# Convolutional Recurrent Network for Road Boundary Extraction

Justin Liang[1*] Namdar Homayounfar[1,2*]
Wei-Chiu Ma[1,3] Shenlong Wang[1,2] Raquel Urtasun[1,2]
[1]Uber Advanced Technologies Group [2]University of Toronto [3] MIT
{justin.liang,namdar,weichiu,slwang,urtasun}@uber.com

## Abstract

*Creating high definition maps that contain precise information of static elements of the scene is of utmost importance for enabling self driving cars to drive safely. In this paper, we tackle the problem of drivable road boundary extraction from LiDAR and camera imagery. Towards this goal, we design a structured model where a fully convolutional network obtains deep features encoding the location and direction of road boundaries and then, a convolutional recurrent network outputs a polyline representation for each one of them. Importantly, our method is fully automatic and does not require a user in the loop. We showcase the effectiveness of our method on a large North American city where we obtain perfect topology of road boundaries* 99.3% *of the time at a high precision and recall.*

## 1. Introduction

High definition maps (HD maps) contain useful information about the semantics of the static part of the scene. They are employed by most self-driving cars as an additional sensor in order to help localization, [9, 30], perception [27, 12] and motion planning. Drawing the maps, is however, a laborious process where annotators look at overhead views of the cities and draw one by one all the elements of the scene. This is an expensive and time consuming process preventing mapping from being done at scale.

Crowd-source efforts such as OpenStreetMaps provide scale, but are not very reliable or precise enough for the safe navigation of the self driving cars. Although they cover most of the globe and provide valuable data such as the road network topology, speed limits, traffic signals, building contours, etc, they suffer from low resolution and are not perfectly aligned with respect to the actual physical clues in the world. This is due to the nature of the map topology creation process which is obtained from GPS trajectories or satellite imagery that could have errors in meters as it is typically very low resolution.

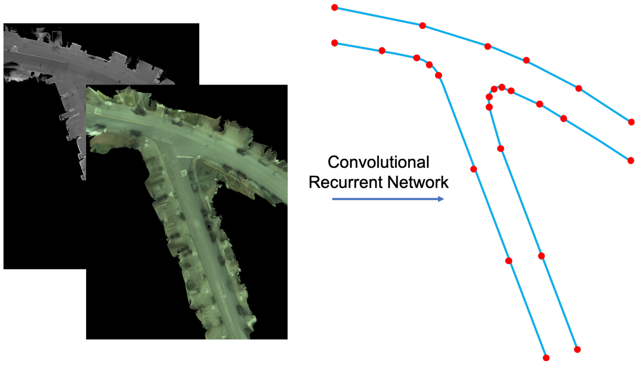

Figure 1. **Overview:** Our Convolutional Reccurrent Network takes as input overhead camera and LiDAR imagery (**left**) and outputs a structured polyline for each road boundary (**right**) that is utilized in creating HD maps for autonomous driving.

Many efforts have been devoted to automate the map creation process to achieve scale. Most early approaches treat the problem as semantic segmentation, either from aerial images [40, 41, 34, 33] or from first person views, where the goal is to capture free space [22, 15, 54, 2, 23]. However, these techniques do not provide a structured representation that is required in order to be consumed by most self driving software stacks.

Automatically estimating the road topology from aerial imagery has been tackled in [35, 52, 10]. In these works, a graph of the road network with nodes being intersections and edges corresponding to the streets connecting them is extracted. Although very useful for routing purposes, these graphs still lack the fine detail and accuracy needed for a safe localization and motion planning of an autonomous car.

In contrast, in this paper we tackle the problem of estimating drivable regions from both LiDAR and camera data, which provide a very high definition information of the surroundings. Towards this goal, we employ convolutional neural networks to predict a set of visual cues that are employed by a convolutional recurrent network to output a variable number of boundary regions of variable size. Im-

[*] Equal contribution.

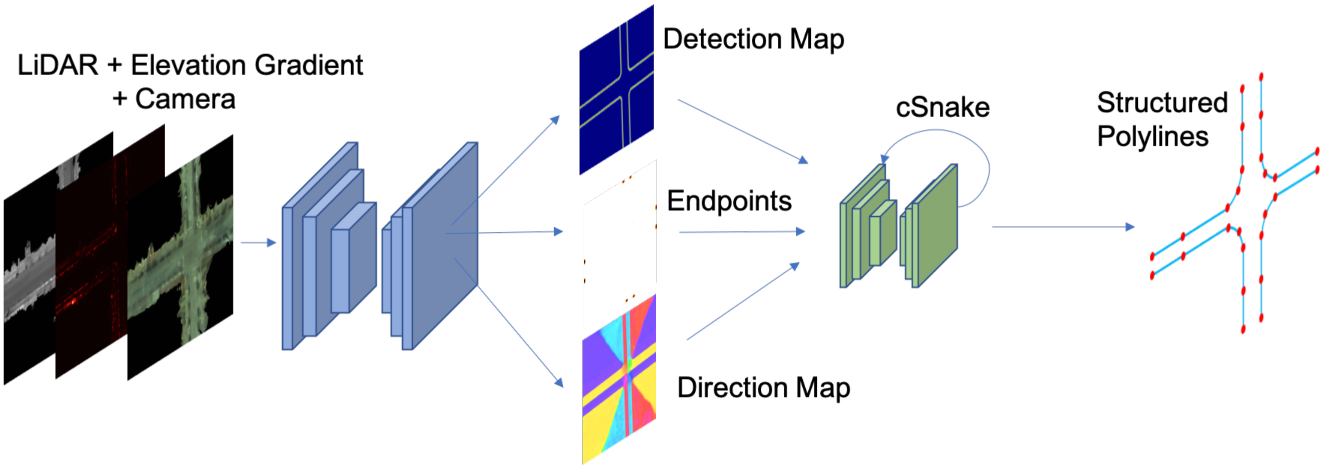

Figure 2. **Model:** Our model takes as input overhead LiDAR and camera imagery as well as the gradient of the LiDAR's elevation value. Next, a convolutional network outputs three feature maps: A truncated inverse distance transform of the location of road boundaries (**Detection Map**), their endpoints (**Endpoints**) and the vector field of normalized normals to the road boundaries (**Direction Map**) shown here as a flow field [7]. Finally, a convolutional recurrent network (**cSnake**) takes these deep features and outputs a structured polyline corresponding to each road boundary.

portantly, our approach is fully automatic and does not require a user in the loop. Inspired by how humans performed this task, our convolutional recurrent network outputs a structured polyline corresponding to each road boundary one vertex at a time.

