

**Monocular Camera-Based Road Segmentation
using Geometric Cues**

Gong CHENG

A DISSERTATION SUBMITTED TO
THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Graduate Program in Computer Science

York University

Toronto, Ontario

November 2025

© Gong Cheng, 2025

Abstract

Vision-based road segmentation aims to identify drivable road regions in images captured by vehicle-mounted cameras. With the rapid development of autonomous driving, such methods have attracted substantial attention from both academia and industry due to their research significance and commercial potential. However, a key challenge lies in achieving robust adaptation—that is, effectively generalizing to diverse environmental and operational conditions.

In my Ph.D. research, I have concentrated on leveraging geometric information to advance monocular-camera-based road segmentation and adaptation. Specifically, I have completed three independent projects: (1) a supervised road segmentation approach that fuses geometric cues with appearance cues to enhance segmentation accuracy, (2) an unsupervised domain adaptation method that employs a novel geometry-guided strategy to reduce domain shift between source and target data, and (3) another unsupervised domain adaptation approach that uses known camera parameters and online estimation of the camera pan and tilt to better align imagery across and within datasets, improving accuracy and generalization. These contributions collectively push forward the development of robust, geometry-driven solutions for road scene understanding.

Acknowledgements

I would like to thank my supervisor, Professor James Elder, for his exceptional guidance. I did not learn only computer vision and machine learning methods but also gained a deeper and more systematic understanding of the research itself. His emphasis on clarity, rigor, and principled methodology has greatly enriched the way I formulate questions, design experiments, and evaluate ideas.

I would also like to thank all members of the Elder Lab, past and present, for their discussions and knowledge sharing. In particular, I would like to thank our senior research scientist, Dr. Bob Hou, for his help and time.

I would like to thank all the members of my oral examination committee, Professor Michael Brown, Professor Hui Jiang, Professor Gunho Song, and Professor Steven Waslander, for their thoughtful feedback, questions, and suggestions during my defense and thesis revision.

I am grateful to the Department of Electrical Engineering and Computer Science at York University for providing an excellent academic environment, resources, and support throughout my doctoral studies.

Last but not least, I would like to thank my parents for their constant encouragement and support during my Ph.D. journey.

Contents

Abstract	ii
Acknowledgements	iii
Table of Contents	iv
List of Tables	viii
List of Figures	ix
1 Introduction	1
1.1 Outline	2
2 Related Works	3
2.1 Hand-Crafted Feature-based Road Segmentation	3
2.1.1 Appearance-based Road Segmentation	3
2.1.2 Geometry-based Road Segmentation	13
2.1.3 Conclusion	23
2.2 DNN-based Road Segmentation	24
2.2.1 Commonly-used Road Segmentation Backbones	24
2.2.2 DNN-based Road Segmentation Methods	28
2.2.3 Conclusion	36
2.3 Unsupervised Domain Adaptation for Road Segmentation	37
2.3.1 CNN-based UDA Methods	38

2.3.2	GAN-based UDA Methods	44
2.3.3	Conclusion	47
3	Fusing-based Road Segmentation	49
3.1	Introduction	49
3.2	Contributions	51
3.3	Related Works	52
3.4	Datasets	53
3.5	Proposed Method	54
3.5.1	Geometric Prior	54
3.5.2	Learning Appearance Models	54
3.5.3	Selecting an Appearance Model	58
3.5.4	Fusion of Geometry with Appearance	59
3.5.5	Segmentation and Refinement	59
3.5.6	Implementation Details	62
3.6	Results	64
3.7	Conclusions and Future Research	65
4	Geometry-guided Domain Adaptation for Road Segmentation	68
4.1	Introduction	68
4.2	Contributions	69
4.3	Related Works	70
4.3.1	Road Segmentation	70
4.3.2	Unsupervised Domain Adaptation	72

4.4	Datasets	73
4.5	Proposed Method	76
4.5.1	Supervised Training	76
4.5.2	Unsupervised Adaptation	84
4.5.3	Inference	85
4.6	Implementation Details	86
4.6.1	Run Time	86
4.7	Evaluation	88
4.8	Conclusion and Future Research	91
5	VCSeg: Virtual Camera Adaptation for Road Segmentation	92
5.1	Introduction	92
5.2	Contributions	93
5.3	Related Works	94
5.4	Datasets	95
5.5	Proposed Method	96
5.5.1	Reprojection to Virtual Camera	98
5.5.2	Within-Camera Shift	104
5.5.3	Projective Completion	105
5.5.4	Supervised Training	106
5.5.5	Geometry-Driven Adaptation	106
5.6	Experimental Details	108
5.7	Results	109

5.7.1	Comparison with Prior Adaptation Methods	109
5.7.2	Incorporating Appearance Adaptation	110
5.7.3	Comparison with Other Architectures	112
5.7.4	Evaluating Projective Completion	113
5.7.5	Visualizing Shift in Geometric Camera Parameters	116
5.7.6	Qualitative Results	117
5.8	Conclusions and Future Research	117
6	Conclusion and Future Work	121
6.1	Conclusion	121
6.2	Future Work	122
	Bibliography	125

List of Tables

1	Quantitative comparison (GTA5 [66] → Cityscapes [21]). Numbers are % IoU. . . .	47
2	Quantitative comparison (SYNTHIA [68] → Cityscapes [21]). Numbers are % IoU.	47
3	Quantitative comparison (IOU for foreground) with SegNet [8] on test set.	65
4	The five road segmentation datasets used in this project.	73
5	Quantitative evaluation (IOU)	89
6	The five cameras and road segmentation datasets used in this work. Estimated values indicated by *.	96
7	Learning parameters used by GeoSeg [20] and our new method, VCSeg.	108
8	Average runtime for inference.	109
9	Quantitative comparison with prior adaptation methods, using a common SegNet architecture. Numbers are % IoU. See Table 7 for old and new training parameters.	110
10	Results of Geometry-driven adaptation in the virtual camera domain. Numbers are % IoU. Geo-Prior indicates segmentation based upon the geometric prior used by GeoSeg [20], mapped to the virtual camera.	111
11	Quantitative comparison with recent semantic segmentation architectures trained on Cityscapes. Numbers are % IoU.	112
12	Performance of VCSeg with and without projective completion. The last col- umn shows for reference the performance when ground truth vanishing points are used at inference. This provides an upper bound on performance gains achievable through improvements to vanishing point detection. Numbers are % IoU.	113

List of Figures

1	Vehicle-mounted camera-based road segmentation. Left: road scene; Right: segmentation result. Images from [18].	1
2	Bayesian Road segmentation example from [42].(a) Input road scene. (b) Road segmentation result.	7
3	Unsupervised texture-based road segmentation example from [98]. (a) Input road image. (b) Strength of texture anisotropy. (c) Unsupervised Segmentation.	10
4	Supervised texture-based road segmentation example from [98]. (a) Input road image. (b) Strength of texture anisotropy. (c) Supervised Segmentation.	10
5	Segmentation examples from [69]. Top: Input road images. Bottom: Segmentation results.	13
6	The panned plan-view image from [52]. Top: Original road image with overlay grid for transformation. Middle: Schematic view of the road in the vehicle's coordinate system, showing road alignment. Bottom: Transformed panned plan-view image.	15
7	Angle-based voting for vanishing point detection.	17
8	VP detection and road segmentation examples from [65]. Ellipses: Gaussian fits of human labels. Crosses: VPs detected by the proposed method.	17
9	Road vanishing point estimation examples from [3].	19

10	Horizon estimation from [3]. (a) Mean normalized luminance of training images of the horizon as a function of the vertical displacement from the horizon location. (b) The first three principal components of the vertical luminance distribution around the horizon location.	20
11	Geometric Priors [18]. (a) Unconditional prior. (b) Road vanishing point-based prior. (c) Road horizon-based prior.	22
12	Vanishing estimation examples from [3]. (a)-(c): Best, median, and worst vanishing point-based road segmentation results. (d)-(f): Best, median, and worst horizon-based road segmentation results	22
13	Architecture of FCN from [51].	25
14	Architecture of U-Net from [67].	26
15	Architecture of SegNet from [8].	27
16	Architecture of Multinet from [81].	28
17	Overview of reverse and boundary attention network from [76]. (a) Overall architecture. ASPP denotes atrous spatial pyramid pooling. (b) Residual refinement module.	29
18	Left: Reverse Attention (RA) unit. Right: Boundary Attention (BA) unit. Details are explained in the following paragraphs. Images are taken from [76].	30
19	Overview of s-FCN-loc from [87].	32
20	Architecture of LaneNet from [58]. $\otimes$ and clustering are described in the following paragraphs.	33
21	Workflow of the proposed road segmentation method from [5].	34
22	Overview of SLL-GAN from [32].	35

23	Flowchart of cross-domain sampling in DACS from [82].	39
24	Cross-domain data augmentation in DACS [82]. Objects are cropped from the source domain image X_S (a GTA5 image) using the ground truth label and then mixed with a target domain image X_T (a Cityscapes image). Finally, an augmented image X_M is generated.	39
25	Overview of Shallow Features Guided-UDA from [12].	42
26	Data augmentation for adaptation examples from [12]. Top-left: a target image and its corresponding prediction. Top-right: A source domain prediction pair on which we aim to do data augmentation. Bottom-left: the cropped person class from the top-right pair. Bottom-right: the cropped objects have been pasted to the target source domain prediction pair and the data augmentation is complete.	44
27	Overview of AdaptSegNet from [83].	45
28	Example images from the dataset that has varying weather conditions. Images taken from [18].	49
29	(a) Unconditional prior. (b) Prior anchored to road vanishing point. (c) Prior anchored to horizon.	55
30	Distributions over the number of mixture components for the optimized foreground k_i^F (a) and background k_i^B (b) models, over the training images.	57
31	Selecting an appearance model. (a) Target test image. (b) Foreground $\hat{Y}^F$ (red) and background $\hat{Y}^B$ (blue) appearance samples. (c-d) Training images from which optimal foreground $\mathcal{M}^F$ and background appearance models $\mathcal{M}^B$ are drawn for this test image.	60

32	Fusion of geometric and appearance cues for an example test image (a). (b) Geometric prior ratio map $p(x_i = F) / p(x_i = B)$. (c) Appearance likelihood ratio map.	61
33	Optimizing segmentation and refinement. (a) Optimal threshold on log posterior odds. (b) Optimal threshold on region size.	63
34	Mean performance on test set. Error bars indicate standard error of the mean. . . .	64
35	Example results. (a) Test image (b) Ground truth segmentation. (c) Geometric method [3]. (d) Fusion method (initial). (e) Fusion method (refined). (f) SegNet method (road + pavement, pretrained weights provided by the original authors). . .	66
36	Example image from each dataset. (a) CamVid. (b) KITTI. (c) Cityscapes. (d) Qian. (e) Toronto-2020.	74
37	Additional ground-truthing. (a) Ground truth vanishing point (b) Ego-vehicle mask	76
38	Overview of training, adaptation and inference processes.	77
39	Vanishing Point Detection. (a) Sample input image. (b) 105 top-ranked line segments detected by MCMLSD [4]. (c) Elongating line segments and producing pair-wise intersections. (d) Kernel Density Estimation on intersections. (e) Estimated vanishing point. (f) Ground truth vanishing point.	81
40	Vanishing Point Estimation Examples. Green dots: estimated vanishing points. Red dots: Ground truth vanishing points.	82

41	Geometry-guided neural network adaptation. (a) Geometric prior generated from the source domain ensemble. (b) Input target image. (c) Geometric prior anchored to the estimated vanishing point in the target image. (d) Tri-mask identifying high-confidence foreground (yellow) and background (blue) regions. Regions of intermediate confidence (red) are ignored and do not influence the adaptation loss. . . .	83
42	Fusing geometry cues with appearance cues. (a) Sample input image. (b) Uncalibrated softmax output of appearance sub-system (SegNet). (c) Temperature-scaled softmax map $p(F a)$. (d) Geometric Prior $p(F g)$. (e) Posterior odds $L \cdot P$ computed from (c) & (d).	87
43	Qualitative results. From top to bottom, target images from CamVid, KITTI, Cityscapes, Qian <i>et al.</i> , Toronto-2020 datasets. (a) Target image (b) Ground truth (c) SegNet (d) Geometry (e) Geo-guided SegNet adaptation (f) Fusion (g) Adapt-SegNet (VGG-16)	90
44	Representative failures from the KITTI test dataset. (a) Input test image. (b) Ground truth. (c) Geometric prior. (d) Adapted Softmax. (e) Fusion. (f) Fusion Segmentation Result.	90
45	Overview of the proposed method.	97
46	Camera geometry. PP denotes the principal point of the camera.	99
47	Ground plane back-projection of the lower quarter of the mean FOV for each camera.	101
48	Projection of physical camera FOVs to the virtual camera plane.	103

49	Projective completion. (a) Reprojected image from the CamVid training partition. Red point indicates ground truth annotated road vanishing point, red dashed lines the source boundary pixels and green lines the completion directions. (b) Projectively completed source image. (c) Projectively completed ground truth mask. . . .	107
50	Projective completion examples 1. Red dot indicates ground truth vanishing point and red lines indicate original image boundaries.	114
51	Projective completion examples 2. Red dot indicates ground truth vanishing point and red lines indicate original image boundaries.	115
52	Demonstration of the importance of projective completion on images from the Cityscapes dataset.	116
53	Averages across training datasets, computed in original source domains and in the virtual camera domain, after reprojection, correcting for both between-camera and within-camera shift. Source domain images have been reshaped to match virtual camera dimensions to ease comparison.	118
54	Successes: Image from each dataset for which our full VCSeg method obtains the highest IoU. Left to right: (1) input image, (2) ground truth, (3) SegNet, (4) GeoSeg, (5) adapt-SegNet, (6) VCSeg Between Cam, (7) VCSeg+Within Cam. Top to Bottom: CamVid, Cityscapes, KITTI, Qian, Toronto-2020.	119
55	Failures: Image from each dataset for which our full VCSeg method obtains the lowest IoU. Left to right: (1) input image, (2) ground truth, (3) SegNet, (4) GeoSeg, (5) adapt-SegNet, (6) VCSeg Between Cam, (7) VCSeg+Within Cam. Top to Bottom: CamVid, Cityscapes, KITTI, Qian, Toronto-2020.	119

56	Medians: Image from each dataset for which our full VCSeg method obtains the median IoU. Left to right: (1) input image, (2) ground truth, (3) SegNet, (4) GeoSeg, (5) adapt-SegNet, (6) VCSeg Between Cam, (7) VCSeg+Within Cam. Top to Bottom: CamVid, Cityscapes, KITTI, Qian, Toronto-2020.	120
----	--	-----

1 Introduction

Image segmentation is one of the fundamental tasks in computer vision and digital image analysis. It aims to partition an image into multiple meaningful and coherent regions, each corresponding to an object or part of a scene. By assigning a semantic label to every pixel, image segmentation provides a foundation for high-level vision applications such as object detection, scene understanding, and visual navigation. Traditional segmentation methods include techniques such as thresholding [61], clustering [54], and region growing [1]. With the advent of deep learning, segmentation has evolved into a data-driven problem, where convolutional neural network (CNN) [41]-based and transformer [85]-based architectures [8, 67, 100, 90] learn hierarchical visual representations from large annotated datasets.



Figure 1: Vehicle-mounted camera-based road segmentation. Left: road scene; Right: segmentation result. Images from [18].

Road segmentation is a subfield of image segmentation that involves extracting road surfaces from road scenes. With advances in deep neural networks (DNNs) and autonomous driving, vehicle-mounted camera-based road segmentation has gained significant attention from both academia and industry. It plays a critical role in applications such as autonomous driving, road

condition assessment, traffic analytics, and urban scene parsing.

A major challenge is unsupervised domain adaptation (UDA), where a segmentation model trained on a labeled source domain must generalize to an unlabeled target domain. Domain shifts arise from variations in appearance (lighting, color, weather) and geometry (camera parameters, camera tilt/pan, road structure). In this thesis, I examine how geometric cues can help mitigate such domain shifts and improve adaptation performance.

1.1 Outline

This dissertation consists of the following chapters:

- **Chapter 2:** Related Works
- **Chapter 3:** Fusing geometry and appearance for supervised road segmentation. ICCV Workshops, 2017.
- **Chapter 4:** Geometry-Guided Adaptation for Road Segmentation. CRV 2020.
- **Chapter 5:** VCSeg: Virtual Camera Adaptation for Road Segmentation. WACV 2022.
- **Chapter 6:** Conclusion.

2 Related Works

2.1 Hand-Crafted Feature-based Road Segmentation

This subsection provides an overview of early road segmentation approaches that rely on hand-crafted features. Before the emergence of deep neural networks, researchers developed methods leveraging carefully designed appearance and geometry cues to detect drivable road regions. These foundational techniques paved the way for modern DNN-based road segmentation methods, which will be discussed in subsequent sections.

Hand-crafted features for road segmentation can be obtained from either appearance or geometry cues. In Section 2.1.1, we review the evolution of appearance-based methods. In Section 2.1.2, we introduce geometry-based road segmentation methods.

2.1.1 Appearance-based Road Segmentation

2.1.1.1 Thresholding Pixels can be segmented into meaningful groups based on various appearance cues such as color, intensity, and texture. A simple method for such segmentation is thresholding, which has been one of the pioneering image segmentation methods. One of the most widely used automatic thresholding methods is Otsu’s method [61], which has found numerous applications in computer vision [72, 71, 57, 35].

Otsu’s Method. Consider a grayscale image in which we want to categorize all the pixels into two classes: foreground (**f**) and background (**b**). Otsu’s method solves this binary categorization by thresholding pixels’ grayscale values. The optimal threshold is defined as the one that can

generate the least intra-class variance within the foreground and background groups. To find this optimal threshold, Otsu's method iteratively computes all the intra-class variances using all the possible threshold values. More concretely, for a given grayscale image, Otsu's method tries all the possible threshold values (i.e., in $[0,255]$) to compute intra-class variance and then picks the one with the minimal intra-class variance as the final threshold.

The intra-class variance can be computed using

$$\sigma_{intra}^2(t) = \omega_f(t)\sigma_f^2(t) + \omega_b(t)\sigma_b^2(t) \quad (1)$$

where t represents the current selected threshold, $\sigma_{intra}^2(t)$ indicates the intra-class variance given t , $\omega_f(t)$ and $\omega_b(t)$ are the probabilities of occurrence in foreground and background groups given threshold t , and $\sigma_f^2(t)$ and $\sigma_b^2(t)$ are the variances of foreground and background.

$\omega_f(t)$ and $\omega_b(t)$ can be computed using:

$$\omega_f(t) = \frac{N_f}{N_{all}} \quad (2)$$

$$\omega_b(t) = \frac{N_b}{N_{all}} \quad (3)$$

where N_f & N_b denote all the pixels that are labelled as foreground and background respectively, and N_{all} denotes the total number of pixels.

Foreground & background variances can be computed using:

$$\sigma_f^2(t) = \frac{\sum_{i=1}^{N_f} (x_i - \bar{x}_f(t))^2}{N_f - 1} \quad (4)$$

$$\sigma_b^2(t) = \frac{\sum_{i=1}^{N_b} (x_i - \bar{x}_b(t))^2}{N_b - 1} \quad (5)$$

where x_i denotes pixel i 's grayscale value, $\bar{x}_f(t)$ & $\bar{x}_b(t)$ denote the estimated mean grayscale value of the foreground and background using threshold t .

The authors proved that intra-class variance is minimized when inter-class variance is maximized.

The inter-class variance can be computed using:

$$\sigma_{inter}^2(t) = \omega_f(t)\omega_b(t)(\bar{x}_f(t) - \bar{x}_b(t))^2 \quad (6)$$

Thresholding-Based Road Segmentation. In 1987, Turk *et al.* [84] proposed one of the earliest road segmentation methods based on thresholding. The inputs to the method are RGB images captured by a color camera mounted on an Autonomous Land Vehicle, and the goal is to segment each input image into two classes: road and non-road. The authors found that the green channel provided little information helpful in road/non-road segmentation and therefore defined the problem in R-B space. They sampled image pixels to determine a roughly optimal linear decision boundary between road and non-road pixels.

This paper has several limitations that need to be addressed. Firstly, like any thresholding-based method, it suffers from the problem of having to rely on a single threshold to segment road and nonroad pixels, which can lead to false positives and negatives. Secondly, the method for finding the optimal threshold is somewhat heuristic, and it is unclear how the method's approach compares to other thresholding methods. Finally, the paper lacks both qualitative and quantitative evaluations to assess the method's performance and compare it to other methods.

2.1.1.2 Probabilistic Methods In 1988, Kuan *et al.* [42] proposed a Bayesian road segmentation method for autonomous robot vehicle navigation. They first converted frames from RGB space to a 1D color space by weighting the R, G, and B channels with the weights (w_r, w_g, w_b) that maximize the discrepancy between the road & non-road sample pixels. Next, the authors defined two priors for road (i.e., $P(\text{road})$) and nonroad (i.e., $P(\text{nonroad})$). For a pixel with value x in the target frame, the posterior odds can be computed using Bayes' theorem:

$$\frac{P(x|\text{road})P(\text{road})}{P(x|\text{nonroad})P(\text{nonroad})} \quad (7)$$

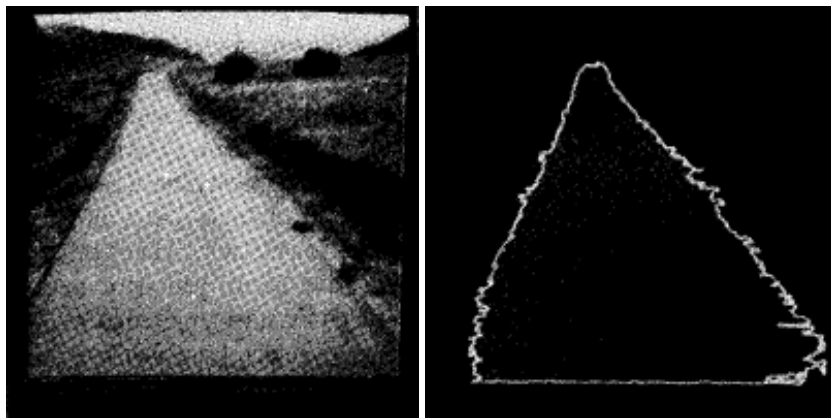
Where $P(x|\text{road})$ and $P(x|\text{nonroad})$ are the likelihoods conditioned on the road and nonroad distributions generated from previous frame segmentation results. To initiate this method, for the first frame, a manually selected reference window at the bottom center of the estimated image is created and sampled to get a probability density function of the road colors. The normalized histogram of the entire image is employed to estimate the probability density function of pixel colors, encompassing both road and nonroad. The conditional probability density function for the nonroad class can be acquired through a weighted linear combination of the above two histograms according to:

$$P(x) = P(x|\text{road})P(\text{road}) + P(x|\text{nonroad})P(\text{nonroad}) \quad (8)$$

To identify a pixel as a road pixel, we need to satisfy:

$$\frac{P(x|\text{road})P(\text{road})}{P(x|\text{nonroad})P(\text{nonroad})} > 1 \quad (9)$$

Road and nonroad histograms are updated iteratively frame by frame. When all the pixels in frame t are classified, those histograms are updated using the new classification results for computing the likelihoods for frame $t + 1$. A qualitative example is shown in Fig. 2



(a)

(b)

Figure 2: Bayesian Road segmentation example from [42].(a) Input road scene. (b) Road segmentation result.

2.1.1.3 Texture Image texture describes the spatial arrangement of intensities or colors in an image. Several early road segmentation methods were based on texture analysis, such as those proposed by [98, 9, 28, 72, 43, 65].

For example, Zhang *et al.* [98] proposed an oriented texture-based road segmentation method for extracting roads without marked lanes or strong differences relative to their surroundings. This method involves two main steps: (1) texel extraction and (2) texture-based road segmentation. The detailed implementation is described below.

1. **Texel Extraction.** The input road image $i(x,y)$ is first filtered using a Gaussian gradient kernel:

$$\nabla f(x, y) = \nabla g(x, y) * i(x, y) \quad (10)$$

where $\nabla g(x, y)$ the Gaussian gradient kernel and $\nabla f(x, y) = [f_x, f_y]^T$ is the output of filtering. The authors then compute at each pixel a local gradient covariance matrix (Eq. 11).

$$\Sigma' = \begin{bmatrix} \sigma_{xx} & \sigma_{xy} \\ \sigma_{yx} & \sigma_{yy} \end{bmatrix} \quad (11)$$

For example:

$$\sigma_{xx} = \sum_{i=-n}^n \sum_{j=-n}^n \omega(i, j) f_x^2(x + i, y + j) - \mu_x^2 \quad (12)$$

$$\mu_x = \sum_{i=-n}^n \sum_{j=-n}^n \omega(i, j) f_x(x + i, y + j) \quad (13)$$

where ω is a Gaussian weighting function in the proposed method, and the covariance is computed within a 5×5 window (i.e., $n = 2$).

Zhang *et al.* observed that road and non-road regions often differ in the degree to which surface texture is oriented (anisotropy). To capture this, Zhang *et al.* [98] proposed a novel

eigenvalue-based analysis. The eigenvalues of Σ' can be computed using:

$$\det(\Sigma' - \lambda * I) = 0 \quad (14)$$

$$(15)$$

where I represents a 2×2 unit matrix. Finally, λ can be computed as:

$$\lambda_1 = \frac{1}{2}(\sigma_{xx} + \sigma_{yy} + \sqrt{(\sigma_{xx} - \sigma_{yy})^2 + 4\sigma_{xy}^2}) \quad (16)$$

$$\lambda_2 = \frac{1}{2}(\sigma_{xx} + \sigma_{yy} - \sqrt{(\sigma_{xx} - \sigma_{yy})^2 + 4\sigma_{xy}^2}) \quad (17)$$

λ_1 is defined as the larger eigenvalue. A measure s of the anisotropy in the local gradient distribution can be computed using the following equation:

$$s = \frac{\lambda_1^2 - \lambda_2^2}{c\lambda_1}, \quad (18)$$

where c is a normalization factor predefined by the authors.

2. **Two-Phase Road Segmentation.** In the first phase, the authors combined this oriented texture feature with the image coordinates: $(s_i, x_i, y_i)^T$ and employed a modified version of the unsupervised isodata clustering algorithm proposed by Duda and Hart [25] to divide the image into three classes (one for road and two for non-road). In the following phase, the authors leveraged a Bayesian classifier for supervised segmentation. This classifier assigns pixels to classes based on a posteriori probability. Notably, these models are derived from the initial phase for the first image and subsequently updated using the prior image's segmentation results.

One notable limitation of this paper is that not all the roads have strong or well-defined textures. For example, gravel roads might pose challenges for this approach. Moreover, texture analysis

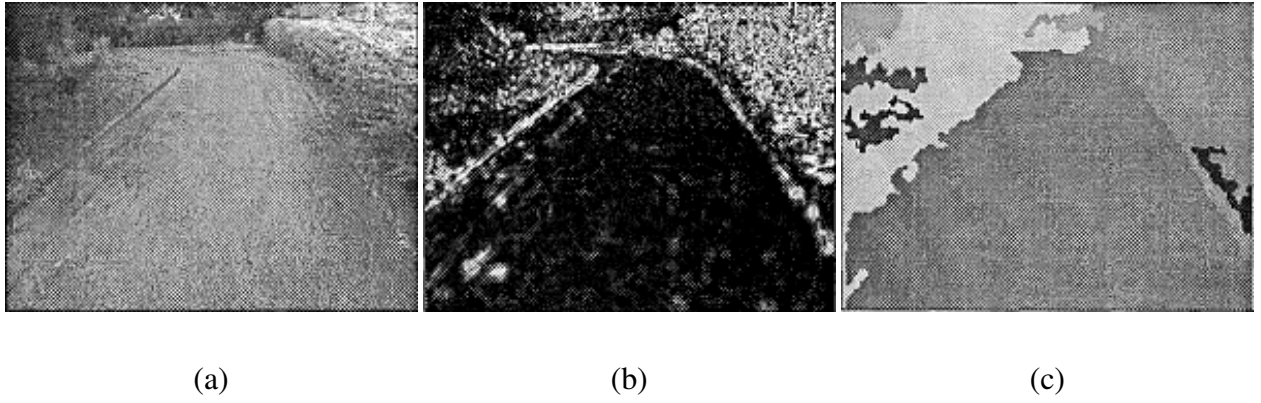


Figure 3: Unsupervised texture-based road segmentation example from [98]. (a) Input road image. (b) Strength of texture anisotropy. (c) Unsupervised Segmentation.

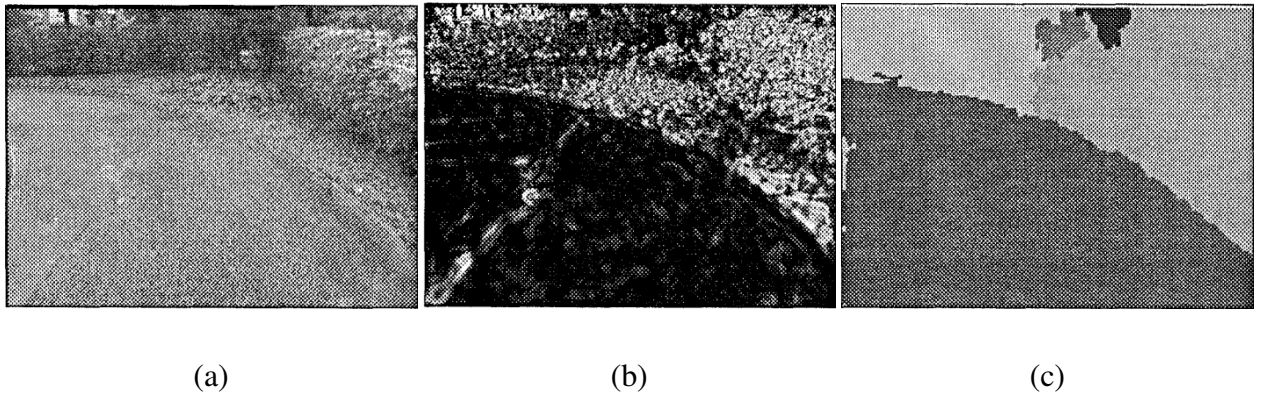


Figure 4: Supervised texture-based road segmentation example from [98]. (a) Input road image. (b) Strength of texture anisotropy. (c) Supervised Segmentation.

might struggle with occlusions like parked cars or shadows cast by objects on the road. These can disrupt the continuity of textures and lead to segmentation errors.

2.1.1.4 Other Color Spaces Converting from RGB to HSI color space in image processing offers several advantages. HSI separates color information from intensity, making it easier to analyze color-specific properties. For example, the saturation (i.e., S) channel is roughly illumination-invariant, reducing sensitivity to lighting changes. It is also useful for color-based feature extraction, and color region segmentation. The intensity (i.e., I) channel isolates the brightness information from the color information in an image.

The conversion between RGB and HSI can be done using the Eqs. 19 - 21.

$$H = \arccos \left(\frac{\frac{1}{2} \times [(R - G) + (R - B)]}{\sqrt{(R - G)^2 + (R - B)(G - B)}} \right) \quad (19)$$

$$S = 1 - \frac{3}{R + G + B} \times \min(R, G, B) \quad (20)$$

$$I = \frac{R + G + B}{3} \quad (21)$$

HSI color space has been applied to multiple image segmentation methods, e.g., [89, 97, 37, 38, 92]. Rotaru *et al.* [69] proposed a segmentation method for autonomous driving using only saturation (S) and intensity (I) channels.

Unlike conventional road segmentation methods, which commonly set the road region as the foreground, their proposed segmentation method focuses on segmenting (1) road regions, (2) lane markings, and (3) skies from frames captured by a camera mounted on a vehicle on a highway. In other words, the foreground was roads + lane markings + sky without distinctions between these three categories. This is because the goal was not road segmentation, but rather detection of possible obstacles.

1. **Outsider & Insider.** The authors used the terms insider and outsider to refer to foreground and background. Insider indicates the roads, lane markings, and skies. Outsider indicates other regions (i.e., other surroundings and vehicles).
2. **Reference Point Selection.** In the initial step, reference points were manually selected from insider and outsider classes.
3. **Customized Distance.** Next, for saturation S and intensity I values at a particular pixel, the proposed method computes distance to outsider values using:

$$F_3(S_o, I_o, S, I) = \frac{(S - S_p)^2 + (I - I_p)^2}{\text{Divider}(S_o, I_o, S_i, I_i)} \quad (22)$$

$$\text{Divider}(S_o, I_o, S_i, I_i) = \frac{(S_i - S_o)^2 + (I_i - I_o)^2}{255} \quad (23)$$

where (S_o, I_o) and (S_i, I_i) denote the cluster centers of outsider and insider pixels respectively. (S_p, I_p) represents the saturation and intensity values of a reference point in the SI plane used for image segmentation.

