InstaGraM: Instance-level Graph Modeling
for Vectorized HD Map Learning

Juyeb Shin[1]   Francois Rameau[2]   Hyeonjun Jeong[1]   Dongsuk Kum[1]

[1]KAIST   [2]SUNY Korea

[1]{juyebshin, hyeonjun.jeong, dskum}@kaist.ac.kr   [2]francois.rameau@sunykorea.ac.kr

Abstract

Inferring traffic object such as lane information is of foremost importance for deployment of autonomous driving. Previous approaches focus on offline construction of HD map inferred with GPS localization, which is insufficient for globally scalable autonomous driving. To alleviate these issues, we propose online HD map learning framework that detects HD map elements from onboard sensor observations. We represent the map elements as a graph; we propose InstaGraM, instance-level graph modeling of HD map that brings accurate and fast end-to-end vectorized HD map learning. Along with the graph modeling strategy, we propose end-to-end neural network composed of three stages: a unified BEV feature extraction, map graph component detection, and association via graph neural networks. Comprehensive experiments on public open dataset show that our proposed network outperforms previous models by up to 13.7 mAP with up to 33.8× faster computation time.

1. Introduction

Understanding static environment of the surroundings is one of the essential techniques for deploying autonomous driving. To automatically associate such type of information from a vehicle on the go, traditional works adopt offline generated global High Definition (HD) map along with the global pose information obtained from GPS equipped on the vehicle. Although its simple and straightforward solution, such traditional approaches require offline generated HD map prior which demands a large amount of resources in construction. To alleviate these issues, recent approaches focus on online detection of HD map elements from onboard sensor observations, named *online HD map detection* [8, 15, 21, 22, 24–26, 32–34, 44, 45].

Online HD map detection network associates onboard sensor observations such as camera and/or LiDAR, and predicts static components of the map. Seminal works [22, 32–34, 44, 45] formulate HD map detection as a raster-

ized segmentation problem, assigning occupancy score to rasterized bird's-eye-view (BEV) grid. While these approaches demonstrated their relevance, such representation remains memory intensive and lacks structural relationships desirable for downstream tasks such as trajectory planning and motion forecasting [12]. To this end, recent works attempt to reconstruct vectorized representation of the maps that are lighter and readily applicable to autonomous driving. [15, 21, 24–26] Despite their compelling performances, the aforementioned vectorized HD map networks remain too computationally demanding for a realistic deployment. In this context, we propose InstaGraM, Instance-level Graph Modeling for fast HD map construction model that predicts map elements as a set of structured polylines from a set of cameras mounted on a vehicle. To reach this goal, our end-to-end network is composed of three stages, as depicted in Figure 1. First, we extract the CNN feature maps from each image – captured by the camera rig – and aggregate them into a single top-down feature map using a 2D-to-BEV transformation network [21, 32]. Then, given this top-down feature map, we detect the road elements' vertex points and edge maps via CNNs. Finally, the vertices' positions and their respective local edge maps are passed to a Graph Neural Network (GNN) to predict their instance-level connections as an adjacency matrix trained in a supervised manner.

Our contributions can be summarized as:

- We propose a novel graph modeling for vectorized polylines of map elements that models geometric, semantic and instance-level information as graph representations.

- On top of the proposed graph modeling, we present InstaGraM, an end-to-end vectorized HD map learning network designed for real-time performance

2. Related Work

Road Detection and Segmentation. Overall, extracting high-level scene semantic information from onboard sen-
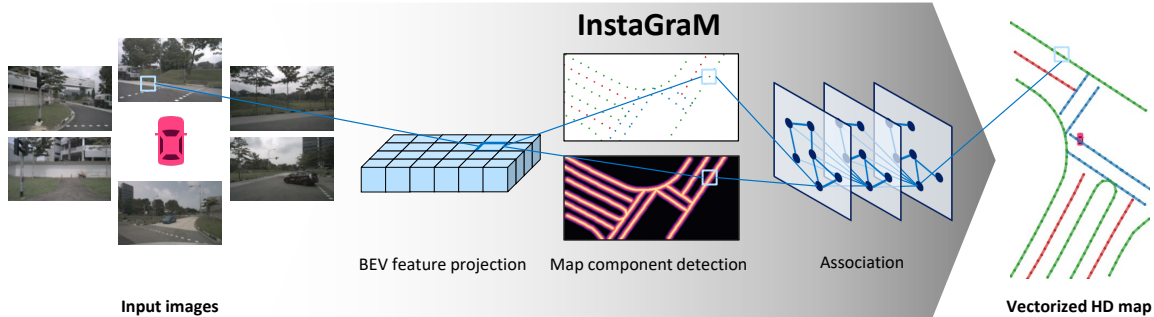
Figure 1. We propose InstaGraM, a hybrid architecture of CNNs and a GNN for real-time HD map learning in bird's-eye-view representation. Starting from the input surround images and camera parameters, a unified BEV representation is extracted by projecting and fusing image features. InstaGraM extracts vertex locations and implicit edge maps of map elements, and final vectorized HD map elements are generated throughout a GNN.

sors has always been of high interest for autonomous driving. In particular, the detection [1, 20] and segmentation [16] of road elements (e.g., lanes, road marks, and road signs) have been highly valuable for a large spectrum of tasks, including localization [2,13], lane keeping [31], autoparking [17], and much more. To achieve this goal, early algorithms for lane detection rely on hand-crafted binary segmentation followed by lines or curves fitting strategies [3]. While such approaches are often fast, they remain brittle to adverse conditions (e.g., lighting, shadow, road wear) and are limited to lane detection. For these reasons, these methods have been progressively substituted by deep learning-based strategies offering more flexibility and robustness to lane and road markings segmentation [23]. While the techniques mentioned above provide rich information from the camera's viewpoint, they do not contain 3D information about the observed elements in the scene.

