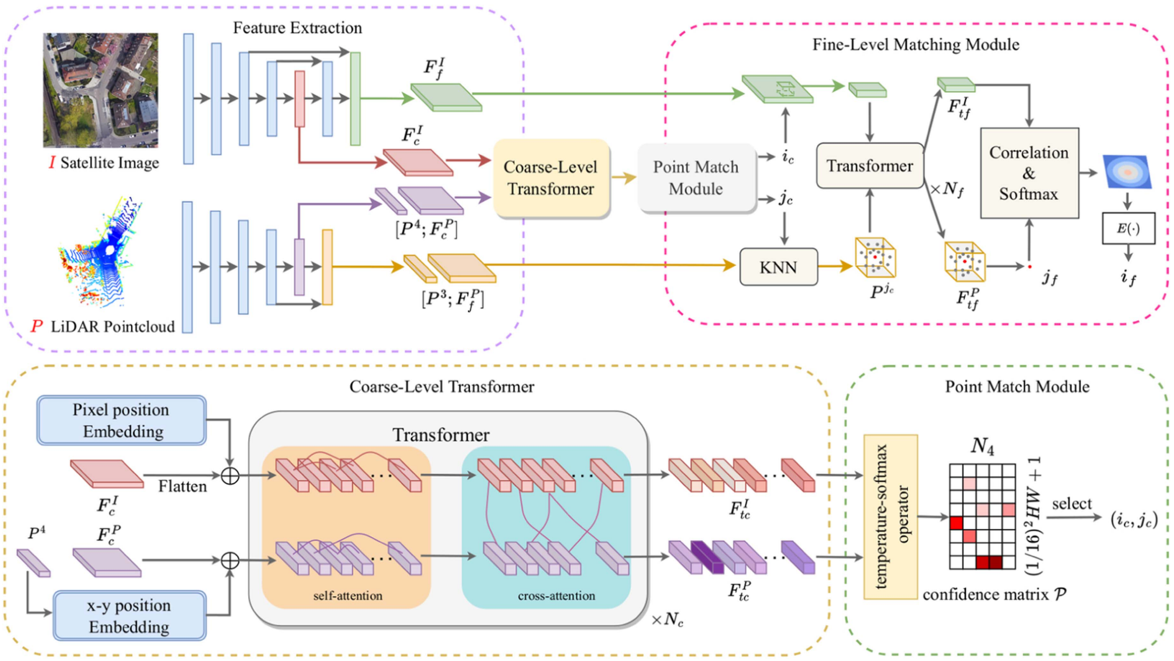

Fig. 2. Overview of the D-GLSNet. There are four components: 1. Feature Extraction extracts coarse features F_c^I, F_c^P and fine features F_f^I, F_f^P from images and point clouds(Section III-A). 2. Coarse-Level Transformer transforms coarse features F_c^I, F_c^P into more unique features F_{tc}^I, F_{tc}^P (Section III-B). 3. Point Matching Module uses coarse features F_{tc}^I, F_{tc}^P to generate coarse matching predictions $M_c = \{(i_c, j_c)\}$ (Section III-C). 4. Fine-Level Matching Module further refines each selected coarse prediction (i_c, j_c) to obtain a finer match (i_f, j_f) (Section III-D).

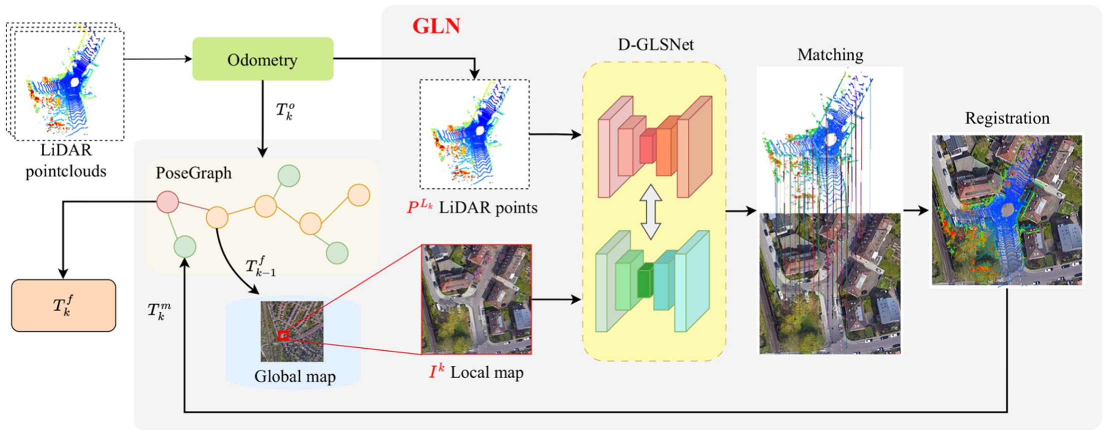

Fig. 3. Overview of the Geo-localization Node (Section III-F). We use the proposed network and pose graph optimization model to correct the cumulative error of the odometry. GLN is independent and can be combined with different odometry.