We demonstrate the effectiveness of our work on a large dataset of a North American city composed of overhead LiDAR and camera imagery of streets and intersections. In comparison to the baselines where a road boundary could be estimated with multiple smaller segments, we predict perfect topology for a road boundary 99.3% of the time with high precision and recall of 87.3% and 87.1% respectively at 5 pixels away. We also perform extensive ablation studies that justify our choices of input representations, training regimes and network architecture. Finally, we propose a novel metric that captures not only precision and recall but also a measure of connectivity of the predictions corresponding to a road boundary.

## 2. Related Work

In the past few decades the computer vision and sensing communities have actively developed a myriad of methods that perform semantic segmentation tasks from aerial and satellite imagery. We refer the reader to [48] for a complete introduction of classical approaches. More recently, deep neural networks [39, 40, 41, 34, 33] have been applied to this task with considerable success. Such output is however not directly usable by self driving vehicles which require a structured representation instead.

Research extracting structured semantic and topological information from satellite and aerial imagery, mainly for consumption in geographic information systems, goes back decades to the earliest works of [50, 6] in the 70s. In these works, the authors grow a road from pixels to edges to line segments iteratively by using thresholds on simple features obtained from geometric and spectral properties of roads. [4] compiles a comprehensive survey of these earlier approaches. Later on, active contour models (ACM) [20, 11] were applied to the task of road extraction from aerial imagery [38, 24, 49, 32]. Here, the authors evolve a snake that captures a road network by minimizing an energy function that specifies geometric and appearance constraints of the roads. The authors in [31] use a deep learning model to define the energy function of an ACM to generate building polygons from aerial imagery but not the road network. [36, 37] apply graphical models on top of deep features in order to enhance Open Street Maps with semantic information such as the location of sidewalks, parking spots and the number and location of lanes. In other work [55, 56, 43] extract the road network from aerial images using a conditional random field, while the works of [35, 52] perform this task by first segmenting the image to road/non-road pixels using a deep network and then performing post process graph optimization. In [10] the authors iteratively grow the road network topology by mixing neural networks and graph search procedures. These approaches extract the road network at a coarser scale and are useful for routing applications, however they lack the fine detail of the surroundings required for the safe navigation of a self driving vehicle.

Predicting the drivable surface is very important for safe navigation of an autonomous vehicle. [42, 25, 57] use graphical models to predict the free space and the road while [51, 28, 45, 3, 22, 15, 54, 2, 23] detect the road using appearance and geometric priors in unsupervised and self-supervised settings.

More recent line of research and industrial work leverage sensors such as camera and LiDAR [44, 19] mounted on cars to create HD maps of the environment. In [47, 46,

|  | Precision at (px) |  |  |  | Recall at (px) |  |  |  | F1 score at (px) |  |  |  | Conn |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
|  | 2 | 3 | 5 | 10 | 2 | 3 | 5 | 10 | 2 | 3 | 5 | 10 |  |
| DT | 47.5 | 66.0 | 85.9 | **96.2** | 47.6 | 65.8 | 84.7 | 93.8 | 47.4 | 65.6 | 85.0 | **94.6** | 89.1 |
| **Ours** | **57.3** | **72.9** | **87.3** | 94.5 | **57.1** | **72.6** | **87.1** | **94.3** | **57.2** | **72.7** | **87.2** | 94.4 | **99.2** |

Table 1. This compares the distance transform (DT) baseline with our model. We show the results for all the models at precision, recall and F1 score thresholds of 2, 3, 5, 10 (4cm/px).

16, 8], multiview and fisheye cameras are used for dense mapping of the static environment using stereo reconstruction and structure from motion techniques. In other work [17, 26] extracted semantic information of the scene such as the precise location and number of the lane boundaries and the crosswalks. These semantics aid the autonomous agent in precise localization and safe navigation. In [17], the authors predict a structured representation of lane boundaries in the form of polylines directly from LiDAR point clouds using a recurrent hierarchical network and in [26], crosswalks are detected using a deep structured model from top down camera and LiDAR imagery. [5] fuses LiDAR and camera to perform dense online lane detection.

In contrast to the aforementioned approaches, in this work we extract road boundaries from LiDAR and camera imagery to create HD maps. Similar to [13, 1], we use a structured output representation in the form of polylines. However, unlike them we propose a fully automatic approach and we tackle a very different setting with different sensors and visual cues.

## 3. Convolutional Recurrent Road Extraction

High definition maps (HD maps) contain useful information encoding the semantics of the static scene. These maps are typically created by having hundreds of annotators manually label the elements on bird's eye view (BEV) representations of the world. Automating this process is key for achieving self driving cars at scale.

In this paper we go one step further in this direction, and tackle the problem of estimating drivable regions from both LiDAR and camera data. We encode these drivable regions with polylines delimiting the road boundaries, as this is the typical representations utilized by commercial HD maps. Towards this goal, we employ convolutional neural networks to predict a set of visual cues that are employed by a convolutional recurrent network to output a variable number of road boundaries of variable size. In particular, our recurrent network attends to rotated regions of interest in the feature maps and outputs a structured polyline capturing the global topology as well as the fine details of each road boundary. Next, we first describe the specifics of the feature maps, followed by our convolutional recurrent network.

### 3.1. Deep Visual Features

We now describe how we obtain deep features that are useful for extracting a globally precise structured polyline representation of the road boundary. As input to our system, we take advantage of different sensors such as camera and LiDAR to create a BEV representation of the area of interest. Note that this can contain intersections or straight portions of the road. We also input as an extra channel the gradient of the LiDAR's height value. This input channel is very informative since the drivable and non-drivable regions of the road in a city are mostly flat surfaces at different heights that are separated by a curb. As shown in our experimental section, these sources of data are complementary and help the road boundary prediction problem. This results in a 5-channel input tensor of size $I \in R^{5\times H\times W}$ that is fed to the multi-task CNN that predicts three types of feature maps: the location of the road boundaries encoded as a distance transform, a heatmap encoding the possible location of the endpoints as well as a direction map encoding the direction pointing towards the closest road boundary. We refer the reader to Fig. 2 for an illustration of these visual cues, which are explained in detail below.