4. If (S_p, I_p) is an outside reference point and F_3 is smaller than a threshold, the pixel will be categorized as outsider (environment) class. The authors select 0.3 as the threshold. Some segmentation examples are shown in Fig. 5.

The proposed method has some notable limitations: (1) No quantitative evaluation was performed to assess the performance of the method. (2) The authors did not provide sufficient details about how to sample points for insiders and outsiders, which may affect the robustness of the method. (3) Similarly, they did not provide enough information about the threshold selection process, which may impact the accuracy of the segmentation. (4) The authors did not justify their decision to ignore the hue channel, which may contain valuable information for road segmentation.



Figure 5: Segmentation examples from [69]. Top: Input road images. Bottom: Segmentation results.

2.1.2 Geometry-based Road Segmentation

In some scenarios, appearance-based methods are not robust enough for road segmentation, such as when the road surface is covered by ice or snow or when nearby objects (e.g., sidewalks or buildings) have a similar appearance to the road. Therefore, alternative methods have been proposed that use geometric cues, such as road vanishing points or road boundaries, for road segmentation.

2.1.2.1 Road Boundary-based Methods Many road boundary-based road segmentation methods have been proposed, e.g., [80, 52, 75, 7].

In 1990, Lotufo *et al.* [52] proposed a road segmentation algorithm that relies on detected lane boundaries within the bird’s-eye view (BEV) space. The initial assumption made by the authors was that all target roads were nearly straight. The other assumptions are the road is flat and the camera intrinsics and extrinsics are known. The proposed method encompasses the following

steps:

1. Begin with a digitized image captured by a vehicle camera and convert it to a BEV image (Fig. 6).
2. Generate the gradient map of the BEV image and then apply a vertical gradient edge detector to identify a group of vertical lines.
3. Convert the detected vertical lines from the BEV space to the original image space. Link the top and bottom endpoints of each line with their adjacent neighbor line's top and bottom endpoints.
4. The contour formed by these linked lines constitutes the road segmentation result.

The algorithms were implemented in Parallel C on an image processing system based on transputers. This system was designed and constructed at the University of Bristol as a component of its Autonomously Guided Road Vehicle project. For testing purposes, a compact tracked battery-operated vehicle navigates a gravel path within the University gardens. There are a couple of notable limitations of this method:

1. No mathematical details about the BEV transformation and feature extraction are provided. For example, there is no explanation about how they did edge detection and what kernels were used.
2. There is no quantitative evaluation.

2.1.2.2 Vanishing Point-based Methods Assuming a straight and flat road, the vanishing point (VP) of the road can be defined as the point in the image where lane boundary lines converge. Many

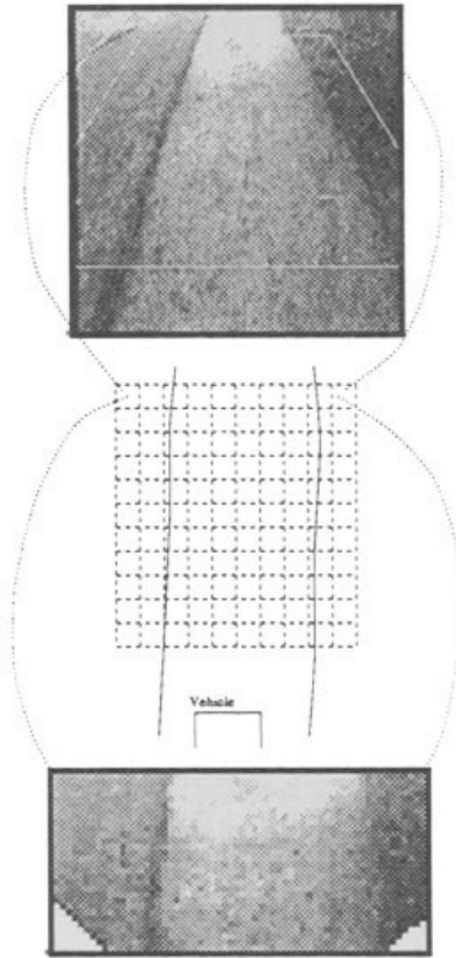


Figure 6: The panned plan-view image from [52]. Top: Original road image with overlay grid for transformation. Middle: Schematic view of the road in the vehicle's coordinate system, showing road alignment. Bottom: Transformed panned plan-view image.

VP-based road segmentation methods [3, 39, 64, 65] have been proposed. Here, we introduce some representative VP-based road segmentation methods.

In 2004, Rasmussen proposed a method [65] for both (1) vanishing point detection in straight and curved roads and (2) road segmentation based on the straight road vanishing point. Here, we focus on the latter, i.e., road segmentation based on the straight road vanishing point. The proposed approach employs texture analysis to detect the vanishing point in scenes in which the texture of the roads are homogeneous and lack distinct road boundaries. The method involves two main steps: (1) detecting the vanishing point on the straight road using Gabor filter-based [47] textures and (2) performing road segmentation with constraints based on the detected vanishing point.

Straight Road VP detection. For each pixel $\mathbf{p} = (x, y)$ in the image, a dominant texture orientation $\theta(p)$ is computed using Gabor filter-based textures. The author proposed a voting scheme based on these orientations to find the road VP. For a selected VP candidate $\mathbf{v}$, all the pixels within a voting region $R(\mathbf{v})$ are allowed to vote:

$$vote(\mathbf{p}, \mathbf{v}) = \begin{cases} 1 & \text{if } |\alpha(\mathbf{p}, \mathbf{v}) - \theta(\mathbf{p})| \leq \frac{n}{2\pi} \\ 0 & \text{otherwise} \end{cases} \quad (24)$$

$$votes(\mathbf{v}) = \sum_{\mathbf{p} \in R(\mathbf{v})} vote(\mathbf{p}, \mathbf{v}) \quad (25)$$

where $\alpha(p, v)$ denotes the angle of the vector (p, v) (Fig. 7), and $\frac{n}{2\pi}$ is an angular threshold for voting. Since road textures like tire tracks converge at the VP due to perspective, pixels vote based on their alignment with the candidate VP. For each candidate $\mathbf{v}$, its voting region $R(\mathbf{v})$ is set to all the pixels below the horizontal line passing through $\mathbf{v}$. Finally, the VP is identified as the candidate that receives the maximum number of votes.

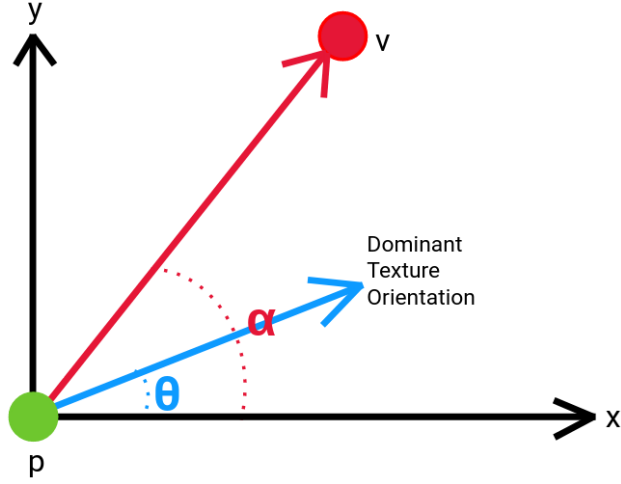


Figure 7: Angle-based voting for vanishing point detection.

Road Segmentation using a Straight Road VP. Left and right road boundaries are defined as line segments from the VP to the image frame (Fig. 8). The road region is defined as the image region below the road boundaries.



Figure 8: VP detection and road segmentation examples from [65]. Ellipses: Gaussian fits of human labels.

Crosses: VPs detected by the proposed method.

To evaluate the performance on straight road VP detection and segmentation, the authors used 16 samples from the "Mojave" road scene dataset (Images were provided by Caltech's Grand Challenge team, no reference). Human labelers were asked to label the vanishing point (VP) positions

on the sampled images, and the proposed algorithm’s performance was compared to that of the human markings. Road segmentation examples are provided in Fig. 8.

In 2009, Kong *et al.*, proposed a vanishing point and road boundary-based road segmentation method [39]. As in [65], this method consists of two main steps: (1) vanishing point detection and (2) vanishing point-based road boundary estimation.

For vanishing point detection, the authors first computed texture orientations using a Gabor filter. Next, pixels with high-orientation responses were grouped according to their texture orientations. Finally, the grouped pixels with texture orientations pointing upward were used to detect the vanishing point. The vanishing point (VP) was identified as the pixel that intersected with the most texture orientations. The voting points have to be below the vanishing point. To avoid hard voting bias towards the points that are higher in the image (since more validated voting pixels are below it), the authors proposed a locally adaptive soft-voting (LASV) technique to balance the voting.

After the vanishing point was detected, for road segmentation, a group of dominant edges constrained by VP was identified. Among all the edges, the two most prominent edges are then selected as the boundaries defining the road. Finally, the road segmentation was identified as the region that was enclosed by the bottom of the image and two road edges.

Compared with [65], one main novelty of this method is that the authors tried to avoid hard voting bias by using the LASV. However, this proposed method also has some notable problems: (1) the authors assumed that all roads are triangular-shaped. (2) The method is too heuristic, with too many free parameters and thresholds that were empirically chosen to obtain the final performance, making this method non-adaptive to other datasets.

In 2016, Almazan *et al.* [3] presented a road segmentation method that relies on vanishing points and the horizon to estimate a triangular segmentation hypothesis for images captured from varying camera positions. The method involves the following steps:

1. **Vanishing Point Estimation.** To estimate the road vanishing point on a given image, the authors first do line detection using the method proposed by Tal *et al.* [78]. Next, all the lines are ranked according to the likelihood ratio which was acquired from the geometric cues. More concretely, geometric cues involve: (1) position ρ , (2) orientation θ , and (3) length l . For a given line i , its likelihood ratio of vanishing point (ON) versus background (OFF) can be computed using a naive Bayes assumption:

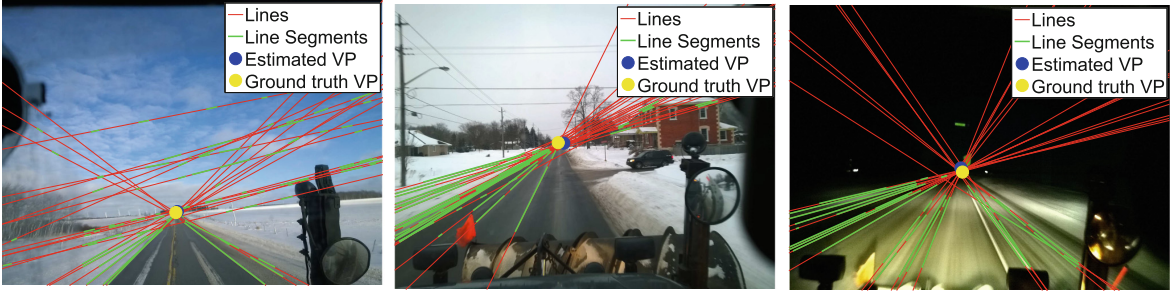


Figure 9: Road vanishing point estimation examples from [3].

$$L_i = \frac{p(\rho_i, \theta_i, l_i | ON)}{p(\rho_i, \theta_i, l_i | OFF)} \approx \frac{p(\rho_i | ON)p(\theta_i | ON)p(l_i | ON)}{p(\rho_i | OFF)p(\theta_i | OFF)p(l_i | OFF)} \quad (26)$$

The 20 lines with the highest likelihood ratio are selected and then the road VP is estimated at the point that minimizes the RMS distance to these 20 lines. The authors further extended this maximum likelihood method to include a prior over the location of p :

$$p(v | L) \propto \prod_i^n p(d_i | v)p(v) \quad (27)$$

Where v denotes the vanishing point and d_i is the distance to the detected line i . $p(v)$ is a spatial prior modeled as a Gaussian distribution and $p(d_i|v)$ is the likelihood. This non-convex optimization is solved using MATLAB `fminsearch`. Some vanishing point estimation examples are shown in Fig. 9.

2. **Vanishing Point Reliability Check.** If the average distance $\bar{D}_k$ of the estimated VP to the top k lines exceeds a threshold t , it is rejected as unreliable. Optimal values for k and t were determined based on labelled training data using `fminsearch`.

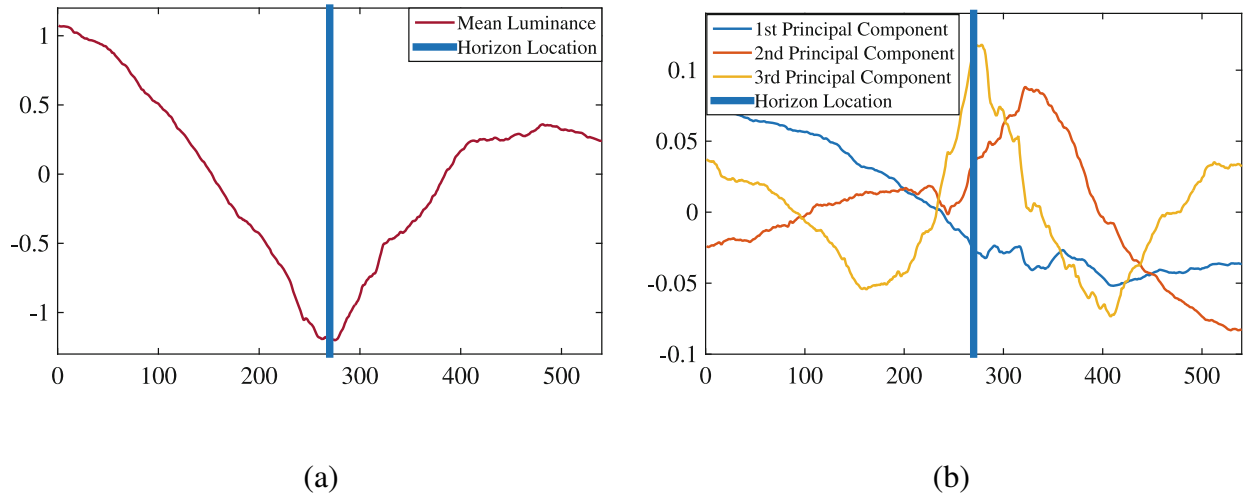


Figure 10: Horizon estimation from [3]. (a) Mean normalized luminance of training images of the horizon as a function of the vertical displacement from the horizon location. (b) The first three principal components of the vertical luminance distribution around the horizon location.

3. **Horizon Estimation.** If there is no reliable vanishing point that can be estimated, a horizontal horizon detection is performed. To estimate the vertical location of the horizon, the authors utilized the ground truth horizons in the training dataset from which vanishing point estimates were deemed unreliable. The following steps were taken: (1) Each image was

normalized to have zero mean and unit variance in all 3 color channels. (2) The images were vertically registered to ensure that their horizons were aligned. (3) The luminance channel was extracted from each image. (4) Luminance was averaged over horizontal position (as shown in Fig. 10 (a)). (5) The resulting averaged luminance was cropped to extract a vector l_i of length n , centered at the horizon. (6) Finally, the first m principal components u_i (as shown in Fig. 10 (b)) of length n were computed over these k vectors.

During the inference phase, the horizon of a target image can be estimated by performing the following steps: (1) Normalize the image to zero mean and unit variance, (2) Compute the luminance of the image, (3) Average the luminance values horizontally, and (4) Convolve the resulting vector, l , with each of the m principal component vectors to generate m projection vectors, $\tilde{l}_i$. By approximating the training data as multivariate normal, the log probability of the horizon lying at vertical location y can be estimated using

$$\log p(y_h = y) \propto - \sum_{i=1}^m \left(\tilde{l}_i(y) - \bar{l}^T u_i \right)^2 / \lambda_i, \quad (28)$$

where $\bar{l}$ denotes the mean over the training vectors, λ_i denotes the i th eigenvalue.

4. Anchoring Geometric Prior for Road Segmentation. Three types of ground truth geometric priors (Fig. 11) were generated from the training set: (1) unconditioned geometric prior, (2) vanishing point-based prior, and (3) horizon-based prior. Given a test image i , after its vanishing point (or horizon) is detected, the corresponding type of geometric prior will be anchored accordingly. The final segmentation results can be generated by thresholding the anchored geometric prior $p(x_i = F)$, where x_i denotes a pixel position and F indicates the road region. The authors used the theoretical optimal threshold, i.e., 0.5 for the prior thresholding. Some segmentation results are shown in Fig. 12.

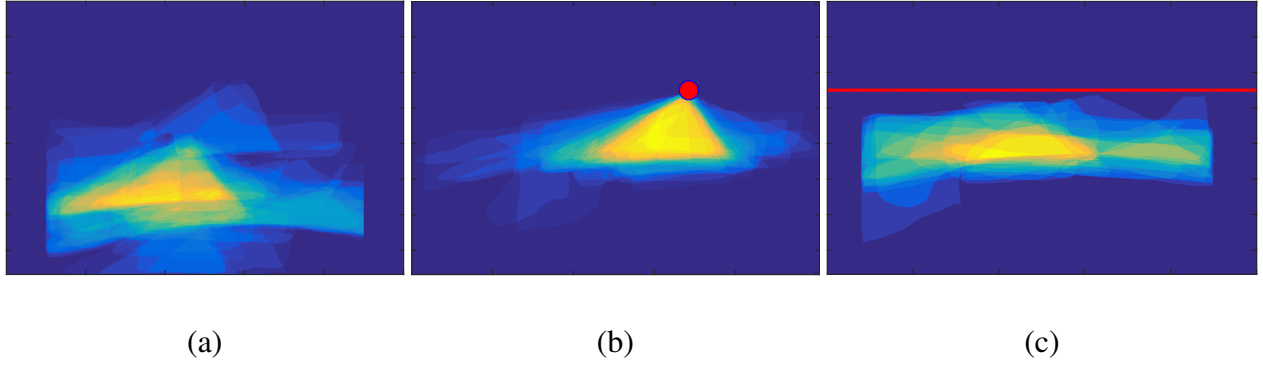


Figure 11: Geometric Priors [18]. (a) Unconditional prior. (b) Road vanishing point-based prior. (c) Road horizon-based prior.

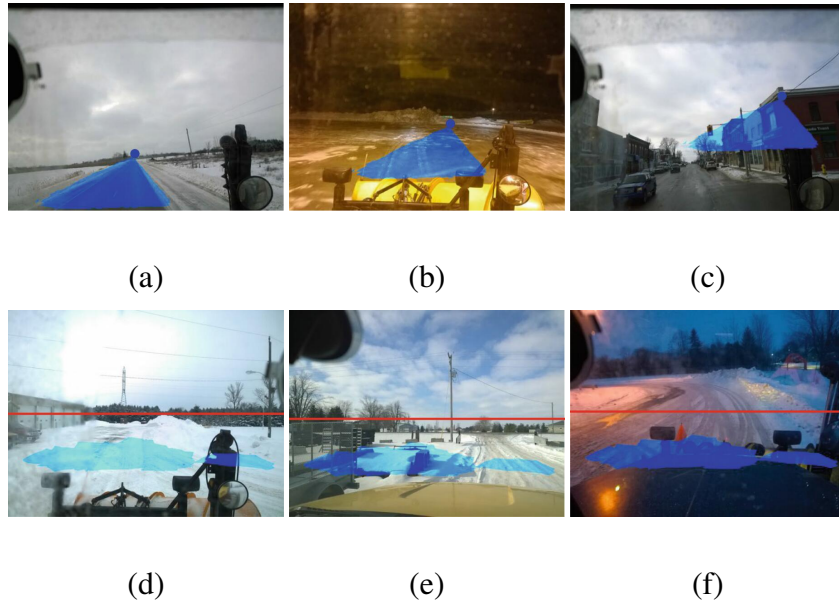


Figure 12: Vanishing estimation examples from [3]. (a)-(c): Best, median, and worst vanishing point-based road segmentation results. (d)-(f): Best, median, and worst horizon-based road segmentation results

Limitations of this paper include: (1) The threshold on the prior was not optimized. (2) It was only evaluated on one small dataset. (3) Since the end goal was evaluating road weather classification, the authors did not provide a quantitative evaluation of the road segmentation step.

2.1.3 Conclusion

In conclusion, hand-crafted feature-based road segmentation methods were commonly used before the rise of deep neural networks. Hand-crafted feature-based road segmentation can generally be divided into two categories: (1) appearance-based methods and (2) geometry-based methods. Each approach has its own advantages and limitations. Appearance-based methods may not perform well on test images with unfamiliar road weather conditions, while geometry-based methods may fail to exclude objects on the road and tend to produce oversimplified segmentations.

2.2 DNN-based Road Segmentation

Classical segmentation techniques are limited in handling large appearance variations; however, they remain important for benchmarking and for providing interpretable baselines. These methods are computationally efficient, require no large-scale training data, and often form the foundation for many real-time systems.

However, the advent of AlexNet [41] sparked the rise of deep neural networks (DNN) and significantly improved the performance of various computer vision tasks, surpassing former hand-crafted feature-based methods. As a result, DNN-based driving scene understanding has made remarkable progress, with road segmentation benefitting greatly from state-of-the-art techniques such as semantic segmentation, instance segmentation, and domain adaptation.

This section provides an overview of DNN-based road segmentation methods. In Section 2.2.1, we describe popular DNN architectures that have been widely used as backbones in road segmentation methods. Additionally, in Section 2.2.2, we review some classic and state-of-the-art representative road segmentation networks.

2.2.1 Commonly-used Road Segmentation Backbones

2.2.1.1 FCN In 2015, Long *et al.* proposed the Fully Convolutional Network (FCN) for semantic segmentation tasks. Unlike the traditional Convolutional Neural Network (CNN) architecture, which uses flattened vectors from the final fully connected layers to predict a label for the entire image, FCN replaces the fully connected layers with fully convolutional layers, allowing for pixel-wise predictions. This design enables the network to produce a dense prediction map for each pixel in the input image. Now FCN has become a widely used backbone for many state-of-the-art

(SOTA) image segmentation networks.

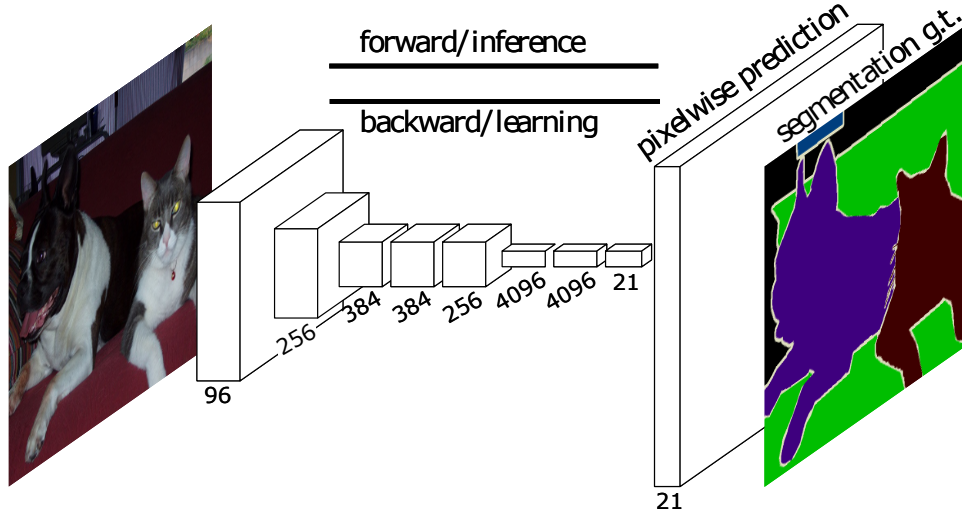


Figure 13: Architecture of FCN from [51].

Figure 13 illustrates the architecture of the FCN. When an input image is fed into the network, tensors are first generated using a conventional Convolutional Neural Network (CNN) architecture, such as AlexNet, GoogleNet [77], or VGG [73]. The fully connected layers in the CNN architecture are then replaced with fully convolutional layers, which allow the FCN to produce output with the same size as the input using upsampling. This design enables pixel-wise labeling for the input image.

2.2.1.2 U-Net In 2015, Ronneberger *et al.*[67] proposed an encoder-decoder style network for biomedical image segmentation, which quickly became a popular backbone for many semantic segmentation networks (e.g., [56, 36, 59]). As for FCN[51], U-Nets include an encoder (contracting path) that downsamples the input images with convolutions and a decoder (expanding path) that upsamples the tensors to the same size as the inputs. The key idea of UNets is to provide to

each decoder layer with direct information from the corresponding encoder layer with matching resolution (see Fig. 14).

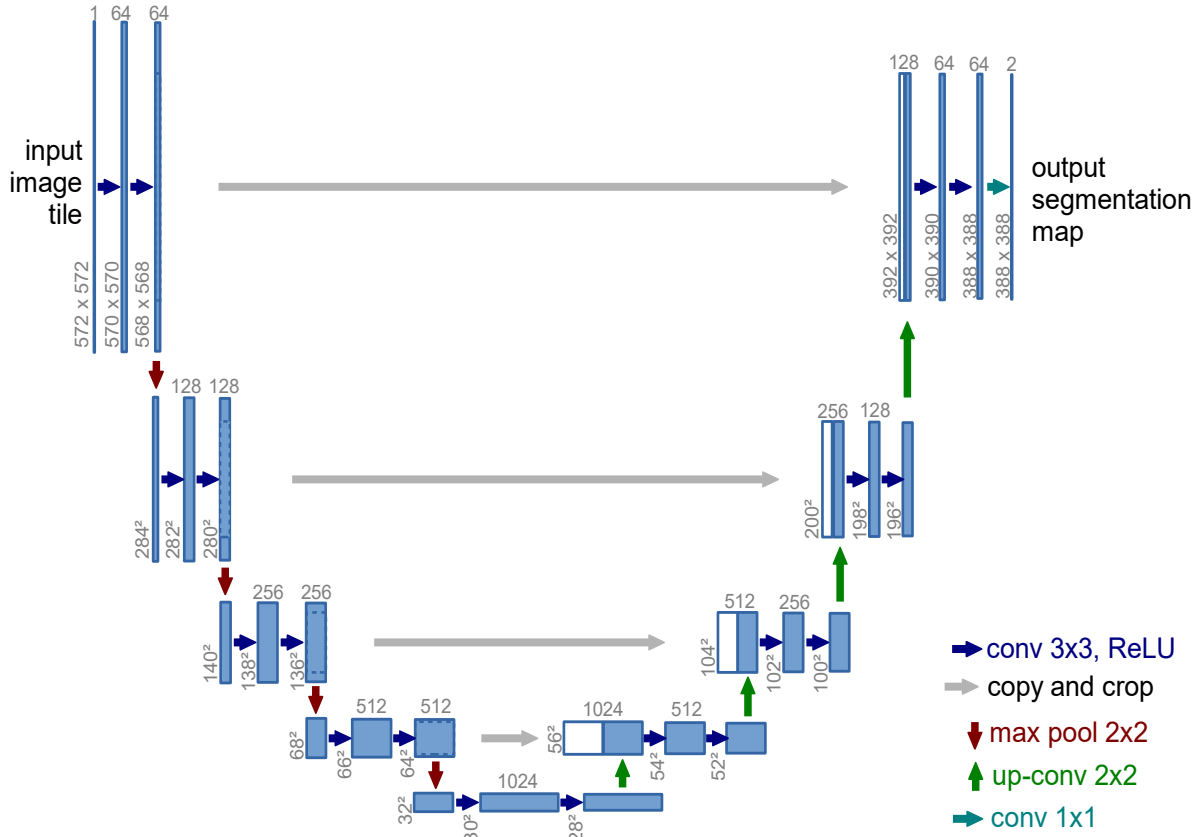


Figure 14: Architecture of U-Net from [67].

2.2.1.3 ENet In 2016, Paszke *et al.* proposed ENet [62], a compact and efficient semantic segmentation network. ENet adopts an encoder-decoder style architecture, but unlike many similar networks (e.g., [51, 8]) which use the large VGG16 architecture, ENet is designed to be lightweight and efficient while achieving state-of-the-art performance in semantic segmentation tasks. Key innovations of ENet include a new bottleneck module that reduces the number of parameters and computational complexity, as well as a novel architecture for the encoder and decoder that enables

efficient information flow and spatial resolution preservation. Since publication, many semantic segmentation networks (e.g. [5, 34]) have used ENet as their backbones.

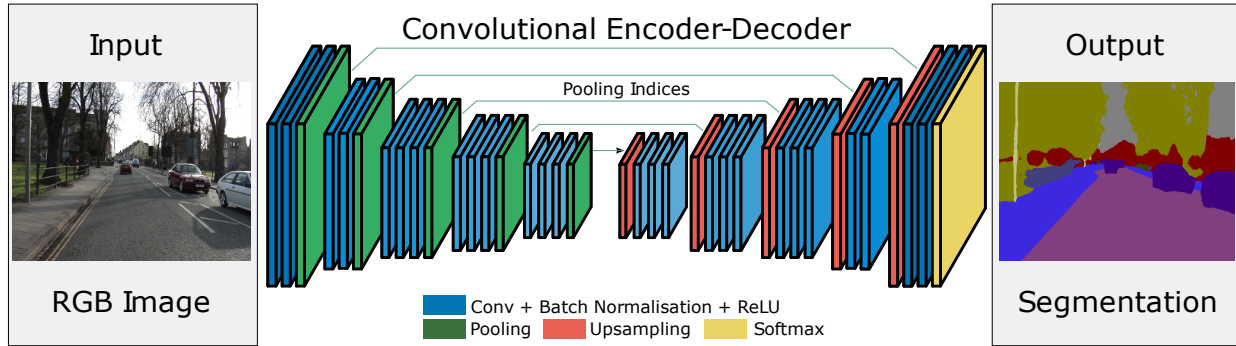


Figure 15: Architecture of SegNet from [8].

2.2.1.4 SegNet In 2017, Badrinarayanan *et al.* [8] introduced SegNet, an encoder-decoder style end-to-end pixel-wise semantic segmentation network. SegNet quickly gained popularity and became a common backbone for many subsequent road segmentation and driving scene understanding networks[18, 17, 83, 87, 76]. Unlike other end-to-end pixel-wise segmentation networks (such as FCN), where the encoder and decoder parts typically have different architectures, the standard SegNet consists of two VGG-16 networks, with one serving as the encoder and the other as the decoder, and all fully connected layers are removed. The encoder is trained using conventional VGG-16 training. In the decoder upsampling step, no learning is performed. However, trainable multi-channel decoder filters are convolved with the upsampled tensors to generate dense output. SegNet has been evaluated on the CamVid [11, 10] road scene dataset. When it was first proposed, SegNet achieved the highest IOU (90.40%) on this dataset.

2.2.2 DNN-based Road Segmentation Methods

Many road segmentation networks have been proposed, e.g., [48, 55, 91, 16, 81, 87, 60, 102, 53, 32, 76]. Here, I focus on highly cited or state-of-the-art (SOTA) road segmentation networks.

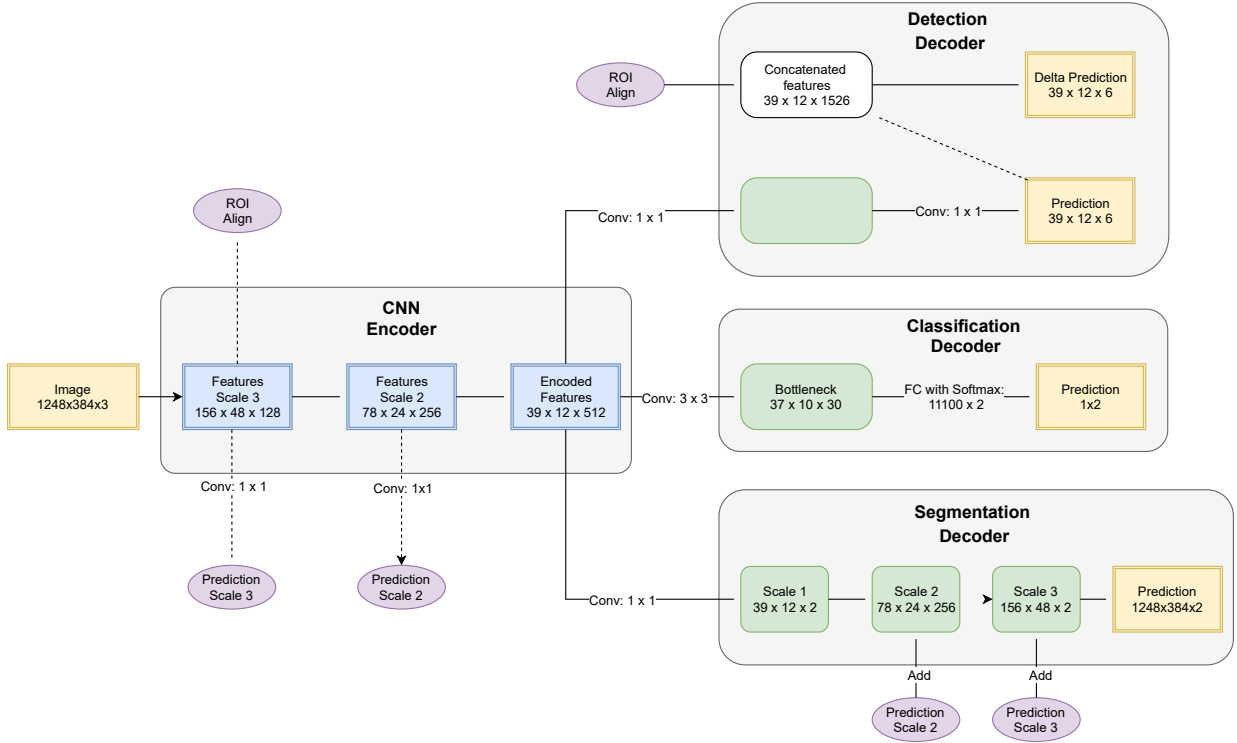


Figure 16: Architecture of Multinet from [81].