$P \in \mathbb{R}^{N\times 3}$, without relying on a separate detector. Our method employs a network architecture depicted in Fig. 2. Based on the proposed network model, we design a Geo-localization Node (shown in Fig. 3) that can be used for localization combined with a LiDAR SLAM system.

A. Feature Extraction

Image features are extracted using the Feature Pyramid Network (FPN) [33]. For each input image I, we extract a coarse feature F_c^I with a spatial resolution of 1/16 of the input image and a fine feature F_f^I with a spatial resolution of 1/4 of the input image. For point cloud feature extraction, we use the KPConv-FPN [34] backbone to extract multi-level features. We perform four down-sampling of the input point cloud P and obtain four point clouds $P^k \in \mathbb{R}^{N_k\times 3}, k = 1, 2, 3, 4$ with different resolutions. Then the point cloud features can be extracted by five encoder layers and one decoder layer. Here, $[P^4; F_c^P]$ denotes the coarse matching feature and $[P^3; F_f^P]$ represents the fine matching feature.

Matching on feature maps that have been down-sampled instead of the original image and point clouds, because registration can be fixed by a coarser correspondence between match subsets. The features of the original resolution are often too dense, and applying them is inefficient and time-consuming. Matching on low-resolution features leads to a noteworthy reduction in the input to the Transformer module, thereby substantially enhancing the matching efficiency while simultaneously preserving accuracy.

B. Coarse-Level Transformer

After obtaining the coarse features, F_c^I and $[P^4; F_c^P]$ are passed through the Transformer to extract position and context-dependent coarse features F_{tc}^I and F_{tc}^P. At the same time, we explicitly encode the geometry and match or estimate the relative geometric relationship between 2D and 3D using global and explicit cues. In this way, the learned features are geometrically discriminative and can effectively resolve the problem of matching ambiguity, thereby reducing the number of outlier matches. The Transformer models typically consist of a series of sequentially connected attention layers. The attention layer typically receives the query Q, key K, and value V as input, completing the work like information retrieval. The attention determines the weight by calculating the similarity between the query Q and the key K and then converts the similarity value V into a set of weights through a softmax layer. Finally, the attention value is obtained by weighting and summing the values according to the product of this set of weights and the corresponding value.

$$\text{Attention}(Q, K, V) = \text{softmax}(QK^\top)V.$$

The attention can be broadly interpreted as a vector of importance weights. To predict an element, the attention vector is used to estimate how it is related to other elements, and the weighted sum of these values is used as an approximation of the target. When the input query Q and value V are both image or point cloud features (F_c^I or F_c^P), this attention mechanism is called self-attention. When the input query Q and value V are image and point cloud features respectively (F_c^I and F_c^P), it is called cross-attention.

To integrate position information into features, the simplest and most effective method is to add or concatenate position embedding vectors directly to features, which is very effective in some Transformer-based models [19], [20], [35], [36]. To embed 2D pixel position and 3D point cloud consistently, we only embed the x and y position of the point cloud and recover the real scale $q(m/pix)$ of the satellite map by $zoomLevel$ ($pixelCoordinate = worldCoordinate * 2^{zoomLevel}$). The main reason is that satellite images are obtained by aerial satellite surveys of the ground, ignoring the height information of surface objects. The points in the LiDAR point cloud with the same horizontal coordinates (x, y) should correspond to the same point on the satellite image and maintain feature invariance along the z-axis.

C. Point Matching Module

This module is designed to accurately determine the pixel coordinates of the point cloud in the image. A single 2D image feature may correspond to multiple 3D point cloud features when conducting coarse-level matching. It means that the projection of multiple points will fall on the same image patch. This many-to-one matching method differs from the one-to-one matching approach used by many traditional images matching [19], [35] and point cloud matching [20], [36] methods. There are two reasons for this. The first is that the resolution of the image and the point cloud are different when downsampling, and the other is that points with the exact horizontal coordinates (x, y) should correspond to the same point on the satellite image due to a viewpoint difference. To achieve better matching results, we choose the dual-softmax operator. First, we normalize the F_{tc}^I and F_{tc}^P features output by the Transformer module and calculate the fractional matrix $S \in \mathbb{R}^{(1/16)^2 HW \times N_4}$ by:

$$S(i, j) =< normalize(F_{tc}^I(i)), normalize(F_{tc}^P(j)) > .$$

Since not all 3D points can be found on the 2D image, we extend the fraction S to $\hat{S}$ by adding a new row as a bin filled with a variable a. Then, we get the matching confidence matrix $\mathcal{P}$ by softmax with temperature t,

$$\mathcal{P}(i, j) = softmax(\hat{S}(\cdot, j)/t)_i.$$

Finally, we select the maximum value on the column of $\mathcal{P}$ and only select the match if the max confidence of the match is higher than the preset threshold th_c. The coarse-level matching predictions denote as:

$$M_c = \{(i_c, j_c) | \mathcal{P}(i_c, j_c) = max(\mathcal{P}(\cdot, j_c)), \mathcal{P}(i_c, j_c) > th_c\}.$$