**Road Boundary Dense Detection Map:** To obtain a dense representation of the location of the road boundaries in $I$, we output an inverse truncated distance transform image $S \in \mathbb{R}^{1\times H\times W}$ that encodes the relative distance of each pixel in $I$ to the closest road boundary [5, 26], with the road boundary pixels having the highest value and decreasing as we move away. In contrast to predicting binary outputs at the road boundary pixels which are very sparse, the truncated inverse distance transforms encodes more information about the locations of the road boundaries.

**Endpoints Heatmap:** We output a heatmap image $E \in \mathbb{R}^{1\times H\times W}$ encoding the probability of the location of the endpoints of the road boundaries. Note that this typically happens at the edges of the image.

**Road Boundary Direction Map:** Finally, we also predict a vector field $D \in \mathbb{R}^{2\times H\times W}$ of normal directions to the road boundaries. We obtain the ground truth by taking the Sobel derivative of the road boundaries' distance transform image followed by a normalization step. This feature map specifies at each pixel the normal direction towards the closest road boundary. The normalization step relieves the network from predicting vectors of arbitrary magnitude. We utilize the direction map as an input to our convolutional recurrent network, as it encourages the polyline vertices to be pulled towards the road boundaries. The direction map

|  |  |  | Precision at (px) |  |  |  | Recall at (px) |  |  |  | F1 Score at (px) |  |  |  | Conn |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| L | E | C | 2 | 3 | 5 | 10 | 2 | 3 | 5 | 10 | 2 | 3 | 5 | 10 |  |
| - | - | ✓ | 42.9 | 57.6 | 74.3 | 86.7 | 42.8 | 57.4 | 74.0 | 86.4 | 42.9 | 57.5 | 74.3 | 86.6 | 98.8 |
| ✓ | - | - | 44.0 | 62.2 | 82.8 | 93.4 | 43.9 | 62.0 | 82.7 | 93.3 | 44.0 | 62.1 | 82.8 | 93.3 | 99.2 |
| ✓ | ✓ | - | 51.4 | 69.1 | 86.4 | **94.8** | 51.3 | 68.9 | 86.2 | **94.6** | 51.3 | 69.0 | 86.2 | **94.7** | 99.2 |
| ✓ | ✓ | ✓ | **57.3** | **72.9** | **87.3** | 94.5 | **57.1** | **72.6** | **87.1** | 94.3 | **57.2** | **72.7** | **87.2** | 94.4 | **99.2** |

Table 2. The abbreviated columns are: L (lidar input), E (elevation input), C (camera input). We show the results for all the models at precision, recall and F1 score thresholded at 2, 3, 5, 10px (4cm/px).

is also used in providing the direction of next rotated ROI when evolving the road boundary polyline as we shall explain in section 3.2.

**Network Architecture:** We use an encoder decoder architecture similar to the feature pyramid networks in [29, 14]. This network was chosen for its efficiency and ability to keep spatial information. In particular, there are skip connections between the encoder and decoder that allows for the recovery of lost spatial information which is useful as we use large images in our application. In the encoder stage, each encoder block contains two residual blocks and each residual block contains three dilated convolutional layers. This effectively increases the receptive field to help the network deal with large imagery. In the decoder stage, we have four convolutional layers and a nearest neighbor upsampling of 2x. Prior to each convolutional layer we perform instance normalization followed by a ReLU nonlinearity . Our network has three output branches performing pixel wise prediction to output our distance transform, endpoints and direction features. These features all have the same spatial resolution as the input image $I$.

### 3.2. Convolutional Snake (cSnake)

In this section, we describe the mechanics of our module that captures the precise topology of a road boundary. In the following we refer to this module as *cSnake*. Note that our module is fully automatic and does not require any user in the loop. At a high level, drawing on the deep detection and directional features obtained from the input image and the location of the endpoints, the cSnake iteratively attends to rotated regions of interest in the image and outputs the vertices of a polyline corresponding to a road boundary. The direction of travel of the cSnake is obtained from the direction map $D$.

In particular, we first compute the local maxima of the endpoints heatmap $E$ to find the initial vertices of the road boundaries.Then for each endpoint we draw a separate polyline as follows: Given an initial vertex $x_0$ of the endpoint and a direction vector $v_0$, we use the Spatial Transformer Network [18] to crop a rotated ROI from the concatenation of the detection and direction maps $S$ and $D$. Intuitively, the detection distance map $S$ and the direction map $D$ encourage the cSnake module to pull and place a vertex on the road boundaries. The direction $v_0$ runs in parallel to the road boundary at position $x_0$ and is obtained by first looking up the vector from the closest pixel in the direction map $D$ and then rotating it by 90 degrees pointing away from the image boundary. This rotated ROI is fed to a CNN that outputs an argmax of the next vertex $x_1$ in the image. The next direction vector $v_1$ is obtained similarly by looking up the direction map $D$ and rotating it by 90 degrees to be in the same direction of $v_0$. We repeat this process until the end of the road boundary where we fall outside of the image. At the end we obtain a polyline prediction $x = (x_i)$ with vertices in $\mathbb{R}^2$ that captures the global topology of the road boundary.

Thus for each input $I$, we obtain a set of polylines emanating from the predicted endpoints. Note that we can assign a score to each polyline by taking the average of the detection scores on its vertex pixels. We use this score for two minimal post-processing steps: i) We remove the low scoring polylines protruding from potentially false negative endpoints. ii) Two predicted endpoints could correspond to the same road boundary giving rise to two different polylines. Thus, we look at all pairs of polylines and if they overlap by more than 30% in a dilated region around them, we only keep the highest scoring one.

**Network Architecture:** For each cropped ROI, we feed it through a CNN. We use the same encoder decoder backbone that we used to predict the deep features but with one less convolutional layer in both the encoder and decoder blocks. The output is a score map that we can take argmax of to obtain the next vertex for cropping.

### 3.3. Learning

**Deep visual features:** To learn the deep visual features, we use a multi-task objective, where the regression loss is used for the distance transform feature maps $S$ and $E$ and the cosine similarity loss is used for the direction map $D$. Thus:

$$\ell(S, E, D) = \ell_{det}(S) + \lambda_1 \ell_{end}(E) + \lambda_2 \ell_{dir}(D) \qquad (1)$$

In our experiments we set the loss weighting parameters $\lambda_1$ and $\lambda_2$ to be 10.

