DAGMapper: Learning to Map by Discovering Lane Topology

Namdar Homayounfar[1,2]   Wei-Chiu Ma[1,3]

Justin Liang[1]   Xinyu Wu [1]   Jack Fan [1]   Raquel Urtasun[1,2]

[1]Uber Advanced Technologies Group   [2]University of Toronto   [3] MIT

{namdar,weichiu,justin.liang,xinyuw,jackwindows,urtasun}@uber.com

Abstract

One of the fundamental challenges to scale self-driving is being able to create accurate high definition maps (HD maps) with low cost. Current attempts to automate this process typically focus on simple scenarios, estimate independent maps per frame or do not have the level of precision required by modern self driving vehicles. In contrast, in this paper we focus on drawing the lane boundaries of complex highways with many lanes that contain topology changes due to forks and merges. Towards this goal, we formulate the problem as inference in a directed acyclic graphical model (DAG), where the nodes of the graph encode geometric and topological properties of the local regions of the lane boundaries. Since we do not know a priori the topology of the lanes, we also infer the DAG topology (i.e., nodes and edges) for each region. We demonstrate the effectiveness of our approach on two major North American Highways in two different states and show high precision and recall as well as 89% correct topology.

1. Introduction

Self-driving vehicles are equipped with a plethora of sensors, including GPS, cameras, LiDAR, radar and ultrasonics. This allows the vehicle to see with a field of view of 360 degrees, potentially having super human capabilities. Despite decades of research, building reliable solutions that can handle the complexity of the real world is still an open problem.

Most modern self-driving vehicles utilize a high definition map (HD map), which contains information about the location of lanes, their types, crosswalks, traffic lights, rules at intersections, etc. This information is very accurate, typically with error on the order of a few centimeters. It can thus help localization [37, 8, 36], perception [60, 32, 12], motion planning [62], and even simulation [16]. Building these maps is however one of the main obstacles to building self driving cars at scale. The typical process is very cumbersome, involving a fleet of vehicles passing multiple
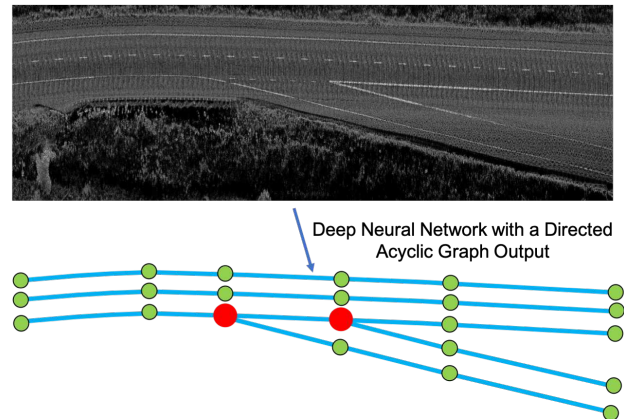

Figure 1: The input to our model is an aggregated LiDAR intensity image and the output is a DAG of the lane boundaries parametrized by a deep neural network.

times across the same area to build accurate geometric representations of the world. Annotators then label the location of all these semantic landmarks by hand from a bird's eye view representation of the world. While building the geometric map is time consuming, manual annotation is costly in terms of capital.

In this paper, we tackle the problem of automatically creating HD maps of highways that are consistent over large areas. Unlike the common practice in the industry, we aim to map the whole region from a *single* pass of the vehicle. Towards this goal, we first capitalize on the LiDAR mounted on our self-driving vehicles to build a bird's eye view (BEV) representation of the world. We then exploit a deep network to extract the exact geometry and topology of the underlying lane network.

The main challenge of this task arises from the fact that highways contain complex topological changes due to forks and merges, where a lane boundary splits into two or two lane boundaries merge into one. We address these challenges by formulating the problem as inference in a directed acyclic graphical model (DAG), where the nodes of the graph encode geometric and topological properties of the
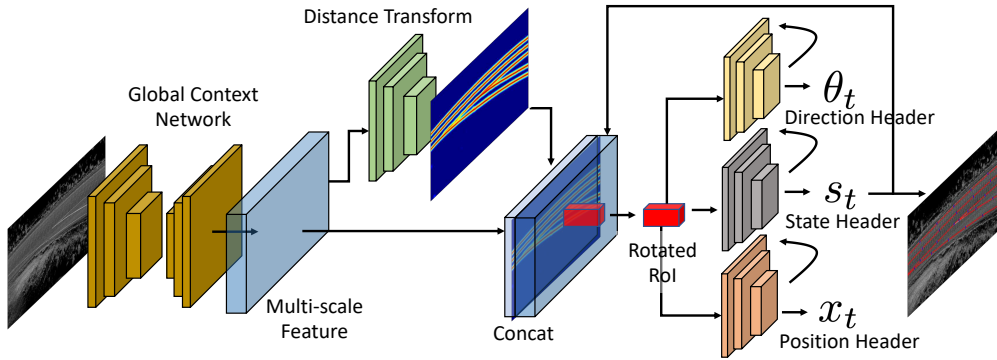
Figure 2: Network Architecture: A deep global feature network is applied on the LiDAR intensity image followed by three recurrent convolutional headers that parametrize a DAG of the lane boundaries. DT image is used for initialization of the lane boundaries as well as recovery from mistakes.

local regions of the lane boundaries. Since we do not know a priori the lane topology, we also infer the DAG topology (i.e., nodes and edges) for each region. We leverage the power of deep neural networks to represent the conditional probabilities in the DAG. We then devise a simple yet effective greedy algorithm to incrementally estimate the graph as well as the states of each node. We learn all the weights of our deep neural network in an end-to-end manner. We call our method DAGMapper.

We demonstrate the effectiveness of our approach on two major North American Highways in two different states 1000 km apart. The dataset consists of LiDAR sensory data and the corresponding lane network topology on areas of high complexity such as forks and merges. Our approach achieves a precision and recall of 89% and 88.7% respectively at a maximum of 15 cm away from the actual location of the lane boundaries. Moreover, our approach obtains the correct topology 89% of the time. Finally, we showcase the strong generalization of our approach by training on a highway and testing on another belonging to a different state. Our experiments show that no domain adaptation is required.