Bird's-eye-view Map Segmentation. To offer a more comprehensive 3D understanding of the vehicle's surroundings, recent studies demonstrate that BEV semantic segmentation is particularly well suited for autonomous driving [29]. When only road markings are considered, such representation can easily be obtained via an Inverse Perspective Mapping (IPM) of the image-level segmentation results [18] – assuming known camera's intrinsics, elevation, and tilt. This IPM warping is valid under the planar assumption but is violated for any object above the ground level (e.g., cars and pedestrians) leading to severe perspective distortion in the resulting BEV. To avoid such stretching effect, [6, 27] predict the vehicles foot-print to respect the planar assumption. To avoid using dedicated cars' footprint datasets, Cam2Bev [35] attempts to correct this distortion directly from warped segmentation masks and [47] proposes a GAN-based approach to directly transform front facing views into BEV on which the segmentation can be performed.

Alternatively, to deal with uneven road surfaces and non-flat objects, some approaches utilize depth information to warp the segmentation results adaptively [39]. Following a similar philosophy, [34] proposes to use depth information to combine the CNN features into the BEV space. Thus the segmentation can be compute directly in this representation, allowing a better integration of multi-camera system. Homographic or depth-aware warping strategies have the advantage of being intuitive, interpretable and to offer good transferability to various camera setups. Despite these advantages, geometric warping solutions face multiple limitations: they rely on strong prior, may suffer from perspective distortions and require successive stages. To circumvent these limitations, another solution consists in using neural networks to learn the image(s) to BEV transformation implicitly. One of the pioneering work employing this strategy is VED [28] which directly employ a variational autoencoder to predict the BEV from an input image without intermediate stages.

To better preserve spatial information and to ease the integration of cross-view information, follow up works rely on more interpretable and elegant approaches to map the transformation between the features in the camera view and the BEV. One of the seminal work is [32] where a Multilayer Perceptron (MLP) is used to learn this mapping. After the image features from multiple views are mapped onto a unified BEV, the segmentation can be learned into this final representation. This approach combines multiple advantages. Unlike its IPM counterpart, it does not require any prior calibration and it is not affected by perspective distortion (global receptive field). As a result, this strategy has influenced numerous works proposing various improvements such as multi-resolution features [36, 37] and learnable bi-directional projection [21]. More recently, to provide more expressive and data dependent mapping, the use of transformer networks has grown [33,44]. The problem of these approaches is their high memory requirement, to alleviate this issue recent works adopt deformable transformer

networks [8, 22, 45].

Vectorized HD Map Detection. The previously introduced literatures [22, 32–34, 44, 45] predict map elements in a rasterized BEV space. The downside of this representation is its lack of structural relations and instance-level information. In order to provide a lighter and more suitable representation for self-driving related downstream tasks [12], recent works [15, 21, 24–26] propose estimating the vectorized HD map elements instead of a segmentation map. InstaGraM belongs to this category. A representative work is HDMapNet [21] which generates a vectorized HD map by post-processing various BEV segmentation maps. Despite promising results, the heuristic post-processing requires large amount of computations. In order to predict a vectorized map in an end-to-end manner, VectorMapNet [26] proposes two successive transformer decoders; the first decoder detects map elements via cross-attention between the BEV feature and element queries while the second transformer adopts auto-regressive decoder to recurrently generate polylines. However, detection from element queries with cross-attention is known for its slow convergence, thus requires longer train epochs [7, 46]. Auto-regressive decoder in polyline generator of VectorMapNet makes its computation heavy, which is not applicable for real-time autonomous driving tasks. In contrast, our proposed architecture does not require large amount of training time nor heavy computation of recurrent model.

A work sharing similarities with our approach is Poly-World [48] which predicts the buildings' contours as a set of polygons from satellite images. Similarly to our strategy, this technique uses a CNN for vertex detection followed by a GNN for association. In contrast with PolyWorld, we adopt the interest point decoder from [10] predicting high-resolution vertices' coordinates. Furthermore, we leverage the distance transform embedding for implicit directional information between vertices to associate. Finally, our strategy is designed for road elements detection requiring both semantic and instance segmentation information.

3. Method

We propose an end-to-end network to compute a BEV vectorized HD map from a set of cameras mounted on a vehicle. To represent road elements (*i.e.*, lane dividers, pedestrian crossing, and road boundaries), HD maps typically consist of 2D polyline vertices and their instance-level adjacency connectivity. To obtain this vector representation, previous works rely on segmentation prediction and heavy post-processing [21], or auto-regressive models [26] known for their high computational cost. In contrast, we propose a lighter pipeline based on a combination of CNNs and a GNN able to predict a set of vertices and their adjacency directly. Our method is three-folded. First, similarly to HDMapNet [21], we utilize an MLP-based approach to

build a unified BEV features map from the CNN features extracted from each image captured from the camera rig – via an EfficientNet [41]. From this BEV feature map, two CNN decoders extract the vertices and edge maps of the observed road elements. Finally, these vertices and their local edge response are fed to an attentional GNN in order to learn the semantic class and the connection between the vertices.

3.1. Neural View Transform

The very first stage of our HD map estimation network is extraction of the top-down BEV features map $\mathcal{F}_{bev}$ by combining the CNN features [41] from each images captured by the camera rig at a given time. The proposed InstaGraM can be adopted to any existing BEV feature transform method. We demonstrate scalability of InstaGraM by reporting experiments with various BEV transformation method in section 4.3.

3.2. Element Detector Heads

From the top-down feature map $\mathcal{F}_{bev}$, we extract the *vertices* and *edges* of the HD map elements using two CNN decoders $\phi_{\mathcal{V}}, \phi_{\mathcal{E}}$ respectively. These two components are predicted in the rasterized BEV space $\mathbb{R}^{W_{bev} \times H_{bev}}$ similar to segmentation tasks. The vertex decoder $\phi_{\mathcal{V}}$ adopts the interest point decoder from [10] and extracts possible position heatmap at every 8×8 local, non-overlapping grid in BEV pixels. It computes $\mathcal{X} \in \mathbb{R}^{\frac{W_{bev}}{8} \times \frac{H_{bev}}{8} \times 65}$, the 65 channels indicating possible position in the local grids with an additional "no vertex" dustbin. After a channel-wise softmax, the dustbin dimension is removed and the vertex heatmap is reshaped from $\mathbb{R}^{W_c \times H_c \times 64}$ to $\mathbb{R}^{W_{bev} \times H_{bev}}$. In parallel with the vertex decoder, the edge map decoder $\phi_{\mathcal{E}}$ predicts the distance transform map $\mathcal{D} \in \mathbb{R}^{W_{bev} \times H_{bev} \times 3}$, the 3 channels indicating the number of class categories of map elements. This edge map of distance transform [4] implicitly provides spatial relations between vertices and directional information of map elements inspired from [15, 24, 25]. We further demonstrate in section 4 that this distance transform representation as an edge map plays a significant role in instance-level association. We apply ReLU and a threshold after the last Conv layer to predict the distance values from 0 to 10 in the rasterized BEV image.