D. Fine-Level Matching Module

Coarse matching is initially applied to the low-resolution inputs, but the accuracy of the matches is not sufficient for precise positioning. To address this limitation, we implement additional processing and refinement to improve the precision of the matching results. For each coarse match (i_c, j_c), we sample the fine features F_f^I and F_f^P from the image and point cloud. Precisely, in the 2D image, we first calculate the corresponding position (u_c, v_c) of i_c on the feature map F_f^I and then crop a window of size $w \times w$ at that position. For the 3D point cloud, We sample a local point cloud P^{j_c} from P^3 based on the position of j_c at Euclidean distance and then obtain the local point cloud feature from the F_f^P.

Similar to the Coarse-Level Transformer, we use the Transformer to obtain the features (F_{tf}^I and F_{tf}^P) to compute the precise point position on the image (Fig. 2: Fine-Level Matching Module). Notably, this part uses only local point features learned by the backbone, without any location embeddings. In fact, after resolving the global ambiguity by coarse matching, point-level matching mainly depends on the proximity between points. Finally, we select the features of the centroid j_f from P^{j_c} and all points in the image block and calculate the matching probability to obtain the heat map. By calculating the expected value of the probability distribution, we get the pixel position i_f as the matching position. After each point obtained by coarse matching is processed, the matches are collected to form the final dense global point correspondence $M_f = (i_f, j_f)$.

E. Loss Functions

In network training, the total loss function consists of two loss functions of coarse-level and fine-level: $\mathcal{L} = \mathcal{L}_c + \mathcal{L}_f$.

The coarse-level loss function is the Focal Loss of the confidence matrix $\mathcal{P}$. We use the known transformation relationship T_{gt} between the image and the point cloud to get the accurate rough match M_c^{gt}, and then calculate $\mathcal{L}_c$ by:

$$\mathcal{L}_c = -\frac{1}{M_c^{gt}} \sum_{(i_c,j_c)\in M_c^{gt}} (1-\mathcal{P}(i_c,j_c))^\gamma \log \mathcal{P}(i_c,j_c)$$

The fine-level matching uses $L2-loss$. For each query point j_f, we also measure its uncertainty by computing the total variance $\sigma^2(j_f)$ of the corresponding heat map. The goal is to optimize the location of the refinement with low uncertainty, resulting in the final weighted loss function:

$$\mathcal{L}_f = \frac{1}{M_f} \sum_{(i_f,j_f)\in M_f} \frac{1}{\sigma^2(j_f)} \|i_f - i_{gt}\|$$

where i_{gt} is obtained by projecting each j_f onto F_f^I. Ignore (i_f, j_f) if the projection of j_f exceeds the local window taken by i_f on F_f^I.

F. Geo-Localization Node

We develop a localization node named Geo-localization Node (GLN) for the odometry (Fig. 3). The node receives the 6-DoF odometry poses $\{T_k^o | k = 1, 2, 3..., n\}$ and LiDAR scans $\{P^{L_k} | k = 1, 2, 3..., n\}$ to correct the odometry trajectory with D-GLSNet and pose graph optimization. D-GLSNet provides feature-matching results between LiDAR point clouds and satellite maps to enable the registration of LiDAR data and maps. We then use the 3-DoF registration results $\{T_k^m | T_k^m = (\Delta x_k, \Delta y_k, \Delta\theta_{yaw_k})\}$ and 6-DoF pose $\{T_k^o\}$ to construct a pose graph, which will be optimized to give more accurate poses $\{T_k^f\}$. As GLN and odometry are separate threads, the node is suitable for arbitrary SLAM systems, and it provides new and accurate calculations without affecting the odometry.

In order to obtain T_k^m, we search for the closest local satellite map I^k from the global map as input to D-GLSNet using the latest pose T_{k-1}^f optimized by the pose graph. We feed the frame of LiDAR points P^{L_k} and the local satellite map I^k into the D-GLSNet network to obtain the alignment relations and then calculate the deviation of the LiDAR coordinates from the global map T_k^m. In practice, we need to measure the vehicle's rough starting position T_{prior}, which can be obtained with only GPS, to intercept the satellite map I^0 quickly.

GLN is a pseudo-GPS signal that also corrects for cumulative errors in positioning and can be more stable and less affected by 'urban canyons' than GPS. In addition, GLN can keep odometry remains stable and accurate, especially in urban scenarios with rich structural information. Although the initial position is required, it does not need to be measured precisely, which makes the system easier to implement and use.