2.2.2.1 Multinet In 2018, Teichmann *et al.* proposed a novel encoder-decoder network called MultiNet [81] for real-time joint semantic reasoning in autonomous driving. MultiNet utilizes one VGG network as the encoder backbone, which is shared by three different decoders for road segmentation, street classification, and vehicle detection tasks. An overview of MultiNet’s architecture can be seen in Fig 16.

MultiNet has several notable strengths, including (1) real-time multi-task road reasoning, which

addresses key challenges in developing autonomous driving systems, (2) achieving the best performance compared to other state-of-the-art methods on the KITTI benchmark at the time of publication, and (3) demonstrating the potential for generalizing one encoder to multiple decoders.

However, MultiNet also has some weaknesses. One notable issue is the requirement for a large amount of annotated data to achieve optimal performance, which can be time-consuming and laborious. Additionally, the network’s architecture is complex, making it challenging to optimize and reach desirable performance for all decoders.

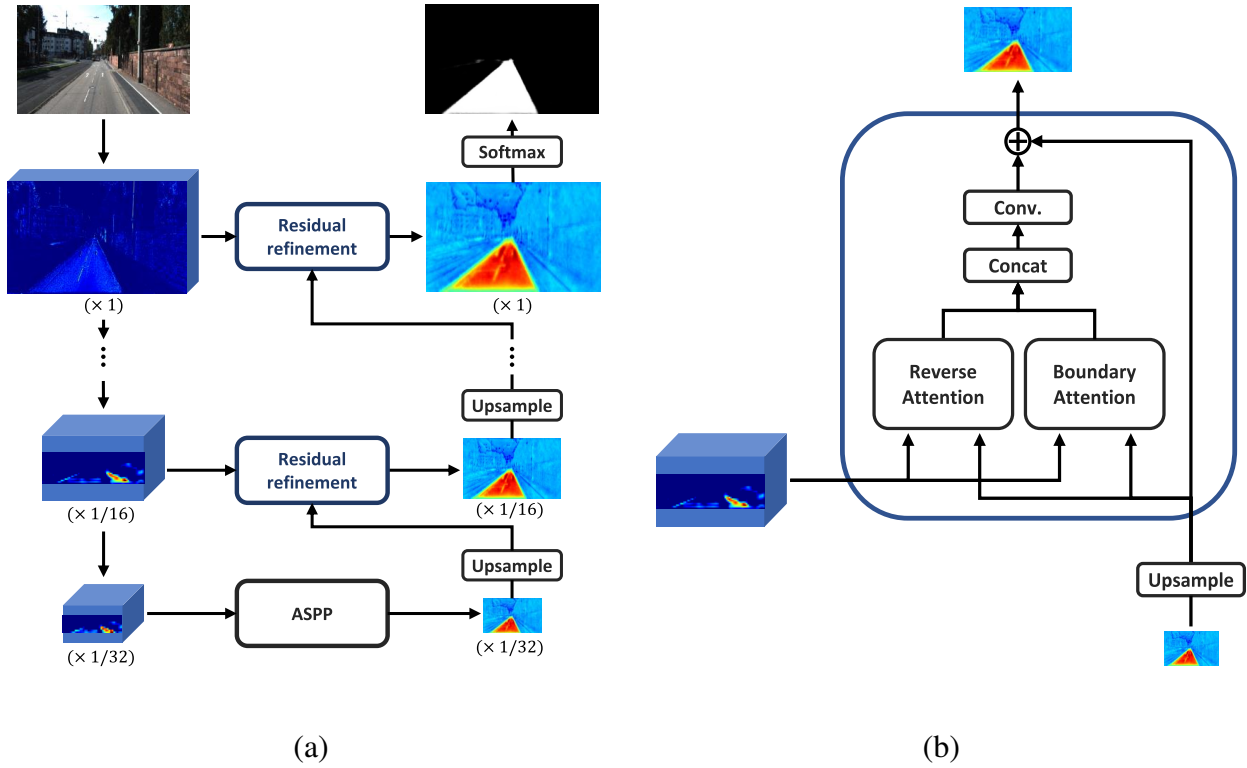


Figure 17: Overview of reverse and boundary attention network from [76]. (a) Overall architecture. ASPP denotes atrous spatial pyramid pooling. (b) Residual refinement module.

2.2.2.2 Reverse and Boundary Attention Network In 2019, Sun *et al.* proposed a residual learning-based network [76] to improve road boundary estimation and handle lighting variations (e.g., shadows or over-exposure) in road segmentation. The overall architecture is shown in Fig. 17 (a). The SegNet architecture serves as the backbone of the network. Initially, the network is trained to predict approximate road regions. Next, this coarse map is progressively refined through the addition of residual predictions obtained from lower-level feature maps in a top-down fashion using reverse attention (RA) and boundary attention (BA) units (Fig. 17 (b)). At the time of publication, this approach achieved superior segmentation accuracy compared to state-of-the-art methods on KITTI, Cityscapes [21], and CamVid for traffic scene understanding.

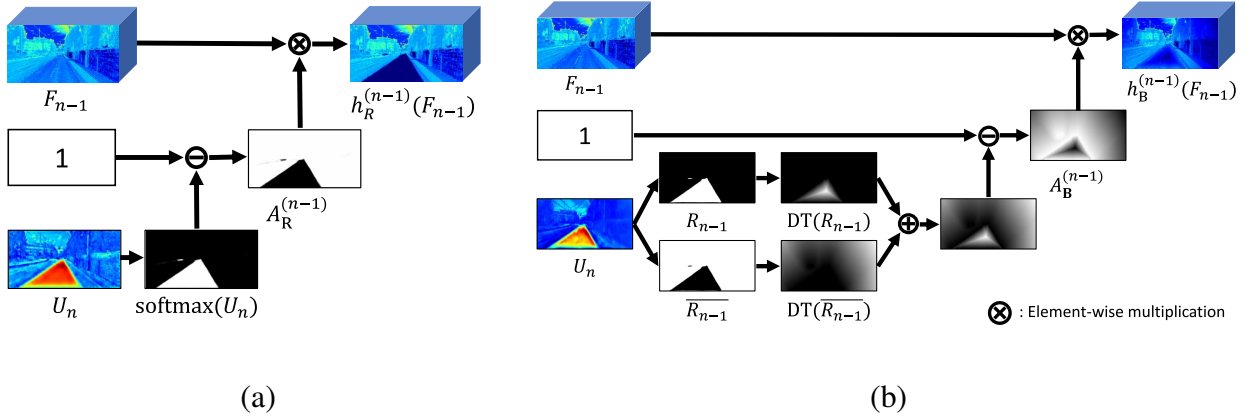


Figure 18: Left: Reverse Attention (RA) unit. Right: Boundary Attention (BA) unit. Details are explained in the following paragraphs. Images are taken from [76].

The RA & BA units are an important novelty.

1. **RA unit** (Fig. 18 (a)). The RA unit was initially proposed by [15] for saliency detection.

Here, the authors used it for encouraging the network to focus on the background. For a

given feature map F_{n-1} at the $(n - 1)$ th level, RA is defined as

$$h_R^{(n-1)}(F_{n-1}) = A_R^{(n-1)} \otimes (F_{n-1}), \quad (29)$$

where $\otimes$ denotes the element-wise multiplication and $A_R^{(n-1)}$ is the RA mask, which is acquired from the softmax of the (n) th level feature map prediction U_n .

2. **BA unit** (Fig. 18 (b)). The BA unit is used to assign attention to the road boundaries. For a given feature map F_{n-1} at the $(n - 1)$ th level, its BA unit is defined as

$$h_B^{(n-1)}(F_{n-1}) = A_B^{(n-1)} \otimes F_{n-1}, \quad (30)$$

where $A_B^{(n-1)}$ denotes the BA mask, which is the complement $DT(\overline{R_{n-1}})$ of the rectified distance transform encoding the distance of each pixel from the current estimate of the road boundary.

In summary, this method has several notable strengths, including: (1) achieving state-of-the-art performance at the time of publication, (2) having an end-to-end architecture that simplifies the training process. However, as with many deep neural network-based methods, it remains uncertain how well it will generalize to novel data.

2.2.2.3 s-FCN-loc In 2018, Wang *et al.* proposed a two-stream FCN, s-FCN-loc [87], for road region segmentation. Fig. 19 shows the architecture of s-FCN-loc, which consists of two streams: (1) an RGB stream and (2) a contour stream. Like the Reverse and Boundary Attention Network [76] introduced previously, one notable novelty of s-FCN-loc is that it also uses road boundaries for a better segmentation. However, it is done in a different way. At the beginning of training, contour maps are generated from the input RGB images using Structured Forests (SF) [24]. Next,

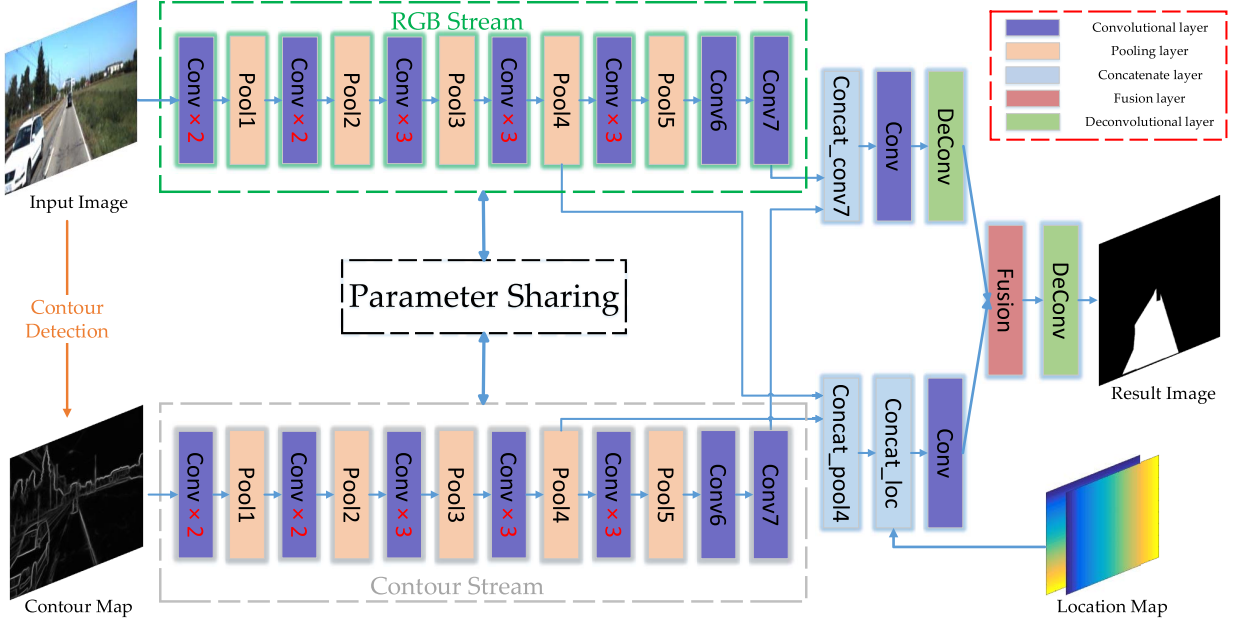


Figure 19: Overview of s-FCN-loc from [87].

two streams are used, each of which utilizes the VGG-16-based FCN [51] as a backbone, with the RGB images and contour maps serving as inputs, respectively. During the training process, the two streams share parameters at the 4th pooling layer (Pool4) and the 7th convolutional layer (Conv7) (Fig 19). A location prior is employed that consists of two normalized (ranging from 0 to 1) maps (one for the x-axis and one for the y-axis) generated from the ground truth road/non-road masks. These location priors are treated feature maps directly appended to the concat_pool4 layer. (i.e., the Location Map shown in Fig. 19). The outputs from the two streams are then fused and up-sampled to generate the final road region segmentation map. s-FCN-loc was evaluated on the KITTI road detection benchmark and the One-Class road detection dataset. At the time of publication, it achieved a competitive performance with SOTA methods.

The s-FCN-loc approach has several strengths, including its ability to consider road boundaries

for improved road segmentation, as well as the use of a road location prior map to reduce false detections in the road region. However, the method also has a significant weakness in that its performance is highly dependent on the accuracy of the contour maps and location priors. If these inputs are not sufficiently accurate, it can negatively impact the road region detection results.

2.2.2.4 Fusion-based Two-stream Network A fusion-based network for road detection was proposed by Almeida *et al.* in 2020, as reported in [5]. In contrast to prior road segmentation approaches that generate segmentation masks, this technique strives to generate a probabilistic map that indicates the probability of each pixel belonging to the road area as an intermediate step of the autonomous driving system.

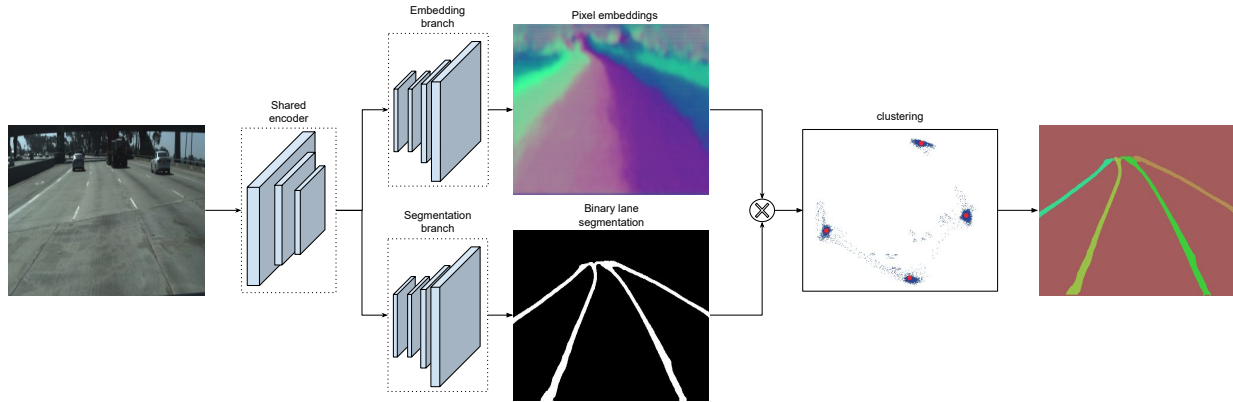


Figure 20: Architecture of LaneNet from [58]. $\otimes$ and clustering are described in the following paragraphs.

The network consists of two streams that integrate the road region segmentation result with the lane detection-based road polygon, resulting in a final probabilistic road confidence map. The road region segmentation uses ENet [62] (Section 2.2.1.3), while LaneNet[58] estimates the lane detection-based road polygon. LaneNet is an instance-segmentation-based lane detection network that consists of two branches: (1) a lane segmentation mask (i.e., the lower branch in Fig. 20) that

aims to detect the binary lane segmentation mask and (2) a lane instance segmentation branch (i.e., the upper branch in Fig. 20) that generates pixel embeddings of the input image. The embedding branch creates an N-dimensional embedding for each lane pixel, aiming to position embeddings of pixels from the same lane close together in the manifold, while distancing the embeddings of pixels from different lanes. Next, this method masks out the background pixels from the pixel embeddings (upper branch) according to the binary segmentation result (lower branch). Then it clusters the pixels into groups according to their distances to the center of each detected lane. $\otimes$ denotes masking by the binary lane segmentation.

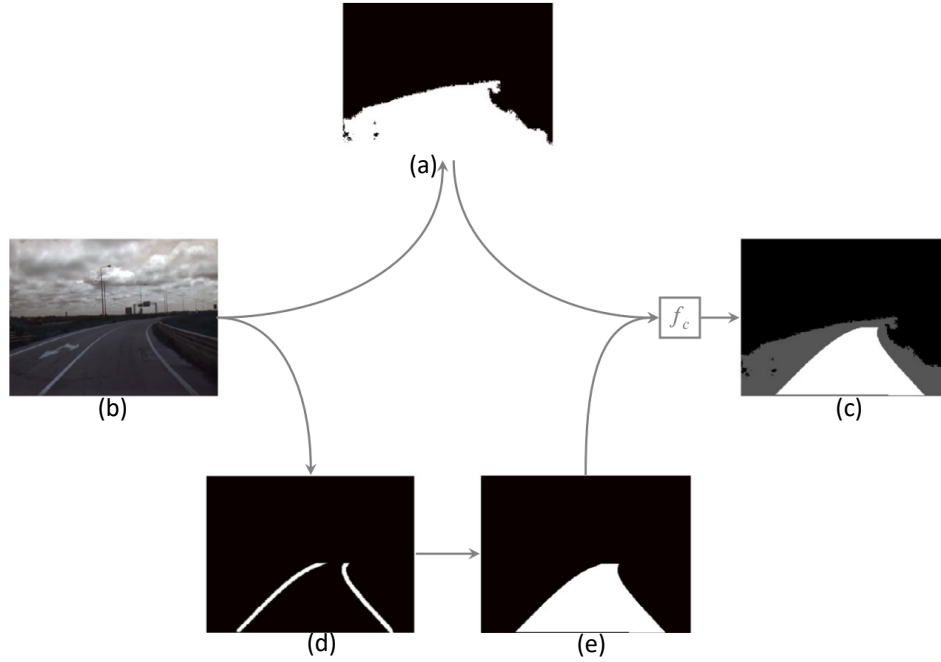


Figure 21: Workflow of the proposed road segmentation method from [5].

Fig.21 illustrates the flow of the Fusion-based Two-stream Network. As depicted in Fig.21, when given an input image of a road scene (Fig.21(b)), it is initially fed into two sub-networks to obtain the road area segmentation (Fig.21(a)) using ENet, and the lane branch estimates the road

region polygon by bridging the endpoints of paired adjacent lanes with straight lines. (Fig.21(d) to (e)). Then, the results of the two branches are averaged generating a confidence map for the road area (Fig.21(c)).

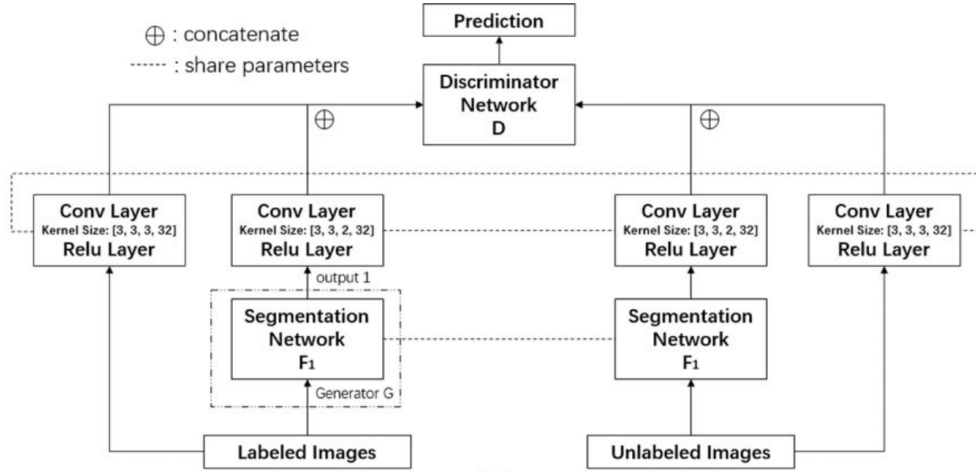


Figure 22: Overview of SLL-GAN from [32].

2.2.2.5 SSL-GAN In summary, the objective of this method is to produce a probabilistic road confidence map for autonomous driving systems. One innovative aspect is the integration of lane markings and semantic road segmentation to enhance the accuracy of the confidence map estimation. However, a notable drawback is that the authors did not conduct any quantitative or qualitative evaluation of the binary road segmentation task.

In addition to utilizing supervised road segmentation methods with the aforementioned CNN architectures, researchers have also explored the use of Generative Adversarial Networks (GANs) to tackle this task. For instance, in 2018, Han *et al.* introduced SSL-GAN [32], a GAN-based approach that employs ResNet-101 [33] and enables a semi-supervised training regime by incorporating both labeled and unlabeled images. The network architecture of SSL-GAN is presented

in Fig. 22. In SSL-GAN, the generator aims to detect road regions in all the labeled and unlabeled images, while the discriminator determines whether an input image belongs to the labeled or unlabeled set. Based on this semi-supervised learning approach, the authors reported that SSL-GAN achieved comparable performance to fully supervised methods trained on all labeled training images with only 400 labeled road images on the KITTI road segmentation dataset.

2.2.3 Conclusion

In conclusion, encoder-decoder style backbones have been widely utilized as the backbone for numerous deep neural network-based road segmentation networks. Like the hand-crafted-feature-based road segmentation methods, many deep neural network-based road segmentation networks also attempt to fuse cues acquired from both appearance and geometry, as evidenced in methods such as [32, 76]. Additionally, some methods such as [81] propose a multi-task network in which road segmentation is one task.

2.3 Unsupervised Domain Adaptation for Road Segmentation

Domain adaptation is a subfield in transfer learning that seeks to train a model on a source domain and then adapt it to a related but different target domain. In some cases, the ground truth (GT) of the target domain is available for better supervision. However, we are interested in a more challenging scenario known as unsupervised domain adaptation (UDA), where only unlabeled data from the target domain is provided.

UDA for semantic segmentation has gained significant attention in recent years. Although it is now easy to capture images, generating accurate ground-truth annotations for segmentation remains a slow and labor-intensive process. UDA offers a potential solution by enabling semantic segmentation using a model that has been previously trained on a related source domain and then adapted to the user’s unlabeled target dataset. The task of adaptation is particularly important for road segmentation, where changing weather, lighting, or viewing conditions can significantly affect segmentation accuracy.

Only a few UDA methods ([18, 17]) focus on road segmentation but a considerable number of UDA methods (e.g., [83, 14, 86, 93, 101, 94, 88]) have been proposed for semantic segmentation. Hence, we will review UDA methods for both semantic segmentation and road segmentation. Usually, these UDA methods focus on adapting the model from synthetic datasets (e.g., GTA5 [66] or SYNTHIA [68]) to real datasets (e.g., KITTI, Cityscapes, or BDD [95]).

We divide the UDA methods according to their network architectures. In Section 2.3.1, we introduce the state-of-the-art (SOTA) CNN-based UDA methods. In Section 2.3.2, we introduce the GAN-based SOTA UDA methods.

2.3.1 CNN-based UDA Methods

2.3.1.1 DACS In 2021, Tranheden *et al.* proposed a CNN-based unsupervised domain adaptation method ‘Domain Adaptation via Cross-domain mixed Sampling (DACS) [82]’ for semantic segmentation. This method applies a self-training scheme to narrow the distribution gap between the source and the target domains.

Self-training is a technique that was first proposed for solving the semi-supervised learning problem [45]. Later, self-training was utilized in the UDA research field. Self-training for UDA aims to first generate pseudo labels for the unlabelled target domain to form the target domain training pairs. More concretely, for the vision-based UDA case, each target domain training pair consists of a target image and its corresponding pseudo label. Next, the target domain training pairs can be used to narrow the distribution gap through adaptations. For example, one simple idea is to fine-tune the source domain-trained network with the target training pairs. One problematic issue with respect to self-training is how to improve the accuracy of pseudo labels. Due to the varieties of domain gaps (e.g., viewing, lighting condition, appearance, or geometry), pseudo labels are likely to be noisy, which can easily hinder the final performance.

To solve this mislabeling problem, DACS introduces a simple yet effective augmentation technique to alleviate pseudo labels noise:

1. **Preparation.** The source domain dataset D_s contains images and corresponding ground truth, and the target domain dataset D_t contains unlabeled images. f_θ denotes a semantic network. For all the experiments conducted by DACS, the authors used the DeepLabv2 [13] framework with ResNet-101 [33] as the backbone [50]. The backbone is pre-trained on ImageNet and MSCOCO.

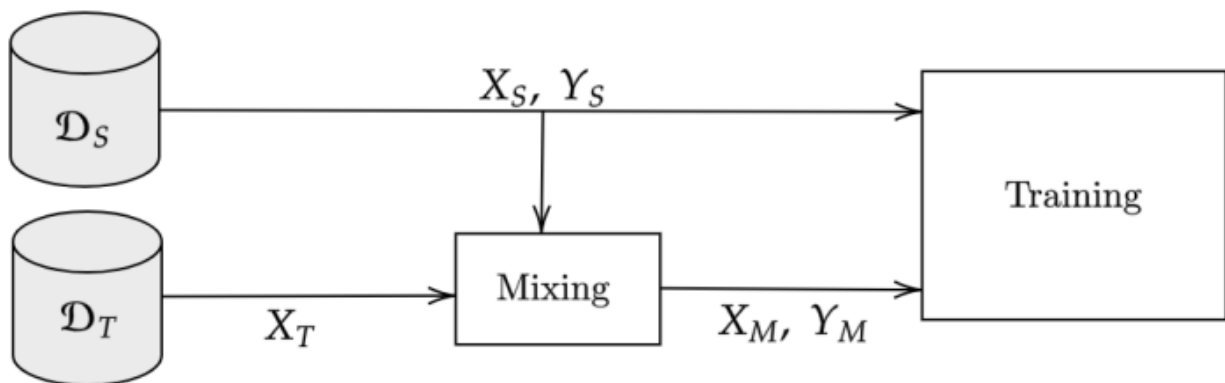


Figure 23: Flowchart of cross-domain sampling in DACS from [82].



Figure 24: Cross-domain data augmentation in DACS [82]. Objects are cropped from the source domain image X_S (a GTA5 image) using the ground truth label and then mixed with a target domain image X_T (a Cityscapes image). Finally, an augmented image X_M is generated.

2. **Adaptation.** A batch of source domain training pairs (X_s, Y_s) are sampled from D_s , where X_s denotes the images while Y_s denotes the ground truth labels. A batch of images X_t are extracted from D_t . Next, X_t are fed to the f_θ to get pseudo labels represented by $\hat{Y}_t$. Then the augmented training pair (X_M, Y_M) can be acquired by mixing X_s with X_t and Y_s with $\hat{Y}_t$ respectively (i.e., placing the objects from S into T). A flowchart of the above-mentioned cross-domain sampling is shown in Fig. 23. The detailed data augmentation is illustrated in Fig. 24. Next, the authors fine-tune f_θ with augmented training pair (X_M, Y_M) in which the loss is computed by the following equation

$$L(\theta) = \mathbb{E} [H(f_\theta(X_S), Y_S) + \lambda H(f_\theta(X_M), Y_M)] \quad (31)$$

,where λ denotes a weight balance term and H denotes the cross-entropy loss.

3. **Inference.** The fine-tuned f_θ will be used to conduct inference on the unlabeled target domain. The performance evaluation of DACS is shown in Sec. 2.3.3.

Overall, DACS works well on various synthetic-to-real domain adaptation benchmarks; however, one notable weakness is that the authors did not give any theoretical analysis or justification for why the proposed augmentation method works,

2.3.1.2 ProDA The performance of self-training can be hindered by noisy pseudo labels. These labels are generated based on the source domain distribution and may be inaccurate due to the gap between the source and target domain distributions. In 2021, Zhang *et al.* introduced a new approach called ProDA [99] that addresses this problem by adjusting the pseudo labels during the training process based on a prototype that is updated online. The prototype is a vector that denotes the average features of a class. To compute it, the segmentation network extracts the features from

the source domain data belonging to that class and takes their mean.

The proposed method involves the following steps:

1. **Compute weighted pseudo labels.** The unlabeled target images were initially fed into the source domain-trained network to produce the softmax output, which was used as the initial pseudo labels. These initial pseudo labels were subsequently weighted based on a confidence score computed from the distance between the pseudo labels and the corresponding class-wise prototype vectors. Higher confidence scores were assigned to the pseudo labels that were closer to the prototype vector of the relevant class.
2. **Prototype formation and update.** To form the prototype for a given class, the segmentation network extracts the features from the source domain data belonging to that class and takes their mean. In the following training iterations, the prototype is updated by computing the weighted average of the features extracted by the segmentation network from the target domain data, using the denoised pseudo labels. The updated prototype more accurately reflects the feature distribution of the target domain data. Consequently, it can improve the weighting of the pseudo labels.

Using the aforementioned steps, UDA semantic segmentation network can be trained using online-learned pseudo labels, resulting in improved segmentation outcomes. The performance of ProDA is summarized in the Section 2.3.3.

Overall, ProDA can generate state-of-the-art performance on three domain adaptive semantic segmentation benchmarks: (1) GTA5 to Cityscapes, (2) SYNTHIA to Cityscapes, and (3) Cityscapes to Cross-City. However, one notable limitation is the authors did not provide ablation studies with respect to how different components affect the final performance.

2.3.1.3 Shallow Features Guided-UDA In 2022, Cardace et al. [12] proposed a novel UDA method that aims to enhance object boundary prediction. This method also employs self-training techniques to achieve UDA.

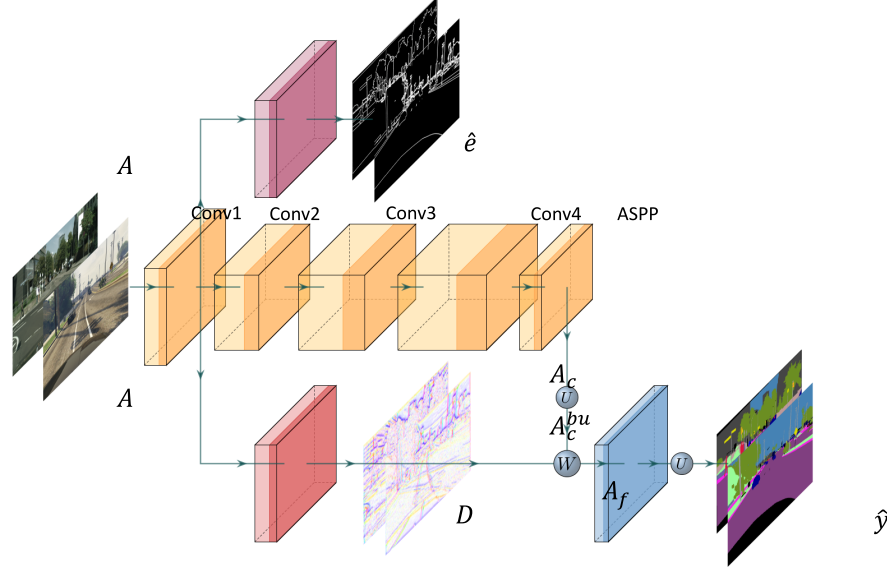


Figure 25: Overview of Shallow Features Guided-UDA from [12].

As can be seen from Fig. 25, the proposed network is a pixel-wise CNN semantic segmentation network that consists of three main parts: (1) a standard semantic segmentation network backbone (the yellowish network), (2) an edge extractor that connects to the first convolution block of the backbone (top branch), and (3) a displacement network (bottom branch), which produces a 2D vector for each spatial location on the predicted coarse feature map. In Figure 25, the predicted coarse feature map is denoted by A_c . This map is the output generated by the yellowish backbone. These 2D vectors are used to warp the down-sampled deep features, allowing for sharper predictions along semantic edges. The edge estimation branch (the top branch) is used to compute the edge loss. Semantic loss is computed from the yellow backbone and the bottom red 2D

displacement unit. The overall loss is the sum of the above two losses. The displacement map D generated from the bottom branch will be used to refine the up-sampled backbone output A_c^{bu} to get the fine-grained feature map A_f . Finally, A_f will be sent to the classifier to predict the pixel-wise class labels according to the corresponding ground truth. The authors employ the Deeplab-v2 [13] architecture, with a dilated ResNet-101 [33] backbone pretrained on ImageNet [23]. The classifier is an ASPP module [13].

The overall UDA of [12] consists of three stages: (1) initialization, (2) adaptation, and (3) distillation.