3.3. Association via Graph Neural Network

The two components extracted from element detector heads are associated via a graph neural network, where all vertices interact throughout an attention scheme [11, 42]. This allows our network to reason about both point-level and instance-level relations between map elements based on various attributes including positions, implicit edge map of distance values and class categories.
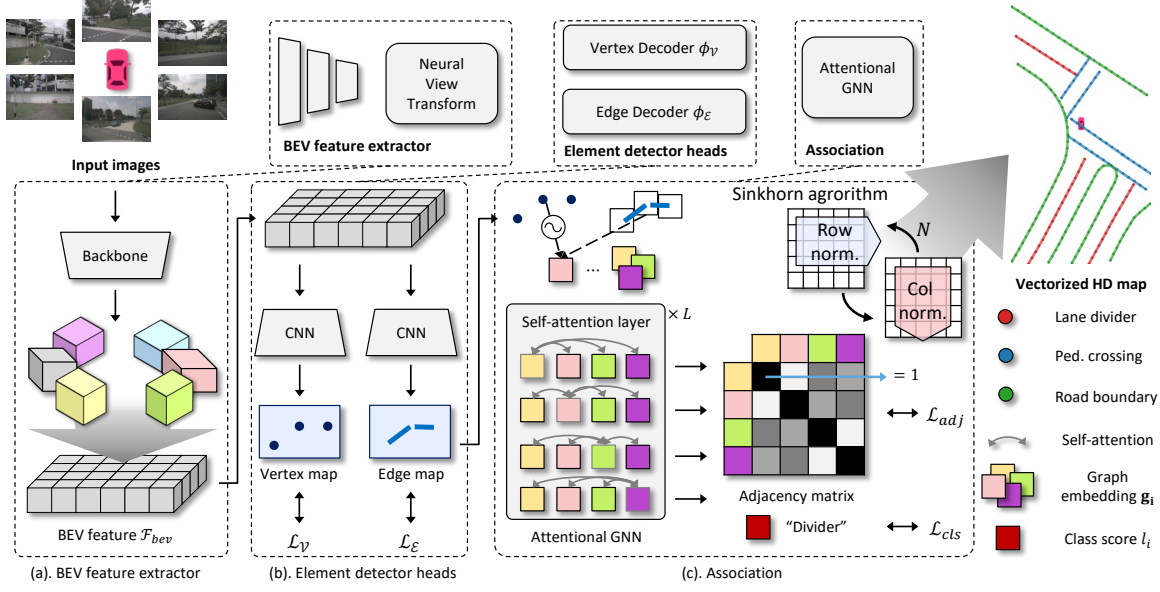
Figure 2. Proposed InstaGraM architecture. The blocks at the top show the overall components of InstaGraM architecture and the bottom blocks show the details of structure and training of each component.

Graph Embeddings: We combine vertex positions and distance transform maps to form initial graph embeddings. We first extract the position of each vertex in rasterized BEV coordinate and their respective confidence from channel-wise softmax in the vertex position heatmap, $\mathbf{v}_i = (x_i, y_i, c)$. We only extract one distinctive vertex position with maximum confidence in each 8×8 grid cell, which is acting similar to Non-Maximum Suppression. After extraction, a ith vertex position $\mathbf{v}_i$ is encoded by a sinusodial positional encoding function γ to augment it into a high-dimensional vector [30]. This positional encoding is further supported by an additional shallow MLP. To complement the positional information of the vertex $\mathbf{v}_i$, we additionally include the local directional information as the embedding of the distance transform patch corresponding to the same grid cell. Then our initial graph consists of D-dimensional embeddings $^0\mathbf{g}_i \in \mathbb{R}^D$, combining both the vertex position and its directional local information, can be formulated as:

$$^{(0)}\mathbf{g}_i = \text{MLP}_{\mathcal{V}}(\gamma(\mathbf{v}_i)) + \text{MLP}_{\mathcal{E}}(d_i). \tag{1}$$

This enables us to associate multiple graph embeddings based on their vertex and edge representation throughout an attention scheme.

Attentional Message Passing from SuperGlue [38]: We start from an initial graph $^{(0)}\mathcal{G}$ with nodes containing both vertex position and edge map embeddings as a high-dimensional vector. This initial graph has bidirectional edges, connecting vertex i to all other vertices. To further enhance the nodes and find the final edges of the ver-

tices, we pass the initial graph to the attentional graph neural network and propagate this graph through message passing [38, 43]. Our objective is to find final bidirectional edges of the vertices as an instance-level information of map elements. We feed our initial graph to attentional graph neural network that aggregates graph embeddings via a message passing consists of MLP and Multi-head Self Attention (MSA):

$$^{(0)}\mathcal{G} = [^{(0)}\mathbf{g}_1; ^{(0)}\mathbf{g}_2; \ldots; ^{(0)}\mathbf{g}_{\mathcal{N}}]$$
$$^{(l)}\mathcal{G} = {}^{(l-1)}\mathcal{G} + \text{MLP}([^{(l-1)}\mathcal{G}\|\text{MSA}(^{(l-1)}\mathcal{G})]), l = 1, \ldots, L \tag{2}$$

Self-attention and aggregation in Equation 2 provides interaction between all the graph embeddings based on their spatial and directional appearance embedded in $\mathbf{g}_1$. Concretely, each vertex node attend to all other nodes to find the next possible vertices that would appear in the map. After L layers of attentional aggregation, class scores $\mathbf{l}_i \in \mathbb{R}^3$ and graph matching embeddings $\mathbf{f}_i \in \mathbb{R}^D$ are obtained:

$$\mathbf{l}_i = \text{MLP}_{cls}(^{(L)}\mathbf{g}_i)$$
$$\mathbf{f}_i = \text{MLP}_{match}(^{(L)}\mathbf{g}_i) \tag{3}$$