IV. Experiments

In this section, we conduct a detailed evaluation of our proposed D-GLSNet on 2D satellite imagery and 3D LiDAR point cloud direct matching tasks. In addition, we also evaluate the accuracy of our designed module GLN combined with different LiDAR odometry. To this end, we train on KITTI-360 [37] and then evaluate it on KITTI odometry datasets. We exclude scenario KITTI 01 as the highway scene is not the focus of our attention and KITTI 03 as GPS ground truth is unavailable.

A. Implementation Details

In order to train our network to achieve pixel-to-point image and point cloud matching, we train on the dataset with images and point clouds that have an accurate pixel-to-point correspondence. We download the satellite images corresponding to each LiDAR point frame from Google Maps using KITTI-360's GPS, and collect images with a pixel resolution of $q = 0.196$ m/pix. Then, we remove LiDAR points farther than 50 m and downsample them with a resolution of 0.4 m. Several types of data augmentation are applied during the training process. We randomly rotate and pan the satellite image and intercept an image of size 480×480 pixels from the collect map as the input to the network. We randomly offset the LiDAR and satellite images horizontally by a maximum of 27.72 meters. In addition, we manipulate the images with photometric variations and add pretzel noise to increase the diversity of the data. We randomly rotate the LiDAR point cloud, where the roll and pitch angles are rotated by no more than 10°, to simulate LiDAR mounting changes and ground undulations. We also add random positional noise to the coordinates of each LiDAR point. We train our neural network on an NVIDIA GeForce RTX 3090. The complete model is trained end-to-end, with the weights being initialized randomly. The model is trained using Adam with an initial learning rate of $l = 5e^{-5}$, which will be dynamically adjusted by Warmup and MultiStepLR. We set the temperature $t = 0.01$, the point selection threshold $th_c = 0.5$ for Point Matching Model, and window size w set to 8. More details can be obtained by reading our code.

B. Evaluation on Matching

To assess the effectiveness of our 2D-3D direct matching network D-GLSNet in the pixel-to-point image and point cloud matching tasks, we use the following metrics commonly adopted in related work: 1) Average matching number (AMN): the number of correct pixel-to-point matches where the distance between a pixel and a point pair is below a threshold (i.e., 2 m) under its ground truth transformation; 2) Inlier Ratio (IR): the percentage of correct pixel matching among all possible matches; 3) Recall (R): the percentage of correct matches to all ground truth matches; 4) Feature Matching Recall (FMR): the percentage of point cloud pairs with an inlier ratio above a certain threshold (i.e., 30%).

None existing works can directly address pixel-to-point matching in 2D and 3D domains without keypoint detection. The closest related work [25] to ours requires manual pixel-to-point supervision, and no data is publicly available. Therefore, we present our results under different configurations, as summarized in Table I, where small, medium, and big indicate different sizes of feature extraction networks, respectively, and $[N_c, N_f]$ denotes the number of layers in the Coarse-Level and Fine-Level Transformer. We train 800 k steps for each configuration and select the best model. During the evaluation, we randomly apply a yaw rotation to the LiDAR point cloud and shift the satellite image, consistent with the training process. Our results show that the number of matching points obtained from the Fine-Level stage is significantly higher than that from the Coarse-Level stage for all performance metrics. Additionally, increasing the size of the feature extraction network and the number of Transformer layers improves the matching performance, but using an additional Transformer yields higher returns.

TABLE I
RESULTS OF OUR MODEL D-GLSNET ON THE KITTI DATASET UNDER DIFFERENT CONFIGURATIONS. THE TABLE ALSO SHOWS BOTH THE COARSE-LEVEL AND FINE-LEVEL MATCHING RESULTS

D-GLSNet			Fine-Level					Coarse-Level				
backbone	$[N_c, N_f]$	size(MB)	AMN	IR	R	FMR	Time(ms)	AMN	IR	R	FMR	Time(ms)
Small	[0,0]	75.18	41.52	56.28	20.42	75.22	32.06	32.12	44.33	15.81	72.28	31.00
	[2,1]	97.71	45.88	58.49	22.47	77.06	40.56	35.14	45.25	17.23	74.44	36.61
	[4,2]	120.25	48.65	57.78	23.71	75.49	48.24	37.25	44.92	18.18	73.15	41.75
Medium	[0,0]	154.95	44.12	58.31	21.62	76.66	37.11	33.94	45.48	16.66	75.76	36.14
	[2,1]	177.48	53.31	60.66	25.97	<u>78.83</u>	45.14	40.70	46.77	19.84	<u>76.87</u>	41.16
	[4,2]	200.02	<u>59.54</u>	**62.57**	<u>28.88</u>	**80.50**	55.14	<u>44.82</u>	<u>47.22</u>	<u>21.75</u>	**77.84**	48.19
Big	[0,0]	362.68	50.14	57.63	24.72	75.18	50.57	37.87	44.32	18.73	71.69	49.55
	[2,1]	385.22	55.93	58.60	27.48	74.93	57.36	42.80	45.43	21.03	72.69	53.43
	[4,2]	407.75	**62.15**	<u>61.73</u>	**30.49**	75.36	65.96	**47.13**	**47.38**	**23.16**	73.80	59.24

The best result is bold, and the underline is the second.