**Convolutional Snake:** Similar to [17], in order to match the edges of a predicted polyline $P$ to its corresponding ground truth road boundary $Q$, we use the Chamfer Dis-

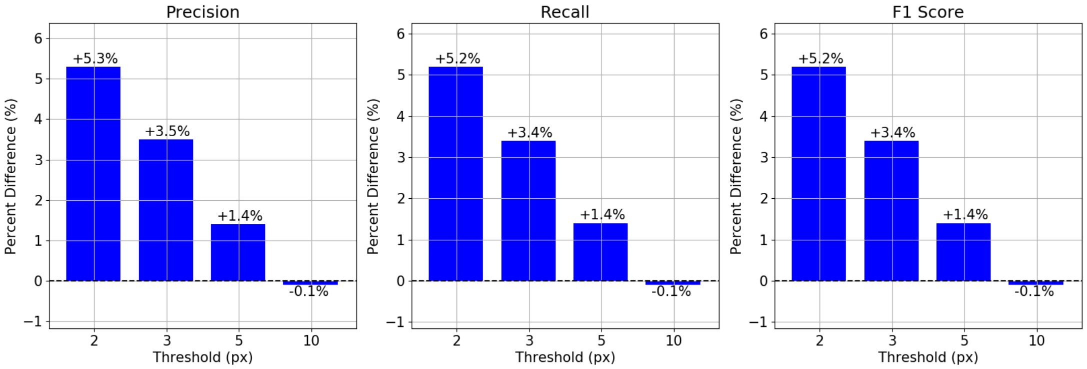

Figure 3. **Amortized learning:** In this figure we show the percent difference between our models for precision, recall and F1 score trained with and without amortized learning. We see amortized learning significantly improves the result at all thresholds of our metric. Connectivity is the same for both models.

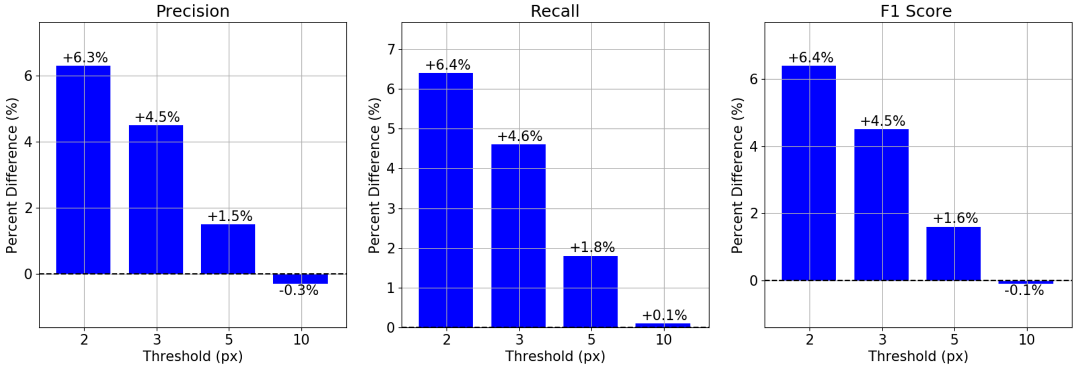

Figure 4. In this figure we show the percent difference between our models trained to predict the direction map versus predicting the dilated normals. We report the difference for precision, recall and F1 score. We see using the direction map significantly improves the result at all thresholds of our metric. Furthermore, the direction map method improves the connectivity by 1%.

tance defined as:

$$L(P,Q) = \sum_i \min_{q \in Q} \|p_i - q\|_2 + \sum_j \min_{p \in P} \|p - q_j\|_2 \quad (2)$$

where $p$ and $q$ are the rasterized edge pixels of the polylines $P$ and $Q$ respectively. This loss function encourages the edges of the predicted polyline to fall completely on its ground truth and vice versa. This is a more suitable loss function for matching two polylines rather than one penalizing the position of vertices. For example a ground truth straight line segment can be redundantly represented by three vertices rather than two and thus misleading the neural network when using a vertex based loss.

# 4. Experimental Evaluation

## 4.1. Experimental Details

**Dataset:** Our dataset consists of BEV projected LiDAR point clouds and camera imagery of intersections and other regions of the road from multiple passes of a self-driving vehicle in a major north American city. In total, 4750km were driven to collect this data from a 50 km$^2$ area with a total of approximately 540 billion LiDAR points. We then tile and split the dataset into 2500, 1000, 1250 train/val/test BEV images that are separated based on 2 longitudinal lines dividing the city. On average, our images are 1927px (±893) x 2162px (±712) with 4cm/px resolution.

In more detail, to create our dataset, we first drive around the areas we want to map several times. We exploit an efficient deep-learning based pointcloud segmentation algorithm [58] to remove moving objects and register the pointclouds using ICP. We then generate a very high resolution bird's eye view image given the pointcloud (i.e., 4cm/pixel). This process allows us to have views of the world without occlusion. We also generate panoramas using the camera rig in a similar fashion. To compute the elevation gradient image, we simply apply a sobel filter on the LiDAR elevation for both x and y directions and then take its magnitude.

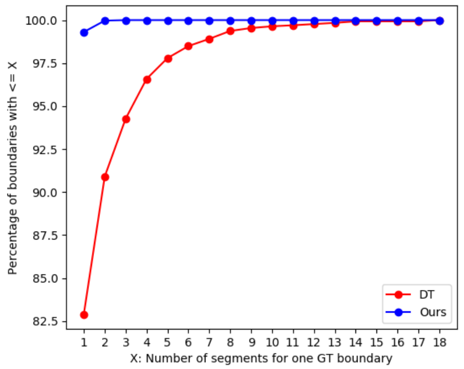

Figure 5. In this figure we show the cumulative percentage of GT boundaries with X number of predicted segments. For our model, we see that 99.3% of the GT boundaries have a single predicted boundary.

**Baseline:** Since this is a new task, we cannot compare to prior work. To create a competitive baseline, we first binarize the distance transform output of our method at a threshold and then skeletonize. For a fair comparison, we use grid search to find the threshold that gives us the best results. Next, we find the connected components and consider each as a predicted polyline.