2. Related Work

Road and lane detection: Detecting the lane markings [56, 11], inferring the lane boundaries [6, 58, 22] as well as finding the drivable surface [18, 5, 54] on the road is of utmost importance for self-driving vehicles. Many methods have been proposed to tackle this problem. The authors in [53, 33, 48, 3, 24, 15, 59, 2, 25] use appearance and geometric cues to detect the road in an unsupervised or semi-supervised fashion, while [46, 28, 61] apply graphical models to estimate free space. With the advent of modern deep learning [26], a new line of research for lane marking detection has been established. In [27], the authors employ vanishing points and train a neural network that detects lane

boundaries. [19] uses generative adversarial networks [20] to further refine the detection and obtain thinner segmentations. The authors in [4] design a symmetric CNN enhanced by wavelet transform in order to segment lane markings from aerial imagery. In [6], camera and LiDAR imagery are employed to predict a dense representation of the lane boundaries in the form of a thresholded distance transform. In [47], the authors treat lane detection as an instance segmentation task where deep features corresponding to lane markings are clustered.

Road network extraction: Leveraging aerial and satellite imagery for the task of road network extraction goes back many decades [17, 49]. In the early days, researchers mainly extracted the road network topology by iteratively growing a graph using simple heuristics on spectral features of the roads [50, 7]. Recently, deep learning and more advanced graph search algorithms have been leveraged [43, 44, 45, 38, 39, 40, 55, 9] to extract the road network from aerial imagery more effectively. For instance, [29] extract building and road network graphs directly in the form of polygons from aerial imagery. [41, 42] enhance Open Street Maps with lane markings, sidewalks and parking spots by applying graphical models on top of deep features. [51] leverages GPS data to enhance road extraction from aerial imagery. [10] predicts both the orientation and the semantic mask of the road to obtain topologically correct and a connected road network from aerial imagery. One however should note that these methods perform road network extraction and semantic labeling at a *coarse* scale. While they can be beneficial for routing purposes, they lack the required resolution for self driving applications.

High Definition maps: Creating HD maps that have centimeter level accuracy is crucial for the safe navigation of self-driving vehicles. Recently, researchers have been devoted to generating HD maps automatically from various

sensory data [23]. For example, the authors in [30] extract crosswalk polygons from top-down LiDAR and camera imagery. [31, 21] employ deep convolutional networks to extract road and lane boundaries in the form of polylines. These lines of mapping work are similar to instance segmentation methods [13, 1, 35] where structured representations of objects such as polygons are obtained. The algorithms are thus amenable to an annotator in the loop.

At a high level, our work shares similarities with [21], which predicts lane boundaries from a top down BEV LiDAR image by exploiting a recurrent convolutional neural network. However, [21] cannot handle changes of topology of lane boundaries and has baked in notions of lane boundary ordering which we address in this paper.

3. Learning to Map

Our goal is to create HD maps directly from 3D sensory data. More specifically, the input to our system is a BEV aggregated LiDAR intensity image $\mathcal{D}$ and the output is a collection of structured polylines corresponding to the lane boundaries. Rather than focusing on local areas, we attempt to extract the exact topology and geometry of the lane network network on a long stretch of the highway. This is extremely challenging as highways contain complex topological changes (*e.g.*, forks and merges) and the area of coverage is very large. Towards this goal, we formulate the underlying geometric topology as a directed acyclic graph and present a novel recurrent convolutional network that can effectively extract such topology from LiDAR data. We start our discussion by describing our output space formulation in terms of the DAG. Next, we explain how we exploit neural networks to parameterize each conditional probability distribution. Finally, we describe how inference and learning are performed.

3.1. Problem Formulation and DAG Extraction

We draw our inspirations from how humans create lane graphs when building maps. Given an initial vertex, the annotators first trace the lane boundary with a sequence of clicks that respect the local geometry. Then, if there is a change of topology during the process, the annotators identify such points and annotate correspondingly. For instance, if the click reaches the end of the road, one simply stops; if there is a fork, one simply needs to create another branch from the click. By using this simple approach, the annotators can effectively label the road network in the form of a DAG. In this work, we design an approach to road network DAG discovery that mimics such an annotation process.

More formally, let $G = (V, E)$ be a DAG, where V and E denote the corresponding set of nodes and edges defining the topology. Each node $v_i = (x_i, \theta_i, s_i)$ in the DAG encodes geometric and topological properties of a local region of the lane boundary. We further use $v_{\mathcal{P}(i)}$ and $v_{\mathcal{C}(i)}$

Algorithm 1: Deep DAG Topology Discovery

Input : Aggregated point clouds, initial vertices
$\{v_{\text{init}} = (\theta_{\text{init}}, x_{\text{init}}, s_{\text{init}})\}$
Output: Highway DAG Topology

1 Initialize *queue* Q with vertices $\{v_{\text{init}}\}$;
2 **while** *Q not empty* **do**
3     $v_i \leftarrow Q.pop()$;
4     $i \leftarrow \mathcal{C}(i)$;
5   **while** $s_{\mathcal{P}(i)}$ *not Terminate* **do**
6        $\theta_i \leftarrow \arg\max\ p(\theta_i | \theta_{\mathcal{P}(i)}, s_{\mathcal{P}(i)}, x_{\mathcal{P}(i)})$;
7        $x_i \leftarrow \arg\max\ p(x_i | \theta_i, s_{\mathcal{P}(i)}, x_{\mathcal{P}(i)})$;
8        $s_i \leftarrow \arg\max\ p(s_i | \theta_i, s_{\mathcal{P}(i)}, x_{\mathcal{P}(i)})$;
9   **if** $s_i = Fork$ **then**
10          $Q.insert(v_i)$;
11   **end**
12        $i \leftarrow \mathcal{C}(i)$;
13   **end**
14 **end**

---

respectively to denote the parent and the child of the node v_i. The geometric component x_i denotes the position of the vertex in global coordinates and θ_i refers to the turning angle at the previous vertex position $x_{\mathcal{P}(i)}$. The state component s_i is a categorical random variable that denotes whether to continue the lane boundary without any change of topology, to spawn an extra vertex for a new lane boundary at a fork, or to terminate the lane boundary (at a merge). Therefore, if s_i is a fork, then $|\mathcal{C}(i)| = 2$; if s_i is stop at the lane end of the lane boundary, then $|\mathcal{C}(i)| = 0$.