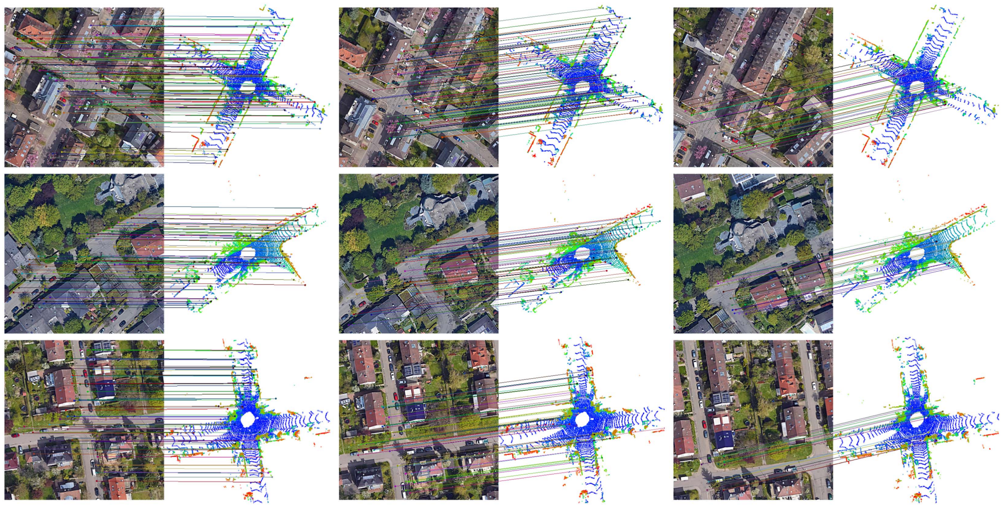

Fig. 4. Diagram depicts the matching of satellite maps and LiDAR point clouds, with lines indicating matching pairs. Results for three scenarios are shown from top to bottom, with image center shifts of 0 m, 20.78 m, and 41.58 m, and LiDAR positions varied accordingly. The last column's shift exceeds our training configuration's maximum.

We present qualitative results in Fig. 4. The figure illustrates that as the distance difference between the satellite map's center and the LiDAR's location increases from left to right, there are still good matches available although the number of matches gradually decreases. Certainly, we can incorporate larger local satellite images and random translation changes during training to achieve better matching, but this may lead to additional computational overhead. Our objective is to integrate matching networks with LiDAR SLAM. During normal LiDAR odometry operations, there will not have a large deviation within a certain period of time. Therefore, it is non-essential to adapt to situations where the distance difference between the center and the LiDAR position is large.

We assess the registration error of our method using three indicators: 1) Relative Rotation Error (RRE): measuring yaw distance from ground truth; 2) Relative Translation Error (RTE), measuring Euclidean distance between the $x - y$ vector and ground live conversion vector; 3) Registration Recall (RR), the percentage of image point cloud pairs with RRE and RTE below a specific threshold. Our D-GLSNet(Medium, [4,2]) model performed well on various KITTI sequences shown in Table II, with an average registration accuracy of 1.43 m. These results demonstrate our method's potential to replace consumer-grade GPS.

TABLE II
REGISTRATION RESULTS ON KITTI ODOMETRY BY THE MODEL D-GLSNET(MEDIUM, [4,2]). WE ENSURE THAT AT LEAST 15 INLINE MATCHES ARE CONSIDERED SUCCESSFUL DURING REGISTRATION

Sequence	RTE	RRE	RR($< 2m, < 5^\circ$)	RR($< 4m, < 10^\circ$)
00	0.79	1.49	96.48	99.07
02	1.46	3.59	86.41	98.32
04	2.09	1.47	69.57	95.65
05	1.44	5.85	91.51	96.75
06	2.10	7.61	79.09	94.27
07	0.77	0.92	98.90	99.45
08	1.98	8.21	79.85	94.71
09	1.01	2.45	91.76	99.34
10	1.24	1.54	81.15	99.65
Average	1.43	3.68	86.08	97.47

C. Localization Accuracy

To test our method's effectiveness, we combine two LiDAR odometry methods, A-LOAM and LIO-SAM, with our

TABLE III
RMSE(M) AND MEAN(M) OF APE RESULTS OF THE TWO-DIMENSIONAL POSES IN METERS ON THE KITTI DATASET. MISSING CELLS INDICATE SCENARIOS THAT WERE NOT SUCCESSFULLY TRACKED