**Implementation Details:** We trained our deep feature prediction model distributed over 16 Titan 1080 Ti GPUs each with a batch size of 1 using ADAM [21] with a learning rate of 1e-4 and a weight decay of 5e-4. We perform data augmentation by randomly flipping and rotating the images during training. The model is trained for 250 epochs over the entire dataset and takes 12 hours. The cSnake is also trained distributed on 16 Titan 1080 Ti GPUs each with a batch size of 1, ADAM, a learning rate of 5e-4 and a weight decay of 1e-4. The model is trained for 60 epochs over the entire dataset. During training, we give the network the ground truth end points and add ±16 pixels of noise. We also use amortized learning and train with ground truth direction and distance transform features 50% of the time. During training, we give the network the number of steps based on the length of the ground truth boundary plus 5 extra steps.

## 4.2. Metrics

Given a set of polyline predictions, we assign each one to a ground truth road boundary that has the smallest Hausdorff distance. Note that multiple predictions could be assigned to the same ground truth boundary but only one ground truth road boundary can be assigned to a prediction polyline. We now specify our metrics.

**Precision and Recall:** For precision and recall, we use the definition of [53] and specify points on the predicted polylines as either true positive or false positive if they are within a threshold of the ground truth polyline. False negatives are the points on the ground truth polyline that fall outside that threshold. We also combine the two metrics and report its result for harmonic mean (f1 score).

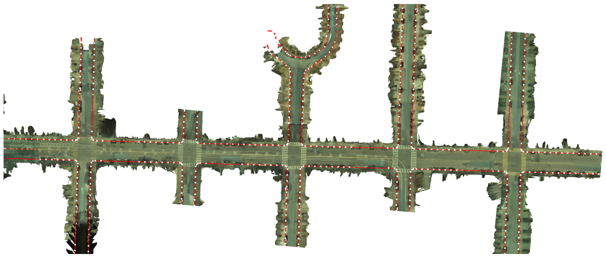

Figure 6. Example of our outputs stitched together (please zoom).

**Connectivity:** For each ground truth boundary, let $M$ be the number of its assigned predicted polylines. We define:

$$Connectivity = \frac{1(M > 0)}{M} \tag{3}$$

This metric penalizes the assignment of multiple small predicted segments to a ground truth road boundary.

**Aggregate Metrics:** We report the mean taken across the ground truth road boundaries at different thresholds.

## 4.3. Results

**Baseline:** As shown in Table 1, our method significantly outperforms the baseline in almost all metrics. The baseline is better at in precision at a threshold of 10px, however, this is because the baseline has lots of small connected components. However, in practice, these would be thrown out when doing actual annotation of road boundaries as they are too small to be useful.

**Sensor Modality:** We explore various sensor input combinations for our model. In Table 2, we show in line (4) that using every sensor input from L (lidar intensity), E (lidar elevation gradient) and C (camera) produces the best result. We perform an ablation studies by removing the camera input and then the elevation input, and also an experiment with camera only and show a significant performance drop.

**Amortized Learning:** The convolutional snake can be trained using either the ground truth or predicted deep features. For our best model, we train using half ground truth and half predicted deep features. We also train a model using only predicted deep features and show the difference in results of the two models in Figure 3. This figure shows the percentage difference between our model trained with

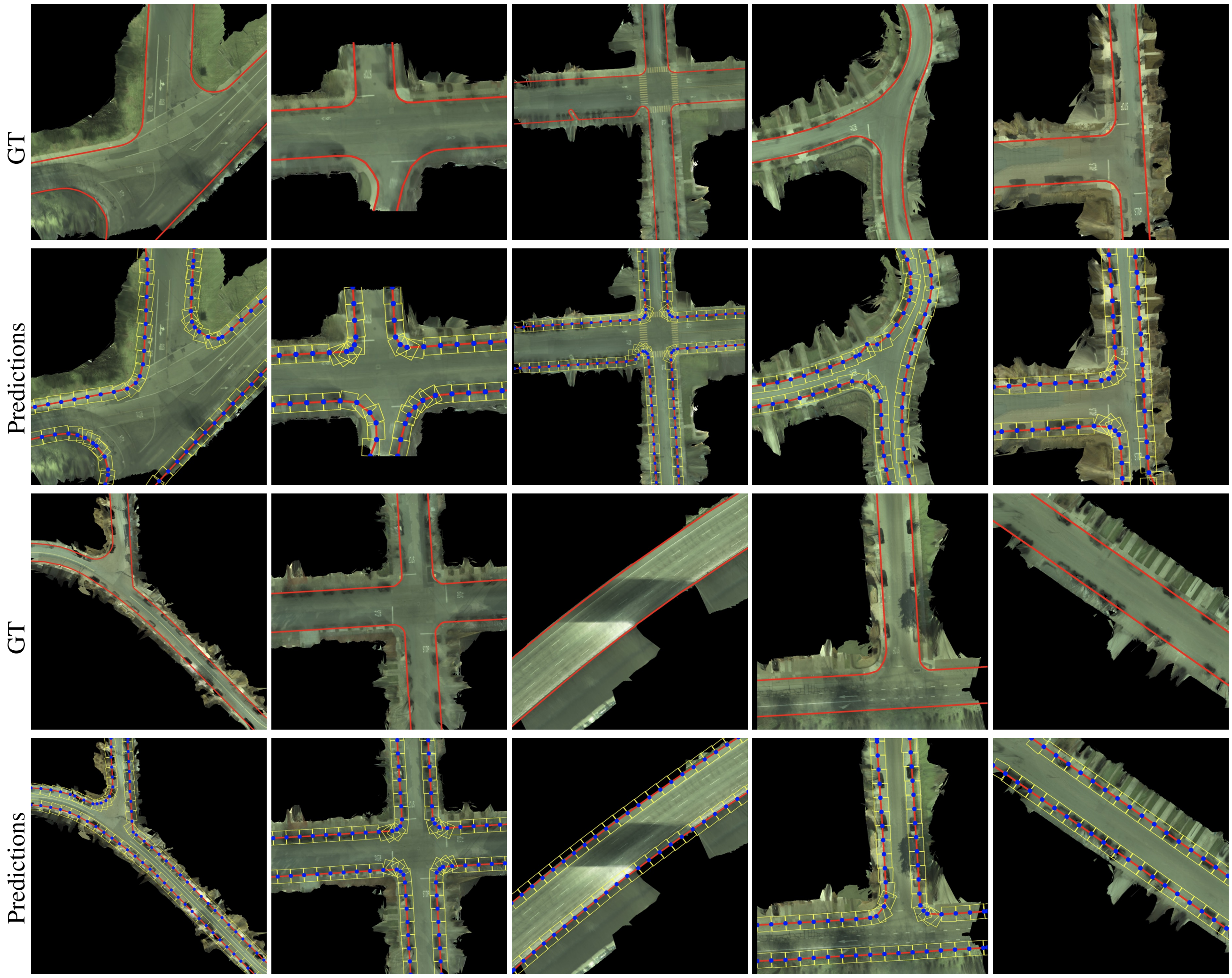

Figure 7. Qualitative results: (**Rows 1,3**) GT road boundaries displayed in red and overlaid on camera imagery. ( **Rows 2,4**) Road boundary prediction polylines. The blue dots correspond to the vertex outputs of the cSnake module. The yellow boxes are the rotated ROIs that the cSnake attends to while drawing the road boundary polylines. Note that we crop our imagery for better visualization. Please refer to the supplementary material for more complete visualization.