Given the aggregated BEV LiDAR data $\mathcal{D}$, our goal is to find the Maximum A Posteriori (MAP) over the space of all possible graphs $\mathcal{G}$:

$$G^* = \arg\max_{G \in \mathcal{G}} p(G|\mathcal{D}). \tag{1}$$

As G is a DAG in our case, the joint probability distribution $p(G|\mathcal{D})$ can be factorized into:

$$p(G|\mathcal{D}) = \prod_{v_i \in V} p(v_i | v_{\mathcal{P}(i)}, \mathcal{D}), \tag{2}$$

where each conditional probability further decomposes into geometric and topological components:

$$\begin{aligned} p(v_i | v_{\mathcal{P}(i)}, \mathcal{D}) = &p(\theta_i | \theta_{\mathcal{P}(i)}, s_{\mathcal{P}(i)}, x_{\mathcal{P}(i)}, \mathcal{D}) \\ &\times p(x_i | \theta_i, s_{\mathcal{P}(i)}, x_{\mathcal{P}(i)}, \mathcal{D}) \\ &\times p(s_i | \theta_i, s_{\mathcal{P}(i)}, x_{\mathcal{P}(i)}, \mathcal{D}). \end{aligned} \tag{3}$$

This conditional distributions are modeled with deep neural networks in order to handle the complexities of the real world. Fig. 3 shows an example BEV image $\mathcal{D}$ and the inference process of a lane boundary where there is a fork.

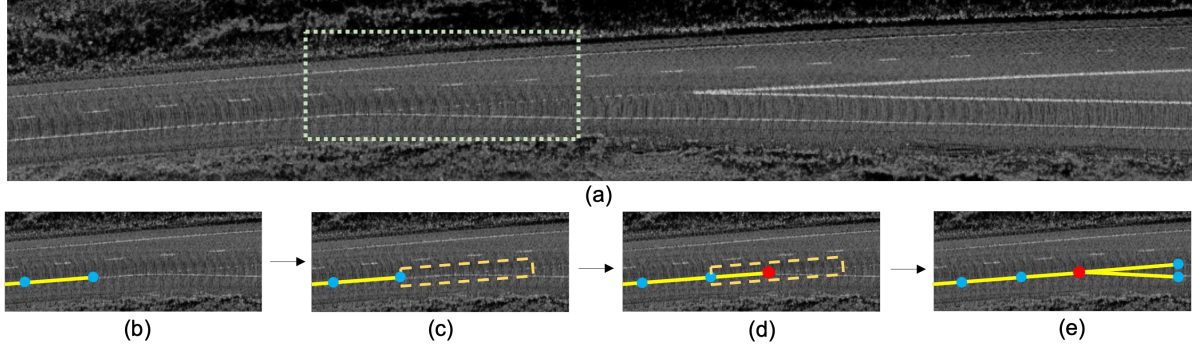We provide in the next section an in depth discussion of the networks.

Figure 3: (a) The BEV aggregated LiDAR image $\mathcal{D}$ of an area of the highway with a fork. The green box shows a zoomed in area for visualization of the inference process. (b) The current subgraph of the inferred bottom lane boundary. (c) The direction θ_i of the new ROI (yellow box) along the lane boundary is predicted. (d) A new node is predicted with position x_i (red dot) within the ROI with the *fork* state s_i. (e) Two lane boundaries emanate from the fork node and the process continues.

Topology Extraction: Unfortunately, the space of $\mathcal{G}$ is exponentially large. It is thus computationally intractable to consider all possible DAGs. To tackle this issue, we design an approach that *greedily* constructs the graph and computes the geometric and topological properties of each vertex. Specifically, we take the $\arg\max$ of each component in Eq. (3) to obtain the most likely topological and geometric states for each node v_i. Given a parent node $v_{\mathcal{P}(i)}$, we first predict along which direction θ_i the node v_i would be. Then we attend to a local region of interest along the lane boundary specified by θ_i and predict the vertex position x_i and the local state s_i. The state specifies that either (1) there is no change of topology and we continue as normal, (2) there is a fork and we continue the current lane boundary as normal but at the same time spawn a new vertex for a new lane boundary, or (3) the current lane boundary has terminated at the merge, and hence the node v_i has no child. By iterating through the procedure, we are able to obtain the structure of the DAG as well as their geometric positions. Our structure discovery algorithm is summarized in Alg. 1. We next describe the backbone network that we used to extract the context features and the header networks that approximate the geometrical and topological probability distribution.

3.2. Network Design

In this work, we exploit neural networks to approximate the DAG probability distributions described in the previous section, also shown in Alg. 1. At a high level, as shown in Fig. 2, we first exploit a *global feature network* to extract multi-scale features and a *distance transform network* to encode explicit lane boundary information. The features are then passed to the *direction header* to predict a rotated Region of Interest (RoI) along the lane boundary. Finally, the *position header* and the *state header* condition on the features within the RoI predict the position and state of the next vertex. We now present the high level details of each head and refer the reader to the supplementary material for

a description of the full architecture.

Global Feature Network: The aim of this network is to build features for the header networks to draw upon. Since the changes of topology are usually very gradual, it will be very difficult to infer them if the features capture merely local observations or the network has a small receptive field (see Fig. 3). For instance, at a fork, a lane boundary gradually splits into two so that vehicles have enough time to exit the highway and slow down in a safe manner. Similarly, at a merge, two lanes become one gradually over a long stretch so that vehicles accrue speed and enter the highway smoothly without colliding with the ongoing flow of traffic.

The input to this network is a BEV projected aggregated LiDAR intensity image $\mathcal{D} \in \mathbb{R}^{1 \times H \times W}$ at 5 cm per pixel. Our images are of dimension 8000 pixels in width by maximum 1200 in height corresponding to 400m by maximum 60m in the direction of travel of the mapping vehicle as shown in Fig. 3. As such, to better infer the state of each vertex, one must exploit larger contextual information. Following [21], we adopt an encoder-decoder architecture [14] built upon a feature pyramid network [34] that encodes the context of the lane boundaries and the scene. The bottom-up, top-down structure enables the network to process and aggregate multi-scale features; the skip links help preserve spatial information at each resolution. The output of this network is a feature map $F \in \mathbb{R}^{C \times \frac{H}{4} \times \frac{W}{4}}$.