Method	00	02	04	05	06	07	08	09	10
	rmse / mean	rmse / mean	rmse / mean	rmse / mean	rmse / mean	rmse / mean	rmse / mean	rmse / mean	rmse / mean
A-LOAM	15.17 / 11.83	-	1.64 / 1.03	10.69 / 8.63	8.34 / 6.18	1.35 / 1.24	31.32 / 26.20	16.73 / 13.59	7.51 / 6.76
A-LOAM + GPS	0.59 / 0.50	-	0.59 / 0.48	0.36 / 0.28	0.63 / 0.48	0.37 / 0.33	1.41 / 0.96	0.77 / 0.59	0.54 / 0.47
A-LOAM + GLN	0.96 / 0.87	-	0.67 / 0.54	0.84 / 0.77	0.88 / 0.69	0.56 / 0.52	1.53 / 1.26	1.22 / 1.10	1.13 / 0.91
LIO-SAM	-	44.41 / 36.37	0.67 / 0.91	11.09 / 9.06	8.89 / 6.54	1.00 / 0.94	37.53 / 30.91	18.04 / 14.40	6.33 / 5.60
LIO-SAM + GPS	-	2.10 / 1.12	0.35 / 0.24	0.34 / 0.26	0.53 / 0.43	0.29 / 0.25	1.41 / 0.92	0.74 / 0.57	1.13 / 0.75
LIO-SAM + GLN	-	1.84 / 1.72	0.38 / 0.28	0.91 / 0.84	0.91 / 0.73	0.47 / 0.44	1.47 / 1.21	1.30 / 1.16	1.19 / 0.93

TABLE IV
MEAN(M) OF APE RESULTS OF THE TWO-DIMENSIONAL POSES IN METERS ON THE KITTI DATASET. LOWER BOUNDS ARE GIVEN FOR METHODS THAT DID NOT REPORT EXACT RESULTS

Method	00	02	04	05	06	07	08	09	10
Brubaker et al. [39]	2.1	4.1	-	2.6	-	1.8	2.4	4.2	3.9
Floros et al. [40]	>10	>20	-	-	-	-	-	-	-
Yan et al. [31]	>10	-	-	>10	>10	>10	-	>10	>10
Miller et al. [6]	2.0	9.1	-	-	-	-	-	7.2	-
Kim et al. [8]	4.65	-	-	-	-	-	-	7.69	-
Zhu et al. [32]	-	-	-	-	-	-	4.45	-	-
Fervers et al. [9]	-	**1.42**	0.66	**0.77**	**0.57**	0.85	2.51	-	0.96
A-LOAM + GLN	**0.87**	-	0.54	**0.77**	0.69	0.52	1.26	**1.10**	**0.91**
LIO-SAM + GLN	-	1.72	**0.28**	0.84	0.73	**0.44**	**1.21**	1.16	0.93

Missing cells indicate scenes that are not evaluated or not tracked successfully. The best result is bold, and the underline is the second.

Geo-localization Node (GLN) III-F. LOAM is a LiDAR SLAM system divided into two threads: odometry at high frequencies and map-building at low frequencies. A-LOAM is based on the principles of LOAM and is openly available in code. LIO-SAM builds on LOAM by tightly coupling IMU measurements in the odometry section and adds GNSS adaptation as a global map optimization. We adjust LIO-SAM parameters to ensure stable operation on most sequences, as the original version of LIO-SAM degenerates on some KITTI dataset sequences. We conduct a real-time system test on the KITTI dataset, initializing the system with the first frame GPS of each sequence. We use GPS/IMU as the ground truth and measure the horizontal offset performance using APE (Absolute Position Error). We test each sequence six times and take the average.

We compare the results with the original odometry and those fused with GPS. To control variables, we assume that the noise level of the GLN and GPS is 1 m and maintains a frequency of approximately 1 Hz during trajectory optimization. The final results are displayed in Table III. As we use KITTI's GPS as ground truth, integrating GPS should theoretically provide the upper limit of accuracy we can achieve. The table reveals that the odometry has been quite precise in several short sequences. However, there are noticeable positioning errors at longer distances due to the impact of accumulated drift. Just like using GPS, our method can reduce cumulative errors, and already approximate the results of GPS on some sequences. Fig. 5 depicts that our approach can eliminate accumulated drift of the odometry, keeping the positioning error low and enhancing the system's overall robustness and stability.