1. **Initialization.** The baseline model (i.e., the yellow backbone shown in Fig. 25) is trained using the source domain data only. Once the backbone is trained, it can be used to produce the pseudo-labels for the target domain by feeding the target images into it. Next, data augmentation takes place. Inspired by [96, 26, 30, 82], the authors utilized the pseudo-labels to crop objects out of target images and then paste them into other images. These images can be either target or source domain images depending on the processing stage (see below Fig. 26).
2. **Adaptation.** At this stage, the pre-trained network (i.e., the one trained in Step 1.) will be fine-tuned using the augmented data which extracts target objects and then pastes them to both source and target images.
3. **Distillation.** The model of step (2) is further trained with augmented data which only pastes to the target domain images, to let the network capture more domain-specific characteristics.

After the above three steps, we then can conduct inference on the unlabeled target domain images.

The performance evaluation of this proposed method is shown in Section 2.3.3.



Figure 26: Data augmentation for adaptation examples from [12]. Top-left: a target image and its corresponding prediction. Top-right: A source domain prediction pair on which we aim to do data augmentation. Bottom-left: the cropped person class from the top-right pair. Bottom-right: the cropped objects have been pasted to the target source domain prediction pair and the data augmentation is complete.

In conclusion, this method achieves state-of-the-art performance on GTA5 to Cityscapes and SYNTHIA-to-Cityscapes.

2.3.2 GAN-based UDA Methods

2.3.2.1 AdaptSegNet In 2018, Tsai et al. proposed AdaptSegNet [83] which is a GAN-based UDA method. As can be seen from Fig. 27, AdaptSegNet involves two modules: (1) a semantic segmentation network G which takes RGB images as input and outputs the pixel-level class predictions, and (2) a discriminator D which takes the output of G as input and identifies whether the input belongs to the source or the target domain. Overall, the generator encoder aims to fool the discriminator, i.e., to generate similar distributions in the output for images from either source or target domains.

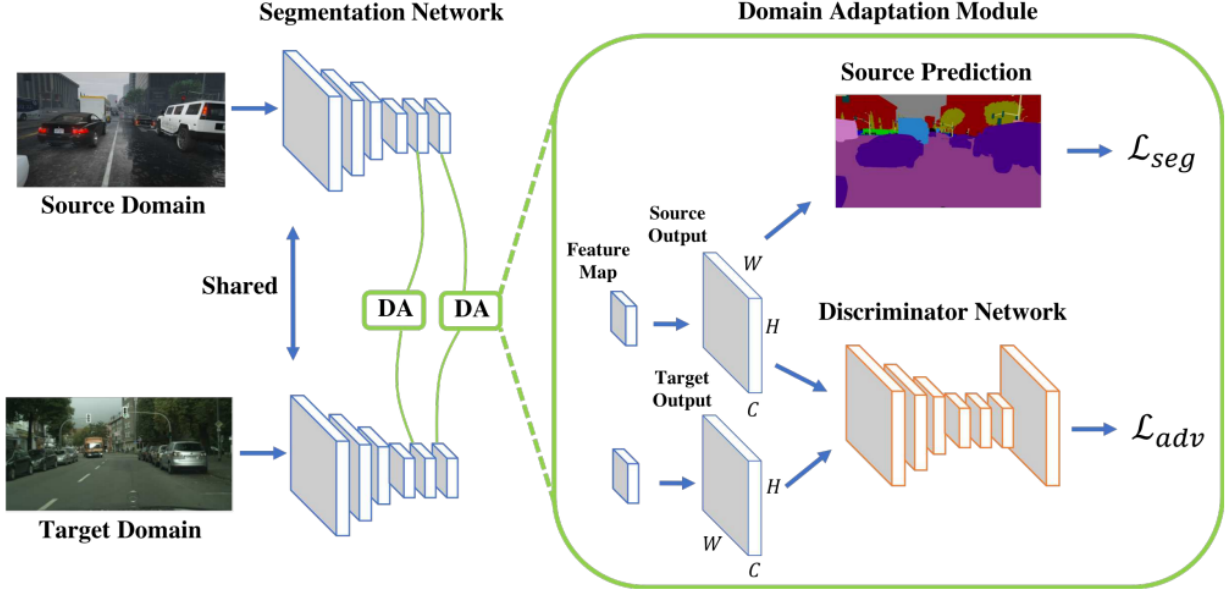


Figure 27: Overview of AdaptSegNet from [83].

There are two types of AdaptSegNet proposed: (1) Single-level AdaptSegNet and (2) Multi-level AdaptSegNet. Single-level AdaptSegNet performs adversarial learning only using the output of G , which means only one discriminator module is used to compute the overall loss. In the single-level AdaptSegNet, low-level features may not be well adapted since they are produced at the early layers of G . To take these low-level features into consideration, the Multi-level AdaptSegNet was proposed, which contains multiple adaptation modules. The loss for Multi-level AdaptSegNet is derived from losses at all layers.

The overall loss of AdaptSegNet can be computed using:

$$L(I_s, I_t) = L_{seg}(I_s) + \lambda_{adv} L_{adv}(I_t) \quad (32)$$

where I_s denotes a source domain image (with ground truth labels) and I_t denotes a target domain image (unlabeled). L_{seg} is the cross-entropy loss for semantic segmentation and L_{adv} is the adversarial loss derived from the discriminator to distinguish semantic maps generated from the source

& target images. λ_{adv} is a weight used to balance the two losses.

For single-level AdaptSegNet, the losses are:

1. **Discriminator loss.** The discriminator loss $L_d(p)$ can be computed by

$$L_d(P) = - \sum_{h,w} [(1-z) \log(D(P)^{(h,w,0)}) + z \log(D(P)^{(h,w,1)})] \quad (33)$$

where P denotes the softmax output generated from G . h and w are the height and width of P . $z = 0$ if P is generated from the target domain and $z = 1$ if P is generated from the source domain.

2. **Segmentation network losses.**

For the source domain images, cross-entropy losses are computed as

$$L_{seg}(I_s) = - \sum_{h,w} \sum_{c \in C} Y_s^{(h,w,c)} \log(P_s^{h,w,c}) \quad (34)$$

where Y_s denotes the source domain images' ground truth labels. $P_s = G(I_s) \in \mathbb{R}^{(H \times W \times C)}$, where C is the number of classes.

For target domain images, adversarial loss L_{adv} is computed using:

$$L_{adv}(I_t) = - \sum_{h,w} \log(D(P_t)^{(h,w,1)}) \quad (35)$$

During the training phase, the network begins by forwarding the source image I_s through the segmentation network to optimize for L_{seg} (Eq. 34), resulting in the generated output P_s . Simultaneously, for the target image I_t , we acquire the segmentation output P_t . This P_t is then combined with P_s and fed into the discriminator to optimize L_d as outlined in Eq. 33. Moreover, the adversarial loss L_{adv} as defined in Eq. 35 are computed for the target prediction P_t . Multi-level AdaptSegNet utilizes the same equations for computing losses, but over multiple layers.

The performance evaluation of AdaptSegNet is shown in Section 2.3.3.

GANs are known to be difficult to train and prone to collapse, which may affect the performance of this method. Additionally, the impact of large class distribution gaps between source and target on the performance of the method has not been explored.

2.3.3 Conclusion

Method	Conference	Road	mIOU (13 classes)
AdaptSegNet [83]	CVPR 2018	86.5	41.4
DACS [82]	WACV 2021	89.9	52.1
ProDA [99]	CVPR 2021	87.8	57.5
Shallow [12]	WACV 2022	91.9	53.5

Table 1: Quantitative comparison (GTA5 [66] $\rightarrow$ Cityscapes [21]). Numbers are % IoU.

Method	Conference	Road	mIOU (13 classes)
AdaptSegNet [83]	CVPR 2018	84.3	46.7
DACS [82]	WACV 2021	80.6	54.8
ProDA [99]	CVPR 2021	87.8	62.0
Shallow [12]	WACV 2022	90.4	48.4

Table 2: Quantitative comparison (SYNTHIA [68] $\rightarrow$ Cityscapes [21]). Numbers are % IoU.

In the previous sections, we have reviewed (1) DACS [82], (2) ProDA [99], (3) Shallow Features Guided Adaption [12], and (4) AdaptSegNet [83]. In this section, we compare their perfor-

mance on two synthetic-to-real adaptation experiments (Tables 1 and 2). Since our report focuses on UDA for road segmentation, only the road and the average IOU (i.e., mIOU) are shown. As can be seen from those tables, in terms of UDA for road segmentation, Shallow Features Guided UDA [12] performs best.

In conclusion, recent UDA methods have primarily focused on multi-class semantic segmentation, with a particular emphasis on urban scene semantic segmentation, which often includes road segmentation. Self-training schemes are commonly used in many state-of-the-art UDA methods, such as [82, 12, 99, 17] and one of these, Shallow Features Guided Adaption (SFGA), achieves the best performance on road segmentation for SYNTHIA and Cityscapes UDA. However, GAN-based architectures are another prevalent alternative used in these methods.

3 Fusing-based Road Segmentation

3.1 Introduction

Vehicle-mounted cameras provides important visual driving scene data that can be applied to many research fields, such as road condition assessment, autonomous driving, traffic analytics, or urban planning. One important initial task is to segment the road region from the given driving scene. This task becomes challenging when the camera parameters (e.g., pan/tilt/zoom, or the camera position) are not known because a region of interest cannot be pre-defined. Moreover, there are other difficulties in tasks: (1) the varying road weather conditions (e.g., rain, snow, dry, or wet. See Figure 28) and (2) the occlusions (e.g., vehicles, motorcycles, or bicycles) on the road surface.

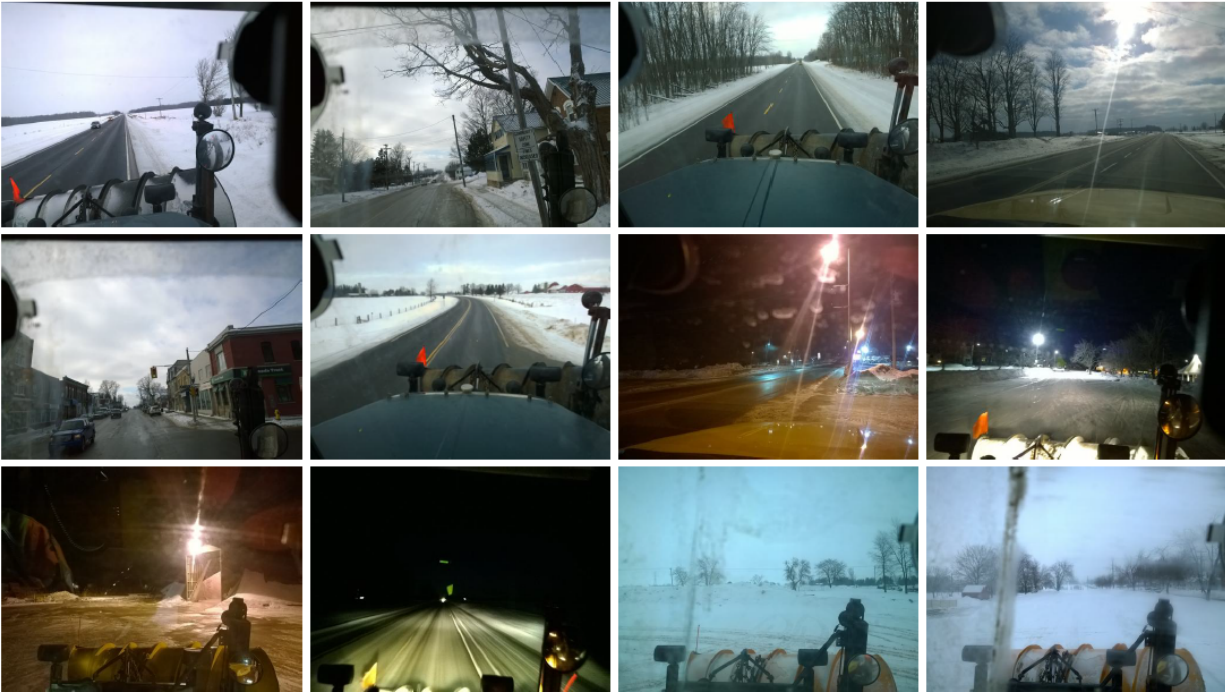


Figure 28: Example images from the dataset that has varying weather conditions. Images taken from [18].

In my first Ph.D. research project, we proposed a novel road segmentation that fuses geometric cues with appearance cues for solving the above-mentioned difficulties. Before our method was proposed, most road segmentation methods are either appearance-based or geometry-based, but not both. However, each type of those method has its own shortcomings: (1) Appearance-based Road segmentation methods are likely to fail when the surroundings have a similar color to the road surface. For example, when the road surface and its neighboring sidewalks are all covered by snow, they share the same appearance; hence, under this condition, it becomes difficult to extract the road region from its neighboring sidewalks. (2) In terms of geometry-based methods, they are not subject to these limitations; however, they can fail when the contour of the road surface is obscured (e.g., due to snow) or less well-defined (e.g., the road surface in the parking lots). Moreover, geometry-based methods cannot exclude non-road objects (e.g., vehicles or motorcycles) on the road.

To model the geometric cues, we apply the approach proposed by Almazan *et al.* [3] which utilizes estimates of the road vanishing point or road horizon to identify a pixel-wise probabilistic map on the given scene indicating how likely each pixel is road or non-road. To model appearance cues, we create a Gaussian mixture model (GMM) using the pixel-wise RGB information provided by the road scene images. Unlike many other methods which model appearance cues with discriminative models, modeling appearance cues using GMM can avoid the ad hoc weights in the fusion step. Finally, we fuse the geometric cues with appearance cues under a Bayesian framework.

There are two essential challenges in modeling a real-world vision problem with GMM: (1) how to adapt to the varying scene conditions and (2) how to the number of Gaussian components. It is worth mentioning that one contribution of our work is that we proposed a novel method for finding the optimal number of GMM components.

3.2 Contributions

This project has been published at the 2017 International Conference on Computer Vision (ICCV-2017) Workshop on Computer Vision for Road Scene Understanding and Autonomous Driving [18]. I am the first author of the published paper.

Individual Contributions. My individual contributions to this project are listed below:

1. Develop the method under the guidance of my advisor Dr. Elder.
2. Implement the proposed project.
3. Re-ground-truthing Qian’s Dataset. We use Qian’s dataset [63] to develop and evaluate our method. Since Qian’s dataset was proposed for a road weather classification project and its road segmentation labels are not very accurate. I first relabeled all the road regions in Qian’s dataset.
4. Evaluate our method and do comparisons with other road segmentation methods.

Scientific and Engineering Contributions. Beyond improving benchmark performance, this project provides scientific and engineering insights into the role of appearance and geometric cues in road scene understanding. By systematically fusing geometric priors with probabilistic appearance GMMs, this study highlights how geometry can compensate for appearance ambiguity under adverse conditions such as snow or occlusion. Conversely, appearance information can correct geometric misestimations when road structure is not well defined.

3.3 Related Works

For a comprehensive overview of traditional hand-crafted feature-based and DNN-based road segmentation approaches, please refer to Chapter 2.1 and Chapter 2.2. In this section, we briefly summarize key research closely related to our work on fusing geometry and appearance for road segmentation.

Prior appearance-based road surface segmentation algorithms have modeled road pixel appearance in HSV [74, 69] or RGB [46, 22, 79] space. However, these methods become problematic when the road surface is covered by ice or snow and when other nearby objects (e.g., sidewalks, buildings) have similar appearance.

An alternative is to use geometric cues such as the road vanishing point [39, 64, 65], horizon [6], and/or road boundaries [40, 75, 80]. Here we employ the recent geometric approach of Almazen [3], which uses either the detected vanishing point of the road or the horizon to identify the probability that each pixel of the image projects from the road surface. While Almazen showed that this geometric segmentation method improved road weather classification performance, the resulting segmentations are approximate and do not exclude objects such as other vehicles that may be occluding the road surface. Indeed, Almazen noted that residual error in road segmentation remains one of the factors limiting road weather classification accuracy.

At the time of publication, Deep Neural Network (DNN) methods have been applied to the road segmentation problem. StixelNet [48] uses a 5-layer convolutional architecture inherited from LeNet [44], trained from on the KITTI dataset [29] for obstacle detection and road segmentation. SegNet [8] is a VGG16 [73]-based deep convolutional encoder-decoder network for semantic road scene segmentation that includes the road surface as one of 12 classes. The StixelNet system has

only been evaluated on the KITTI dataset which lacks the more realistic and challenging diversity in camera pose, weather, illumination and road environments present in the Qian *et al.* dataset [63], and the StixelNet code is not publicly available. However, the SegNet code is publicly available, and we compare its performance against our proposed method on the Qian *et al.* dataset (Section 3.6).

Our proposed method uses the posterior probability map generated by the geometry-based method of Almazan [3] as a prior and combines with appearance likelihoods based upon our mixture models to generate a fused posterior that combines both geometric and appearance cues. A final segmentation can then be generated by thresholding the posterior.

3.4 Datasets

To train and evaluate the proposed method, we use the dataset of road images introduced by Qian *et al.* [63], obtained directly from the authors. The dataset consists of 100 2048×1536 pixel images in a 50/50 training/test split. For each of the images the portion of the image projecting from the road has been segmented by hand. It covers a broad range of road weather conditions, including dry, wet, snow and ice-covered as well as a broad range of illumination conditions, from full sunlight to night. The pose of the camera varies considerably - note that the horizon may appear either toward the top or toward the bottom of the image. Finally, the images were obtained from a diversity of road maintenance vehicles, and large portions of the road surface can be occluded by the hood of the vehicle and mounted snow-clearing equipment.

All parameter tuning was performed on the training dataset: the test dataset was used only for final evaluation.

3.5 Proposed Method

3.5.1 Geometric Prior

Let $x_i \in \{F, B\}$ represent the provenance of pixel i , where F indicates foreground (i.e., the road surface) and B indicates background, and let y_i represent the colour of this pixel, in RGB space. Almazen *et al.* [3] observed that while the unconditional probability map $p(x_i = F)$ estimated from the training subset of the Qian *et al.* dataset [63] is quite broad due to the variability in camera pose and road geometry (Fig. 29(a)), anchoring the prior to a detected vanishing point (Fig. 29(b)) or horizon (Fig. 29(c)) tightens the distribution considerably. With this motivation, their method proceeds by first using dominant lines in the image to estimate the road vanishing point. If the resulting estimate is judged to be reliable, the vanishing point prior (Fig. 29(b)) is employed to estimate the road segmentation. Otherwise the horizon line is estimated and the horizon prior (Fig. 29(c)) is employed. In either case, Almazen *et al.* threshold the prior at 0.5 to deliver a final segmentation, which is then used to estimate road conditions. For our purposes we will retain the real-valued probability map to allow probabilistic fusion with appearance cues.

3.5.2 Learning Appearance Models

The training subset of the Qian *et al.* dataset [63] provides ground truth segmentations of foreground (road) and background from which appearance models can be learned. Here we elect to model the appearance y_i of a pixel conditioned on its provenance x_i (foreground or background) as Gaussian mixture model (GMM) in RGB space. Unlike discriminative methods this approach delivers a probability that allows principled fusion with our geometric prior.

In prior approaches [74, 69, 46, 22, 79], a single model for road appearance is learned from

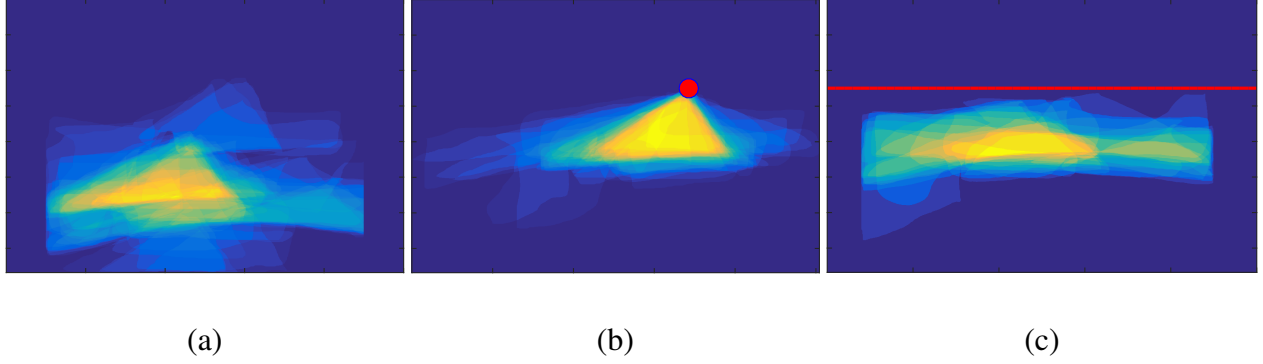


Figure 29: (a) Unconditional prior. (b) Prior anchored to road vanishing point. (c) Prior anchored to horizon.

training data, and we could certainly follow this approach by learning a single GMM for foreground and a single GMM for background over all training data. However, this naïve approach would fail to account for the non-stationarity of these conditional distributions due to diversity in geography, weather and illumination conditions (Fig. 28). To address this challenge we instead adopt a nearest-neighbour approach. Specifically, we learn separate foreground and background models for each image in the training dataset. At inference time we then apply each model to the test image and select the model with highest likelihood. This allows the model to adapt to changing geography, weather and illumination conditions. Note that foreground and background models can be and often are drawn from different training images.

A second issue that arises with all mixture models is how to choose the number of components. Maximizing likelihood leads to overfitting, while maximizing performance on training data can be slow and lead to models that are too task-specific and do not generalize. Here we adopt a novel cross-validation approach that is simple but effective. Let Y_i^F and Y_i^B represent the set of appearance vectors for all pixels in foreground and background regions of training image i and let

$\mathcal{M}_{ik}^F$ and $\mathcal{M}_{ik}^B$ represent k -component maximum likelihood GMMs for these appearance vectors.

We first determine these models for $k = [1 \dots 20]$ and all training images. Our goal is then to identify, for each training image i , the optimal model complexity k_i^F and k_i^B for foreground and background regions, respectively. Here we define optimality in terms of the ability of the model to generalize, selecting the model that maximizes the log likelihood over the other images in the training dataset, assuming conditional independence of appearance observations over pixels:

$$k_i^F = \underset{j \neq i}{k} \sum \log p(Y_j^F | \mathcal{M}_{ik}^F), \quad (36)$$

$$\text{where } p(Y_j^F | \mathcal{M}_{ik}^F) = \prod_{\mathbf{y}_l \in Y_j^F} p(\mathbf{y}_l | \mathcal{M}_{ik}^F) \quad (37)$$

$$k_i^B = \underset{j \neq i}{k} \sum \log p(Y_j^B | \mathcal{M}_{ik}^B), \quad (38)$$

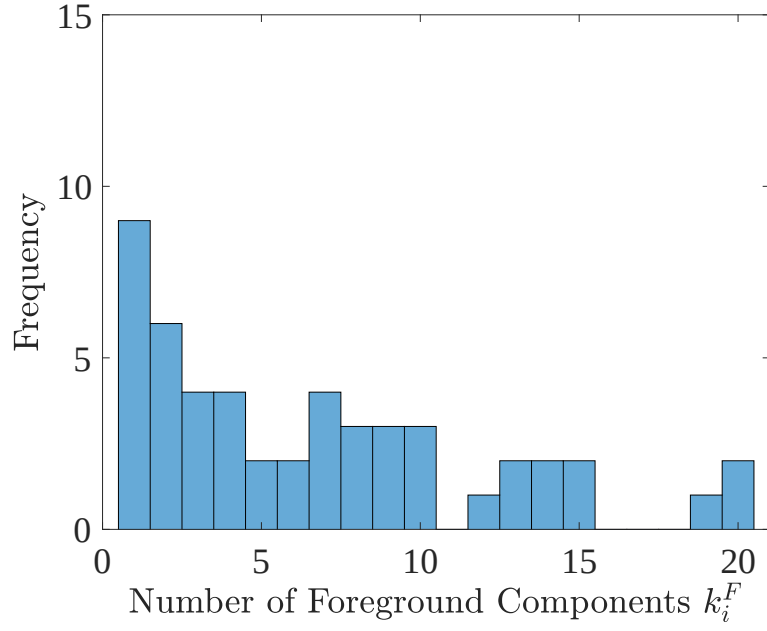
$$\text{where } p(Y_j^B | \mathcal{M}_{ik}^B) = \prod_{\mathbf{y}_l \in Y_j^B} p(\mathbf{y}_l | \mathcal{M}_{ik}^B) \quad (39)$$

In this way we identify a dictionary of optimal GMMs, one for each training image:

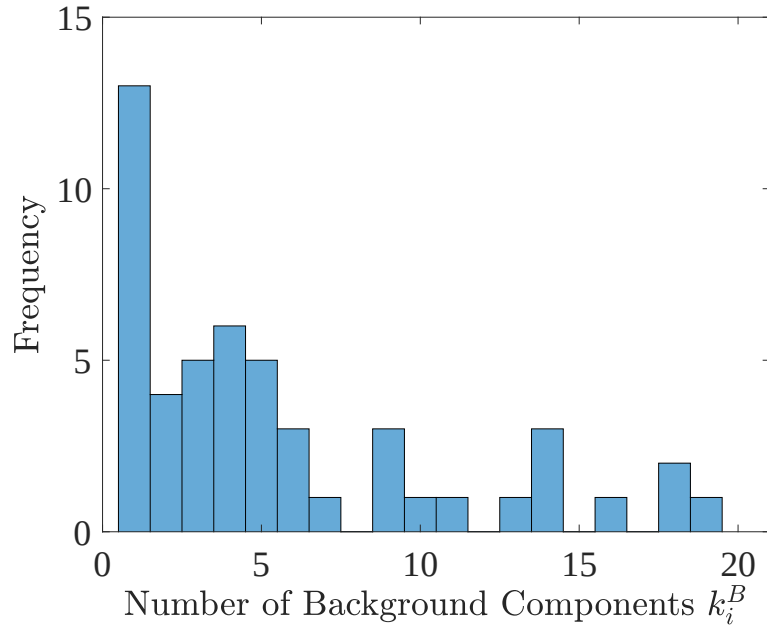
$$\mathcal{M}_i^F = \mathcal{M}_{ik_i^F}^F \quad (40)$$

$$\mathcal{M}_i^B = \mathcal{M}_{ik_i^B}^B \quad (41)$$

In effect this approach uses the other training images as surrogates for test images, optimizing the complexity of the model to maximize its predictive capacity. Fig. 30 shows the distributions over the number of mixture components k_i^F and k_i^B for the optimized foreground and background models derived from the training images.



(a)



(b)

Figure 30: Distributions over the number of mixture components for the optimized foreground k_i^F (a) and background k_i^B (b) models, over the training images.

3.5.3 Selecting an Appearance Model

Given a novel test image, we wish to select appropriate foreground and background appearance models from our GMM dictionary learned from training data. This is a chicken-and-egg problem: If we knew which test pixels were foreground and background, we could evaluate each of the models from our foreground dictionary on the set of foreground test pixels and each of the models from our background dictionary on the set of background test pixels, selecting the models with maximal likelihood. However, to discriminate foreground from background pixels we first need to apply our appearance model.

We resolve this chicken-and-egg problem with a simple bootstrap approach. First, we use the methods of Section 3.5.1 to compute the geometric prior $p(x_i = F)$ over all pixels of the test image. We then use this geometric prior to identify samples $\hat{Y}^F$ and $\hat{Y}^B$ of the colours in the test image that we believe to be drawn from foreground or background regions:

$$\hat{Y}^F = \{\mathbf{y}_i : p(x_i = F) > 0.5\} \quad (42)$$

$$\hat{Y}^B = \{\mathbf{y}_i : p(x_i = F) = 0\} \quad (43)$$

The asymmetry between these two thresholds arises from the asymmetry in the geometric priors (Fig. 29) derived from the training data. While there are many pixels with a foreground probability of zero, there are very few with a foreground probability of one.

Finally, we select from our dictionary foreground and background appearance models that maximize the likelihood over these samples, again assuming conditional independence of appearance

observations over pixels:

$$\mathcal{M}^F = \mathcal{M}_{\hat{i}}^F, \quad (44)$$

$$\text{where } \hat{i} = {}_i p(\hat{Y}^F | \mathcal{M}_i^F) \quad (45)$$

$$\text{and } p(\hat{Y}^F | \mathcal{M}_i^F) = \prod_{\mathbf{y}_i \in \hat{Y}^F} p(\mathbf{y}_i | \mathcal{M}_i^F) \quad (46)$$

$$\mathcal{M}^B = \mathcal{M}_{\hat{i}}^B, \quad (47)$$

$$\text{where } \hat{i} = {}_i p(\hat{Y}^B | \mathcal{M}_i^B) \quad (48)$$

$$\text{and } p(\hat{Y}^B | \mathcal{M}_i^B) = \prod_{\mathbf{y}_i \in \hat{Y}^B} p(\mathbf{y}_i | \mathcal{M}_i^B) \quad (49)$$

These models allow us to compute appearance likelihoods $p(\mathbf{y}_i | x_i = F, \mathcal{M}^F)$ and $p(\mathbf{y}_i | x_i = B, \mathcal{M}^B)$ for each pixel in the test image.

Fig. 31 shows foreground $\hat{Y}^F$ and background $\hat{Y}^B$ appearance samples for an example test image, and the training images from which optimal foreground $\mathcal{M}^F$ and background appearance models $\mathcal{M}^B$ are drawn for this test image.

3.5.4 Fusion of Geometry with Appearance

Given a novel image, geometric and appearance cues are fused by computing the posterior odds of the pixel labels:

$$\frac{p(x_i = F | \mathbf{y}_i, \mathcal{M}^F)}{p(x_i = B | \mathbf{y}_i, \mathcal{M}^B)} = \frac{p(\mathbf{y}_i | x_i = F, \mathcal{M}^F) p(x_i = F)}{p(\mathbf{y}_i | x_i = B, \mathcal{M}^B) p(x_i = B)} \quad (50)$$

An example is shown in Fig. 32.

3.5.5 Segmentation and Refinement

We segment road from non-road regions by thresholding the posterior odds. To optimize the threshold, we apply our appearance model selection method to each of the training images, using a



(a)



(b)



(c)

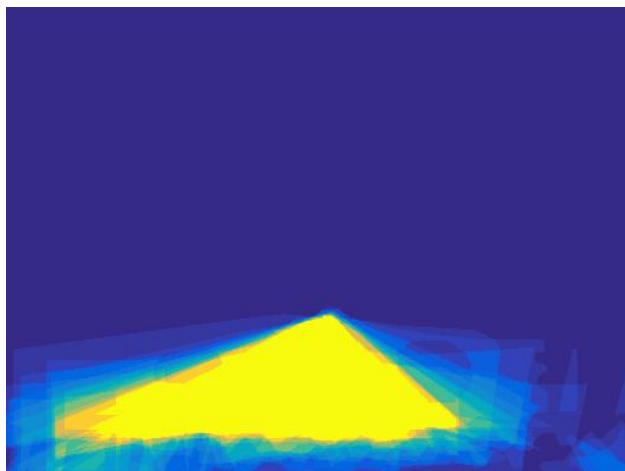


(d)

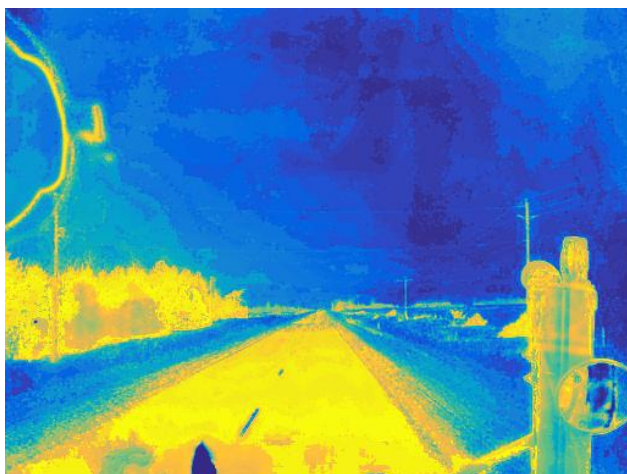
Figure 31: Selecting an appearance model. (a) Target test image. (b) Foreground $\hat{Y}^F$ (red) and background $\hat{Y}^B$ (blue) appearance samples. (c-d) Training images from which optimal foreground $\mathcal{M}^F$ and background appearance models $\mathcal{M}^B$ are drawn for this test image.



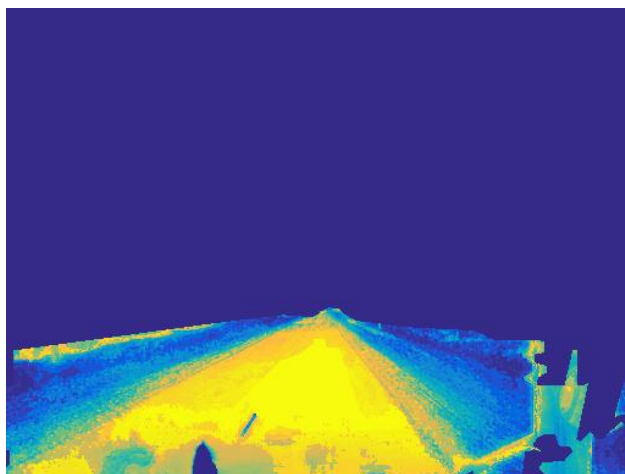
(a)



(b)



(c)



(d)

Figure 32: Fusion of geometric and appearance cues for an example test image (a). (b) Geometric prior ratio map $p(x_i = F) / p(x_i = B)$. (c) Appearance likelihood ratio map.

dictionary drawn from the remaining training images, and compute the log odds of the posterior. Sweeping over a broad range of thresholds, we select the threshold that maximizes the average intersection-over-union (IOU) of the segmented road region with ground truth over all training images.