Distance Transform Network: The distance transform (DT) and more specifically the thresholded inverse DT has been proven to be an effective feature for mapping [30, 31]. It encodes at each point in the image the relative distance to the closest lane boundary. As such, we employ a header that consists of a sequence of residual layers that takes as input the global feature map F and outputs a thresholded inverse DT image $D \in R^{1 \times \frac{H}{4} \times \frac{W}{4}}$. We use this DT image D for three purposes: 1) We stack it to the global feature map F

	Precision at (px)				Recall at (px)				F1 at (px)			
	2	3	5	10	2	3	5	10	2	3	5	10
DT baseline [6]	68.1	87.9	**96.3**	**98.0**	65.8	85.0	93.5	**96.2**	66.9	86.4	**94.9**	**97.0**
HRAN [21] with GT init	55.6	71.0	84.0	90.9	45.9	58.1	68.3	74.0	50.1	63.7	75.1	81.4
Ours	**76.4**	**89.0**	94.6	96.6	**76.2**	**88.7**	**94.2**	96.1	**76.3**	**88.8**	94.4	96.3

Table 1: Comparison to the baselines from [6, 21]. We highlight the precision, recall and F1 metrics at thresholds of 2, 3, 5 and 10 px (5cm/px).

as an additional channel and feed it to the other headers. Our aim is to use D as a form of attention on the position of the lane boundaries. 2) We threshold, binarize and skeletonize D and obtain the endpoints of the skeleton as the initial vertices of the graph. 3) After inferring the graph using Alg. 1, we draw the missed lane boundaries by initializing points on the regions of the D that are not covered by our graph.

Direction Header: This network serves as an approximation to $p(\theta_i|\theta_{\mathcal{P}(i)}, s_{\mathcal{P}(i)}, x_{\mathcal{P}(i)})$. Given the geometrical and topological information of the parent vertex, this header predicts the direction of the rotated RoI along the lane boundary where the current vertex lies. The input to this header is a bilinearly interpolated axis aligned crop from the concatenation of F, D centered at $x_{\mathcal{P}(i)}$ and the channel-wise one hot encoding of the state $s_{\mathcal{P}(i)}$. At a fork or a merge, the two lane boundaries are very close to each other. Conditioning on the state vector signals to the header to predict the correct direction corresponding to its lane boundary. This input is fed into a simple convolutional RNN that outputs a direction vector of the next rotated ROI. We employ an RNN to encode history of the previous directions and states.

Position Header: This network can be viewed as an approximation to $p(x_i|\theta_i, s_{\mathcal{P}(i)}, x_{\mathcal{P}(i)})$. Given the state and position of the previous vertex, the network predicts a probability distribution over all possible positions within the rotated ROI along the lane boundary generated by the direction header. This RoI is bilinearly interpolated from the concatenation of F, D and channel-wise one-hot encoding of $s_{\mathcal{P}(i)}$. After the interpolation, we upsample the region to the original image dimension and pass it to a convolutional recurrent neural network (RNN). The output of the RNN is fed to a lightweight encoder-decoder network that outputs a softmax probability map of the position x_i that is mapped to the global coordinate frame of the image.

State Header This header can be regarded as an approximation to $p(s_i|\theta_i, s_{\mathcal{P}(i)}, x_{\mathcal{P}(i)})$, which infers the local topological state of the lane boundaries. Specifically, the network predicts a categorical distribution specifying whether to continue drawing normally, to stop drawing, or to fork a new lane boundary as we have arrived to a merge. The input to this model is the same rotated RoI of the position header.

A convolutional RNN is employed to encode the history and the output is the softmax probability over the three states.

3.3. Learning

We employ a multi task objective to provide supervision to all the different parts of the model. Since all the components are differentiable we can learn our model end-to-end. In particular, similar to [21] we employ the symmetric Chamfer distance to match each GT polyline Q to its prediction P :

$$L(P, Q) = \sum_i \min_{q \in Q} \|p_i - q\|_2 + \sum_j \min_{p \in P} \|p - q_j\|_2 \quad (4)$$

where p and q are the densely sampled coordinates on polylines P and Q respectively. To learn the topology states, we use multi label focal loss with a slight modification; Rather than taking the mean of all the individual losses, we add them up and divide by the sum of the focal weights. Here, the intuition is that the wrong predictions are more emphasized and are not dampened by the over-sampled class corresponding to the *normal* state. Finally, we employ cosine similarity loss and the ℓ_2 loss to learn the directions and the distance transform. We refer the reader to the supplementary material for the full training and implementation details.

4. Experimental Evaluation

Dataset: Our dataset consists of LiDAR point clouds captured by driving a mapping vehicle multiple times through a major North American Highway. For each run of the vehicle, we aggregate the point clouds using IMU odometry data in a common coordinate frame with an arbitrary origin. In our setting of offline mapping, the aggregation of point clouds in a local area is taken from all the past as well future LiDAR sweeps that hit that area. Next, we project the point clouds onto BEV and rasterize at 5 cm per pixel resolution by taking the intensity of the returned point with the lowest elevation value. Since we are interested in HD mapping of lane boundaries that fall on the surface of the road, by taking the lowest elevation point we aim to filter out the LiDAR return from other moving vehicles.