Adjacency Matrix: We predict optimal edges by computing score matrix $\hat{S} \in \mathbb{R}^{\mathcal{N} \times \mathcal{N}}$ between nodes of the graph $^{(L)}\mathcal{G}$. The adjacency score between nodes i and j can be computed as cosine similarity of embedding vectors

$$\hat{\mathbf{S}}_{ij} = <\mathbf{f}_i, \mathbf{f}_j>, \forall\{i, j\} \in \mathcal{N} \times \mathcal{N}, \tag{4}$$

where $< \cdot, \cdot >$ is an inner product of two embeddings. Following SuperGlue, we augment this score matrix to
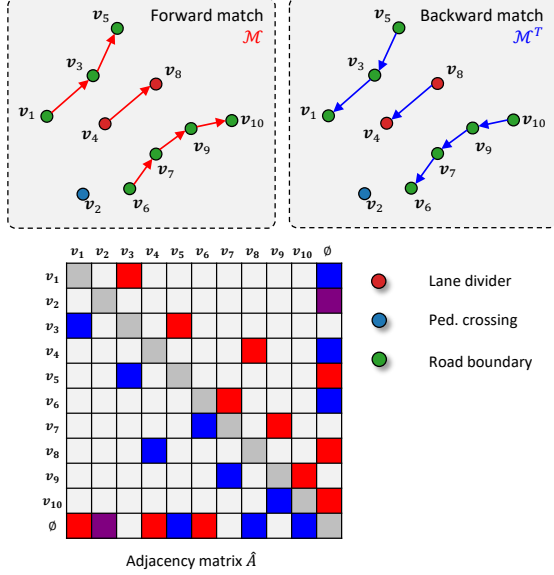
4

Figure 3. We model polylines of map elements by a graph with bidirectional edges. Attentional graph neural network and optimal matching in InstaGraM computes symmetric adjacency matrix $\hat{A} \in \mathbb{R}^{(\mathcal{N}+1)\times(\mathcal{N}+1)}$ with both forward and backward connections. With augmented dustbin vertex $\varnothing$, adjacency matrix provides instance-wise prediction of the map elements. Note that for vertex that does not have any matched vertex (i.e. $\mathbf{v}_2$ above) only has connection to dustbin vertex in both forward and backward, thus colored in purple.

$\bar{\mathbf{S}} \in \mathbb{R}^{(\mathcal{N}+1)\times(\mathcal{N}+1)}$ with dustbin node for vertices that might not have any match, *i.e.* a vertex at the end of an element instance. The Sinkhorn algorithm [9, 40] that iteratively normalizes $\exp(\bar{\mathbf{S}})$ along rows and columns is used to compute final adjacency matrix of the graph. Adjacency matrix $\hat{\mathbf{A}} \in \mathbb{R}^{(\mathcal{N}+1)\times(\mathcal{N}+1)}$ with instance-level edges can be computed throughout this optimal matching with augmented score $\hat{\mathbf{S}}$.

3.4. Losses

We design the whole network to be differentiable, allowing us to train it in a fully supervised manner with combinations of losses at multiple branches. For supervision of element detector heads, cross-entropy with softmax loss and L2 loss are used for vertex location heatmap and distance transform map respectively.

$$\mathcal{L}_{\mathcal{V}}(\mathcal{X}, \mathcal{Y}) = \frac{1}{H_c, W_c} \sum_{h=1, w=1}^{H_c, W_c} l_p(\mathbf{x}_{hw}; y_{hw})$$
$$\mathcal{L}_{\mathcal{E}}(\hat{\mathcal{D}}, \mathcal{D}) = \frac{1}{N} \sum_{d_p \in \mathcal{D}} \|d_p - \hat{d}_p\|^2, \tag{5}$$

where $H_c = \frac{H_{bev}}{8}$ and $W_c = \frac{W_{bev}}{8}$ are the indexing dimension of 8×8 local cells.

The coordinates from the vertex location heatmap prediction may not align perfectly to the ground truth vertex coordinates, specifically in the early stage of training, resulting in ambiguity of the ground truth adjacency and class label. To address this we find the nearest pairs between the ground truth vertices and predicted vertices to provide the ground truth for the output of the graph neural network, adjacency matrix and class predictions. The nearest ground truth vertex $\sigma(i)$ to the predicted vertex i is obtained that minimizes the Chamfer distance cost with threshold D_0:

$$\sigma = \underset{D(\mathbf{v}_i, \mathbf{v}_{\sigma(i)}) < D_0}{\arg\min} \sum_i^{\mathcal{N}} D(\mathbf{v}_i, \mathbf{v}_{\sigma(i)}) \tag{6}$$

From the matching pairs $[\mathbf{v}_i, \mathbf{v}_{\sigma(i)}]$, the ground truth adjacency pairs $\mathcal{M} = \{(i,j)\} \in \mathbb{1}^{\mathcal{N}\times(\mathcal{N}+1)}$ between the vertices i and j are obtained by observing connection between the ground truth vertices $\sigma(i)$ and $\sigma(j)$. The predicted vertices that do not have the ground truth pair within σ fall into a dustbin vertex $\varnothing$ as ground truth adjacency (i.e. $\mathbf{v}_2$ in Figure 3). Since the vectorized representation of the map has bidirectional edges, we compute negative log-likelihood of adjacency loss for both forward and backward direction $\mathcal{M} = \{(i,j)\} \in \mathbb{1}^{\mathcal{N}\times(\mathcal{N}+1)}$ and $\mathcal{M}^T = \{(j,i)\} \in \mathbb{1}^{\mathcal{N}\times(\mathcal{N}+1)}$.

$$\mathcal{L}_{adj} = -\frac{1}{2}\left(\sum_{(i,j)\in\mathcal{M}} \log \bar{\mathbf{A}}_{ij} + \sum_{(i,j)\in\mathcal{M}^T} \log \bar{\mathbf{A}}_{ij} \right). \tag{7}$$

We further supervise the graph neural network with negative log-likelihood for vertex classification. Through this supervision, our graph neural network can reason about the vertex label categories in addition to the instance-level information.

$$\mathcal{L}_{cls} = \sum_i^{\mathcal{N}} \log l_{\sigma(i)} \tag{8}$$

Combining above losses, our final loss function is formulated as:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{\mathcal{V}} + \lambda_2 \mathcal{L}_{\mathcal{E}} + \lambda_3 \mathcal{L}_{adj} + \lambda_4 \mathcal{L}_{cls}, \tag{9}$$

where we set $\lambda_3, \lambda_4, \ll \lambda_1, \lambda_2$ to ensure the graph neural network have enough vertex predictions to associate, specifically during the early stage of training.