amortized learning and the model trained with only predicted features. At a threshold of 2px (8cm), the model trained with amortized learning is 5% better across all our metrics. In terms of connectivity, both methods of training achieve around the same result.

**Exploring Direction Prediction Alternatives:** We explore another method to predict the direction feature used by the convolutional snake. Here, we predict a pixel-wise direction at the boundary of the road that is the normal pointing into the road. Since this notion only exists at the road boundaries, we dilate the road boundary by 16 pixels and each pixel's direction will be equal to the normal of the closest road boundary pixel. However, the problem with this is that outside of this dilation, there will be no direction. We show in Figure 4 that our predicted direction map performs much better than these dilated normals. Here we show the percentage difference between the two models for all our metrics. Not shown in these figures is that our direction map method also produces a connectivity score that is 1% higher.

**Cumulative Distribution of Connectivity:** In Figure 5 we compare the number of predicted connected boundaries for each ground truth boundary for our model and the baselines. In this figure, we plot the cumulative distribution of the percentage of ground truth boundaries with X number of predicted segments. We show that our model significantly outperforms the baselines. For our model, 99.3% of the ground truth boundaries have a single predicted boundary whereas for the baselines this number is around 80%.

**Qualitative Results:** In Figure 8, we visualize the learnt features of our network given the input modalities. In particular, given the camera, LiDAR and the elevation gradient, our model outputs the dense location of the road boundaries, their endpoints as well as the vector field of the normal-

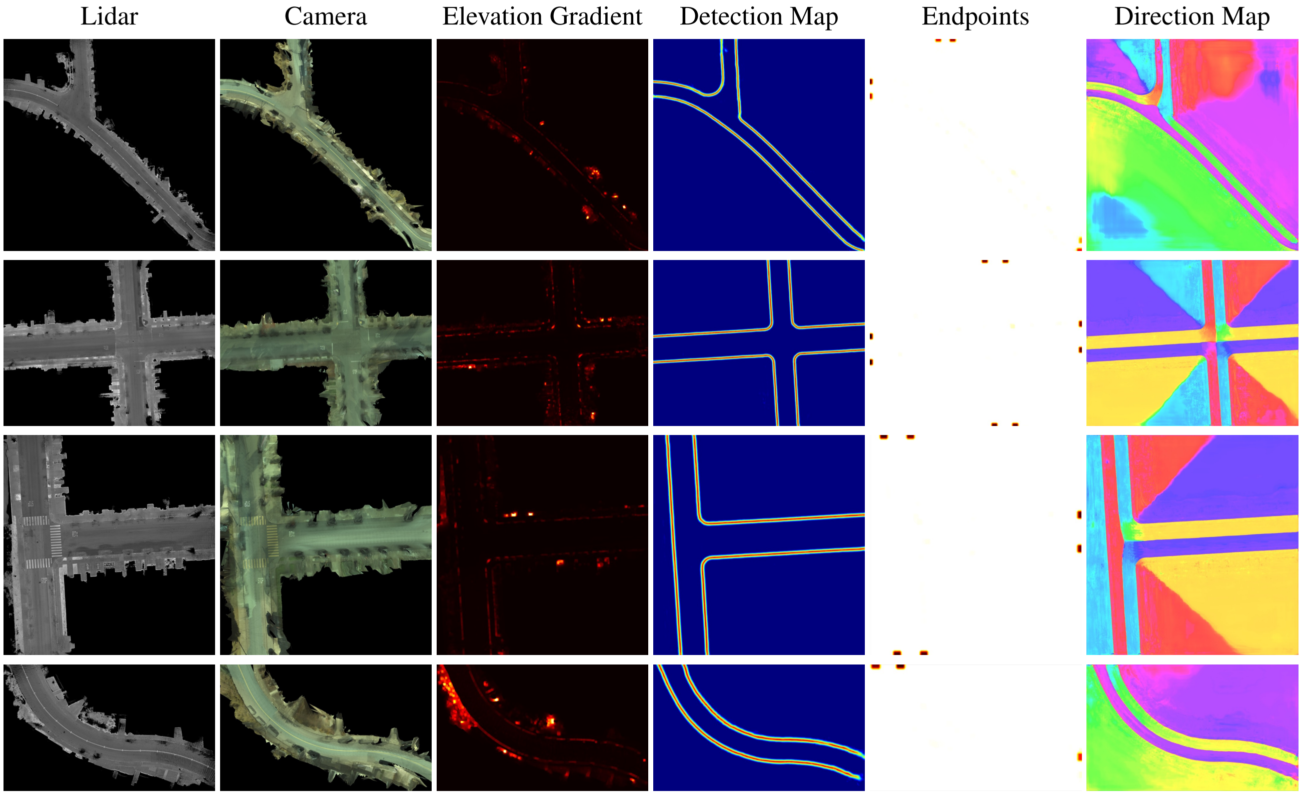

Figure 8. Deep Features: Columns **(1-3)** correspond to the inputs and columns **(4-6)** correspond to the deep feature maps. The direction map shown here as a flow field [7].