We are interested in mapping difficult areas on the highway such as forks and merges. As such we locate where these interesting topologies occur and crop a rectangular

G	M	R	S	Precision at (px)				Recall at (px)				F1 at (px)			
				2	3	5	10	2	3	5	10	2	3	5	10
-	-	-	✓	75.4	87.4	91.6	93.5	62.1	72.0	75.6	77.4	68.1	79.0	82.8	84.7
✓	-	-	✓	74.1	86.6	93.0	95.9	65.6	76.5	82.0	84.5	69.5	81.2	87.2	89.8
✓	-	✓	✓	73.3	85.9	92.6	95.4	72.8	85.2	91.6	94.4	73.1	85.6	92.1	94.9
✓	✓	✓	-	69.8	83.0	90.6	93.4	70.2	83.5	91.2	94.2	70.0	83.3	90.9	93.8
✓	✓	-	✓	**77.0**	**89.5**	**95.1**	**96.9**	75.0	87.2	92.4	94.2	76.0	88.3	93.7	95.5
✓	✓	✓	✓	76.4	89.0	94.6	96.6	**76.2**	**88.7**	**94.2**	**96.1**	**76.3**	**88.8**	**94.4**	**96.3**

Table 2: We evaluate the contribution of the different components of our model. In particular, we assess the global feature network (G), the conv-RNN (M), the notion of state (S), and recovery using the DT image (R) in case of missing a lane boundary.

	2 px	3 px	5 px	10 px
Precision	57.4	72.1	83.4	88.6
Recall	53.0	66.2	76.1	80.9
F1	55.1	69.0	79.5	84.6

Table 3: Generalization to a new highway: we report precision, recall and F1 metrics of training on all the images of one highway and testing on a completely new highway.

region of 8000 pixels by a maximum of 1200 pixels with the long side of the rectangle perpendicular to the trajectory of the mapping vehicle. Note that the height of the image varies in each example depending on the curvature of the road. This crop corresponds to a 400m by maximum 60m region.

We annotate the ground truth by drawing polyline instances associated to each lane boundary. At a fork we assign a new polyline to the new lane boundary that appears. In total we create 3000 images from 68 unique fork/merge areas on the highway.

Since our aim is scalable high definition mapping of highways, we evaluate the effectiveness of our method on a different highway in a different geographical region. As such we drive our mapping vehicle multiple times in this new highway and collect and annotate 336 images from 114 unique fork/merge areas on the highway. We use this smaller dataset only for evaluation.

To create our train, val and test splits, we first divide our 68 fork/merge areas randomly with the ratio of 70:15:15 and place the images of each area in the corresponding split. This way the same fork/merge area is not present in different splits. To gain statistical significance in our reported metrics, we repeat this procedure three time to obtain three cross validation splits. As such, for baseline comparisons and ablation studies, we train and test each model three times.

Baselines: Since there are no baselines in the literature that perform offline mapping on long stretches of the highway using BEV LiDAR imagery, we create two baselines based on [6] and [21]. For the first baseline based on [6] denoted by (DT baseline) , we use the same backbone of the global feature network with additional upsampling and residual layers to only predict the the inverse thresholded DT image at the original image dimension. We threshold the distance transform at 32 pixels on each side of the lane boundary. Next, we binarize the DT image and skeletonize [52] to obtain a dense representation of the lane boundaries. This is a very strong baseline since the whole capacity of the backbone network is devoted to predicting the DT image. However, it differs in our method in that we output structured representations of the lane boundaries in the form of polylines in an end-to-end manner that is amenable to an annotator in the loop for correction tasks. We create a second baseline based on [21] denoted by (HRAN). The recurrent lane counting module of HRAN architecture has the baked in notion of attending to new lane boundaries from left to right of the road in the bottom of the image which breaks down for the general case of having new lane boundaries spawning at forks and merges. As such we made this baseline stronger by removing the lane counting module during training and inference and instead providing the ground truth starting points for initialization. Note that our method infers the initial points automatically from the DT image.

Precision/Recall: We report the precision and recall metric of [57]. Precision is defined as the number of densified predicted polyline points that fall within a threshold of the GT polylines divided by the total length of the predicted polylines. The sums and the divisions are taken over all the images rather than per lane boundary or per image similar to object detection task metrics. Recall is the same but the role of prediction and GT polylines are reversed. We evaluate at 2,3,5, and 10 pixels corresponding to 10, 15, 25 and 50 cm. As we can see in Table 1, we outperform both baselines. For DT baseline, we obtain higher precision and recall by a wide margin at 2 and 3 pixels corresponding to 10 and 15. However, this baseline performs better at precision and comparable at recall at 5 and 10 pixels corresponding to 25 and 50 cm. For self driving applications where cm accu-
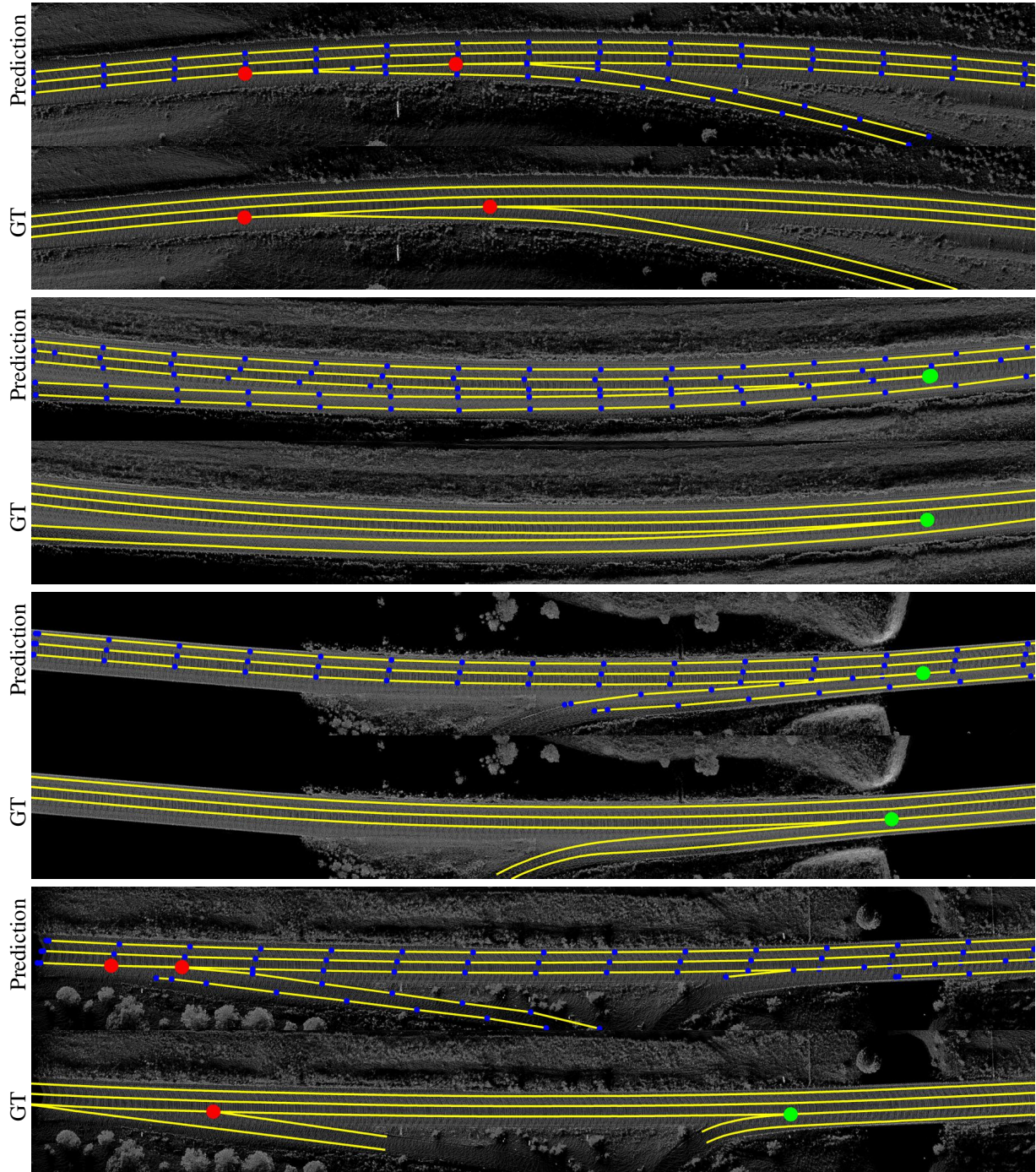
Figure 4: Qualitative results: Rows 1-3 showcase the effectiveness of our method. Row 4 is a failure case where two lane boundaries are bottom are drawn partially.

