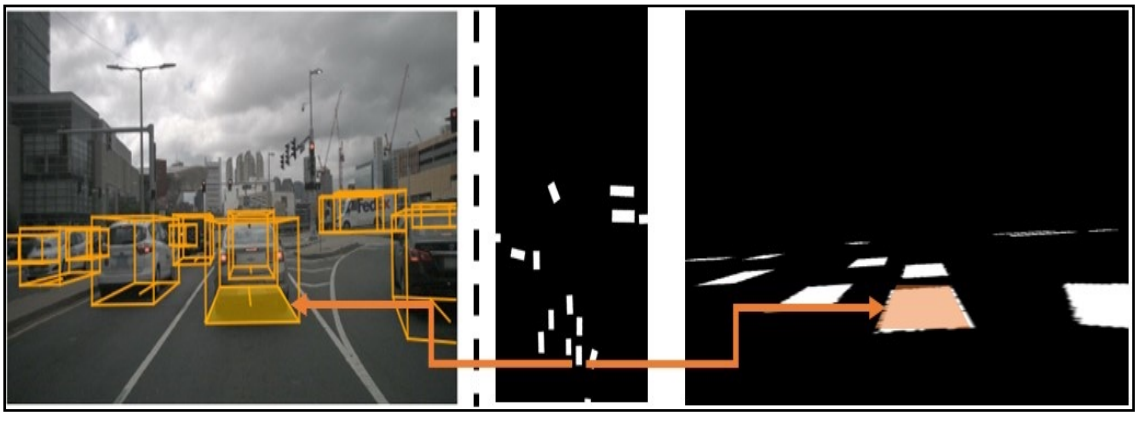
Deep convolutional neural networks for scene understanding and motion planning for self-driving vehicles

Abdelhak Loukkal

Par **Abdelhak LOUKKAL**

Deep convolutional neural networks for scene understanding and motion planning for self-driving vehicles

Thèse présentée
pour l'obtention du grade
de Docteur de l'UTC



Soutenue le 5 mai 2021
Spécialité : Informatique : Unité de recherche Heudyasic (UMR-7253)

D2607

Université de technologie de Compiègne

PhD Thesis

Deep convolutional neural networks for scene understanding and motion planning for self-driving vehicles

Spécialité : Informatique

Author:
Abdelhak LOUKKAL

Supervisor:
Yves GRANDVALET

A thesis submitted in partial fulfilment of the requirements for the degree of Doctor

May 05, 2021

Remerciements

Mes remerciements s'adressent tout d'abord à mon directeur de thèse Yves Grandvalet. Je lui exprime toute ma reconnaissance pour tous ses précieux conseils et sa patience durant ces trois années de thèse. Aux membres du jury va ensuite ma reconnaissance, en particulier aux rapporteurs, David Filliat et Cedric Demonceaux, qui ont consacré du temps à l'étude critique de mon manuscrit et à Véronique Cherfaoui qui a accepté de présider le jury. J'aimerais tout particulièrement remercier Tom Drummond pour son accueil au sein du laboratoire de l'Université de Monash ainsi que ses très précieux conseils en matière de vision robotique. Ma sincère amitié va ensuite aux membres permanents et non permanents du laboratoire Heudiasyc ainsi que tous mes collègues au sein de Renault. je les remercie sincèrement d'avoir rendu mon expérience plus agréable et appréciable.

A présent, je me dois de rendre hommage aux membres de ma famille qui m'ont soutenu sans jamais faillir. Mon frère Abdou et moi avons toujours été fascinés par les super-héros, et nous devons notre vie à ces quatre fantastiques. Le leader, le modèle, l'érudit, le guide, notre Grand-père Baba. Derrière chaque grand homme, se cache une femme. Ma grand-mère Yemma pour qui nous donnerions notre vie et qui donnerait la sienne pour nous, celle qui nous a aimés, éduqués et fait de nous les hommes que nous sommes. Que serions-nous sans Mama, la plus exceptionnelle des mères, une lionne qui a sacrifié sa vie et défié tous les augures pour amener ses lionceaux à l'age adulte. Et enfin, notre cher oncle Khalou, qui a éveillé notre imagination avec ses contes fantastiques et qui a rendu notre enfance si merveilleuse. Aussi loin que je m'en rappelle, j'ai toujours marché dans les pas de mon grand frère Abdou. Il sera toujours mon frère, ma figure paternelle, mon protecteur, mon modèle. Nous resterons à jamais inséparables et donnerons tout l'un pour l'autre.

J'aimerais également remercier une personne essentielle à mon bonheur et ma réussite, ma femme Sabrina. Elle a enduré mes peines, partagé mes joies et a participé à faire de moi l'homme que je suis aujourd'hui. Elle a surtout porté

en elle et donné naissance à mon ange Hana-Louisa, la prunelle de mes yeux, celle pour qui je décrocherais tous les astres. Je remercie également ma belle famille de m'avoir accueilli chaleureusement et soutenu dès le premier jour.