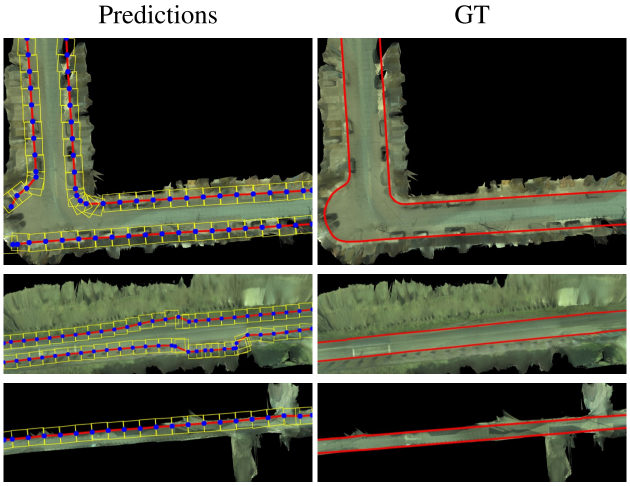

Figure 9. Failure Modes: **(Left)** predictions **(right)** GT.

ized normals to the road boundaries. These features aid the cSnake to output structured polylines for each road boundary as demonstrated in Figure 7.

**Stitched Results:** Our model is fully convolutional and independent of the image dimensions. Fig. 6 shows stitched examples of different crops on larger areas.

**Speed:** On a Titan 1080 Ti GPU, the average compute per image is 196ms for the deep features and 1.76s for the cSnake module.

**Failure Cases:** In Figure 9, we demonstrate a few failure cases of our model. On the first row, we can see that two disconnected polylines have been assigned to the same road boundary. In the second row, lower and bottom predictions deviate from the road boundaries before coming back. In the last row, one road boundary has not been captured.

## 5. Conclusion

In this paper, we proposed a deep fully convolutional model that extracts road boundaries from LiDAR and camera imagery. In particular, a CNN first outputs deep features corresponding to the road boundaries such as their location and directional clues. Then a our cSnake module, which is a convolutional recurrent network, outputs polylines corresponding to each road boundary. We demonstrated the effectiveness of our approach through extensive ablation studies and comparison with a strong baseline. In particular, we achieve F1 score of $87.2\%$ at 5 pixels away from the ground truth road boundaries with a connectivity of $99.2\%$. In the future, we plan to leverage aerial imagery alongside our LiDAR and camera sensors as well as extend our approach to other static elements of the scene.

## References

[1] David Acuna, Huan Ling, Amlan Kar, and Sanja Fidler. Efficient interactive annotation of segmentation datasets with polygon-rnn++. 2018. 3

[2] Jose M Alvarez, Theo Gevers, Yann LeCun, and Antonio M Lopez. Road scene segmentation from a single image. In *European Conference on Computer Vision*, pages 376–389. Springer, 2012. 1, 2

[3] José M Álvarez Alvarez and Antonio M Lopez. Road detection based on illuminant invariance. *IEEE Transactions on Intelligent Transportation Systems*, 12(1):184–193, 2011. 2

[4] Marie-Flavie Auclair-Fortier, Djemel Ziou, and Costas Armenakis. Survey of work on road extraction in aerial and satellite images. 12 2002. 2

[5] Min Bai, Gellert Mattyus, Namdar Homayounfar, Shenlong Wang, Kowshika Lakshmikanth, Shrinidhi, and Raquel Urtasun. Deep multi-sensor lane detection. In *IROS*, 2018. 3

[6] Ruzena Bajcsy and Mohamad Tavakoli. Computer recognition of roads from satellite pictures. *IEEE Transactions on Systems, Man, and Cybernetics*, 6:623–637, 1976. 2

[7] Simon Baker, Daniel Scharstein, J. P. Lewis, Stefan Roth, Michael J. Black, and Richard Szeliski. A database and evaluation methodology for optical flow. *International Journal of Computer Vision*, 92, 2011. 2, 8

[8] Ioan Andrei Bârsan, Peidong Liu, Marc Pollefeys, and Andreas Geiger. Robust dense mapping for large-scale dynamic environments. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7510–7517. IEEE, 2018. 3

[9] Ioan Andrei Barsan, Shenlong Wang, Andrei Pokrovsky, and Raquel Urtasun. Learning to localize using a lidar intensity map. In *Proceedings of The 2nd Conference on Robot Learning*, 2018. 1

[10] 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 *Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 2

[11] Matthias Butenuth and Christian Heipke. Network snakes: Graph-based object delineation with active contour models. *Mach. Vis. Appl.*, 23:91–109, 08 2012. 2

[12] Sergio Casas, Wenjie Luo, and Raquel Urtasun. Intentnet: Learning to predict intention from raw sensor data. In *Conference on Robot Learning*, pages 947–956, 2018. 1

[13] Lluis Castrejon, Kaustav Kundu, Raquel Urtasun, and Sanja Fidler. Annotating object instances with a polygon-rnn. In *CVPR*, 2017. 3

[14] Abhishek Chaurasia and Eugenio Culurciello. Linknet: Exploiting encoder representations for efficient semantic segmentation. *CoRR*, abs/1707.03718, 2017. 4

[15] Hsu-Yung Cheng, Bor-Shenn Jeng, Pei-Ting Tseng, and Kuo-Chin Fan. Lane detection with moving vehicles in the traffic scenes. *IEEE Transactions on intelligent transportation systems*, 7(4):571–582, 2006. 1, 2

[16] Andreas Geiger, Julius Ziegler, and Christoph Stiller. Stereoscan: Dense 3d reconstruction in real-time. In *Intelligent Vehicles Symposium (IV), 2011 IEEE*, pages 963–968. Ieee, 2011. 3

[17] Namdar Homayounfar, Wei-Chiu Ma, Shrinidhi Kowshika Lakshmikanth, and Raquel Urtasun. Hierarchical recurrent attention networks for structured online maps. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 3, 4

[18] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Advances in Neural Information Processing Systems*, pages 2017–2025, 2015. 4

[19] Soren Kammel and Benjamin Pitzer. Lidar-based lane marker detection and mapping. In *Intelligent Vehicles Symposium, 2008 IEEE*, pages 1137–1142. IEEE, 2008. 2

[20] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *International journal of computer vision*, 1(4):321–331, 1988. 2

[21] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *ICLR*, 2015. 6

[22] Hui Kong, Jean-Yves Audibert, and Jean Ponce. General road detection from a single image. *IEEE Transactions on Image Processing*, 19(8):2211–2220, 2010. 1, 2

[23] Tobias Kühnl, Franz Kummert, and Jannik Fritsch. Spatial ray features for real-time ego-lane extraction. In *Intelligent Transportation Systems (ITSC), 2012 15th International IEEE Conference on*, pages 288–293. IEEE, 2012. 1, 2