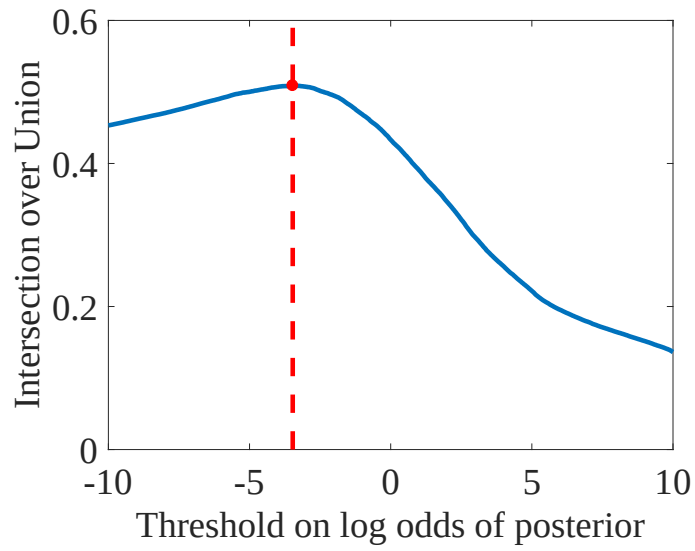
Fig. 33 (a) shows the results of this optimization. Note that the optimal empirical threshold is close to the theoretically optimal value of 0.

Since our appearance model lacks a smoothness term this initial segmentation contains small foreground fragments and holes. We therefore incorporated a refinement stage in which we filled all holes and removed segments below a threshold size, again optimizing the threshold by maximizing IOU of the detected foreground with ground truth over the training dataset. Fig. 33 (b) shows the results of this optimization.

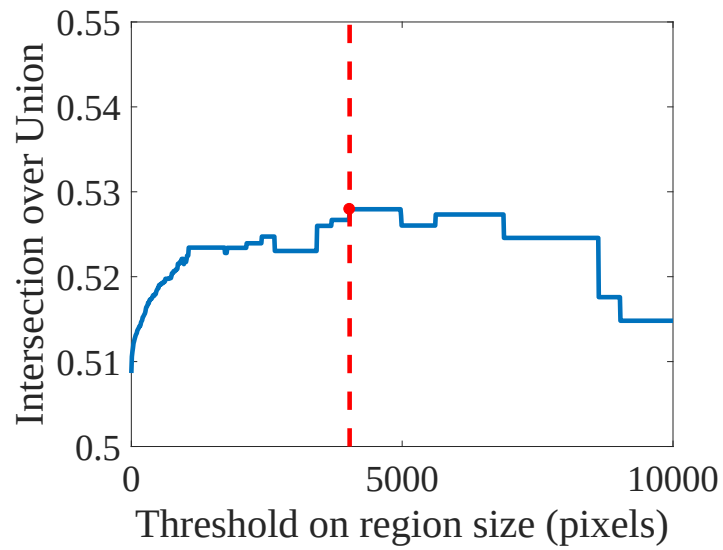
3.5.6 Implementation Details

We implemented our method in MATLAB. To fit the GMMs, we sampled 5,000 pixels from both foreground and 5,000 pixels of each training image. Full-covariance GMMs were estimated using the EM algorithm (MATLAB's `fitgmdist`) with 10 replications per model, initializing with k-means, and employing a small regularization term (10^{-6}) to ensure covariance matrices were positive definite. Morphological close operations (MATLAB `imclose`) with progressively larger structuring elements were applied until all holes were eliminated. This was assessed by counting the number of foreground regions (MATLAB function `bwconncomp`) and comparing with the Euler number (MATLAB function `bweuler`).

All experiments were conducted on a 2.6 GHZ Pentium i7 with 16GB RAM. The geometric method of Almazan *et al.* [3] takes about 18 sec per image. Total run time for appearance model



(a)



(b)

Figure 33: Optimizing segmentation and refinement. (a) Optimal threshold on log posterior odds. (b) Optimal threshold on region size.

selection, fusion and refinement is roughly 3.7 sec per image.

3.6 Results

Fig. 34 compares the performance of our fusion method against the geometric method of Almazen *et al.* [3] on the test dataset, in terms of foreground IOU with ground truth. We find that fusing appearance cues with the geometric method of Almazen *et al.* [3] increases IOU by about 15% (from 43.8 to 50.3). Matched-sample t-tests between the mean IOU for our fusion method (refined) and the geometric method of Almazen *et al.* [3] confirm that these improvements are statistically significant ($t(49) = 2.6, p = 0.012$).

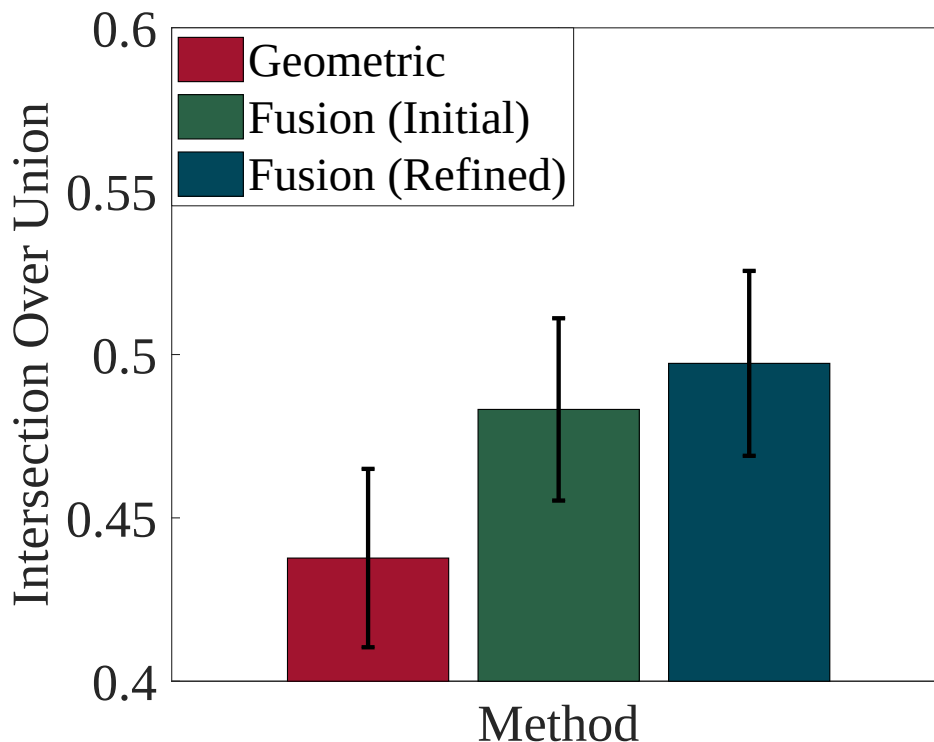


Figure 34: Mean performance on test set. Error bars indicate standard error of the mean.

Table 3 compares these results against SegNet [8]. We compared with SegNet using the pre-

trained weights made publicly available by the original authors. We noticed that SegNet sometimes confuses the road and pavement categories and therefore assess both of these individually as well as their union. SegNet achieves an average IOU of only 29.3, performing best when road and pavement categories are combined.

Method	Mean	Std. Err.
Geometric	43.8	2.7
Fusion (initial)	48.6	2.9
Fusion (refined)	50.3	3.1
SegNet (road)	28.9	3.5
SegNet (pavement)	10.0	2.1
SegNet (road + pavement)	29.3	2.7

Table 3: Quantitative comparison (IOU for foreground) with SegNet [8] on test set.

Fig. 35 shows sample results from the test set.

3.7 Conclusions and Future Research

SegNet [8] did not perform well on the Qian *et al.* test set [3]. Some of the errors may be due to lack of familiarity with wet and snowy ground conditions and occlusions by the vehicle, although it also commits errors where the road surface seems relatively clear (e.g., Fig. 35, second row). Its performance might be improved by fine-tuning on the Qian *et al.* training dataset; this is future work. However, it is still striking that our relatively simple geometric and fusion methods outperform SegNet by such a large margin.

Since our current training sets consist of only 50 images each it is feasible to include a model



(a) (b) (c) (d) (e) (f)

Figure 35: Example results. (a) Test image (b) Ground truth segmentation. (c) Geometric method [3]. (d) Fusion method (initial). (e) Fusion method (refined). (f) SegNet method (road + pavement, pretrained weights provided by the original authors).

from each of the training images in our foreground and background dictionaries. Scaling up to larger and more diverse datasets will require a little more thought in forming the dictionary if the algorithm is to remain efficient. One option is to simply cluster the models, representing each cluster by its centre. Another is to organize the models in a decision tree so that only a logarithmic number of models need be evaluated for a given test image.

4 Geometry-guided Domain Adaptation for Road Segmentation

4.1 Introduction

Building upon the integration of geometric and appearance cues for supervised road segmentation presented in Chapter 3, the following chapter explores how geometric information can be utilized for unsupervised domain adaptation. This shift addresses the challenges of adapting segmentation models to diverse, real-world environments where labeled data is scarce.

Automatic road segmentation is an important computer vision problem for assisted and autonomous vehicle systems and for road assessment, clearing and maintenance technologies. Deep fully-convolutional neural networks [51, 8] can perform this task well under favourable weather conditions and when trained and evaluated on disjoint partitions of the same dataset. However, performance drops substantially when training and test data are drawn from distinct datasets, particularly when weather, lighting or viewing geometry shift. Here we explore whether classical geometric methods can be used to automatically adapt neural networks to new viewing and/or weather conditions. The key idea is that these geometric methods embody an expertise that is to some degree independent of the expertise encoded in the network. This complementary expertise can therefore potentially be used as a teaching signal. This novel method for domain adaptation is evaluated across five diverse datasets, including a novel dataset with highly diverse weather conditions that we make publicly available as part of this effort.

In summary, we make three contributions:

1. A novel geometric algorithm that generates a road segmentation estimate from the estimated road vanishing point.
2. A novel geometry-guided domain adaptation algorithm for adapting a neural network to novel weather, lighting or viewing conditions.
3. A novel road segmentation dataset with diverse weather, lighting, and viewing conditions that we make publicly available to drive future research.

4.2 Contributions

This work is a collaboration between me, my colleague Yiming Qian, our advisor Dr. James Elder, and a visiting graduate student Yue Wang. I am the first author of the published paper.

Individual Contributions. my personal contributions have been listed below:

- Design and develop the method under the guidance of my advisor Dr. James Elder.
- Create the Toronto-2020 dataset with my colleagues. My job involves (1) splitting the images into training/test sets and (2) generating ground truth.
- Generate additional ground truth for four existed datasets that we used in the proposed method. My job consists of (1) creating the ground truthing tools for gathering vanishing point and ego-vehicle pixels, (2) generating ground truth.
- Implement the proposed method.
- Evaluate the proposed method and do comparisons with other state-of-the-art (SOTA) methods.

Scientific and Engineering Contributions. This project advances both the scientific understanding and engineering practice of domain adaptation for road segmentation by demonstrating that explicit geometric reasoning through vanishing-point-anchored priors can guide deep neural network adaptation with unlabeled target data. It shows that geometry provides interpretable, complementary teaching signals for DNN adaptation. Engineering-wise, the proposed geometry-guided adaptation offers a lightweight, and data-efficient alternative to GAN-based approaches (e.g., [83]), enabling low-cost, reproducible adaptation suitable for real-world autonomous and road-maintenance systems. The accompanying Toronto-2020 dataset further establishes a valuable benchmark for studying adaptation under diverse weather and viewing conditions.

4.3 Related Works

A detailed review of prior research, including foundational and advanced methods in road segmentation as well as geometry-driven unsupervised domain adaptation techniques, can be found in Chapter 2.

4.3.1 Road Segmentation

Most road segmentation algorithms are based on supervised machine learning from labelled datasets. Prior to the current emphasis on deep learning, most algorithms relied either on appearance (colour, texture) [74, 69, 46, 22, 79] or geometric cues (e.g., vanishing point, horizon) [39, 64, 65, 6, 3].

Appearance-based methods tend to fail when the road surface is covered or partially covered with snow or ice, and when sidewalk and roadway surfaces are similar. Since they do not rely upon surface appearance, geometric methods can potentially generalize across diverse lighting and

weather conditions, however they fail to exclude pixels from objects on the road, such as motor vehicles and pedestrians. Cheng *et al.* [18] proposed a Bayesian road segmentation method that fuses geometry with Gaussian mixture models over appearance. However, the method was only evaluated on a single, small dataset.

Recent road segmentation research has focused on deep neural networks (DNNs). The Stixel-Net [48] algorithm divides the image into narrow columns and assumes that the road extends from the bottom of the image up to some upper bound in each column. A 5-layer convolutional neural network (CNN) derived from LeNet [44] is used to estimate this upper bound independently in each column, and a simple Markov model is used to compute an optimal upper road boundary across columns. The approach is straightforward but unfortunately unable to handle multiple disjoint occlusions of the road within a column.

More recent DNN-based approaches for road segmentation have employed VGG-based architectures. SegNet [8] is a fully convolutional encoder-decoder architecture involving two adapted VGG16 [73] networks connected end-to-end that estimates the road segmentation over the entire image within a single pass. s-FCN-loc [87] is a two-stream Siamese VGG-based network that takes as input both the RGB image and an edge saliency map. MultiNet [81] is a multi-task (road segmentation, street classification, and vehicle detection) DNN in which a single encoder is shared by three distinct decoders trained for different tasks. Encoders based on both VGG16 and ResNet [33] were assessed. These methods have mainly been trained and evaluated on the KITTI road segmentation dataset [29].

4.3.2 Unsupervised Domain Adaptation

With the help of DNN, researchers are extending the conventional supervised road segmentation to other high-level practical tasks, e.g., domain adaptation, and domain generalization.

Domain adaptation is a sub-topic in the transfer-learning research area which aims to learn a model from the source domain and then adapt it to a different but related target domain. In general, the ground truth (GT) of the target domain can be accessed for better supervision; however, we are focusing on a more complicated situation, i.e., unsupervised domain adaptation (UDA), in which only unlabeled target domain data are provided.

UDA is a highly effective and practical technique that can be applied to a variety of research topics. In particular, UDA for semantic segmentation is getting much attention. This attention is because, nowadays, it is easy to capture or collect images; however, accurate multi-class ground-truthing is still a laborious and slow task. With the help of UDA, semantic segmentation can be conducted by first selecting a model that has been trained on a related source domain and then adapting this model to the user's unlabeled dataset. This adaptation task is also a common problem in the road segmentation research field. For example, it is common that we get some pre-trained road segmentation models, then we try to apply them to a novel unlabeled target domain in which weather, lighting, or viewing conditions are different from the source domain.

To our knowledge, there are very few UDA methods (e.g., [18, 17]) only focusing on road segmentation although a considerable number of UDA methods (e.g., [83, 14, 86, 93, 101, 94, 88]) have been proposed for solving the semantic segmentation problem. Usually, these UDA methods focus on adapting the model from the synthetic datasets (e.g., GTA5 [66] or SYNTHIA [68]) to the datasets (e.g., KITTI, Cityscapes, or BDD [95]).

Dataset Name	Images (Training/Test)	Resolution Conditions	Ego-Vehicle	Weather	Camera
CamVid [11, 10]	367 / 233	960×720	Yes	dry	Fixed
KITTI (training set) [29]	145 / 144	1382×512	No	dry	Fixed
Cityscapes (training set) [21]	1528 / 1406	2048×1024	Yes	dry	Fixed
Qian [3]	34 / 31	2048×1536	Yes	dry, wet, ice-covered, snow-covered	Variable
Toronto-2020 [20]	400 / 400	1024×576	Yes	dry, wet ice-covered, snow-covered	Variable

Table 4: The five road segmentation datasets used in this project.

4.4 Datasets

We employ five diverse datasets to study the problem of domain adaptation for road segmentation (Table 4). Three of these are public datasets, one we obtained directly from the authors, and one is a new dataset we have created and published [19]. For all of these datasets, pixel-level road/non-road labels are available. We describe each of the datasets below; sample images are shown in Fig. 36.

CamVid [11, 10] is a semantic segmentation dataset captured with a fixed vehicle-mounted camera. The dataset consists of 700 960×720 -pixel images partitioned into a 367/233/100 training/test/validation split. Weather conditions are consistently dry but there are substantial variations in illumination. The dataset provides 32 semantic labels; these we binarize into road/non-road.



Figure 36: Example image from each dataset. (a) CamVid. (b) KITTI. (c) Cityscapes. (d) Qian. (e) Toronto-2020.

KITTI [29] consists of 579 1382×512 -pixel images captured with a fixed vehicle-mounted camera and partitioned into a 289/290 training/test split. Weather conditions are clear and roads are dry. Since ground truth is provided only for the training set, we randomly partitioned the training set into a 145/144 training/test split.

Cityscapes [21] is a larger-scale dataset for urban scene understanding, containing 5,000 2048×1024 -pixel images captured by vehicle-mounted cameras across 50 different cities in Germany. The dataset is partitioned into a 2975/1525/500 training/test/validation split. Since ground truth is provided only for the training set, we selected from the training set only those images containing roads and then partitioned it into a 1528/1406 training/test split.

Qian *et al.* Dataset. Qian *et al.* [3] created a dataset to study the problem of estimating road weather conditions. The dataset is small - only 100 2048×1536 -pixel images, partitioned into a 50/50 training/test split, but the images span a wide range of weather (dry,wet,snow,ice), illumination and viewing geometries. Unlike CamVid, CitySCapes and KITTI, the camera is a dash-cam manually placed on the windshield, so height and camera tilt vary considerably. We have obtained the dataset directly from the authors. Some images in this dataset are captured in parking lots and not roads. We have filtered these out to produce a 34/31 training/test split.

Toronto-2020. Our goal in this work is to study unsupervised adaptation across large changes

in weather, lighting and viewing geometry. In the larger existing datasets the weather is generally fair, the roads are dry, and the camera geometry is fixed. The Qian *et al.* dataset provides much more variation, but it is relatively small. To address these limitations, we introduce here a new, much larger ground-truthed road segmentation dataset we have named Toronto-2020, as all images are of street scenes in the Greater Toronto Area. Like the Qian *et al.* dataset, the Toronto 2020 dataset provides substantial variability in weather, illumination and viewing geometry, but is much larger - 800 1024×576 -pixel images partitioned into a 400/400 training/test split. In addition to pixel-level road/non-road labels, we provide manually-estimated ground truth vanishing points and horizon lines. The images were obtained in both urban and highway scenes under varying weather conditions, including dry, wet, and snow-covered roads, and during both daytime and nighttime conditions.

To study the problem of domain adaptation for road segmentation, we conduct five experiments. In each, four of the datasets are employed as source datasets with ground truth labels and the fifth is employed as a hold-out target dataset without labels. To facilitate transfer, we first resize images in all datasets to 640×480 pixels.

To support geometry learning, we augment the pixel-level road/non-road labelling with manual annotations of road vanishing point locations (Fig. 37 (b)).

For all datasets except KITTI, parts of the ‘ego vehicle’, i.e., the vehicle on which the camera is mounted, are sometimes visible within the image. We view this as a nuisance that generally should be taken care of by manual application of a mask provided in a calibration stage or automatically using motion information in a burn-in stage of pre-processing. As this is beyond the scope of the present study, we provide manual masks for these image regions (Fig. 37 (c)), which are then ignored during learning, adaptation and evaluation. Training/validation splits and all additional



Figure 37: Additional ground-truthing. (a) Ground truth vanishing point (b) Ego-vehicle mask

annotations for the four public datasets are available at [19].

4.5 Proposed Method

Fig 38 shows an overview of our system, which consists of supervised training, unsupervised adaptation and inference stages. We detail each in turn below.

4.5.1 Supervised Training

The availability of ground truth pixel-level labels and vanishing points for source domain images allows us to pre-train both appearance- and geometry-based road segmentation subsystems (Stage 1 of Fig. 38). For the appearance subsystem we employ SegNet [8], but in principal any data-driven supervised algorithm could be employed. For the geometry subsystem we develop a novel method for vanishing point estimation which will allow the formation and application of a geometric road prior anchored on the estimated road vanishing point.

Appearance Subsystem. We used the original PyTorch implementation of SegNet [73],

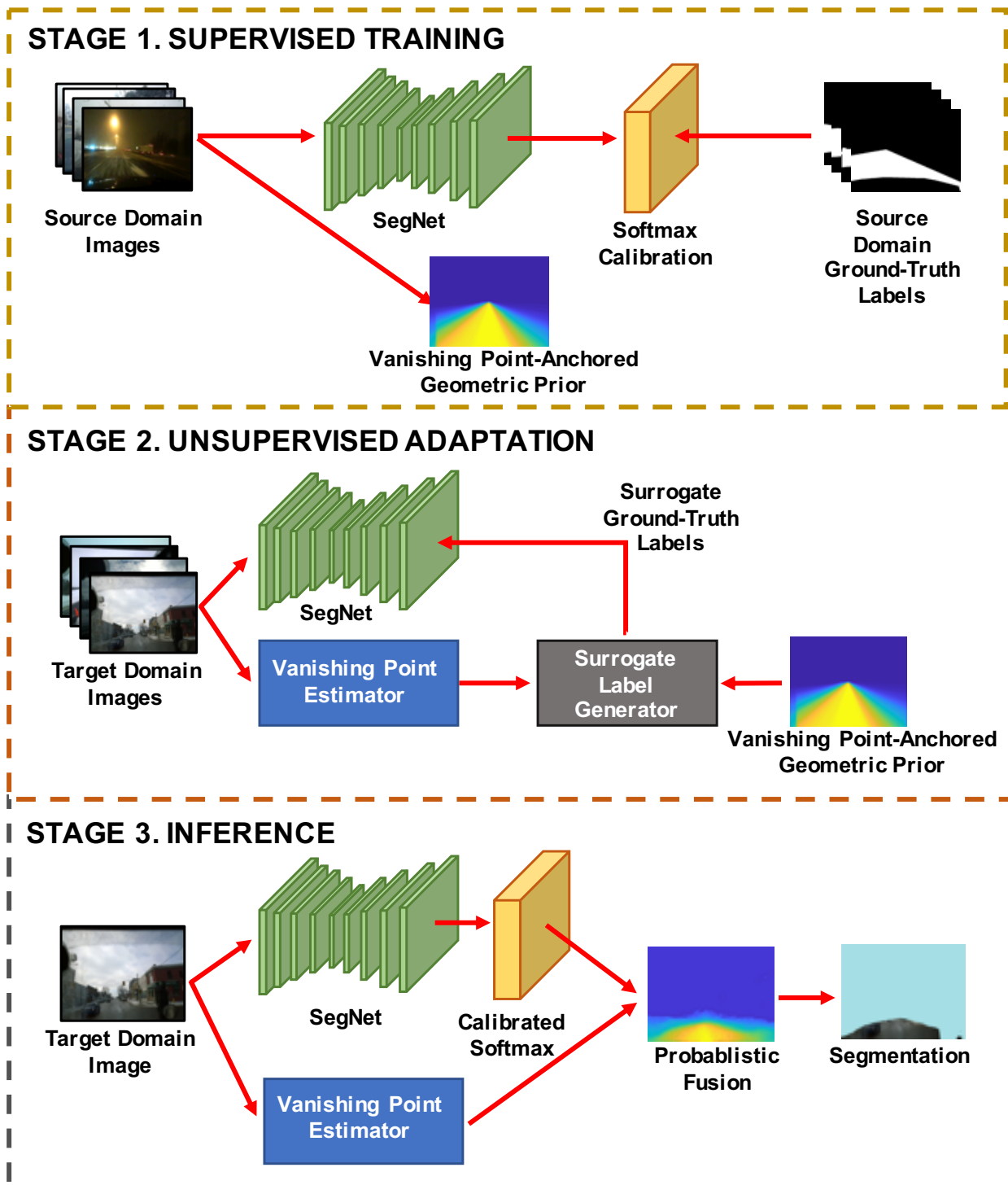


Figure 38: Overview of training, adaptation and inference processes.

initialized with Imagenet-trained VGG weights, replacing only the last softmax layer with a 2-channel softmax layer. We employed a batch size of 16, learning rate of 0.001, momentum of 0.9, and weight decay of 0.0001. The network was trained for 600 epochs.

Optimal fusion of appearance and geometry will depend upon a probabilistic representation which fairly reflects the uncertainty in both cues. Unfortunately, softmax outputs from DNNs are typically not representative of empirical probabilities, being far too confident one way or the other. In order to convert our network output to something closer to probability, we employ temperature scaling [31] over the source domain ensembles.

Geometry Subsystem. The pixel-level ground-truth road labels for the source ensemble can be used to form a prior for the location of the road in the target domain. However, direct estimation of this prior from the pixel labels suffers from two problems. First, variation in the camera parameters within and between source datasets will lead to a diffusion of the prior, weakening its predictive power. Second, variation in the camera parameters between source and target datasets and within the target dataset will lead to bias in application of the prior to the target domain.

To address these issues, we propose a novel vanishing-point anchored prior that will help to mitigate the impact of variation in camera parameters. This will involve the development of a novel road vanishing point estimation algorithm that yields an estimate of the vanishing point for both source and target domain images.

Vanishing Point Estimation

Our novel vanishing point estimation algorithm consists of three stages:

- *Line detection.* We employ the MCMLSD algorithm [4] to recover a ranked list of line segments in the image. (Code available at [2].) Fig. 39 (b) shows an example of the line segments detected in an example image. While not all of the segments detected in an image are likely to align with the road, we hypothesize that for many images, road lines will be ranked high and will generate strong evidence for the road vanishing point. However, the number K of detected segments to use in estimating the vanishing point is an important parameter: small values of K will lead to noisy estimates, while large values may swamp the genuine cues from road lines with distractor cues from background lines generated by other scene structures.

To optimize K , we perform a grid search over the range $(5, 10, \dots, 150)$, minimizing the RMS error in vanishing point estimation using four-fold leave-one-out cross validation over the four source datasets. We find that the optimal value of K ranges from 85 to 125 over the five source ensembles.

- *Intersection Detection.* Given the top K line segments, we identify all pairwise intersections of the lines passing through the segments that fall within the image frame (Fig. 39(c)).
- *Kernel density estimation (KDE).* We estimate the road vanishing point as the point within the image frame with the highest density of pairwise line intersections. We estimate the density using Gaussian KDE (Fig. 39(c)), with bandwidth σ selected adaptively for each image and separately for x and y dimensions based upon the number N of intersections detected, using Scott’s rule [70]: $\sigma = \hat{\sigma} N^{-1/6}$, where $\hat{\sigma}$ is a robust estimate of the standard deviation of the points: $\hat{\sigma} = \text{MAD}/0.6745$, and MAD is the median absolute deviation of the

points. For efficiency, we localize intersections to the nearest pixel and implement the KDE as $x - y$ separable 1D convolutions. Fig. 39 (d) shows the estimated intersection density for an example image. Fig. 39 (e) shows the location of the peak of this distribution, which forms our estimate of the road vanishing point. Fig. 39 (f) shows the ground truth vanishing point. Fig. 40 shows additional examples.

Geometric Prior Formation

In this work we assume that road vanishing point estimates lie within the visible image for all datasets. (Although note that a more relaxed assumption could be employed if necessary.) Based on this assumption we create a virtual prior image with twice the maximum horizontal and vertical dimension of the images in our source ensemble. The empirical prior is then formed by shifting the ground-truth road pixels by the negative of the estimated vanishing-point offset from the centre of the image, prior to accumulating the road pixel labels into the empirical prior. (Note that use of the estimated vanishing point rather than the ground truth vanishing point will account for error in the vanishing point estimated for target domain images.)

Specifically, labelling road and non-road pixels as 1 and 0, respectively, the geometric road prior $p_g(x, y)$ computed from the n images in the source ensemble becomes

$$p_g(x, y) = \frac{1}{n} \sum_{i=1}^n t_i(x - u_i, y - v_i) \quad (51)$$

where $t_i(x, y)$ is the ground-truth label map for the i th source image and (u_i, v_i) is the offset from the centre of the image of the estimated vanishing point. An example of the prior, anchored to the centre of the image, is shown in Fig. 41 (a).

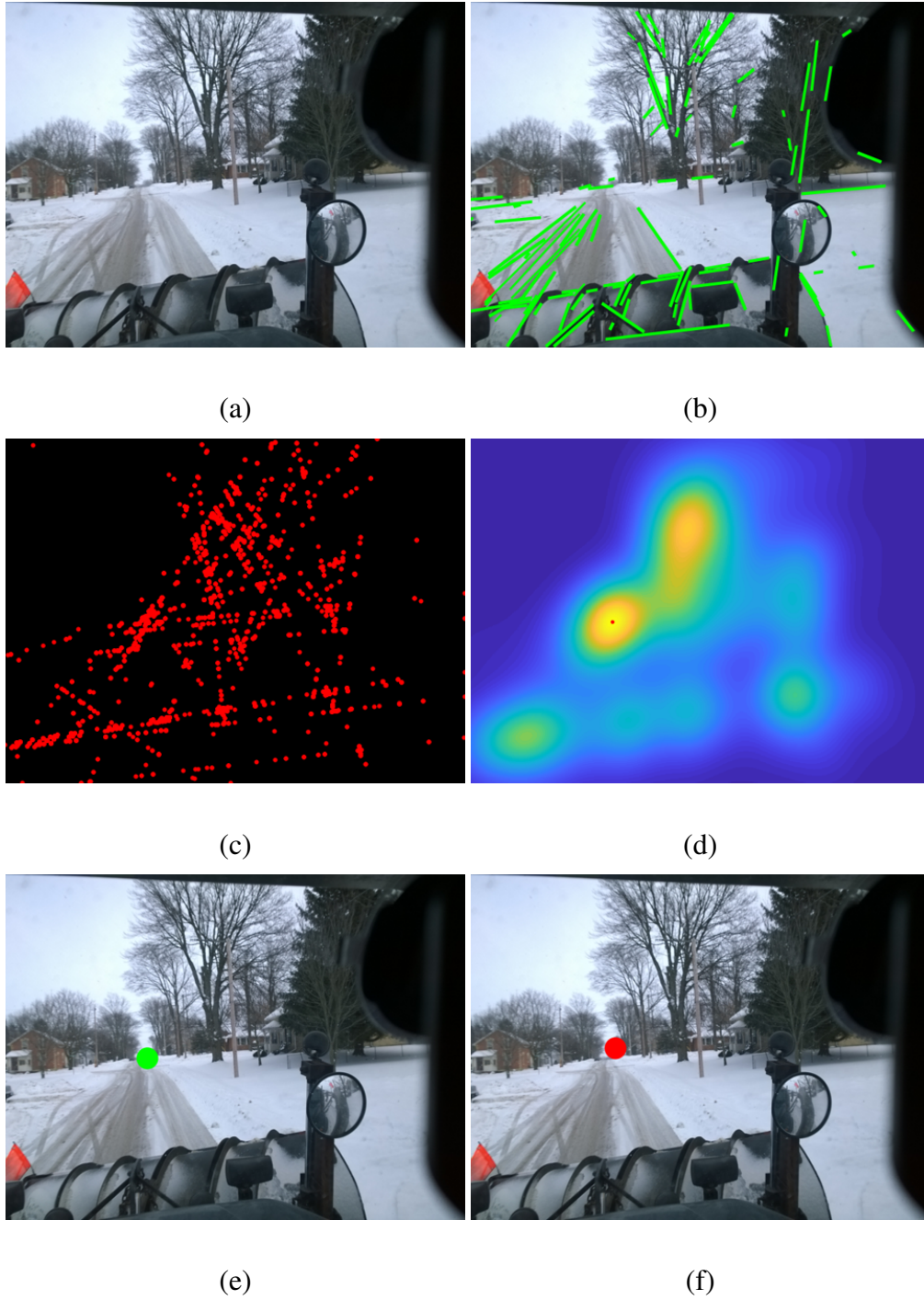


Figure 39: Vanishing Point Detection. (a) Sample input image. (b) 105 top-ranked line segments detected by MCMLSD [4]. (c) Elongating line segments and producing pair-wise intersections. (d) Kernel Density Estimation on intersections. (e) Estimated vanishing point. (f) Ground truth vanishing point.



Figure 40: Vanishing Point Estimation Examples. Green dots: estimated vanishing points. Red dots: Ground truth vanishing points.