4. Experiments

4.1. Dataset and Evaluation

We train and evaluate our network on the nuScenes dataset [5], which consists of 1000 scenes of 20 seconds

Method	Modality	Backbone	Epochs	$AP_{divider}$	AP_{ped}	$AP_{boundary}$	mAP	FPS
HDMapNet	L	PP	30	24.1	10.4	37.9	24.1	-
HDMapNet	C+L	EffiNet-B0 + PP	30	29.6	16.3	46.7	31.0	-
HDMapNet	C	EffiNet-B0	**30**	21.7	14.4	33.0	23.0	0.6
InstaGraM	C	EffiNet-B0	**30**	**40.8**	**30.0**	**39.2**	**36.7**	**20.3**
VectorMapNet	L	PP	110	37.6	25.7	38.6	34.0	-
VectorMapNet	C+L	R-50 + PP	110	50.5	37.6	47.5	45.2	-
VectorMapNet	C	R-50	110	**47.3**	**36.1**	39.3	40.9	3.0
InstaGraM	C	EffiNet-B4	**30**	47.2	33.8	**44.0**	**41.7**	**18.2**

Table 1. Evaluation results on nuScenes dataset. We only experiment and compare models with camera input modality, denoted as C. LiDAR L and camera-LiDAR fusion C+L modality of the baselines are shown for reference. PP, EffiNet and R denote PointPillars [19], EfficientNet and ResNet, respectively. For VectorMapNet, results with single stage training are used for fair comparison. FPS of baseline models are provided by VectorMapNet authors. We measure FPS of InstaGraM on RTX 3090 GPU, with batch size 1 and GPU warm-up.

captured from a vehicle equipped with cameras, LIDARs, IMU, and GPS. Aside from sensory data, this dataset also has the specificity to provide fully annotated HD maps. From each sequence in the dataset, keyframes containing: the pose in the global HD map, 3D point clouds, and images are extracted at 2Hz. For our application, we solely use the images captured by the six surrounding cameras mounted on the vehicle and the HD map. In order to train our pipeline with the local HD map expressed in the vehicle's referential, we use the data processing scheme proposed in [21]. From this data, three kinds of road elements are kept in the sampled local HD maps: lane dividers, pedestrian crossings, and road boundaries. Additionally, to train the intermediate representation of our network, we generate the ground truth vertex map and distance transform map from the vectorized map. Regarding the ground-truth vertices generation, we follow [10] by subsampling one vertex for each non-overlapping 8×8 grid region in the BEV image. We use the OpenCV library to generate the distance transform map within a distance range $[0, 10]$. For a fair comparison with the baselines, we adopt the Average Precision (AP) of instance prediction as the evaluation metric with the following Chamfer distance thresholds $\{0.5, 1.0, 1.5\}$.

4.2. Implementation Details

Our model is trained and evaluated using PyTorch, on RTX 3090 GPUs with batch size 32. We define the perception range $[-15.0m, 15.0m]$ along the X-axis and $[-30.0m, 30.0m]$ along the Y-axis in the vehicle's referential. We rasterize this range with resolution $0.15m$ resulting in $H_{bev} = 200$ and $W_{bev} = 400$ as output of the element detector heads. We adopt EfficientNet [41] as backbone. Neural view transform maps 6 image features to a BEV feature map $\mathcal{F}_{bev} \in \mathbb{R}^{\frac{W_{bev}}{2} \times \frac{H_{bev}}{2} \times 256}$. The element detector head utilizes 3 Residual blocks [14] and upsampling to predict the vertex location heatmap and the distance transform map. We extract the top $\mathcal{N} = 400$ vertices of confidences higher than threshold 0.01 from softmax-ed vertex location

InstaGraM	DT	PE	$AP_{divider}$	AP_{ped}	$AP_{boundary}$	mAP
A	✓	✗	8.6	8.5	3.8	7.0
B	✗	✓	30.0	10.2	22.5	20.9
C	✓	✓	47.2	33.8	44.0	41.7

Table 2. Ablation studies of the graph embeddings **g** design choices. DT, PE denote the distance transform embedding and PE the positional embedding, respectively. For embedding with no PE, we only take the distance transform embedding, and vice versa for embedding with no DT.

heatmap. The local distance transform patch of 8×8 grid is extracted that corresponds to vertex $\mathbf{v}_i$. For each detected vertex, their respective local distance transform patch of size 8×8 is extracted to build the graph embeddings. If the number of extracted vertices is inferior to $\mathcal{N}$, we pad the embeddings with zeros and adopt a mask operation within the graph neural network and the Sinkhorn iterations. We force the diagonal entities of the score matrix $\hat{S}$ to zeros before it is fed into the Sinkhorn algorithm – since the map elements do not have "self-loop". For training, we set the loss scales $\lambda_1, \lambda_2, \lambda_3$ and λ_4 as $1.0, 1.0, 5e^{-3}$ and $1e^{-2}$ respectively. During inference, we follow the flow of vertex connection in adjacency matrix to produce final polylines of the map elements.