We compare our method with previous work, as shown in Table IV. Our approach outperforms previous state-of-the-art methods on most sequences. Specifically, our method achieves the best results in 7 out of 9 sequences and performs well in the

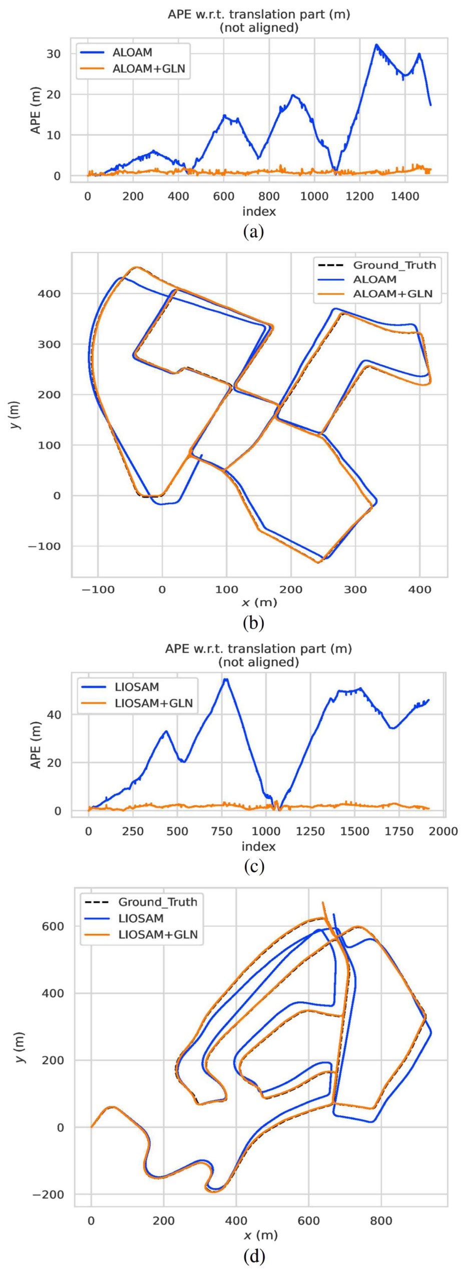

Fig. 5. Quantitative and qualitative results for Sequence 00 and 02 of KITTI. (a) Is the localization error comparison between the A-LOAM and A-LOAM + GLN on Sequence 00. (b) Is the estimated trajectories of the two methods. (c) Is the localization error comparison between the LIO-SAM and LIO-SAM + GLN on Sequence 02. (d) The estimated path of the two methods.

remaining two. Notably, our approach only requires LiDAR data, eliminating the need for ground image inputs such as [9], which makes our method less hardware-dependent and easier to use. Furthermore, unlike [6], which requires semantic segmentation of the map as a preprocessing step, our method only requires the original satellite image and LiDAR point cloud input. Our modular design also ensures the flexibility and universality of our implementation, enabling easy integration with any front-end odometry method, such as A-LOAM and LIO-SAM, as we demonstrated.

V. Conclusion and Future Work

We propose D-GLSNet, a novel 2D-3D matching network for registering LiDAR point clouds and satellite images. Our GLN can be integrated with SLAM to generate pseudo-GPS signals for odometric trajectory correction. We experiment extensively on KITTI to validate our D-GLSNet and the odometry combined with GLN, comparable to low-accuracy GPS and not limited by external disturbances such as signal strength, building blockage, etc. However, our method cannot work in some degraded scenes, such as dense woods and highways, and we will consider combining multiple sensors to solve the degraded scenes in our subsequent work.

References

[1] J. Zhang and S. Singh, "LOAM: Lidar odometry and mapping in real-time," in *Proc. Robot.: Sci. Syst.*, 2014, pp. 1–9.

[2] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "LIO-SAM: Tightly-coupled lidar inertial odometry via smoothing and mapping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 5135–5142.

[3] W. Lu, Y. Zhou, G. Wan, S. Hou, and S. Song, "L3-Net: Towards learning based LiDAR localization for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 6389–6398.

[4] E. Javanmardi, Y. Gu, M. Javanmardi, and S. Kamijo, "Autonomous vehicle self-localization based on abstract map and multi-channel LiDAR in urban area," *IATSS Res.*, vol. 43, no. 1, pp. 1–13, 2019.

[5] M. Karaim, M. Elsheikh, and A. Noureldin, "GNSS error sources," in *Multifunctional Operation and Application of GPS*. R. B. Rustamov and A. M. Hashimov, Eds. Rijeka, Croatia: IntechOpen, 2018, ch. 4, pp. 69–85, doi: 10.5772/intechopen.75493.

[6] I. D. Miller et al., "Any way you look at it: Semantic crossview localization and mapping with LiDAR," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 2397–2404, Apr. 2021.

[7] F. Castaldo, A. Zamir, R. Angst, F. Palmieri, and S. Savarese, "Semantic cross-view matching," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops*, 2015, pp. 9–17.

[8] D.-K. Kim and M. R. Walter, "Satellite image-based localization via learned embeddings," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2017, pp. 2073–2080.

[9] F. Fervers, S. Bullinger, C. Bodensteiner, M. Arens, and R. Stiefelhagen, "Continuous self-localization on aerial images using visual and lidar sensors," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 7028–7035.

[10] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231–1237, 2013.

[11] I. Rocco, R. Arandjelović, and J. Sivic, "Efficient neighbourhood consensus networks via submanifold sparse convolutions," in *Proc. 16th Eur. Conf. Comput. Vis.*, 2020, pp. 605–621.