racy in HD maps is desirable for the safe navigation of the self driving vehicle, our method proves valuable. For the [21] HRAN baseline, where we made it stronger by using GT initial points, we observe that its performance is significantly worse. This is due to HRAN not having an explicit notion of state in the model which prevents the lane bound-

aries to split at forks and terminate at merges.

Topology: For an annotator in the loop to make the least number of clicks, it is desirable that only one predicted polyline segment is assigned to a GT polyline. To assess this, we assign each predicted polyline to the GT lane
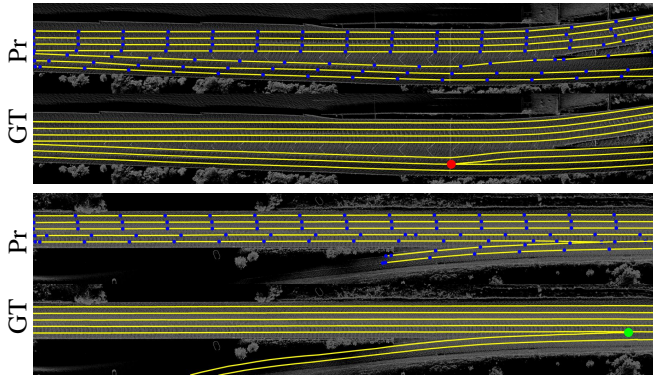
Figure 5: Generalization to a new highway: We train on all the images of one highway and test on another highway in a completely different geographical region to test the generalization of our model. Please zoom.

boundary with the largest intersection at a distance of 20 pixels within the GT. We find that 89% of GT lane boundaries have the correct topology.

Ablation Study: In our ablation studies, we evaluate the importance of the global feature network (G), the Convolutional RNN memory (M) for the headers as well as re-initializing from the DT prediction (R) when we miss drawing lane boundaries. Finally, we evaluate the importance of the notion of state (S) by removing it completely from our model. That is the node v_i will not have a state component s_i and the position x_i and direction θ_i will not be conditioned on s_i.

To assess the importance of the global feature network, we apply the direction, state and position header directly on the LiDAR image at a local ROI. This way the receptive field of each header is restricted to the attended ROI and lacks a global context of lane boundaries. By comparing rows 1 and 2 of Table 2, we see that just adding the global feature network increases the recall by a wide margin. The precision remains competitive as the model can still do a good job predicting the position of the lane boundaries in a local region.

Furthermore, we observe in rows 3 and 5 of Table 2, that adding memory and DT recovery to the global feature network, again boosts the performance by a large margin especially in recall. However, we note that the largest gain is obtained from the memory components. If we remove the notion of state and keep all the other components, both precision and recall suffer dramatically as shown in row 4. This highlights the importance of having state in our model. Finally, we note that our full model with all the components has slightly lower precision than the model with no recovery (row 5) and better recall but overall performs the best in terms of F1. We note that these ablations are the average on the test sets of the three cross validation splits.

Qualitative Results: Please refer to Figure 4 for qualitative results. In row 1 we showcase how our model correctly infers the change of topology by spawning two new lane boundaries at a fork. In rows 2 and 3, we demonstrate the behavior of our model at a merge where two lane boundaries become one and either one or both terminate. In row 4. we showcase a failure mode. At the fork, although we have detected a change of topology, the direction header has failed to infer the correct ROI. At the merge, the upper merging lane boundary overlaps a straight one without stopping. Here an annotator has to manually fix these problematic cases rather than drawing everything from scratch.

Generalization to a New Highway: The ultimate goal of our method is to enable large scale high definition mapping of highways to in turn facilitate safe self driving at scale. To evaluate the generalization of our method, we train a new model on all the images in the train,val, and test splits of our dataset and we evaluate on a different highway located in a different geographical region. In Table 3, we evaluate our model in terms of precision and recall. Although the performance is dropped as expected in comparison to Table 1 where we evaluated on the same highway, we still obtain very promising results. In particular, at 2 pixels or 5 cm, we have 57.4% precision and 53.0% recall while at a wider threshold of 10 pixels or 50 cm, we obtain precision and recall values of 88.6% and 80.9% respectively. In Figure 5, we demonstrate our predictions on this new highway.