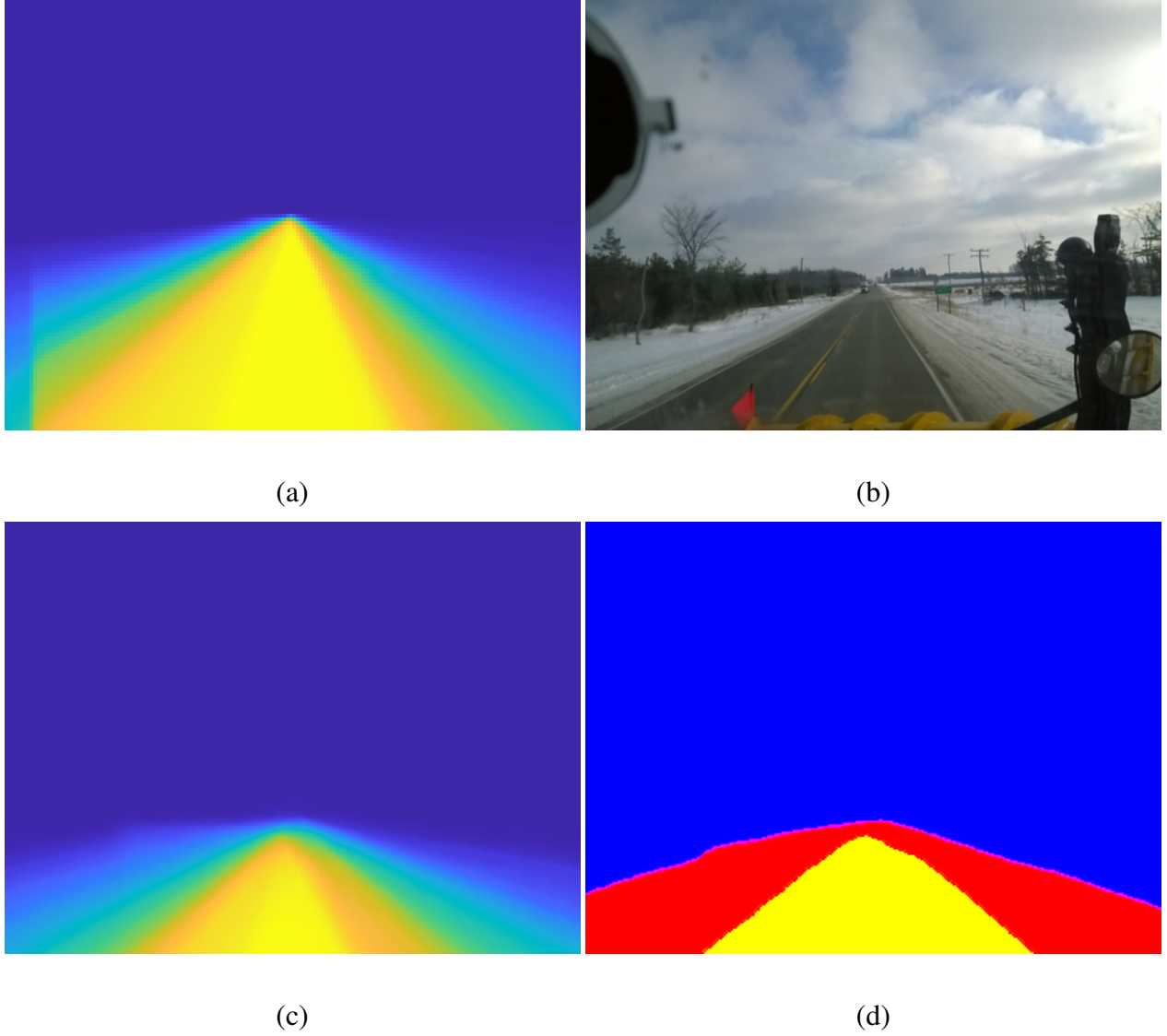


Figure 41: Geometry-guided neural network adaptation. (a) Geometric prior generated from the source domain ensemble. (b) Input target image. (c) Geometric prior anchored to the estimated vanishing point in the target image. (d) Tri-mask identifying high-confidence foreground (yellow) and background (blue) regions. Regions of intermediate confidence (red) are ignored and do not influence the adaptation loss.

4.5.2 Unsupervised Adaptation

Once appearance and geometry subsystems have been pre-trained on the source ensemble, the system is ready to adapt to the target domain (Stage 2 of Fig. 38). Target domain images are input to the appearance sub-system (SegNet) to generate an initial appearance-based segmentation estimate, as well as the vanishing point estimator of our geometric subsystem to estimating the vanishing points for the target images. Anchoring the geometric prior (Fig. 41 (a)) on the estimated vanishing point yields a geometric probability estimate p_g of the road pixel labels for the target image (Fig. 41 (c)).

The key hypothesis of this work is that this geometry-driven estimate embodies knowledge that is to some degree independent of the expertise learned by the network, and thus can serve as a teaching signal to adapt the network to the new domain. Since the geometric estimate p_g is only an approximation to the ground truth, we hypothesize that more accurate domain adaptation will be achieved if only the more confident regions of the geometric estimate factor into the adaption loss as surrogate ground truth. In particular, we define a pair of probability thresholds (p_l, p_u) , $0 \leq p_l < p_u \leq 1$ that together define a tri-mask for the target image (Fig. 41(d)), consisting of: (1) a region of pixels (blue) where $p \leq p_l$ that will be used as surrogate ground truth for background (non-road); (2) an intermediate region (red) where $p_l < p < p_u$ that is deemed uncertain and therefore not used in the loss and (3) a region (yellow) where $p \geq p_u$ that will be used as surrogate ground truth for foreground (road).

Thresholds p_l and p_u were selected to maximize road segmentation accuracy on the source ensemble using an intersection-over-union (IOU) performance measure. Specifically, we conducted a grid search over $p_l \in (0, 0.1, \dots, 0.5)$ and $p_u \in (0.5, 0.1, \dots, 1)$, using four-fold leave-one-out

cross-validation over the four datasets in the source ensemble. In fine-tuning the network, we employed the same parameters used to train on the source ensemble, but reduced the duration to 1,000 iterations. To reduce overfitting to the geometric prior we randomly cropped and shifted the prior horizontally by ± 20 pixels.

4.5.3 Inference

Once the network is adapted to the target domain it can be used to generate probabilistic appearance-based estimates of road segmentation for target domain images (Stage 3 of Fig. 38). The output can then be thresholded at 0.5 to yield a road segmentation estimate. However, even after the network has been adapted based upon the geometric prior, it is conceivable that the geometric and appearance subsystems still carry complementary information. This could be due to ignoring lower-confidence regions in the prior, but also to differences in what is easily computable using these two very different styles of computation. We thus explored whether probabilistic combination of the geometric and adapted appearance estimates might yield better results than either sub-system used alone. To convert the adapted SegNet output to probabilities, we employ the calibrated softmax stage trained on the source ensemble (Fig. 38).

We interpret the output of adapted appearance and geometric sub-systems as posterior probabilities $p(F|a)$ and $p(F|g)$, where F represents the foreground state (road) and a and g represent the observable appearance and geometric cues, respectively. Note that the posterior probability of the background state B is simply the complement: $p(B|a) = 1 - p(F|a)$, $p(B|g) = 1 - p(F|g)$.

Approximating appearance and geometric cues as conditionally independent, the likelihood ratio L can be written as

$$L = \frac{p(g|F)p(a|F)}{p(g|B)p(a|B)} \quad (52)$$

Using Bayes theorem, this can be re-expressed in terms of the posteriors output by the two sub-systems as

$$L = \frac{p(F|g)p(F|a)p(B)^2}{p(B|g)p(B|a)p(F)^2} \quad (53)$$

Thus the posterior odds becomes

$$L \cdot P = \frac{p(F|g)p(F|a)p(B)^2}{p(B|g)p(B|a)p(F)^2} \cdot \frac{p(F)}{p(B)} \quad (54)$$

$$= \frac{p(F|g)p(F|a)p(B)}{p(B|g)p(B|a)p(F)} \quad (55)$$

where the prior ratio $P = p(F)/p(B)$ can be learned from the source ensemble. We threshold the posterior odds using the optimal threshold computed from the source domain to generate a segmentation based on a Bayesian fusion of geometric and appearance cues. One example of geometry-appearance fusion is shown in Fig. 42.

4.6 Implementation Details

4.6.1 Run Time

All experiments were conducted on a 3.4 GHz Pentium i7 with 64 GB RAM and an Nvidia GeForce GTX 1080 Ti graphics card. The MCMLSD line detection method takes about 6.2 sec per image. However, once lines are detected, identification of line intersections and kernel density estimation of the vanishing point location takes only around 25 msec. Real-time performance could be

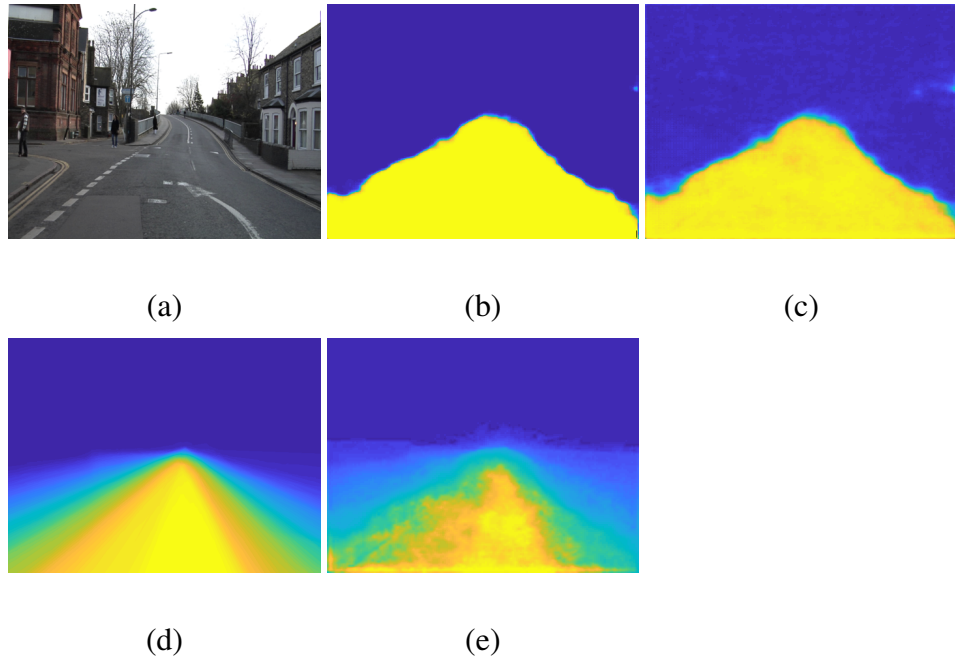


Figure 42: Fusing geometry cues with appearance cues. (a) Sample input image. (b) Uncalibrated softmax output of appearance sub-system (SegNet). (c) Temperature-scaled softmax map $p(F|a)$. (d) Geometric Prior $p(F|g)$. (e) Posterior odds $L \cdot P$ computed from (c) & (d).

achieved by swapping in a real-time line detector.

4.7 Evaluation

We evaluate results using IOU on the target datasets, ignoring masked-out ego-regions, and comparing with the state-of-the-art GAN-based AdaptSegNet [83], using the VGG16 version to form a fair comparison with our VGG16-based Segnet approach. We modify only the final softmax output layer of AdaptSegNet to have only two classes (road/non-road).

Table 5 shows quantitative results. Without adaptation, Segnet generalizes poorly, with worst results on the difficult Qian *et al.* dataset, and achieving an average IOU over all 5 targets of only 43.0%. Interestingly, the much simpler geometric method generalizes much better, performing substantially better on each of the five target datasets, and achieving an average IOU of 57.4%. Our hypothesis that this geometry approach can be used as a complementary teaching signal for SegNet is borne out by the vast improvement of the geo-adapted SegNet model over the original SegNet for all five datasets, and the substantial improvement over using the Geometry algorithm alone, on four of the five target datasets. Although our geometry algorithm performs substantially better than the unadapted SegNet algorithm on KITTI, we do not find an improvement with geo-guided SegNet adaptation. The precise reasons for this failure on KITTI are unclear, however it may be due in part to failures of the geometric prior which introduce errors in the surrogate ground truth used to adapt SegNet. (See below for more discussion of these failure modes.)

We also fail to see any substantial improvement from subsequent Bayesian fusion of the adapted SegNet and geometry solutions, suggesting that the geometry-guided adaptation of SegNet has drawn out most of the complementary information from both approaches.

Dataset	SegNet [8]	Geometry	Geo-Adapted SegNet	Fusion	AdaptSegNet (VGG-16) [83]	Fusion-Adapt (Major Vote)
CamVid	39.2	56.0	60.7	61.0	65.8	61.8
KITTI	39.1	47.6	47.7	46.9	56.7	48.7
Cityscapes	60.8	70.7	74.5	74.7	71.1	76.2
Qian	31.2	55.6	57.9	58.3	52.4	60.1
Toronto-2020	44.7	57.3	65.4	65.6	60.7	67.2
Mean	43.0	57.4	61.2	61.3	61.3	62.8

Table 5: Quantitative evaluation (IOU)

While on average our geo-adaptation method performs comparably to the GAN-based AdaptSegNet approach, it is interesting to see how much they differ on individual datasets: While AdaptSegNet produces substantially superior results on CamVid and KITTI, our geo-adaptation method is substantially better on CityScapes, Qian *et al.* and our new Toronto-2020 dataset. We conjecture that the superior performance of our approach on these last three datasets may be due in part to their greater variability in geometric parameters, which is handled explicitly by anchoring to the vanishing point.

This complementarity of the geometric and GAN approaches led us to wonder whether the two could be synergistically combined. While a strong fusion of the two approaches is beyond the scope of the present work, we report in the final column of Table 5 the IOUs that result from a majority vote between geometry, geo-adapted SegNet and AdaptSegNet segmentations, and we do see an improvement on average, which suggests that a more careful study of how these distinct approaches can be optimally combined may be worthwhile.

The representative examples shown in Fig. 43 demonstrate that in some cases the improvement achieved by geo-guided adaptation can be dramatic, and far better than achieved with the GAN-based AdaptSegNet approach.

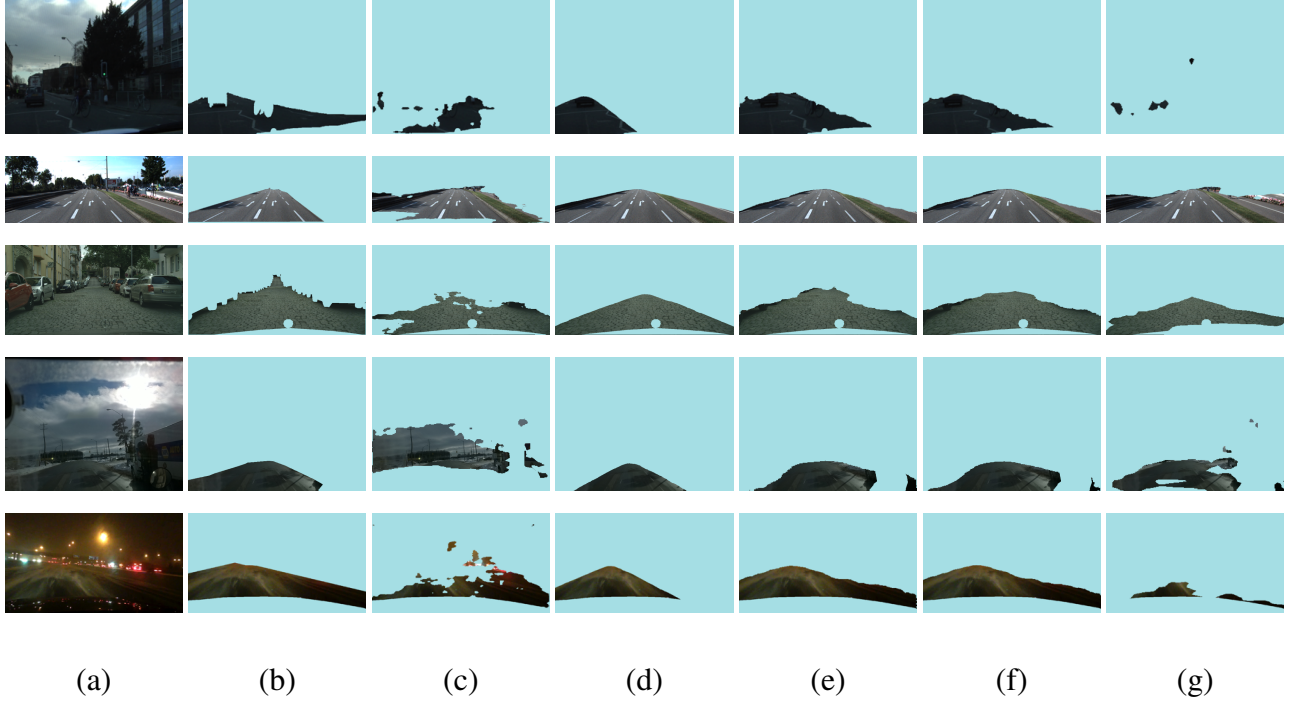


Figure 43: *Qualitative results. From top to bottom, target images from CamVid, KITTI, Cityscapes, Qian et al., Toronto-2020 datasets. (a) Target image (b) Ground truth (c) SegNet (d) Geometry (e) Geo-guided SegNet adaptation (f) Fusion (g) AdaptSegNet (VGG-16)*

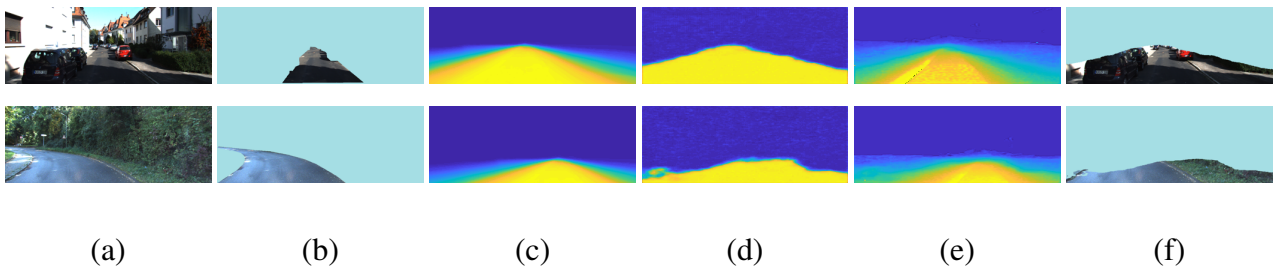


Figure 44: *Representative failures from the KITTI test dataset. (a) Input test image. (b) Ground truth. (c) Geometric prior. (d) Adapted Softmax. (e) Fusion. (f) Fusion Segmentation Result.*

4.8 Conclusion and Future Research

The five datasets employed in this study were generated from cameras with diverse extrinsic (height, tilt) and intrinsic (focal length, resolution) parameters. We ignored this diversity in re-formatting all images to a standard 640×480 pixel format. In our view, for cameras with fixed focal lengths and mounts these parameters should be accounted for, at least roughly, and images mapped homographically to a common virtual image. We expect that this will reduce the geometric deviations between datasets and lead to better transfer.

We are encouraged by the evidence reported here that geometry can be used to drive adaptation in deep networks and are excited about exploring more sophisticated geometric models and adaptation protocols to further improve the adaptability of segmentation systems.

5 VCSeg: Virtual Camera Adaptation for Road Segmentation

5.1 Introduction

Having demonstrated the efficacy of geometry-guided techniques for domain adaptation in Chapter 4, this chapter proposes a UDA method that takes camera-specific parameters into consideration. By projecting images into a virtual camera domain, it introduces a novel framework for addressing domain shifts, enabling finer-grained adaptation to both between-camera and within-camera variations.

For the unsupervised domain adaptation for road segmentation, one of the essential causes of domain shift is the variation of geometric camera settings. Due to the different camera settings, misregistration of road scene structure happens between the source domain and the target domain. For example, in the geometry-guided road adaptation project (GeoSeg in short, illustrated in Chapter 4), we encountered a severe misregistration issue when inferencing on the KITTI dataset. This was caused by the focal length differences between the target domain (i.e., the KITTI dataset) and the source domain (i.e., all the other four datasets).

To address the above-mentioned misregistration issue that is caused by the camera shift, we first decompose the overall camera shift into two categories: (1) Between-camera shift and (2) Within-camera shift. Briefly speaking, the between-camera shift is the shift across given road datasets in which both intrinsic and extrinsic camera parameters are different from one to the other. Within-camera shift denotes the shift across different images within one certain road dataset; therefore, it is only affected by the variations of extrinsic camera parameters. More detailed illustration about between-camera shift and within-camera shift is given in the Section 5.5. After reprojecting all the

images to the virtual camera space, gaps are generated which will be problematic for the following road segmentation network training. To address this problem, we propose a novel projective image completion method that enables the gap fillings in a plausible way.

Using five diverse and challenging road segmentation datasets, we demonstrate that our virtual camera method dramatically improves road segmentation performance when generalizing across cameras, and propose that this be integrated as a standard component of road segmentation systems to improve generalization.

5.2 Contributions

This project has been published at the 2022 Winter Conference on Applications of Computer Vision (WACV-2022) [17]. This work is accomplished by me and my advisor Dr. James Elder. I am the first author of the published paper.

Individual Contributions. my personal contributions are shown below.

- Design the method under the guidance of my PhD advisor.
- Method implementation.
- Performance evaluation and Comparisons with SOTA.

Scientific and Engineering Contributions. This project advances scientific understanding of geometric domain adaptation by decomposing domain shift into between-camera and within-camera variations and correcting both through reprojection to a unified virtual-camera domain. Scientifically, it demonstrates how normalizing geometric camera parameters enhances cross-domain generalization, revealing the quantitative impact of camera geometry on segmentation reliability.

This study helps to clarify the importance of geometric shift in domain adaptation for road segmentation. Specifically, the results demonstrate geometric shift is probably even more important than appearance shift - as the performance boost we get from VCSeg is greater than the performance boost we get from GeoSeg. Further, most of that shift is between-camera shift, that can be calibrated out in advance. However, both geometric and photometric shifts are important and need to be addressed for any system that is to be widely deployed.

As an engineering contribution, VCSeg provides a lightweight adaptation framework based on reprojection to a virtual camera frame that requires only camera parameters and vanishing point estimation. A simple but novel projective completion technique fills gaps created by reprojection, critical for stable fine-tuning. We believe these techniques will be valuable for improving the reliability of deployed ground segmentation systems.

5.3 Related Works

For a comprehensive review of prior research relevant to this work, including both classical and state-of-the-art methods in road segmentation and unsupervised domain adaptation for road segmentation, please refer to Chapter 2.

The work proposed in Chapter 4 laid the groundwork for domain adaptation by introducing a geometry-guided adaptation technique, effectively narrowing the domain gap using geometric cues. While this approach demonstrated substantial improvements in adapting segmentation models to diverse real-world conditions, it primarily focused on leveraging geometric information without explicitly accounting for the inherent variability introduced by camera-specific parameters.

The method proposed in this chapter advances the framework presented in Chapter 4 by in-

corporating camera-specific parameters into the adaptation process. By projecting images into a virtual camera domain, this approach decomposes the domain shift into between-camera and within-camera variations, enabling more targeted and finer-grained adaptation. This advancement improves performance and generalizability compared to the geometry-guided approach in Chapter 4.

5.4 Datasets

We use five diverse road segmentation datasets (Table 6, Figs. 54,55: CamVid [11, 10], Cityscapes [21], KITTI [29], Qian [3], and Toronto-2020 [20]. We follow the GeoSeg[20] partitioning of these datasets into training and test [19].

All five datasets were captured with vehicle-mounted cameras. CamVid was captured in the UK, CityScapes and KITTI in Germany and Qian & Toronto-2020 in Toronto, Canada. While for KITTI, CamVid & CityScapes datasets weather is fair and road surfaces dry, for Qian & Toronto-2020 weather ranges widely and roads may be dry, wet, or partially covered with snow an ice. Nonlinear distortions have been removed from all images.

Table 6 also lists the intrinsic and approximate average extrinsic geometric camera parameters for the five datasets. Where camera height and tilt were not specified they were estimated by visual inspection of the images. Note the wide variation in parameters across datasets. In the following we will initially assume zero roll and pan angles, but will later relax this assumption for pan.

For CamVid, CityScapes and KITTI datasets the camera pose was fixed relative to the vehicle - variations in camera pose relative to the road thus arise primarily from vehicle motion. For the Qian & Toronto-2020 datasets, on the other hand, the cameras were removable dash-cams, and

Dataset	Images	Weather	Resolution	Principal	Focal Length	FOV [H,V]	Camera	Height	Tilt
	Training/Test	Conditions	(px)	Point (px)	[x,y] (px)	(deg)	Position	(m)	(deg)
CamVid [11, 10]	367 / 233	Dry	[960, 720]	[485, 365]	[862, 1141]	[58.2, 35.0]	Fixed	1.21*	0*
Cityscapes [21]	1528 / 1406	Dry	[2048, 1024]	[1082, 514]	[2262, 2252]	[48.7, 25.6]	Fixed	1.21	-2.4
KITTI [29]	145 / 144	Dry	[1242, 375]	[610, 173]	[721, 721]	[81.5, 29.2]	Fixed	1.65	0
Qian [3]	34 / 31	Variable	[2048, 1536]	[1024, 768]	[1845, 1845]	[58.1, 45.2]	Variable	1.50*	0*
Toronto-2020 [20]	400 / 400	Variable	[1024, 576]	[508, 328]	[776, 701]	[66.8, 44.7]	Variable	1.35	0*
Virtual	-	-	[5500, 2800]	[2750, 1400]	[2262, 2262]	[101.1, 63.5]	Fixed	1.38	0

Table 6: The five cameras and road segmentation datasets used in this work. Estimated values indicated by

*.

thus contain wider variations in camera pose due to variations in how the cameras was mounted each day.

Pixel-wise road/non-road labels were obtained from the websites for each dataset. Additional ground truth vanishing point positions and ego-vehicle mask annotations were obtained from the GeoSeg website [19].

Following GeoSeg, we train and evaluate using 5-fold leave-one-out across datasets. In other words, the network is trained on the training partitions of four source datasets and then evaluated on the test partition of the fifth, left-out target dataset. As in GeoSeg, we ignore the ego-vehicle regions when doing source training, target adaptation, and performance evaluation.

5.5 Proposed Method

Fig. 45 shows an overview of the VCseg training pipeline. Source images and ground truth are first reprojected to a standard virtual camera image plane. This will lead to undefined regions in

both the virtual camera image and ground truth mask. To address this, we employ a novel projective completion method to fill these gaps, allowing the segmentation network to be trained over multiple datasets with diverse camera parameters. To apply the network to a novel target domain, target images are processed through the same reprojection and completion pipeline, using the target camera parameters, prior to applying the network. The resulting segmentation is reprojected back to the original target camera plane for evaluation.

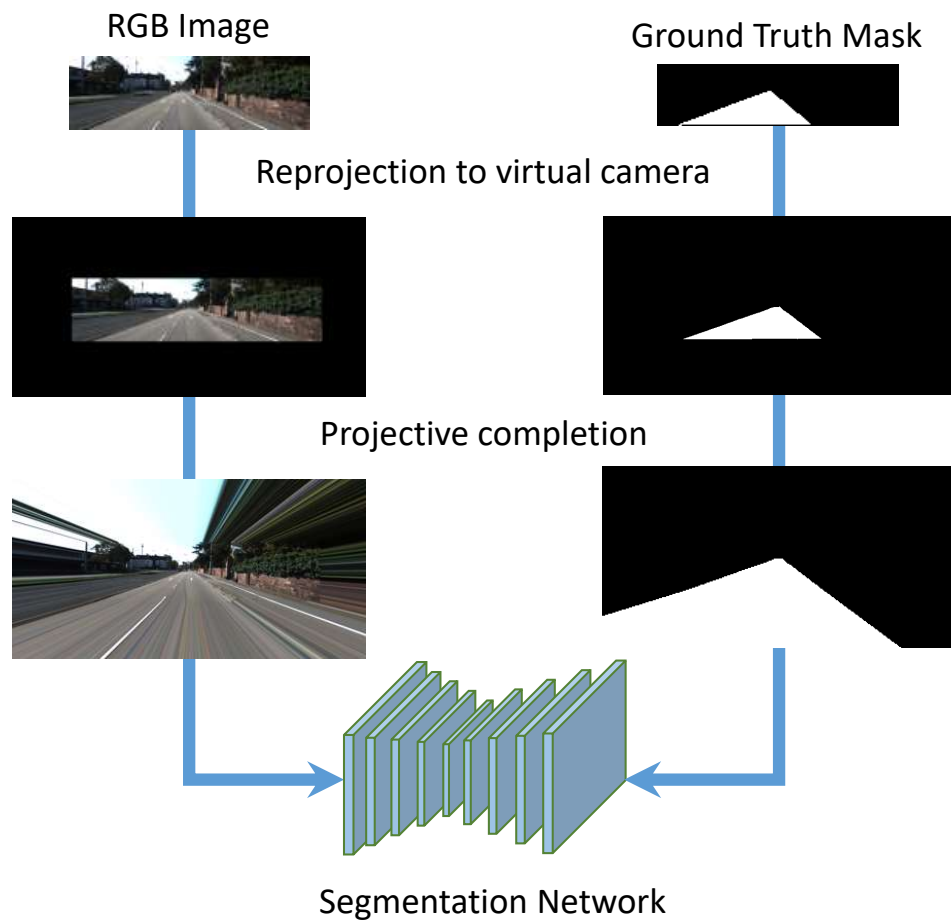


Figure 45: Overview of the proposed method.

5.5.1 Reprojection to Virtual Camera

To address geometric domain shift between and within datasets, we define a virtual standard camera with normative camera parameters, and approximate the mapping from real to virtual camera planes as a homography. This is a reasonable approximation for several reasons. First, the mapping is exact for planar surfaces and road surfaces are typically approximately planar. Second, much of the variation in the extrinsic camera parameters is in the camera rotation, particularly tilt and pan, and for pure rotations, the mapping from image to image is homographic for general scenes, not just planes. Third, distortions will occur primarily for objects with significant vertical extent, and these are of lesser importance for road segmentation.

Projection of Physical Camera to Ground Plane. We employ a world coordinate frame aligned with and lying on the road, directly beneath the camera (Fig. 46). We ignore horizontal translations of the camera and only model translations in the vertical (Y) direction. This allows us to fully specify the pose of each camera relative to a common world frame.

The general projection equation is

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = K[R|t] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad (56)$$

where (X, Y, Z) is a point in 3D space, (x, y) are the image pixel coordinates of the projection, K is the intrinsic matrix, R and t are the rotation and translation of the world frame relative to the

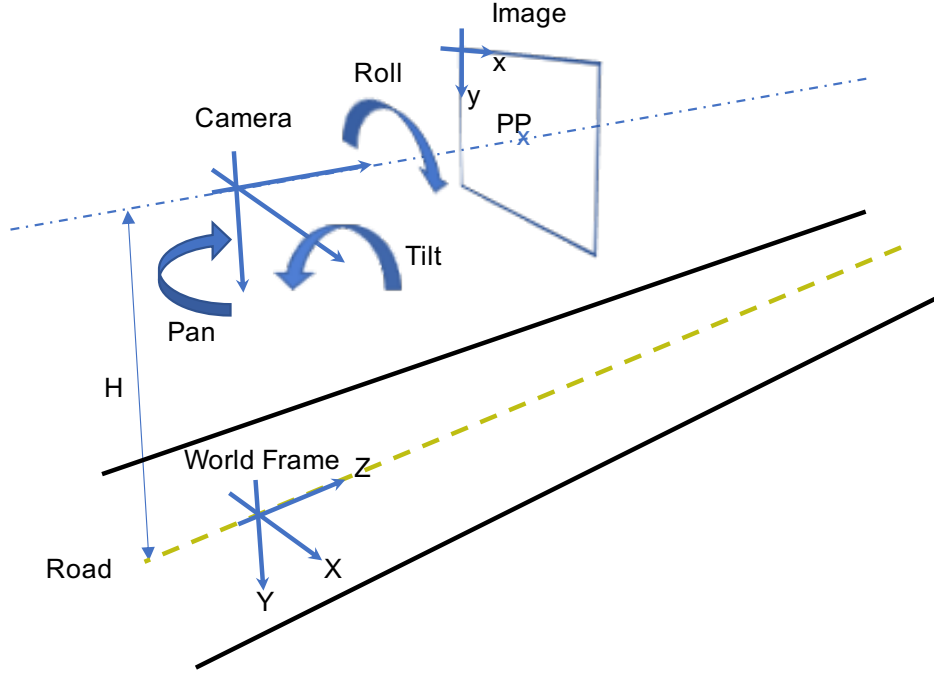


Figure 46: Camera geometry. PP denotes the principal point of the camera.

camera, respectively, and λ is a scaling parameter. Since in our case $Y = 0$ on the ground plane, the projection of a 3D point on the road to the physical camera image coordinates (x_p, y_p) can be modeled as

$$\lambda \begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix} = K_p [R_p^1 | R_p^3 | t_p] \begin{bmatrix} X \\ Z \\ 1 \end{bmatrix} \quad (57)$$

$$\rightarrow \begin{bmatrix} X \\ Z \\ 1 \end{bmatrix} = \lambda (K_p [R_p^1 | R_p^3 | t_p])^{-1} \begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix} \quad (58)$$

where K_p is the intrinsic matrix of the physical camera, R_p^1 and R_p^3 are the first and third columns

of its rotation matrix, and t_p is the translation of the world frame relative to the camera. Note that four of the five physical cameras are, on average, aligned with the world frame, so that R_p is the identity matrix and $t_p = [0, H_p, 0]^\top$, where H_p is the height of the camera above the road. The Cityscapes camera, however, is on average tilted slightly down, so that the third column R_p^3 of the rotation matrix becomes $[0, -\sin(\beta), \cos(\beta)]^\top$ and $t_p = H_p [0, \cos(\beta), \sin(\beta)]^\top$, where $\beta = 2.4$ deg is the average tilt angle of the camera.