[12] I. Rocco, M. Cimpoi, R. Arandjelović, A. Torii, T. Pajdla, and J. Sivic, "Neighbourhood consensus networks," in *Proc. 32nd Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 1658–1669.

[13] Y. Wang and J. M. Solomon, "Deep closest point: Learning representations for point cloud registration," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 3523–3532.

[14] H. Xu, S. Liu, G. Wang, G. Liu, and B. Zeng, "OMNet: Learning overlapping mask for partial-to-partial point cloud registration," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 3132–3141.

[15] Z. J. Yew and G. H. Lee, "RPM-Net: Robust point matching using learned features," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11824–11833.

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

[17] H. Deng, T. Birdal, and S. Ilic, "PPF-FoldNet: Unsupervised learning of rotation invariant 3D local descriptors," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 602–618.

[18] H. Deng, T. Birdal, and S. Ilic, "PPFNet: Global context aware local features for robust 3D point matching," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 195–205.

[19] J. Sun, Z. Shen, Y. Wang, H. Bao, and X. Zhou, "LoFTR: Detector-free local feature matching with transformers," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 8922–8931.

[20] Z. Qin, H. Yu, C. Wang, Y. Guo, Y. Peng, and K. Xu, "Geometric transformer for fast and robust point cloud registration," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 11143–11152.

[21] Y. Li, H. Su, C. R. Qi, N. Fish, D. Cohen-Or, and L. J. Guibas, "Joint embeddings of shapes and images via CNN image purification," *ACM Trans. Graph.*, vol. 34, no. 6, pp. 1–12, 2015.

[22] X. Xing, Y. Cai, T. Lu, S. Cai, Y. Yang, and D. Wen, "3DTNet: Learning local features using 2D and 3D cues," in *Proc. Int. Conf. 3D Vis.*, 2018, pp. 435–443.

[23] Q.-H. Pham, M. A. Uy, B.-S. Hua, D. T. Nguyen, G. Roig, and S.-K. Yeung, "LCD: Learned cross-domain descriptors for 2D-3D matching," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 11856–11864.

[24] M. Feng, S. Hu, M. H. Ang, and G. H. Lee, "2D3D-MatchNet: Learning to match keypoints across 2D image and 3D point cloud," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2019, pp. 4790–4796.

[25] B. Wang et al., "P2-Net: Joint description and detection of local features for pixel and point matching," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 16004–16013.

[26] X. Gao, S. Shen, Z. Hu, and Z. Wang, "Ground and aerial meta-data integration for localization and reconstruction: A review," *Pattern Recognit. Lett.*, vol. 127, pp. 202–214, 2019.

[27] Y. Shi, X. Yu, L. Liu, T. Zhang, and H. Li, "Optimal feature transport for cross-view image geo-localization," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 11990–11997.

[28] Y. Shi, X. Yu, D. Campbell, and H. Li, "Where am i looking at? Joint location and orientation estimation by cross-view matching," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 4064–4072.

[29] A. Gawel, C. D. Don, R. Siegwart, J. Nieto, and C. Cadena, "X-view: Graph-based semantic multi-view localization," *IEEE Robot. Automat. Lett.*, vol. 3, no. 3, pp. 1687–1694, Jul. 2018.

[30] F. Yan, O. Vysotska, and C. Stachniss, "Global localization on openstreetmap using 4-bit semantic descriptors," in *Proc. IEEE Eur. Conf. Mobile Robots*, 2019, pp. 1–7.

[31] M. Zhu, Y. Yang, W. Song, M. Wang, and M. Fu, "AGCV-LOAM: Air-ground cross-view based LiDAR odometry and mapping," in *Proc. Chin. Control Decis. Conf.*, 2020, pp. 5261–5266.

[32] A. Viswanathan, B. R. Pires, and D. Huber, "Vision-based robot localization across seasons and in remote locations," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2016, pp. 4815–4821.

[33] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 936–944.

[34] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. Guibas, "KPConv: Flexible and deformable convolution for point clouds," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 6410–6419.

[35] P.-E. Sarlin, D. DeTone, T. Malisiewicz, and A. Rabinovich, "SuperGlue: Learning feature matching with graph neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 4938–4947.

[36] Z. J. Yew and G. H. Lee, "REGTR: End-to-end point cloud correspondences with transformers," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 6677–6686.

[37] Y. Liao, J. Xie, and A. Geiger, "KITTI-360: Anovel dataset and benchmarks for urban scene understanding in 2D and 3D," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 3, pp. 3292–3310, Mar. 2023.

[38] M. A. Brubaker, A. Geiger, and R. Urtasun, "Lost! Leveraging the crowd for probabilistic visual self-localization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2013, pp. 3057–3064.

[39] G. Floros, B. van der Zander, and B. Leibe, "OpenStreetSLAM: Global vehicle localization using OpenStreetMaps," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2013, pp. 1054–1059.