Failure Cases: The failure modes consists when a lane boundary is partially not drawn or missed completely and is reflected in our recall metric. In these cases, the annotators have to correct only the problematic cases rather than draw everything from scratch.

5. Conclusion

In this paper we have shown how to draw the lane boundaries of complex highways with many lanes that contain topology changes due to forks and merges. Towards this goal, we have formulated the problem as inference in a DAG, where the nodes of the graph encode geometric and topological properties of the local regions of the lane boundaries. We also derived a simple yet effective greedy algorithm that allow us to infer the DAG topology (i.e., nodes and edges) for each region. We have demonstrated the effectiveness of our approach on two major North American Highways in two different states and show high accuracy in terms of the line drawing and the topology recovered. We plan to extend our approach to utilize also cameras that are typically available onboard self driving vehicles to further improve the performance.

References

[1] David Acuna, Huan Ling, Amlan Kar, and Sanja Fidler. Efficient interactive annotation of segmentation datasets with polygon-rnn++. In *CVPR*, 2018.

[2] Jose M Alvarez, Theo Gevers, Yann LeCun, and Antonio M Lopez. Road scene segmentation from a single image. In *ECCV*, 2012.

[3] José M Álvarez Alvarez and Antonio M Lopez. Road detection based on illuminant invariance. *ITS*, 2011.

[4] Seyed Majid Azimi, Peter Fischer, Marco Körner, and Peter Reinartz. Aerial lanenet: Lane-marking semantic segmentation in aerial imagery using wavelet-enhanced cost-sensitive symmetric fully convolutional neural networks. *GRSL*, 2018.

[5] Hernán Badino, Uwe Franke, and Rudolf Mester. Free space computation using stochastic occupancy grids and dynamic programming. In *ICCV Workshop on Dynamical Vision*, 2007.

[6] Min Bai, Gellert Mattyus, Namdar Homayounfar, Shenlong Wang, Kowshika Lakshmikanth, Shrinidhi, and Raquel Urtasun. Deep multi-sensor lane detection. In *IROS*, 2018.

[7] Ruzena Bajcsy and Mohamad Tavakoli. Computer recognition of roads from satellite pictures. *SMC*, 1976.

[8] Ioan Andrei Barsan, Shenlong Wang, and Raquel Urtasun. Learning to localize using a lidar intensity map. In *CORL*, 2018.

[9] Favyen Bastani, Songtao He, Sofiane Abbar, Mohammad Alizadeh, Hari Balakrishnan, Sanjay Chawla, Sam Madden, and David DeWitt. Roadtracer: Automatic extraction of road networks from aerial images. In *CVPR*, 2018.

[10] Anil Batra, Suriya Singh, Guan Pang, Saikat Basu, C.V. Jawahar, and Manohar Paluri. Improved road connectivity by joint learning of orientation and segmentation. In *CVPR*, 2019.

[11] Karsten Behrendt and Jonas Witt. Deep learning lane marker segmentation from automatically generated labels. In *IROS*, 2017.

[12] Sergio Casas, Wenjie Luo, and Raquel Urtasun. Intentnet: Learning to predict intention from raw sensor data. In *CORL*, 2018.

[13] Lluis Castrejón, Kaustav Kundu, Raquel Urtasun, and Sanja Fidler. Annotating object instances with a polygon-rnn. In *CVPR*, 2017.

[14] Abhishek Chaurasia and Eugenio Culurciello. Linknet: Exploiting encoder representations for efficient semantic segmentation. *VCIP*, 2017.

[15] Hsu-Yung Cheng, Bor-Shenn Jeng, Pei-Ting Tseng, and Kuo-Chin Fan. Lane detection with moving vehicles in the traffic scenes. *ITS*, 2006.

[16] Jin Fang, Feilong Yan, Tongtong Zhao, Feihu Zhang, Dingfu Zhou, Ruigang Yang, Yu Ma, and Liang Wang. Simulating lidar point cloud for autonomous driving using real-world scenes and traffic flows. *arXiv*, 2018.

[17] M.-F. Auclair Fortier, Djemel Ziou, and Costas Armenakis. Survey of work on road extraction in aerial and satellite images. *Tech Report*, 2002.

[18] Uwe Franke and I. Kutzbach. Fast stereo based object detection for stop&go traffic. In *IV*, 1996.

[19] Mohsen Ghafoorian, Cedric Nugteren, Nóra Baka, Olaf Booij, and Michael Hofmann. El-gan: embedding loss driven generative adversarial networks for lane detection. In *ECCV*, 2018.

[20] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, 2014.

[21] Namdar Homayounfar, Wei-Chiu Ma, Shrinidhi Kowshika Lakshmikanth, and Raquel Urtasun. Hierarchical recurrent attention networks for structured online maps. In *CVPR*, 2018.

[22] Yuenan Hou, Zheng Ma, Chunxiao Liu, and Chen Change Loy. Learning lightweight lane detection cnns by self attention distillation. In *ICCV*, 2019.

[23] Soren Kammel and Benjamin Pitzer. Lidar-based lane marker detection and mapping. In *IV*, 2008.

[24] Hui Kong, Jean-Yves Audibert, and Jean Ponce. General road detection from a single image. *IP*, 2010.

[25] Tobias Kühnl, Franz Kummert, and Jannik Fritsch. Spatial ray features for real-time ego-lane extraction. In *ITS*, 2012.

[26] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 2015.

[27] Seokju Lee, Junsik Kim, Jae Shin Yoon, Seunghak Shin, Oleksandr Bailo, Namil Kim, Tae-Hee Lee, Hyun Seok Hong, Seung-Hoon Han, and Inso Kweon. Vpgnet: Vanishing point guided network for lane and road marking detection and recognition. In *ICCV*, 2017.

[28] Dan Levi, Noa Garnett, Ethan Fetaya, and Israel Herzlyia. Stixelnet: A deep convolutional network for obstacle detection and road segmentation. In *BMVC*, 2015.

[29] Zuoyue Li, Jan Dirk Wegner, and Aurélien Lucchi. Polymapper: Extracting city maps using polygons. *arXiv*, 2018.

[30] Justin Liang, , and Raquel Urtasun. End-to-end deep structured models for drawing crosswalks. In *ECCV*, 2018.