Given the intrinsic and average extrinsic parameters of each physical camera, Equation 58 allows each pixel of the average image to be mapped to ground plane coordinates (X, Z) . Fig. 47 shows a mapping of the bottom quarter of each camera FOV to the groundplane: Note the significant differences in the ‘footprints’ of each camera.

Defining the Virtual Camera. We align the virtual camera with our world frame (i.e., 0 pan, tilt and roll, Fig. 46) and set its height to the mean height of the five cameras considered here: 1.38m.

We assume equal x and y focal lengths for our virtual camera, and set these to the maximum focal length (in pixels) over our datasets to preserve resolution. (Larger virtual focal lengths could be chosen to accommodate future cameras.) We selected the horizontal and vertical FOVs of the virtual camera to ensure that at least the lower half of all physical camera images reprojected to lie within the virtual camera image, taking into account expected variations in pan (± 20 deg) and tilt (± 15 deg). This resulted in horizontal and vertical FOVs of 101.1 and 63.5 deg, respectively. Combined with the focal length, this determined the resolution of the virtual camera ($5,500 \times 2,800$ pixels). We employ a central principal point.

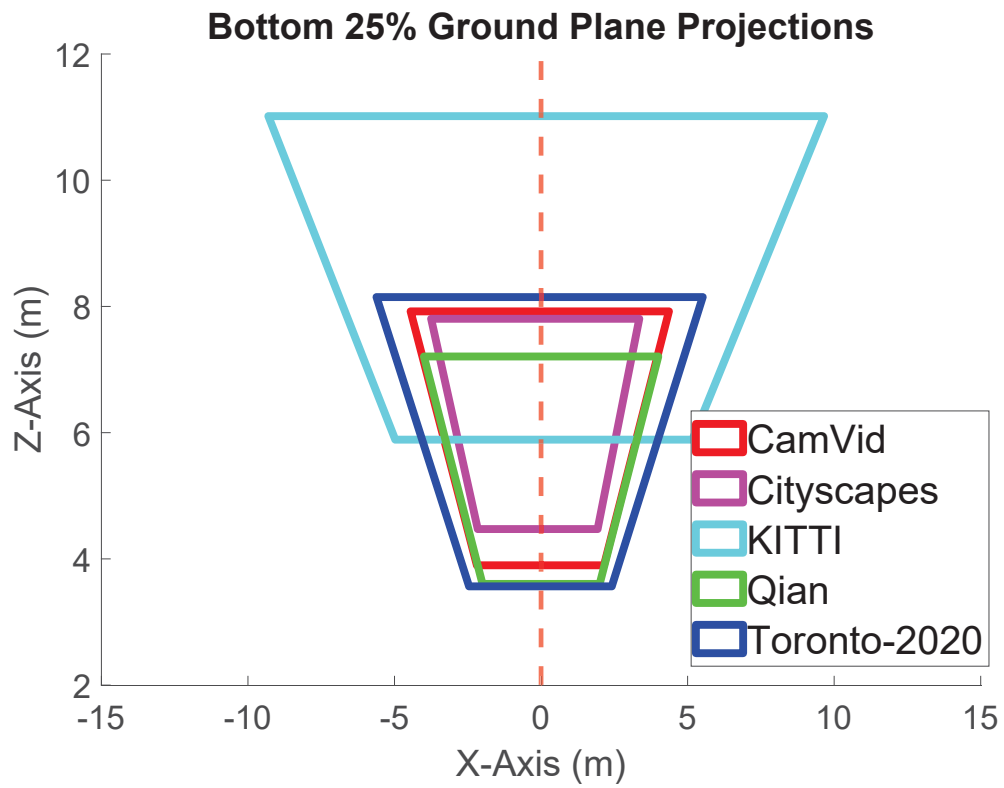


Figure 47: Ground plane back-projection of the lower quarter of the mean FOV for each camera.

Projection of Ground Plane to Virtual Camera. Since the virtual camera is aligned with the world frame, the projection equation (56) simplifies to

$$\lambda \begin{bmatrix} x_v \\ y_v \\ 1 \end{bmatrix} = K_v [I^{13}|t_v] \begin{bmatrix} X \\ Z \\ 1 \end{bmatrix} \quad (59)$$

where K_v is the intrinsic matrix of the virtual camera, I^{13} is the 3×2 matrix formed from the first and third columns of the 3×3 identity matrix, $t_v = [0, H_v, 0]^\top$ is the translation of the virtual camera, and H_v is its height above the road.

Note that the first homography (Eqn. 58) maps image points above the principal point to ground plane points behind the camera. However, since the parameters of the physical and virtual cameras will be relatively similar, by and large the second homography (Eqn. 59) will correctly map these points to the upper portion of the virtual image plane.

Combining Equations 58 and 59 yields the direct homography relating the physical camera to the virtual camera:

$$\lambda \begin{bmatrix} x_v \\ y_v \\ 1 \end{bmatrix} = K_v [I^{13}|t_v] (K_p [R_p^1|R_p^3|t_p])^{-1} \begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix} \quad (60)$$

Fig. 48 shows the reprojection of the average FOVs of each of the five cameras modeled here to the virtual camera plane. Note that the frame-to-frame FOVs will shift as the exact pose of each physical cameras varies. We turn to this issue now.

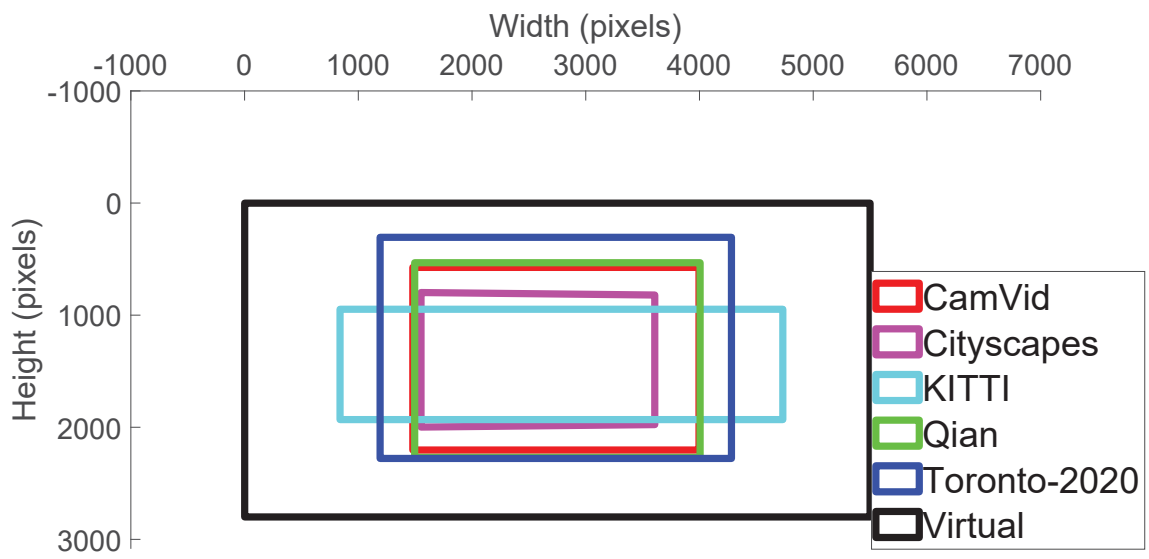


Figure 48: Projection of physical camera FOVs to the virtual camera plane.

5.5.2 Within-Camera Shift

Reprojection to the virtual camera serves to greatly reduce geometric shift between datasets. However, there can also be significant shifts in the extrinsic camera parameters *within* datasets. For the CamVid, Cityscapes and KITTI datasets, these within-camera shifts are primarily in pan, due to the yaw of the car, but other shifts due to vibration etc. also occur. For the Qian and Toronto-2020 datasets there are more profound shifts because the dashcams employed in these datasets were repositioned numerous times.

Here we focus specifically on within-camera shifts in pan and tilt, approximating the roll as 0 and neglecting translational shifts. Taking the world frame as our global frame-of-reference, we decompose the rotation matrix R_p into a sequence of two rotations, first a tilt of β about the X-axis and then a pan of α about the Y-axis. This results in:

$$R_p^1 = \begin{bmatrix} \cos \alpha \\ 0 \\ -\sin \alpha \end{bmatrix}, \quad R_p^3 = \begin{bmatrix} \sin \alpha \cos \beta \\ -\sin \beta \\ \cos \alpha \cos \beta \end{bmatrix} \quad (61)$$

R_p^3 represents the direction of the road vanishing point (x_{vp}, y_{vp}) in camera coordinates, and so can easily be expressed in terms of this vanishing point:

$$R_p^3 = \lambda_3 \begin{bmatrix} (x_{vp} - x_0) / f_x \\ (y_{vp} - y_0) / f_y \\ 1 \end{bmatrix} \quad (62)$$

where λ_3 is a normalizing constant. Finally, Eqns 61 and 62 can be combined to yield $\tan \alpha =$

$(x_{vp} - x_0) / f_x$, which leads to

$$R_p^1 = \lambda_1 \begin{bmatrix} 1 \\ 0 \\ -(x_{vp} - x_0) / f_x \end{bmatrix} \quad (63)$$

where λ_1 is a normalizing constant.

These corrections to the reprojection equation will serve to better align the source images for training. At inference, we employ the line-based (MCMLSD [4]) vanishing point estimation algorithm from [20] to correct for within-camera shift within the target dataset. Vanishing point estimates generating pan shifts of more than 20 deg or tilt shifts of more than 15 deg are rejected as unreliable and replaced by the mean of the estimated vanishing point locations judged to be valid.

5.5.3 Projective Completion

Fig. 48 highlights a complication with our virtual camera approach. Since none of the physical cameras span the entire virtual FOV, when training in the virtual camera domain, each (input, ground truth) pair will include undefined regions, which means that some network activations and back-propagated gradients are undefined. Empirically we find that filling these gaps with nominal values (e.g., zeros) or using standard image boundary handling methods (e.g., pixel replication) leads to huge errors in segmentation. To solve this problem, we introduce a novel projective completion method that exploits the typical approximate 3D symmetry in the road surface to generate more plausible image completions to support accurate learning.

Fig. 49 illustrates the approach. Recall that for each source image the vanishing point of the road has been identified. We establish a polar coordinate system centred on the vanishing point and identify the angle of each source boundary pixel. We then assign RGB and ground truth

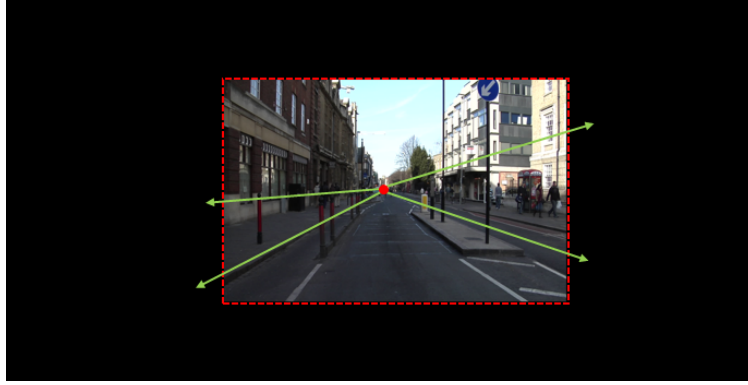
mask values to all pixels outside the source image based on an angular linear interpolation of the RGB and ground truth mask values for these boundary pixels. We find that results of this simple completion method are surprisingly reasonable for a large majority of scenes. (See supplementary file for more examples.) At inference, the estimated vanishing point [20] is used to complete the target image.

5.5.4 Supervised Training

Once all source images have been reprojected and projectively completed in the virtual camera plane, the network can be trained. To compare fairly with prior methods we employ a VGG16 [73] hourglass segmentation architecture for all methods, including VCSeg.

5.5.5 Geometry-Driven Adaptation

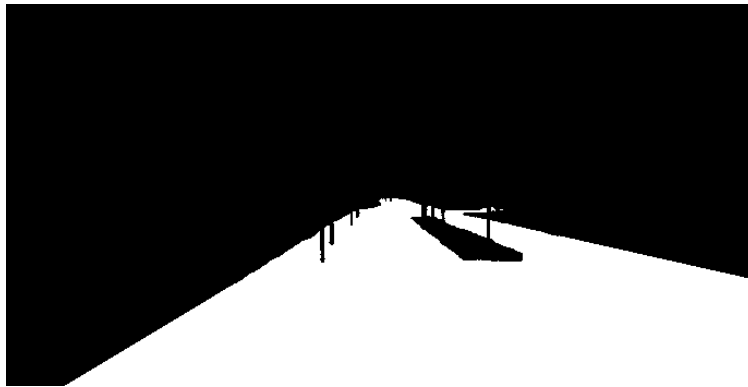
We also evaluate the Geometry-driven adaptation method introduced by GeoSeg [20], but in the virtual camera domain. Briefly, this method learns a prior for the road segmentation from the source images, anchored on the ground-truth vanishing points. In GeoSeg, this prior is used as a weak source of ground truth in the target image domain to fine-tune the segmentation network. Here we do the same, but derive the prior in the virtual camera domain by reprojecting the ground-truth masks from each source dataset. As in GeoSeg, we define a ground-truth tri-mask based on the prior: Letting $p_g(x, y)$ represent the prior geometric probability that virtual camera pixel (x, y) projects from the road, we set the ground truth to road where $p_g(x, y) > 0.75$, to background where $p_g(x, y) < 0.25$ and to unknown for all other pixels. This surrogate ground truth is then used to fine-tune the network.



(a)



(b)



(c)

Figure 49: Projective completion. (a) Reprojected image from the CamVid training partition. Red point indicates ground truth annotated road vanishing point, red dashed lines the source boundary pixels and green lines the completion directions. (b) Projectively completed source image. (c) Projectively completed ground truth mask.

5.6 Experimental Details

As in GeoSeg [20], we score segmentations by intersection over union (IoU), ignoring ego-vehicle regions. We compare against three alternative methods: SegNet [8], the GAN-based AdaptSegNet, [83] and GeoSeg [20].

As in [20], we used a cross-entropy loss and trained for 600 epochs. However, we noticed that VCSeg benefits from a smaller batch size and faster learning rate - Table 7. To control for this difference, we list the SegNet and GeoSeg performance as reported in [20] and also performance when trained with the parameters used to train VCSeg.

Parameter	Old [20]	New
Batch size	16	8
Learning rate	0.001	0.005
Rate decay	0.0001	0.0005

Table 7: Learning parameters used by GeoSeg [20] and our new method, VCSeg.

Experiments were conducted with a Pentium 3.4 GHz i7 processor, 64 GB RAM, and an Nvidia GTX 1080 Ti GPU. Line detection, vanishing point detection and projective completion were performed in MATLAB, while networks were implemented in PyTorch. Table 8 lists the run time at inference for each component of the method. Line detection currently dominates the run time: Future work will focus on speeding line detection and further optimizations to make the method real time.

Stage	Runtime per image
Line detection (MCMLSD) [4]	6200 msec
Vanishing point detection [20]	25 msec
Projection to virtual camera	182 msec
Projective completion	85 msec
Network inference	43 msec
Back-projection to physical camera	43 msec

Table 8: Average runtime for inference.

5.7 Results

5.7.1 Comparison with Prior Adaptation Methods

Table 9 compares quantitative results for VCSeg and prior adaptation methods on a common SegNet architecture. We find that the effect of the training parameters (Table 7) is variable across datasets, but overall, the new parameters adopted for VCSeg improve the performance of SegNet but have negligible effect on the performance of GeoSeg.

We find that our between-camera virtual camera reprojection method leads to a dramatic 12.2% improvement in segmentation performance over SegNet (average of 62.3% IoU vs 50.1% IoU) - note that we see an improvement for every target dataset and the only difference between these methods is that in VCSeg training is performed in the virtual camera domain. Even without adaptive fine-tuning, our between-camera VCSeg method outperforms on average the more complex GeoSeg and AdaptSegNet methods, both of which require costly (though unsupervised) fine-tuning on the target dataset.

Target Dataset	SegNet [8]		GeoSeg [20]		AdaptSegNet [83]	VCSeg	VCSeg
	Old parms	New parms	Old parms	New parms		Between-Cam	+Within-Cam
CamVid	39.2	57.5	61.0	60.4	65.8	73.6	73.9
Cityscapes	60.8	65.7	74.7	75.3	71.1	73.5	76.9
KITTI	39.1	52.9	46.9	47.0	56.7	71.5	80.6
Qian	31.2	29.6	58.3	56.2	52.4	43.7	56.8
Toronto-2020	44.7	34.2	65.6	65.7	60.7	49.0	67.5
Mean	43.0	50.1	61.3	60.9	61.3	62.3	71.1

Table 9: Quantitative comparison with prior adaptation methods, using a common SegNet architecture.

Numbers are % IoU. See Table 7 for old and new training parameters.

We observe a second dramatic (8.8%) improvement in performance when we also correct for within-camera shifts in tilt and pan angles between images (VCSeg + Within-Cam): (average of 71.1% IoU vs 62.3% IoU when accounting only for between-camera shifts). Again, we see an improvement for every target dataset, but the improvement is particularly dramatic for the dashcam datasets (Qian and Toronto-2020), where the camera was repositioned frequently.

5.7.2 Incorporating Appearance Adaptation

We note that our full VCSeg method performs close to but does not beat the GeoSeg method for the Qian dataset, suggesting that, for this dataset at least, GeoSeg is capturing some domain shift that is not accounted for by shift in the camera parameters. To further explore this, we assessed the performance of GeoSeg adaptation, but in the virtual camera domain (Section 5.5.5), accounting for both between- and within-camera shifts. Table 10 shows the results. We find that adaptation improves results for Qian and Toronto-2020 datasets, demonstrating that for these datasets, the

GeoSeg and virtual camera approaches can be fruitfully combined. However, for the other three datasets we find that adaptation hurts performance.

We believe these mixed results are due to two factors. First, since performance for VCSeg is already relatively high for CamVid, Cityscapes and KITTI datasets, the geometric prior derived from source ground truth (fourth column in Table 10) is likely too weak to provide sufficiently reliable supervision to improve results further. Second, while these datasets contain relatively modest appearance variations, Qian and Toronto-2020 datasets contain massive variations in weather, including snow and ice; GeoSeg is able to handle these appearance shifts that are not handled by VCSeg. Our recommendation is thus to apply the two methods together in cases when the target domain includes significant non-geometric shifts, including weather and illumination.

Target Dataset	VCSeg	VCSeg +Geo-Adaptation [20]	Geo-Prior [20]
CamVid	73.9	63.7	59.0
Cityscapes	76.9	76.2	73.6
KITTI	80.6	70.1	62.0
Qian	56.8	60.1	59.3
Toronto-2020	67.5	69.2	65.5
Mean	71.1	67.9	63.9

Table 10: Results of Geometry-driven adaptation in the virtual camera domain. Numbers are % IoU. Geo-Prior indicates segmentation based upon the geometric prior used by GeoSeg [20], mapped to the virtual camera.

5.7.3 Comparison with Other Architectures

To focus on the issue of geometric adaptation our comparisons thus far have been based on a common SegNet architecture. Here we compare against two more recent semantic segmentation approaches (Table 11): STDC-Seg [27] and GALDNet [49], both pretrained on the Cityscapes dataset. (Since our Cityscapes test partition is drawn from this dataset we do not compare on Cityscapes.)

Target Dataset	VCSeg	STDC	STDC2	GALDNet	GALDNet
	(Ours)	Seg50	Seg75	ResNet50	ResNet101
		[27]	[27]	[49]	[49]
CamVid	73.9	74.2	76.4	79.3	78.7
KITTI	80.6	62.4	61.9	70.8	65.8
Qian	56.8	51.8	65.5	56.7	68.0
Toronto-2020	67.5	33.6	36.9	31.8	37.2
Mean	69.7	55.6	60.2	56.7	62.4

Table 11: Quantitative comparison with recent semantic segmentation architectures trained on Cityscapes.

Numbers are % IoU.

We find that despite being based upon a much simpler architecture, our VCSeg approach performs substantially better overall than these more recent Cityscapes-trained architectures, beating the nearest competitor by 7.3% on average. KITTI and Toronto-2020 datasets have much larger FOVs than Cityscapes, which creates a geometric mismatch for these non-adaptive Cityscapes-trained architectures. Correctly aligning the data in the virtual camera domain allows VCSeg

to generalize much better, leading to much higher performance on KITTI and Toronto-2020 test datasets. CamVid and Qian datasets, on the other hand, have FOVs that are more similar to Cityscapes and we see here that GALDNet performs somewhat better. This suggests that combining VCSeg with this more powerful architecture could lead to even further improvements.

5.7.4 Evaluating Projective Completion

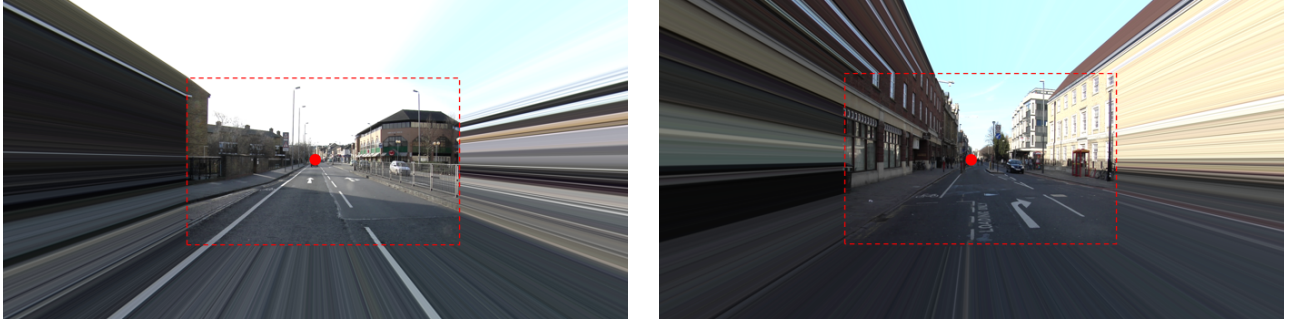
VCSeg depends upon a novel projective completion method to fill-in missing image and ground-truth information outside the projection of the physical image. Table 12 quantifies the importance of this completion step. Substituting zero-padding for projective completion leads to a consistent drop in performance across all five test datasets, averaging 13.4%.

Target Dataset	Zero Padding	Projective Completion (Est. VP)	Projective Completion (GT VP)
CamVid	64.0	73.9	75.1
Cityscapes	69.7	76.9	77.6
KITTI	51.1	80.6	81.5
Qian	53.4	56.8	56.9
Toronto-2020	50.4	67.5	69.7
Mean	57.7	71.1	72.2

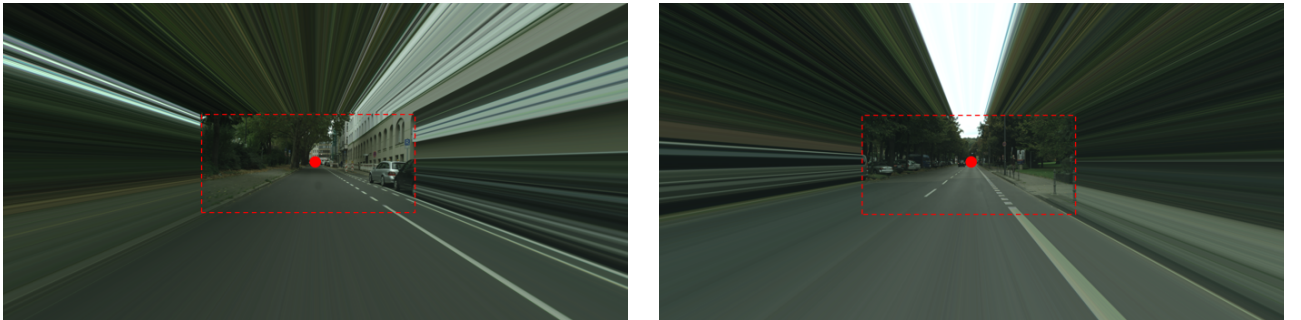
Table 12: Performance of VCSeg with and without projective completion. The last column shows for reference the performance when ground truth vanishing points are used at inference. This provides an upper bound on performance gains achievable through improvements to vanishing point detection. Numbers are % IoU.

Correction of within-camera variation and projective completion both depend on accurate estimation of vanishing points. To understand the impact of vanishing point errors on performance, the last column lists VCSeg performance when the *ground truth* vanishing points are used at inference. We do see a consistent, if relatively modest, improvement across all five target datasets, averaging 1.1%. The greatest improvement is seen for Toronto-2020, which contains relatively large variations in camera parameters.

Fig. 50 & Fig. 51 show more examples of the projectively completed images.

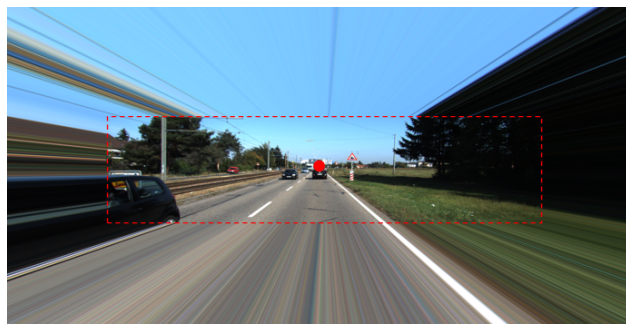
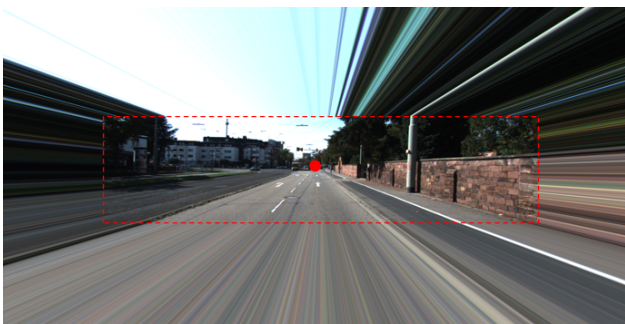


CamVid

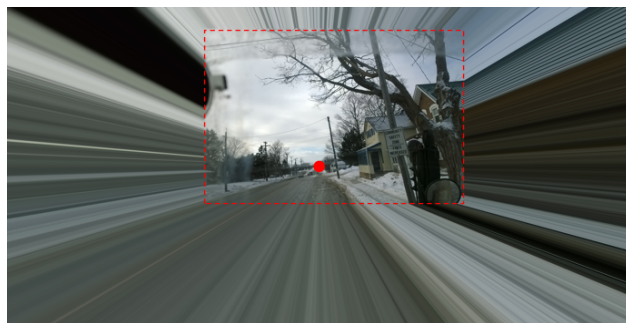
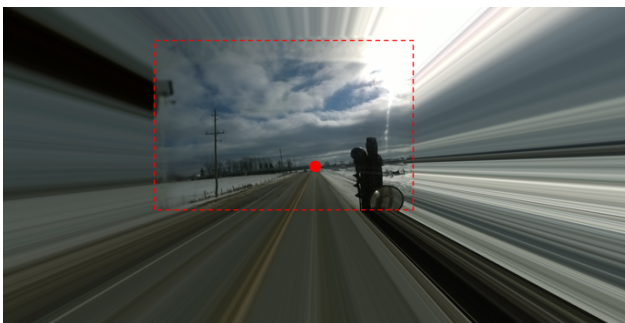


Cityscapes

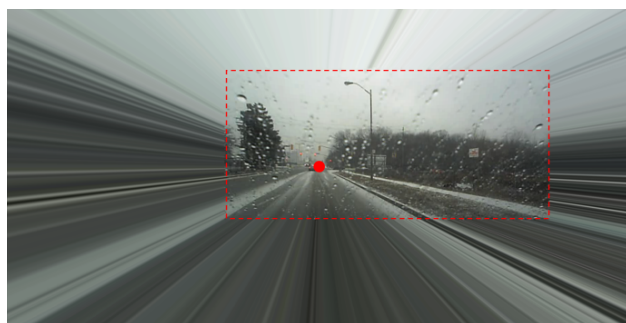
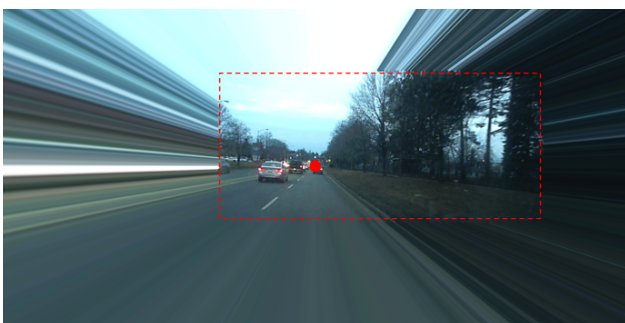
Figure 50: Projective completion examples 1. Red dot indicates ground truth vanishing point and red lines indicate original image boundaries.



KITTI



Qian



Toronto 2020

Figure 51: Projective completion examples 2. Red dot indicates ground truth vanishing point and red lines indicate original image boundaries.

To demonstrate the impact on performance, Fig. 52 shows examples of VCSeg segmentations on Cityscapes images computed with and without projective completion. We see that segmentations without projective completion are consistently more fragmented and less accurate.

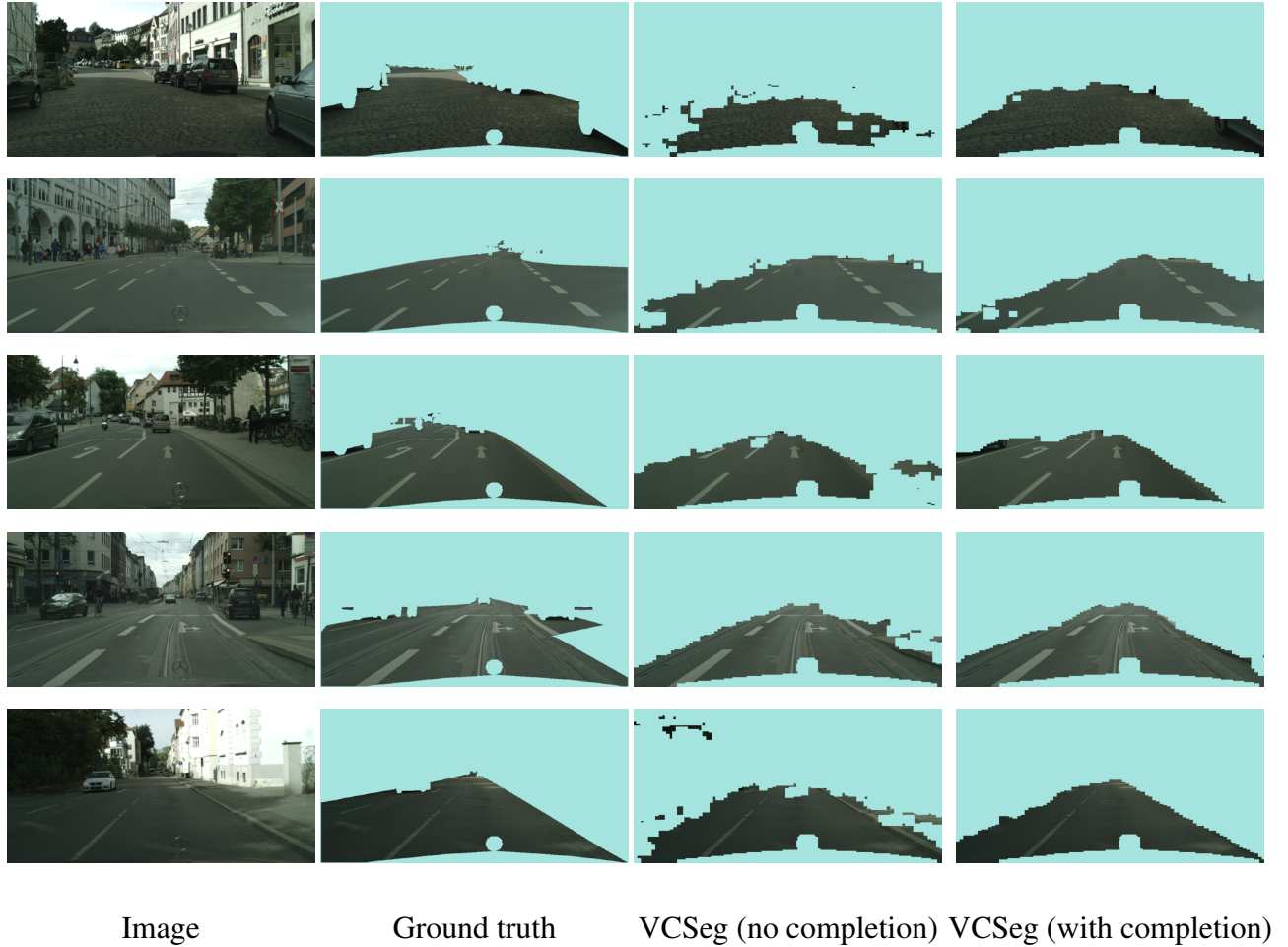


Figure 52: Demonstration of the importance of projective completion on images from the Cityscapes dataset.