4.3. Results

Comparison with baseline methods. We compare our methods against the state-of-the-art vectorized HD map learning pipelines – HDMapNet [21] and VectorMapNet [26]. HDMapNet predicts 3 types of segmentation map – semantic segmentation, instance embedding, and direction prediction – and utilizes heuristic post-processing to generate vectorized map elements. VectorMapNet utilizes transformer decoder from DETR [7] to predict the map elements, and autoregressive transformer decoder to generate detailed geometrical shape of the map elements. We train and evaluate InstaGraM with EfficientNet-B0 and
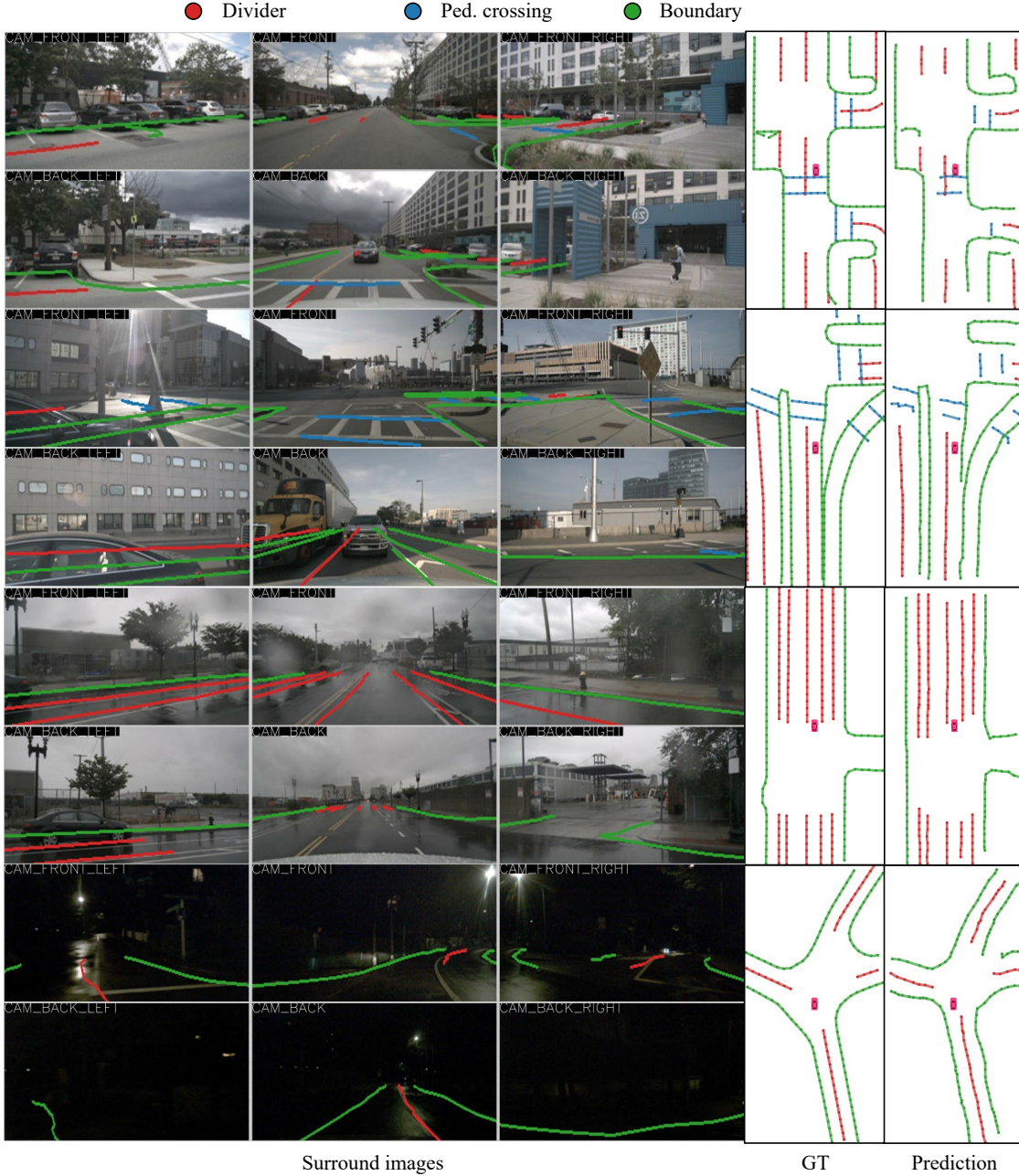
Figure 4. Qualitative results of InstaGraM on complex traffic scenes under various conditions. Beginning from the top row, cloud, sun, rain and night conditions.

EfficientNet-B4, which are comparable backbones with HDMapNet and VectorMapNet (ResNet-50) respectively. Table 1 shows that InstaGraM acheives 13.7 higher mAP with 33× faster inference speed compared to HDMapNet, and 0.6 higher mAP with 6× faster inference speed compared to VectorMapNet, under the camera modality. In both baselines, LiDAR modality provides explicit 3D location of the road boundary, resulting in improved AP. However,

our model under camera modality outperforms HDMapNet and VectorMapNet under the LiDAR modality especially observing on the boundary class. We observe that the distance transform provides strong spatial relations to locate the map elements. Furthermore, specifically in comparison with VectorMapNet, InstaGraM achieves competitive performance with significantly faster convergence time in training (train epochs of 110 for VectorMapNet and 30 for

Method	mAP	FPS	Param.
Warping (IPM)	6.1	15.9	30.7M
Depth (LSS [34])	41.9	15.4	30.7M
MLP (HDMapNet [21])	41.7	18.2	95.5M

Table 3. Ablation studies on BEV transform methods.

# layers	$AP_{divider}$	AP_{ped}	$AP_{boundary}$	mAP	Param.
0 (no GNN)	44.4	29.3	36.6	36.8	90.9
3	45.5	32.7	43.7	40.6	92.9
5	45.9	34.0	42.3	40.7	94.2
7	47.2	33.8	44.0	41.7	95.5
9	47.6	33.4	44.8	41.9	96.8
11	47.7	34.0	44.2	42.0	98.2

Table 4. Ablation studies on the number of GNN layers.

InstaGraM).