La thèse est un exercice difficile et un chemin jonché d'obstacles sur lequel avoir des amis sincères et fidèles est indispensable pour préserver sa motivation. Merci à vous également.

Enfin, j'aimerais terminer cette page de remerciements en dédiant cette thèse à mon défunt grand père, Baba. Mon frère et moi ferons tout pour que lors de nos retrouvailles tu sois fier de nous et nos enfants.

Acknowledgements

First of all, my thanks go to my thesis supervisor Yves Grandvalet. I express to him all my gratitude for all his invaluable advice and his patience during these three years of thesis. I am grateful to the members of the jury, in particular to the reviewers, David Filliat and Cedric Demonceaux, who devoted time to the critical study of my manuscript, and to Véronique Cherfaoui who agreed to chair the jury. I would especially like to thank Tom Drummond for his welcome to the Monash University laboratory and his invaluable advice on robotic vision. My sincere friendship then goes to the permanent and non-permanent members of the Heudiasyc laboratory as well as all my colleagues within Renault. I sincerely thank them for making my experience more pleasant and enjoyable.

Now it is my duty to pay tribute to the members of my family who have never failed me. My brother Abdou and I have always been fascinated by superheroes, and we owe our lives to these fantastic four. The leader, the model, the scholar, the guide, our Grandfather Baba. Behind every great man, there is a great woman. My grandmother Yemma for whom we would give our life and who would give her life for us, the one who loved us, educated us and made us the men that we are. What would we be without Mama, the most exceptional of mothers, a lioness who sacrificed her life and defied all omens to bring her cubs to adulthood. And finally, our dear uncle Khalou, who stirred our imagination with his fantastic tales and who made our childhood so wonderful. As far back as I can remember, I have always walked in my big brother Abdou footsteps. He will always be my brother, my father figure, my protector, my model. We will forever remain inseparable and give our all for each other.

I would also like to thank someone essential to my happiness and success, my wife Sabrina. She endured my sorrows, shared my joys and helped make me the man I am today. Above all, she carried within her and gave birth to my angel Hana-Louisa, the apple of my eye, the one for whom I would unhook

all the stars. I also thank my family in law for welcoming me warmly and supporting me from day one.

The thesis is a difficult exercise and a path strewn with obstacles on which having sincere and faithful friends is essential to preserve one's motivation. Thanks to you too.

Finally, I would like to end this page of thanks by dedicating this thesis to my late grandfather, Baba. My brother and I will do everything so that during our reunion you are proud of us and our children.

Contents

List of Figures

List of Tables

Abbreviations

CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
MLP	Multi-Layer Perceptron
BEV	Bird-Eye-View
CPV	Camera Projective View
OGM	Occupancy Grid Map

Chapter 1

Introduction

1.1 General context

According to a recent National Highway Traffic Safety Administration study (Singh, 2015), 94% of road accidents are caused by human errors which makes safety the main reason for the pursuit of autonomous vehicles. However, self-driving vehicles also bring the promise of reducing emissions, helping disabled persons and even helping less-privileged people as a study revealed that they are 4.3 times more prone to get involved in a car accident as they possess older vehicles with less driving assistance technology. This dream of self-driving or so called autonomous vehicles has captured human imagination for nearly a century with early engineers and futurists from General Motors imagining a radio-controlled vehicle propelled by magnetic fields induced by devices embedded in the road itself. Among the first serious attempts to create a self-driving car was the European project Eureka PROMETHEUS that was carried-out in the late 80s through the mid 90s and led to the development of the VITA II by Daimler-Benz. While the VITA II only showcased self-driving abilities in highways (Ulmer, 1994), the DARPA urban challenge (Buehler et al., 2009) in 2007 brought together research teams from both academia and industry and was first to pose the problem of self-driving in a pseudo-urban environment with 6 teams managing to complete the challenge (Bacha et al., 2008, Montemerlo et al., 2008, Urmson et al., 2007).

The series of DARPA challenges was also the theater of a major turning point in self-driving vehicles research when the winning team of the 2005 challenge used machine learning to successfully navigate in the off-road circuit in the least amount of time without any human intervention. Since then, machine learning has become an essential component in all modern autonomous vehicles. The rebirth of deep convolutional neural networks in 2012 (Krizhevsky et al., 2012a) along with the release of large-scale datasets allowed major breakthroughs in computer vision that cascaded to many applications, among them autonomous mobile platforms for which the perception task is of paramount importance. Indeed, self-driving vehicles pipelines are composed of different modular building blocks with the perception block being the one that processes the raw information coming from the vehicles' sensors. Perception is then arguably the most critical task as every following one heavily depends on its ability to detect the key elements of the driving scene such as the drivable space, the other vehicles and the vulnerable road users such as pedestrians or cyclists.