5.7.5 Visualizing Shift in Geometric Camera Parameters

In the standard approach to road segmentation (e.g., SegNet, GeoSeg, AdaptSegNet), images are fed into a fixed network without regard to the intrinsic or extrinsic parameters of the camera. VCSeg attempts to improve on this by correcting for geometric shift prior to input. To visualize the effect of this intervention on the input, we can average the imagery both within and between

datasets, with and without the VCSeg correction (Fig. 53). Note that the scene structure is much more crisply apparent with geometric correction than without, particularly for the Qian dataset, where variations in camera parameters are most extreme.

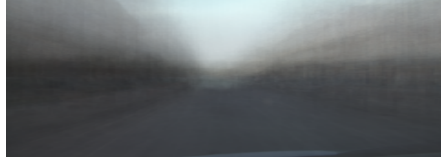
5.7.6 Qualitative Results

To get a sense of some advantages of our virtual camera approach, Fig. 54 compares results of all algorithms for the image from each dataset that generate the *highest* IoU for our full VCSeg method. It is encouraging to see VCSeg performing well even for the snowy road from the Toronto 2020 dataset. Comparing performance of VCSeg (Between-Cam) with VCSeg (Within-Cam), it is apparent that modeling within-camera shifts in pan and tilt is crucial to locking on to the left and right boundaries of the road and avoiding fragmentation. Note that while GeoSeg and AdaptSegNet also avoid much of the fragmentation seen in the SegNet results, these methods are generally not as accurate at capturing the left and right road boundaries.

Fig. 55 shows failure modes, i.e., results for the images that generate the *lowest* IoU for our full VCSeg method. We note that for four of the five cases (CamVid, KITTI, Qian, Toronto-2020), our version of VCSeg that only models between-camera shifts performs relatively well, suggesting that these failures may arise from errors in vanishing point estimation. Improvements to this component of the pipeline would likely address some of these failure modes. Fig. 56 shows the median performance results of each dataset.

5.8 Conclusions and Future Research

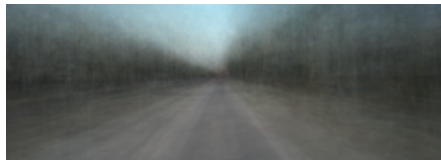
Shift in camera parameters is a major reason road segmentation systems fail to generalize to new domains. Our results show that these geometric camera parameter shifts can be addressed by



CamVid



Cityscapes



KITTI



Qian



Toronto-2020

Averaged in Source Domain

Averaged in Virtual Camera Domain

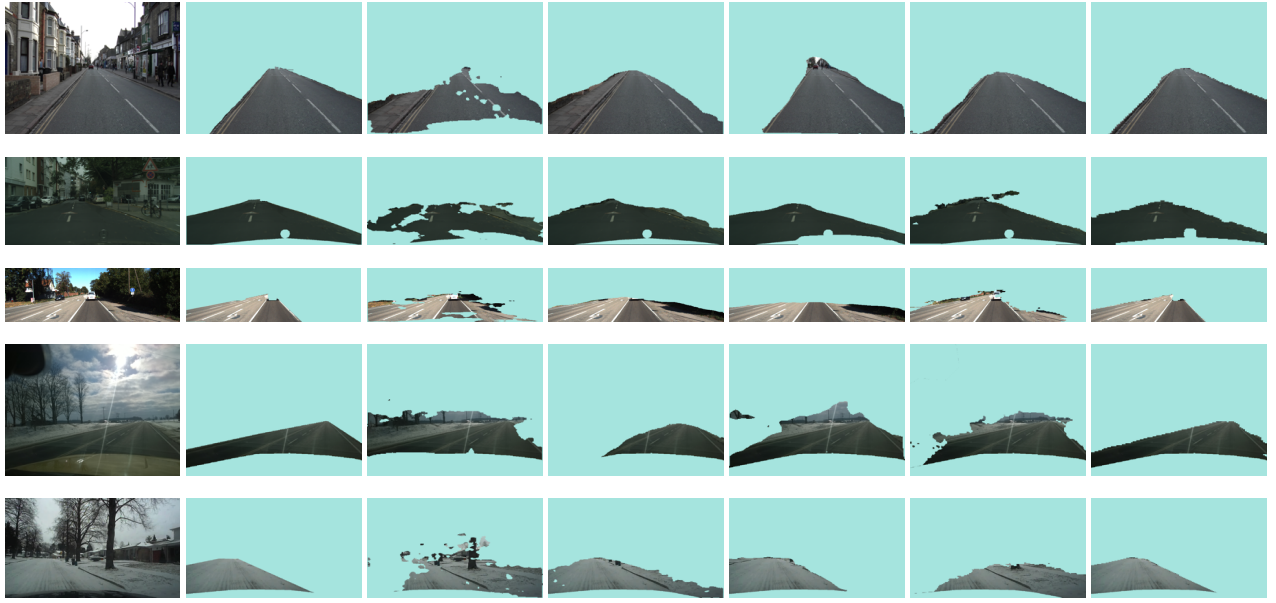


Figure 54: **Successes:** Image from each dataset for which our full VCseg method obtains the highest IoU.

Left to right: (1) input image, (2) ground truth, (3) SegNet, (4) GeoSeg, (5) adapt-SegNet, (6) VCseg Between Cam, (7) VCseg+Within Cam. **Top to Bottom:** CamVid, Cityscapes, KITTI, Qian, Toronto-2020.

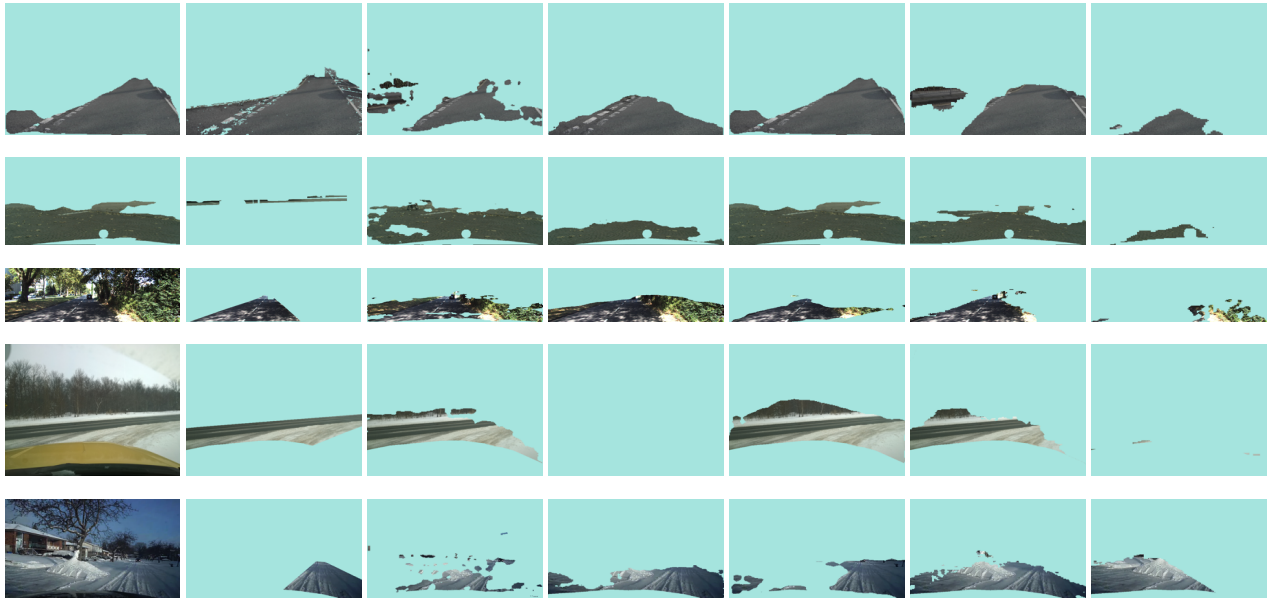


Figure 55: **Failures:** Image from each dataset for which our full VCseg method obtains the lowest IoU. **Left to right:** (1) input image, (2) ground truth, (3) SegNet, (4) GeoSeg, (5) adapt-SegNet, (6) VCseg Between Cam, (7) VCseg+Within Cam. **Top to Bottom:** CamVid, Cityscapes, KITTI, Qian, Toronto-2020.

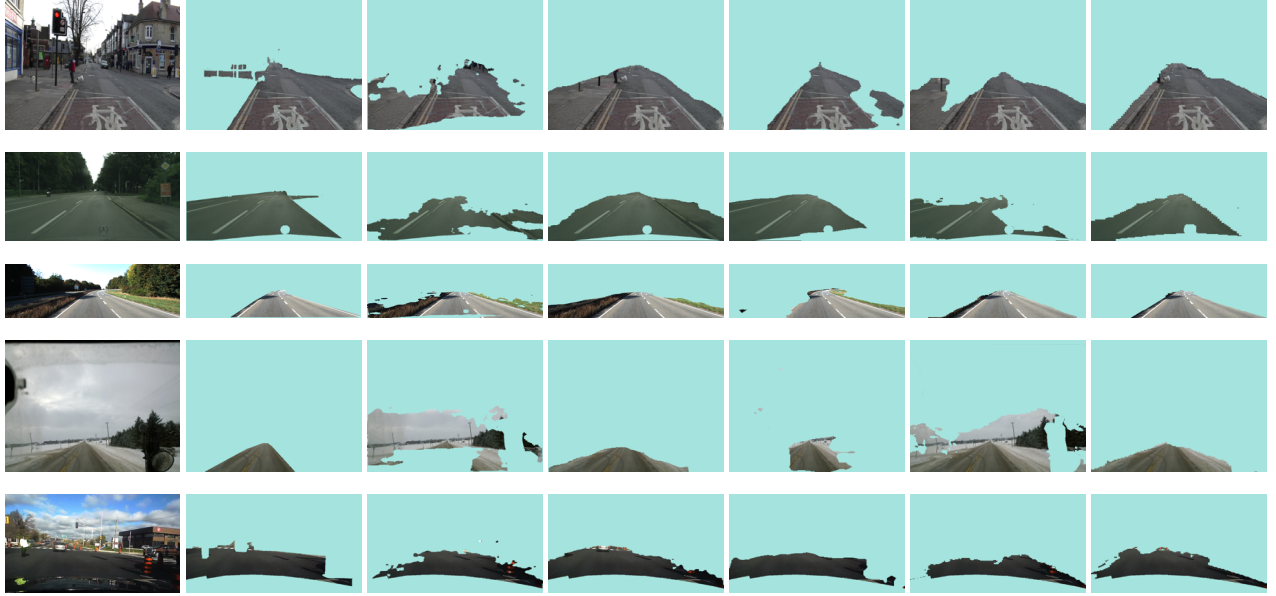


Figure 56: **Medians:** Image from each dataset for which our full VCSeg method obtains the median IoU. **Left to right:** (1) input image, (2) ground truth, (3) SegNet, (4) GeoSeg, (5) adapt-SegNet, (6) VCSeg Between Cam, (7) VCSeg+Within Cam. **Top to Bottom:** CamVid, Cityscapes, KITTI, Qian, Toronto-2020.

reprojecting images to a common virtual camera plane and using a projective completion method to fill resulting gaps in the imagery prior to training. Correcting for between-camera shifts leads to a 12.2% improvement in IoU performance, while correcting for within-camera shifts leads to an additional 8.8% improvement, for a total of 21% (50.1% to 71.1%). Appearance adaptation [20] can further boost performance for datasets with substantial weather variations.

Future work will examine more reliable methods for vanishing point estimation, to handle curved roads, for example, and whether modeling of camera roll and lateral translation can further improve performance. Ultimately, better methods for fusing these geometric adaptation methods with state-of-the-art appearance adaptation methods will lead to road segmentation systems that can be deployed reliably on various platforms and in diverse environments.

6 Conclusion and Future Work

6.1 Conclusion

In conclusion, my Ph.D. research focused on advancing road scene understanding by leveraging geometric information across three distinct projects.

In my first project, a supervised road segmentation study [18], I explored the fusion of appearance and geometric cues to develop a robust road segmentation model capable of adapting to varying weather conditions. This work demonstrated the value of integrating multiple modalities to improve segmentation accuracy and resilience.

Building on this foundation, I expanded my research to address the more challenging problem of unsupervised domain adaptation (UDA) for road segmentation, where domain shifts between source and target datasets significantly complicate the task. In my second project [20], we proposed a geometry-guided adaptation method that leveraged surrogate ground truth masks and target domain RGB images to fine-tune a source-trained segmentation network. This approach highlighted the potential of geometric cues in facilitating effective model transfer.

Finally, in my third project [17], we introduced a virtual camera-based domain adaptation method that uses known camera parameters to better align imagery for both training and inference. This method addressed domain shifts by decomposing them into two components: (1) between-camera shift and (2) within-camera shift. By transferring images from different datasets to a unified virtual camera space, this approach significantly enhanced segmentation performance. Evaluation results demonstrated its superiority over other state-of-the-art methods.

Together, these projects contribute both scientific and engineering advances: scientifically,

these projects illustrate the power of geometry-driven techniques in advancing road scene understanding and overcoming domain adaptation challenges; and engineering-wise, by producing low-cost and reliable frameworks that are suitable for real-world autonomous driving systems. Overall, this thesis demonstrates that the use of geometric cues provides a promising path toward more generalizable and trustworthy vision systems for understanding the road scene.

More specific contributions are listed below.

1. Identifying potential contributing factors to correct segmentation and failure modes including appearance and geometry and particularly shifts in geometry and appearance between contexts.
2. Alternative explainable methods that can potentially be correlated against modern high-capacity networks to understand what principles the latter are implicitly using.
3. Low-cost methods that can be used as independent checks or verification for performance-critical tasks.

6.2 Future Work

Building upon the findings of the three projects, several promising directions can be pursued to further advance the robustness and applicability of road segmentation:

1. **Deployment in Real-World Autonomous Driving Systems:** A natural next step for all three projects [18, 20, 17] is to transition from experimental validation to real-world deployment in autonomous driving platforms. This involves integrating the segmentation frameworks onboard systems for real-world road segmentation tasks. Beyond quantitative per-

formance, this direction will focus on latency, resource efficiency, and robustness in safety-critical scenarios, ensuring that these systems meet the reliability demands of real-world self-driving applications.

2. **End-to-End System Automation and Integration:** While the current pipelines require modular processing and manual intervention, a key future goal is to automate and streamline the entire framework for end-to-end usability. In Project 2 [18], this involves integrating a learnable surrogate label generation module that dynamically learns the geometric prior from the source domain. For Project 3 [17], automation of the virtual camera space creation should be embedded as a pre-processing module that automatically estimates and applies the appropriate geometric mappings based on estimated camera parameters. These enhancements will increase the adaptability and scalability of both systems, making them more suitable for large-scale deployment across varying domains.
3. **Improved Vanishing Point Estimation for Challenging Scenarios:** Accurate vanishing point (VP) detection is a foundational component across all three projects [18, 20, 17]. In edge cases such as severely curved roads, blurred scenes from adverse weather, or poor lighting, estimation may become unreliable. Future work will focus on developing more robust vanishing point detection algorithms for those scenarios.
4. **Multi-Modal Integration for Enhancing the Road Segmentation:** To further boost segmentation accuracy and robustness, future work will explore integrating all three image-based projects [18, 20, 17] with additional sensor modalities such as LiDAR, radar, and HD maps. By leveraging complementary spatial and depth information, this multi-modal fusion can mitigate the limitations of monocular vision and improve performance in adverse

conditions.

Bibliography

- [1] R. Adams and L. Bischof. Seeded region growing. *IEEE Transactions on pattern analysis and machine intelligence*, 16(6):641–647, 1994.
- [2] E. Almazan, R. Tal, Y. Qian, and J. H. Elder. MCMLSD homepage. <http://www.elderlab.yorku.ca/resources/>.
- [3] E. J. Almazan, Y. Qian, and J. H. Elder. Road segmentation for classification of road weather conditions. In *European Conference on Computer Vision*, pages 96–108. Springer, 2016.
- [4] E. J. Almazan, R. Tal, Y. Qian, and J. H. Elder. MCMLSD: A dynamic programming approach to line segment detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2031–2039, 2017.
- [5] T. Almeida, B. Lourenço, and V. Santos. Road detection based on simultaneous deep learning approaches. *Robotics and Autonomous Systems*, 133:103605, 2020.
- [6] J. M. Alvarez, T. Gevers, and A. M. Lopez. 3D scene priors for road detection. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 57–64. IEEE, 2010.
- [7] N. Apostoloff and A. Zelinsky. Robust vision based lane tracking using multiple cues and particle filtering. In *IEEE IV2003 Intelligent Vehicles Symposium. Proceedings (Cat. No. 03TH8683)*, pages 558–563. IEEE, 2003.
- [8] V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12):2481–2495, 2017.

- [9] M. R. Blas, M. Agrawal, A. Sundaresan, and K. Konolige. Fast color/texture segmentation for outdoor robots. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4078–4085. IEEE, 2008.
- [10] G. J. Brostow, J. Fauqueur, and R. Cipolla. Semantic object classes in video: A high-definition ground truth database. *Pattern Recognition Letters*, 30(2):88–97, 2009.
- [11] G. J. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla. Segmentation and recognition using structure from motion point clouds. In *European Conference on Computer Vision*, pages 44–57. Springer, 2008.
- [12] A. Cardace, P. Z. Ramirez, S. Salti, and L. Di Stefano. Shallow features guide unsupervised domain adaptation for semantic segmentation at class boundaries. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1160–1170, 2022.
- [13] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, 2017.
- [14] M. Chen, H. Xue, and D. Cai. Domain adaptation for semantic segmentation with maximum squares loss. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2090–2099, 2019.
- [15] S. Chen, X. Tan, B. Wang, and X. Hu. Reverse attention for salient object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 234–250, 2018.
- [16] Z. Chen and Z. Chen. Rbnet: A deep neural network for unified road and road boundary detection. In *International Conference on Neural Information Processing*, pages 677–687.

Springer, 2017.

- [17] G. Cheng and J. H. Elder. Vcseg: Virtual camera adaptation for road segmentation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 277–286, January 2022.
- [18] G. Cheng, Y. Qian, and J. H. Elder. Fusing geometry and appearance for road segmentation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops*, Oct 2017.
- [19] G. Cheng, Y. Wang, Y. Qian, and J. H. Elder. Geometry-guided road segmentation homepage. www.elderlab.yorku.ca/resources/geometry-guided-road-adaptation/.
- [20] G. Cheng, Y. Wang, Y. Qian, and J. H. Elder. Geometry-guided adaptation for road segmentation. In *2020 17th Conference on Computer and Robot Vision (CRV)*, pages 46–53. IEEE, 2020.
- [21] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3213–3223, 2016.
- [22] H. Dahlkamp, A. Kaehler, D. Stavens, S. Thrun, and G. R. Bradski. Self-supervised monocular road detection in desert terrain. In *Robotics: Science and Systems*, volume 38. Philadelphia, 2006.
- [23] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern*

- Recognition*, pages 248–255. Ieee, 2009.
- [24] P. Dollár and C. L. Zitnick. Structured forests for fast edge detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1841–1848, 2013.
 - [25] R. O. Duda, P. E. Hart, et al. *Pattern Classification and Scene Analysis*, volume 3. Wiley New York, 1973.
 - [26] D. Dwibedi, I. Misra, and M. Hebert. Cut, paste and learn: Surprisingly easy synthesis for instance detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1301–1310, 2017.
 - [27] M. Fan, S. Lai, J. Huang, X. Wei, Z. Chai, J. Luo, and X. Wei. Rethinking bisenet for real-time semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9716–9725, 2021.
 - [28] C. Fernandez-Maloigne and W. Bonnet. Texture and neural network for road segmentation. In *Proceedings of the Intelligent Vehicles’ 95. Symposium*, pages 344–349. IEEE, 1995.
 - [29] J. Fritsch, T. Kuehnl, and A. Geiger. A new performance measure and evaluation benchmark for road detection algorithms. In *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*, pages 1693–1700. IEEE, 2013.
 - [30] G. Ghiasi, Y. Cui, A. Srinivas, R. Qian, T.-Y. Lin, E. D. Cubuk, Q. V. Le, and B. Zoph. Simple copy-paste is a strong data augmentation method for instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2918–2928, 2021.
 - [31] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, pages 1321–1330. PMLR, 2017.

- [32] X. Han, J. Lu, C. Zhao, S. You, and H. Li. Semisupervised and weakly supervised road detection based on generative adversarial networks. *IEEE Signal Processing Letters*, 25(4):551–555, 2018.
- [33] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.
- [34] Y. Hou, Z. Ma, C. Liu, and C. C. Loy. Learning lightweight lane detection CNNs by self attention distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1013–1021, 2019.
- [35] D.-Y. Huang, T.-W. Lin, and W.-C. Hu. Automatic multilevel thresholding based on two-stage Otsu’s method with cluster determination by valley estimation. *International Journal of Innovative Computing, Information and Control*, 7(10):5631–5644, 2011.
- [36] V. Iglovikov and A. Shvets. Terausnet: U-net with VGG11 encoder pre-trained on imagenet for image segmentation. *arXiv preprint arXiv:1801.05746*, 2018.
- [37] S. J. Impey and J. A. Bangham. Image segmentation by area decomposition of HSV components. In *1996 8th European Signal Processing Conference (EUSIPCO 1996)*, pages 1–3. IEEE, 1996.
- [38] Y. Kanai. Image segmentation using intensity and color information. In *Visual Communications and Image Processing’98*, volume 3309, pages 709–720. International Society for Optics and Photonics, 1998.
- [39] H. Kong, J.-Y. Audibert, and J. Ponce. Vanishing point detection for road detection. In *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition*,

pages 96–103. IEEE, 2009.

- [40] H. Kong, J.-Y. Audibert, and J. Ponce. General road detection from a single image. *IEEE Transactions on Image Processing*, 19(8):2211–2220, 2010.
- [41] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, 25:1097–1105, 2012.
- [42] D. Kuan, G. Phipps, and A.-C. Hsueh. Autonomous robotic vehicle road following. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(5):648–658, 1988.
- [43] W. W. L. Lam, C. C.-C. Pang, and N. H. C. Yung. Highly accurate texture-based vehicle segmentation method. *Optical Engineering*, 43(3):591–603, 2004.
- [44] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- [45] D.-H. Lee et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on Challenges in Representation Learning, ICML*, volume 3, page 896, 2013.
- [46] J. Lee and C. D. Crane. Road following in an unstructured desert environment based on the em(expectation-maximization) algorithm. In *2006 SICE-ICASE International Joint Conference*, pages 2969–2974, 2006.
- [47] T. S. Lee. Image representation using 2d gabor wavelets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(10):959–971, 1996.
- [48] D. Levi, N. Garnett, E. Fetaya, and I. Herzlyia. Stixelnet: A deep convolutional network for obstacle detection and road segmentation. In *BMVC*, volume 1, page 4, 2015.

- [49] X. Li, L. Zhang, A. You, M. Yang, K. Yang, and Y. Tong. Global aggregation then local distribution in fully convolutional networks. In *BMVC*, 2019.
- [50] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, pages 740–755. Springer, 2014.
- [51] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.
- [52] R. d. A. Lotufo, B. T. Thomas, and E. L. Dagless. Road following algorithm using a panned plan-view transformation. In *Computer Vision—ECCV 90: First European Conference on Computer Vision Antibes, France, April 23–27, 1990 Proceedings 1*, pages 231–235. Springer, 1990.
- [53] Y. Lyu and X. Huang. Roadnet-v2: A 10 ms road segmentation using spatial sequence layer. *CoRR*, 2018.
- [54] J. B. McQueen. Some methods of classification and analysis of multivariate observations. In *Proc. of 5th Berkeley Symposium on Math. Stat. and Prob.*, pages 281–297, 1967.
- [55] C. C. T. Mendes, V. Frémont, and D. F. Wolf. Exploiting fully convolutional neural networks for fast road detection. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3174–3179. IEEE, 2016.
- [56] F. Milletari, N. Navab, and S.-A. Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 Fourth International Conference on 3D vision (3DV)*, pages 565–571. Ieee, 2016.

- [57] A. Mizushima and R. Lu. An image segmentation method for apple sorting and grading using support vector machine and Otsu’s method. *Computers and Electronics in Agriculture*, 94:29–37, 2013.
- [58] D. Neven, B. De Brabandere, S. Georgoulis, M. Proesmans, and L. Van Gool. Towards end-to-end lane detection: An instance segmentation approach. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 286–291. IEEE, 2018.
- [59] O. Oktay, J. Schlemper, L. L. Folgoc, M. Lee, M. Heinrich, K. Misawa, K. Mori, S. McDonagh, N. Y. Hammerla, B. Kainz, et al. Attention u-net: Learning where to look for the pancreas. *arXiv preprint arXiv:1804.03999*, 2018.
- [60] G. L. Oliveira, W. Burgard, and T. Brox. Efficient deep models for monocular road segmentation. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4885–4891. IEEE, 2016.
- [61] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, 1979.
- [62] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello. Enet: A deep neural network architecture for real-time semantic segmentation. *arXiv preprint arXiv:1606.02147*, 2016.
- [63] Y. Qian, E. J. Almazan, and J. H. Elder. Evaluating features and classifiers for road weather condition analysis. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 4403–4407. IEEE, 2016.
- [64] C. Rasmussen. Grouping dominant orientations for ill-structured road following. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 1, pages I–I. IEEE, 2004.

- [65] C. Rasmussen et al. Texture-based vanishing point voting for road shape estimation. In *BMVC*, pages 1–10. Citeseer, 2004.
- [66] S. R. Richter, V. Vineet, S. Roth, and V. Koltun. Playing for data: Ground truth from computer games. In *European Conference on Computer Vision*, pages 102–118. Springer, 2016.
- [67] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention—MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015.
- [68] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3234–3243, 2016.
- [69] C. Rotaru, T. Graf, and J. Zhang. Color image segmentation in HSI space for automotive applications. *Journal of Real-Time Image Processing*, 3(4):311–322, 2008.
- [70] D. W. Scott. *Multivariate Density Estimation: Theory, Practice, and Visualization*. John Wiley & Sons, 2015.
- [71] C. Sha, J. Hou, and H. Cui. A robust 2d Otsu’s thresholding method in image segmentation. *Journal of Visual Communication and Image Representation*, 41:339–351, 2016.
- [72] A. C. She and T. S. Huang. Segmentation of road scenes using color and fractal-based texture classification. In *Proceedings of 1st International Conference on Image Processing*, volume 3, pages 1026–1030. IEEE, 1994.

- [73] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
- [74] M. A. Sotelo, F. J. Rodriguez, and L. Magdalena. Virtuous: Vision-based road transportation for unmanned operation on urban-like scenarios. *IEEE Transactions on Intelligent Transportation Systems*, 5(2):69–83, 2004.
- [75] B. Southall and C. J. Taylor. Stochastic road shape estimation. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 1, pages 205–212. IEEE, 2001.
- [76] J.-Y. Sun, S.-W. Kim, S.-W. Lee, Y.-W. Kim, and S.-J. Ko. Reverse and boundary attention network for road segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019.
- [77] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015.
- [78] R. Tal and J. H. Elder. An accurate method for line detection and manhattan frame estimation. In *Asian Conference on Computer Vision*, pages 580–593. Springer, 2012.
- [79] C. Tan, T. Hong, T. Chang, and M. Shneier. Color model-based real-time learning for road following. In *2006 IEEE Intelligent Transportation Systems Conference*, pages 939–944. IEEE, 2006.
- [80] C. J. Taylor, J. Malik, and J. Weber. A real-time approach to stereopsis and lane-finding. In *Proceedings of Conference on Intelligent Vehicles*, pages 207–212. IEEE, 1996.

- [81] M. Teichmann, M. Weber, M. Zoellner, R. Cipolla, and R. Urtasun. Multinet: Real-time joint semantic reasoning for autonomous driving. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 1013–1020. IEEE, 2018.
- [82] W. Tranheden, V. Olsson, J. Pinto, and L. Svensson. DACS: Domain adaptation via cross-domain mixed sampling. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1379–1389, 2021.
- [83] Y.-H. Tsai, W.-C. Hung, S. Schuler, K. Sohn, M.-H. Yang, and M. Chandraker. Learning to adapt structured output space for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7472–7481, 2018.
- [84] M. A. Turk and M. Marra. Color road segmentation and video obstacle detection. In *Mobile Robots I*, volume 727, pages 136–142. International Society for Optics and Photonics, 1987.
- [85] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
- [86] T.-H. Vu, H. Jain, M. Bucher, M. Cord, and P. Pérez. Advent: Adversarial entropy minimization for domain adaptation in semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2517–2526, 2019.
- [87] Q. Wang, J. Gao, and Y. Yuan. Embedding structured contour and location prior in siamesed fully convolutional networks for road detection. *IEEE Transactions on Intelligent Transportation Systems*, 19(1), 2018.
- [88] Z. Wang, M. Yu, Y. Wei, R. Feris, J. Xiong, W.-m. Hwu, T. S. Huang, and H. Shi. Differential treatment for stuff and things: A simple unsupervised domain adaptation method for

- semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12635–12644, 2020.
- [89] A. R. Weeks and G. E. Hague. Color segmentation in the HSI color space using the k-means algorithm. In *Nonlinear Image Processing VIII*, volume 3026, pages 143–154. SPIE, 1997.
- [90] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. *Advances in neural information processing systems*, 34:12077–12090, 2021.
- [91] S. Yadav, S. Patra, C. Arora, and S. Banerjee. Deep CNN with color lines model for unmarked road segmentation. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 585–589. IEEE, 2017.
- [92] D. Yagi. Segmentation of color aerial photo graphs using HSV color models. *Proc. MVA’92*, 1992.
- [93] J. Yang, W. An, C. Yan, P. Zhao, and J. Huang. Context-aware domain adaptation in semantic segmentation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 514–524, 2021.
- [94] Y. Yang and S. Soatto. Fda: Fourier domain adaptation for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4085–4095, 2020.
- [95] F. Yu, W. Xian, Y. Chen, F. Liu, M. Liao, V. Madhavan, and T. Darrell. Bdd100k: A diverse driving video database with scalable annotation tooling. *arXiv preprint arXiv:1805.04687*, 2(5):6, 2018.

- [96] S. Yun, D. Han, S. J. Oh, S. Chun, J. Choe, and Y. Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6023–6032, 2019.
- [97] C. Zhang and P. Wang. A new method of color image segmentation based on intensity and hue clustering. In *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*, volume 3, pages 613–616. IEEE, 2000.
- [98] J. Zhang and H.-H. Nagel. Texture-based segmentation of road images. In *Proceedings of the Intelligent Vehicles’ 94 Symposium*, pages 260–265. IEEE, 1994.
- [99] P. Zhang, B. Zhang, T. Zhang, D. Chen, Y. Wang, and F. Wen. Prototypical pseudo label denoising and target structure learning for domain adaptive semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12414–12424, 2021.
- [100] S. Zheng, J. Lu, H. Zhao, X. Zhu, Z. Luo, Y. Wang, Y. Fu, J. Feng, T. Xiang, P. H. Torr, et al. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6881–6890, 2021.
- [101] Z. Zheng and Y. Yang. Rectifying pseudo label learning via uncertainty estimation for domain adaptive semantic segmentation. *International Journal of Computer Vision*, 129(4):1106–1120, 2021.
- [102] F. Zohourian, B. Antic, J. Siegemund, M. Meuter, and J. Pauli. Superpixel-based road segmentation for real-time systems using CNN. In *VISIGRAPP (5: VISAPP)*, pages 257–265, 2018.