Qualitative Results. We visualize the vectorized HD map predictions in Figure 4. It demonstrates that InstaGraM generalizes well under various weather conditions. InstaGraM computes, in an end-to-end manner without post-processing nor heavy computation, the semantic and instance-level information of the complex map elements. Our graph modeling strategy shown in Figure 3 enables accurate and fast prediction of the map elements. The distance transform and positional embedding provides precise primitives to associate through the graph neural network, able to predict various curve shapes, specifically in the first and second row of Figure 3. The top 3 rows further show that InstaGraM is capable of uniformly detecting vertices and connections of occluded map elements.

Graph Embeddings. We conduct ablation studies for design choices of the graph embeddings as shown in Table 2. We find that from A and C in Table 2, positional embedding plays significant role since the distance transform distribution appear the same in overall vertices (see Figure 1). It provides the graph neural network where to look for nearby vertices to associate. The distance transform embedding further improves by providing strong directional information of the vertices (see B and C in Table 2).

Ablation Studies. We further conduct ablation studies with different BEV transformation methods and layers of our graph neural network. Proposed InstaGraM can be adapted several BEV transformation methods as shown in Table 3. Table 4 further analyzes affects of the number of attention layers in our graph neural network. Increasing the number of attention layers improves the accuracy of our model with saturation at 7 layers.

Distance Transform. We further analyze the distance transform embedding by comparing with visual descriptor embedding similar to the embeddings used in PolyWorld [48]. PolyWorld extracts a vector from image feature map corresponding to the predicted vertex coordinate, and
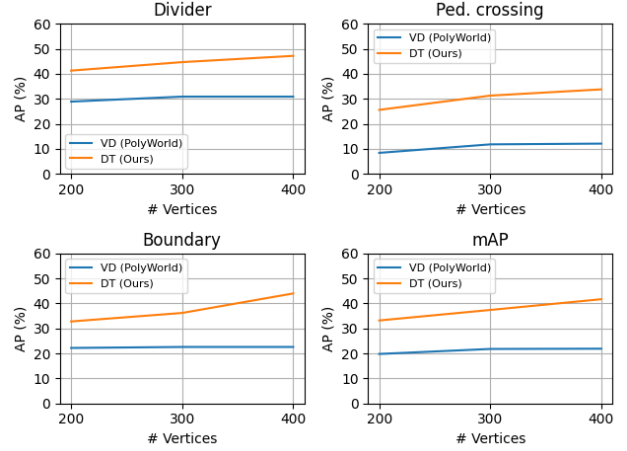

Figure 5. We compare InstaGraM with two different embeddings: the distance transform embeddings and the visual descriptor embeddings extracted from feature map. VD denotes that the visual descriptor is used for graph embeddings similar to PolyWorld [48]. We compare two embeddings over various number of vertices extracted from vertex location heatmap.

refer to it as "visual descriptor". Since the spatial size of the visual BEV feature map does not match the spatial size of the vertex heatmap passed by the interest point decoder, we interpolate the feature map to align the spatial size. We take the visual descriptor from the interpolated feature map corresponding to the vertex coordinate, then encode to form the graph embedding. Figure 5 compares the performance of InstaGraM with different graph embeddings — one with the distance transform embeddings (DT) and one with the visual descriptor embeddings (VD) — over various number $\mathcal{N}$ of vertices extraction. Positional embeddings are adopted in both setups. Although visual descriptor from direct feature map contains high-level semantic information of the scene, the distance transform provides strong prior of the map elements, resulting in better performance. We further emphasize that by adding auxiliary task of the distance transform regression, it implicitly provides the network the geometry of the ground truth map, outperforming baselines with fast convergence.

5. Conclusion

We propose InstaGraM, an end-to-end vectorized HD map learning pipeline that is applicable for real-time autonomous driving. We model the polylines of HD map elements as a graph with vertices and adjacency matrix that composes instance-level edges. Compared to previous works that require heuristic post-processing or a large amount of computational cost, InstaGraM computes in real-time polylines of the map elements achieving better performance compared to the state-of-the-art method.

Acknowledgement

References

[1] Oleksandr Bailo, Seokju Lee, Francois Rameau, Jae Shin Yoon, and In So Kweon. Robust road marking detection and recognition using density-based grouping and machine learning techniques. In *2017 IEEE winter conference on applications of computer vision (WACV)*, pages 760–768. IEEE, 2017. 2

[2] Oleksandr Bailo, Francois Rameau, and In So Kweon. Lightweight place recognition and loop detection using road markings. *arXiv preprint arXiv:1710.07434*, 2017. 2

[3] Aharon Bar Hillel, Ronen Lerner, Dan Levi, and Guy Raz. Recent progress in road and lane detection: a survey. *Machine vision and applications*, 25(3):727–745, 2014. 2

[4] Gunilla Borgefors. Distance transformations in digital images. *Computer Vision, Graphics, and Image Processing*, 34(3):344–371, 1986. 3

[5] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *CVPR*, 2020. 5

[6] Yigit Baran Can, Alexander Liniger, Ozan Unal, Danda Paudel, and Luc Van Gool. Understanding bird's-eye view of road semantics using an onboard camera. *IEEE Robotics and Automation Letters*, 7(2):3302–3309, 2022. 2

[7] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020. 3, 6

[8] Shaoyu Chen, Tianheng Cheng, Xinggang Wang, Wenming Meng, Qian Zhang, and Wenyu Liu. Efficient and robust 2d-to-bev representation learning via geometry-guided kernel transformer. *arXiv preprint arXiv:2206.04584*, 2022. 1, 3

[9] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26, 2013. 5

[10] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 224–236, 2018. 3, 6

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 3

[12] Jiyang Gao, Chen Sun, Hang Zhao, Yi Shen, Dragomir Anguelov, Congcong Li, and Cordelia Schmid. Vectornet: Encoding hd maps and agent dynamics from vectorized representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11525–11533, 2020. 1, 3

[13] Chengcheng Guo, Minjie Lin, Heyang Guo, Pengpeng Liang, and Erkang Cheng. Coarse-to-fine semantic localization with hd map for autonomous driving in structural scenes. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1146–1153. IEEE, 2021. 2

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 6

[15] Namdar Homayounfar, Wei-Chiu Ma, Justin Liang, Xinyu Wu, Jack Fan, and Raquel Urtasun. Dagmapper: Learning to map by discovering lane topology. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2911–2920, 2019. 1, 3