[24] Ivan Laptev, Helmut Mayer, Tony Lindeberg, Wolfgang Eckstein, Carsten Steger, and Albert Baumgartner. Automatic extraction of roads from aerial images based on scale space and snakes. *Machine Vision and Applications*, 12(1):23–31, 2000. 2

[25] Dan Levi, Noa Garnett, Ethan Fetaya, and Israel Herzlyia. Stixelnet: A deep convolutional network for obstacle detection and road segmentation. In *BMVC*, pages 109–1, 2015. 2

[26] 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. 3

[27] Ming Liang, Bin Yang, Shenlong Wang, and Raquel Urtasun. Deep continuous fusion for multi-sensor 3d object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 641–656, 2018. 1

[28] David Lieb, Andrew Lookingbill, and Sebastian Thrun. Adaptive road following using self-supervised learning and reverse optical flow. In *Robotics: Science and Systems*, pages 273–280, 2005. 2

[29] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. *arXiv preprint arXiv:1612.03144*, 2016. 4

[30] 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 *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 6292–6299. IEEE, 2017. 1

[31] Diego Marcos, Devis Tuia, Benjamin Kellenberger, Lisa Zhang, Min Bai, Renjie Liao, and Raquel Urtasun. Learn-

ing deep structured active contours end-to-end. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8877–8885, 2018. 2

[32] Ramesh Marikhu, Matthew N Dailey, Stanislav Makhanov, and Kiyoshi Honda. A family of quadratic snakes for road extraction. In *Asian Conference on Computer Vision*, pages 85–94. Springer, 2007. 2

[33] 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. *ISPRS Journal of Photogrammetry and Remote Sensing*, 135:158–172, 2018. 1, 2

[34] Dimitrios Marmanis, Jan D Wegner, Silvano Galliani, Konrad Schindler, Mihai Datcu, and Uwe Stilla. Semantic segmentation of aerial images with an ensemble of cnss. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 2016*, 3:473–480, 2016. 1, 2

[35] Gellért Máttyus, Wenjie Luo, and Raquel Urtasun. Deeproadmapper: Extracting road topology from aerial images. In *The IEEE International Conference on Computer Vision (ICCV)*, 2017. 1, 2

[36] Gellert Mattyus, Shenlong Wang, Sanja Fidler, and Raquel Urtasun. Enhancing road maps by parsing aerial images around the world. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1689–1697, 2015. 2

[37] Gellért Máttyus, Shenlong Wang, Sanja Fidler, and Raquel Urtasun. Hd maps: Fine-grained road segmentation by parsing ground and aerial images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3611–3619, 2016. 2

[38] Helmut Mayer, Ivan Laptev, and Albert Baumgartner. Multi-scale and snakes for automatic road extraction. In *European Conference on Computer Vision*, pages 720–733. Springer, 1998. 2

[39] 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. *Pattern recognition letters*, 26(9):1201–1220, 2005. 2

[40] Volodymyr Mnih and Geoffrey E Hinton. Learning to detect roads in high-resolution aerial images. In *European Conference on Computer Vision*, pages 210–223. Springer, 2010. 1, 2

[41] Volodymyr Mnih and Geoffrey E Hinton. Learning to label aerial images from noisy data. In *ICML*, 2012. 1, 2

[42] Rahul Mohan. Deep deconvolutional networks for scene parsing. *arXiv preprint arXiv:1411.4101*, 2014. 2

[43] Javier A Montoya-Zegarra, Jan D Wegner, L'ubor Ladickỳ, and Konrad Schindler. Mind the gap: modeling local and global context in (road) networks. In *German Conference on Pattern Recognition*, pages 212–223. Springer, 2014. 2

[44] Hans P Moravec and Alberto Elfes. High resolution maps from wide angle sonar. In *Proceedings of the IEEE Conference on Robotics and Automation*, pages 19–24, 1985. 2

[45] Lina Maria Paz, Pedro Piniés, and Paul Newman. A variational approach to online road and path segmentation with monocular vision. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 1633–1639. IEEE, 2015. 2

[46] Taihú Pire, Rodrigo Baravalle, Ariel D'Alessandro, and Javier Civera. Real-time dense map fusion for stereo slam. *Robotica*, pages 1–17, 2018. 3

[47] Marc Pollefeys, David Nistér, J-M Frahm, Amir Akbarzadeh, Philippos Mordohai, Brian Clipp, Chris Engels, David Gallup, S-J Kim, Paul Merrell, et al. Detailed real-time urban 3d reconstruction from video. *International Journal of Computer Vision*, 78(2-3):143–167, 2008. 3

[48] John A. Richards and Xiuping Jia. *Remote Sensing Digital Image Analysis*. Springer, 2013. 2

[49] Marie Rochery, Ian H Jermyn, and Josiane Zerubia. Higher order active contours. *International Journal of Computer Vision*, 69(1):27–42, 2006. 2

[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. 2

[51] Ceryen Tan, Tsai Hong, Tommy Chang, and Michael Shneier. Color model-based real-time learning for road following. In *Intelligent Transportation Systems Conference, 2006. ITSC'06. IEEE*, pages 939–944. IEEE, 2006. 2

[52] Carles Ventura, Jordi Pont-Tuset, Sergi Caelles, Kevis-Kokitsi Maninis, and Luc Van Gool. Iterative deep learning for road topology extraction. *arXiv preprint arXiv:1808.09814*, 2018. 1, 2

[53] 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. 6

[54] 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. *IEEE Transactions on Intelligent Transportation Systems*, 10(4):572–583, 2009. 1, 2

[55] Jan D Wegner, Javier A Montoya-Zegarra, and Konrad Schindler. A higher-order crf model for road network extraction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1698–1705, 2013. 2

[56] Jan Dirk Wegner, Javier Alexander Montoya-Zegarra, and Konrad Schindler. Road networks as collections of minimum cost paths. *ISPRS Journal of Photogrammetry and Remote Sensing*, 108:128–137, 2015. 2

[57] Jian Yao, Srikumar Ramalingam, Yuichi Taguchi, Yohei Miki, and Raquel Urtasun. Estimating drivable collision-free space from monocular video. In *Applications of Computer Vision (WACV), 2015 IEEE Winter Conference on*, pages 420–427. IEEE, 2015. 2

[58] Chris Zhang, Wenjie Luo, and Raquel Urtasun. Efficient convolutions for real-time semantic segmentation of 3d point clouds. In *2018 International Conference on 3D Vision (3DV)*, pages 399–408. IEEE, 2018. 5