[31] Justin Liang, Namdar Homayounfar, Wei-Chiu Ma, Shenlong Wang, and Raquel Urtasun. Convolutional recurrent network for road boundary extraction. In *CVPR*, 2019.

[32] Ming Liang, Bin Yang, Yun Chen, Rui Hu, and Raquel Urtasun. Multi-task multi-sensor fusion for 3d object detection. In *CVPR*, 2019.

[33] David Lieb, Andrew Lookingbill, and Sebastian Thrun. Adaptive road following using self-supervised learning and reverse optical flow. In *RSS*, 2005.

[34] Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie. Feature pyramid networks for object detection. In *CVPR*, 2016.

[35] Huan Ling, Jun Gao, Amlan Kar, Wenzheng Chen, and Sanja Fidler. Fast interactive object annotation with curve-gcn. In *CVPR*, 2019.

[36] Wei-Chiu Ma, Ignacio Tartavull, Ioan Andrei Bârsan, Shenlong Wang, Min Bai, Gellert Mattyus, Namdar Homayounfar, Shrinidhi Kowshika Lakshmikanth, Andrei Pokrovsky, and Raquel Urtasun. Exploiting sparse semantic hd maps for affordable localization. *IROS*, 2019.

[37] Wei-Chiu Ma, Shenlong Wang, Marcus A Brubaker, Sanja Fidler, and Raquel Urtasun. Find your way by observing the sun and other semantic cues. In *ICRA*, 2017.

[38] Dimitrios Marmanis, Konrad Schindler, Jan Dirk Wegner, Silvano Galliani, Mihai Datcu, and Uwe Stilla. Classification with an edge: Improving semantic image segmentation with boundary detection. *arXiv*, 2016.

[39] Dimitrios Marmanis, Jan D Wegner, Silvano Galliani, Konrad Schindler, Mihai Datcu, and Uwe Stilla. Semantic segmentation of aerial images with an ensemble of cnns. *ISPRS*, 2016.

[40] Gellért Máttyus, Wenjie Luo, and Raquel Urtasun. Deeproadmapper: Extracting road topology from aerial images. In *ICCV*, 2017.

[41] Gellert Mattyus, Shenlong Wang, Sanja Fidler, and Raquel Urtasun. Enhancing road maps by parsing aerial images around the world. In *ICCV*, 2015.

[42] Gellért Máttyus, Shenlong Wang, Sanja Fidler, and Raquel Urtasun. Hd maps: Fine-grained road segmentation by parsing ground and aerial images. In *CVPR*, 2016.

[43] Juan B. Mena and José A. Malpica. An automatic method for road extraction in rural and semi-urban areas starting from high resolution satellite imagery. *IAPR*, 2005.

[44] Volodymyr Mnih and Geoffrey E. Hinton. Learning to detect roads in high-resolution aerial images. In *ECCV*, 2010.

[45] Volodymyr Mnih and Geoffrey E. Hinton. Learning to label aerial images from noisy data. In *ICML*, 2012.

[46] Rahul Mohan. Deep deconvolutional networks for scene parsing. *arXiv*, 2014.

[47] Davy Neven, Bert De Brabandere, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. Towards end-to-end lane detection: an instance segmentation approach. *IV*, 2018.

[48] Lina Maria Paz, Pedro Piniés, and Paul Newman. A variational approach to online road and path segmentation with monocular vision. In *ICRA*, 2015.

[49] John A. Richards. *Remote Sensing Digital Image Analysis*. Springer, 2013.

[50] David S Simonett, Floyd M Henderson, and Dwight D Egbert. On the use of space photography for identifying transportation routes: A summary of problems. 1970.

[51] Tao Sun, Zonglin Di, Pengyu Che, Chun Liu, and Yin Wang. Leveraging crowdsourced gps data for road extraction from aerial imagery. In *CVPR*, 2019.

[52] Satoshi Suzuki and Keiichi Abe. Topological structural analysis of digitized binary images by border following. *CVGIP*, 1985.

[53] Ceryen Tan, Tsai Hong, Tommy Chang, and Michael Shneier. Color model-based real-time learning for road following. In *ITS*, 2006.

[54] Marvin Teichmann, Michael Weber, Marius Zoellner, Roberto Cipolla, and Raquel Urtasun. Multinet: Real-time joint semantic reasoning for autonomous driving. In *IV*, 2018.

[55] Carles Ventura, Jordi Pont-Tuset, Sergi Caelles, Kevis-Kokitsi Maninis, and Luc Van Gool. Iterative deep learning for road topology extraction. In *BMVC*, 2018.

[56] Rafael Peixoto Derenzi Vivacqua, Massimo Bertozzi, Pietro Cerri, Felipe Nascimento Martins, and Raquel Frizera Vassallo. Self-localization based on visual lane marking maps: An accurate low-cost approach for autonomous driving. *ITS*, 2017.

[57] Shenlong Wang, Min Bai, Gellert Mattyus, Hang Chu, Wenjie Luo, Bin Yang, Justin Liang, Joel Cheverie, Sanja Fidler, and Raquel Urtasun. Torontocity: Seeing the world with a million eyes. In *ICCV*, 2017.

[58] Ze Wang, Weiqiang Ren, and Qiang Qiu. Lanenet: Real-time lane detection networks for autonomous driving. *arXiv*, 2018.

[59] Andreas Wedel, Hernán Badino, Clemens Rabe, Heidi Loose, Uwe Franke, and Daniel Cremers. B-spline modeling of road surfaces with an application to free-space estimation. *ITS*, 2009.

[60] Bin Yang, Ming Liang, and Raquel Urtasun. Hdnet: Exploiting hd maps for 3d object detection. In *CORL*, 2018.

[61] Jian Yao, Srikumar Ramalingam, Yuichi Taguchi, Yohei Miki, and Raquel Urtasun. Estimating drivable collision-free space from monocular video. In *WACV*, 2015.

[62] Wenyuan Zeng, Wenjie Luo, Simon Suo, Abbas Sadat, Bin Yang, Sergio Casas, and Raquel Urtasun. End-to-end interpretable neural motion planner. In *CVPR*, 2019.