[16] Yuenan Hou, Zheng Ma, Chunxiao Liu, Tak-Wai Hui, and Chen Change Loy. Inter-region affinity distillation for road marking segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12486–12495, 2020. 2

[17] Chulhoon Jang and Myoungho Sunwoo. Semantic segmentation-based parking space detection with standalone around view monitoring system. *Machine Vision and Applications*, 30(2):309–319, 2019. 2

[18] Wonje Jang, Junhyuk Hyun, Jhonghyun An, Minho Cho, and Euntai Kim. A lane-level road marking map using a monocular camera. *IEEE/CAA Journal of Automatica Sinica*, 9(1):187–204, 2021. 2

[19] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12697–12705, 2019. 6

[20] Seokju Lee, Junsik Kim, Jae Shin Yoon, Seunghak Shin, Oleksandr Bailo, Namil Kim, Tae-Hee Lee, Hyun Seok Hong, Seung-Hoon Han, and In So Kweon. Vpgnet: Vanishing point guided network for lane and road marking detection and recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 1947–1955, 2017. 2

[21] Qi Li, Yue Wang, Yilun Wang, and Hang Zhao. Hdmapnet: An online hd map construction and evaluation framework. *ArXiv*, abs/2107.06307, 2021. 1, 2, 3, 6, 8

[22] Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Qiao Yu, and Jifeng Dai. Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. *arXiv preprint arXiv:2203.17270*, 2022. 1, 3

[23] Dun Liang, Yuan-Chen Guo, Shao-Kui Zhang, Tai-Jiang Mu, and Xiaolei Huang. Lane detection: a survey with new results. *Journal of Computer Science and Technology*, 35(3):493–505, 2020. 2

[24] Justin Liang, Namdar Homayounfar, Wei-Chiu Ma, Shenlong Wang, and Raquel Urtasun. Convolutional recurrent network for road boundary extraction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9512–9521, 2019. 1, 3

[25] Justin Liang and Raquel Urtasun. End-to-end deep structured models for drawing crosswalks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 396–412, 2018. 1, 3

[26] Yicheng Liu, Yue Wang, Yilun Wang, and Hang Zhao. Vectormapnet: End-to-end vectorized hd map learning. *arXiv preprint arXiv:2206.08920*, 2022. 1, 3, 6

[27] Abdelhak Loukkal, Yves Grandvalet, Tom Drummond, and You Li. Driving among flatmobiles: Bird-eye-view occupancy grids from a monocular camera for holistic trajectory planning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 51–60, 2021. 2

[28] Chenyang Lu, Marinus Jacobus Gerardus van de Molengraft, and Gijs Dubbelman. Monocular semantic occupancy grid mapping with convolutional variational encoder–decoder networks. *IEEE Robotics and Automation Letters*, 4(2):445–452, 2019. 2

[29] Yuexin Ma, Tai Wang, Xuyang Bai, Huitong Yang, Yuenan Hou, Yaming Wang, Yu Qiao, Ruigang Yang, Dinesh Manocha, and Xinge Zhu. Vision-centric bev perception: A survey. *arXiv preprint arXiv:2208.02797*, 2022. 2

[30] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 4

[31] Davy Neven, Bert De Brabandere, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. Towards end-to-end lane detection: an instance segmentation approach. In *2018 IEEE intelligent vehicles symposium (IV)*, pages 286–291. IEEE, 2018. 2

[32] Bowen Pan, Jiankai Sun, Ho Yin Tiga Leung, Alex Andonian, and Bolei Zhou. Cross-view semantic segmentation for sensing surroundings. *IEEE Robotics and Automation Letters*, 5(3):4867–4873, 2020. 1, 2, 3

[33] Lang Peng, Zhirong Chen, Zhangjie Fu, Pengpeng Liang, and Erkang Cheng. Bevsegformer: Bird's eye view semantic segmentation from arbitrary camera rigs. *arXiv preprint arXiv:2203.04050*, 2022. 1, 2, 3

[34] Jonah Philion and Sanja Fidler. Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. In *European Conference on Computer Vision*, pages 194–210. Springer, 2020. 1, 2, 3, 8

[35] L. Reiher, B. Lampe, and L. Eckstein. A sim2real deep learning approach for the transformation of images from multiple vehicle-mounted cameras to a semantically segmented image in bird's eye view. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, 2020. 2

[36] Thomas Roddick and Roberto Cipolla. Predicting semantic map representations from images using pyramid occupancy networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11138–11147, 2020. 2

[37] Avishkar Saha, Oscar Mendez, Chris Russell, and Richard Bowden. Enabling spatio-temporal aggregation in birds-eye-view vehicle estimation. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5133–5139. IEEE, 2021. 2

[38] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4938–4947, 2020. 4

[39] Samuel Schulter, Menghua Zhai, Nathan Jacobs, and Manmohan Chandraker. Learning to look around objects for top-view representations of outdoor scenes. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 787–802, 2018. 2

[40] Richard Sinkhorn and Paul Knopp. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics*, 21(2):343–348, 1967. 5

[41] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019. 3, 6

[42] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 3

[43] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017. 4

[44] Weixiang Yang, Qi Li, Wenxi Liu, Yuanlong Yu, Yuexin Ma, Shengfeng He, and Jia Pan. Projecting your view attentively: Monocular road scene layout estimation via cross-view transformation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15536–15545, 2021. 1, 2, 3

[45] Brady Zhou and Philipp Krähenbühl. Cross-view transformers for real-time map-view semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13760–13769, 2022. 1, 3

[46] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020. 3

[47] Xinge Zhu, Zhichao Yin, Jianping Shi, Hongsheng Li, and Dahua Lin. Generative adversarial frontal view to bird view synthesis. In *2018 International conference on 3D Vision (3DV)*, pages 454–463. IEEE, 2018. 2

[48] Stefano Zorzi, Shabab Bazrafkan, Stefan Habenschuss, and Friedrich Fraundorfer. Polyworld: Polygonal building extraction with graph neural networks in satellite images. In *Proceedings of the IEEE/CVF Conference on Computer Vi-*

sion and Pattern Recognition, pages 1848–1857, 2022. 3, 8