The sensors also play an essential role in the self-driving vehicles perception abilities with the main sensors being cameras and range sensors such as LiDARs or Radars. Cameras mimic the way animals perceive their environment. They provide information about the shape and color of the elements in the observed scene whereas range sensors rely on the emission/reception of a radio wave or infrared light pulses to very precisely estimate the distance of surrounding objects. The debate about the supremacy of a sensor over the other ones is an endless discussion with the great majority of industry and academia actors relying on sensor fusion to ensure redundancy and combine the qualities of each type of sensor. However, Elon Musk, the eccentric CEO of Tesla, which is a major actor in the efforts to develop a fully autonomous vehicle, has a different view than his pairs. He has been advocating for a system that relies mostly on cameras and went to the extreme of qualifying the LiDAR as being "lame". While we do not agree with this statement, we do try in this work to develop fully monocular based perception solution and bridge the gap between camera-based and LiDAR-based solutions. Even though we do not consider any sensor as being better than another and we consider that

sensor fusion is the way to go at the time of writing, it is hard to argue against the fact that cameras are by far the cheapest and most discrete sensors when compared with LiDARs.

Cameras are not intrinsically able to estimate depth but CNN-based depth estimation is achieving more and more impressive results with the availability of large-scale datasets. Having access to a depth estimate for the elements of a driving scene is highly important, because ultimately the information extracted by the different sensors in different geometric spaces needs to be projected to a common space, which is usually the 3D space, in order to be processed by the planning and navigation building block. This space is preferred to the camera projective space for example because the size of objects in this space does not depend on the distance to the sensor. In this thesis, we argue that the fact that closer objects appear bigger in the camera-view actually contains a valuable information. In camera-view, bigger means closer and the closest objects to the vehicle are in most cases those that are the most critical as they are in the immediate environment of the vehicle and are more susceptible to cause a collision. It has even been shown that the difference in performance in 3D tasks between camera-based methods and LiDAR-based ones is not only due to the intrinsic nature of each sensor but also to the choice of representation (Wang et al., 2019a) which is confirmed in our findings.

The modular approach to self-driving vehicles pipelines is the most popular but researchers have also been trying to replace part of the pipeline or even the whole pipeline with a single neural network with the objective to avoid laborious hand-engineering and error accumulation along the pipeline. Camera-based end-to-end approaches for example take raw sensor information as input and output directly driving commands which induces a lack of interpretability. Therefore, intermediate representation were then the natural addition to end-to-end networks in order to be able to delve their driving decisions. This also led to mid-to-mid or mid-to-end networks that take as input Bird-Eye-View representations of the scene obtained from the perception building block output and rasterized HD map portions.

1.2 Framework and objectives

During this thesis, some perception approaches for self-driving vehicles were developed using deep convolutional neural networks applied to monocular camera images and High-Definition map (HD-map) rasterized images. HD-maps are centimeter-level precise maps that give information about the road layout and features. They often come in a vectorial format and are then rasterized into images in order to be efficiently processed by CNNs. We focused on camera-only solutions instead of leveraging sensor fusion with range sensors because cameras are the most cost-effective and discrete sensors. The objective was also to show that camera-based approaches can perform at par with LiDAR-based solutions on certain 3D vision tasks. Real-world data was used for training and evaluation of the developed approaches but simulation was also leveraged when annotated data was lacking or for safety reasons when evaluating driving capabilities.

HD-map are very often used to help the localization of self-driving vehicles in their environment. We have tried to leverage these in the form of rasterized images in order to provide a spatial context to perception-oriented neural networks. Instead of only using these HD-map rasterized images as a prior knowledge, we have also developed an approach that allows to output these maps from a monocular camera. The HD-maps are a snapshot of an area at a given time so changes to the road layout would make these maps obsolete, hence having a neural network able to output the road layout based on a camera image is an appealing asset.

Cameras provide visual information in a projective space where the perspective effect does not preserve the distances homogeneity. Scene understanding tasks such as semantic segmentation are then often operated in the camera-view space and then projected to 3D using a precise depth sensor such as a LiDAR. Having this scene understanding in the 3D space is useful because the vehicles evolve in the 3D world and the navigation algorithms reason in this space. Our focus was then to leverage the geometric knowledge about the camera parameters and its position in the 3D world to develop an approach that

allows scene understanding in the 3D space using only a monocular image as input.

Neural networks have also proven to be useful for more than just perception and are more and more used for the navigation and planning tasks that build on the perception outputs. Being able to output 3D scene understanding information from a monocular camera has also allowed us to explore the possibility of having an end-to-end holistic neural network that takes a camera image as input, extracts intermediate semantic information in the 3D space and then plans the vehicle's trajectory.

This thesis was developed within SIVAlab (LABoratoire des Systèmes Intègres pour le Véhicule Autonome), a joint laboratory between Renault SAS and the Heudiasyc Laboratory, UMR UTC/CNRS 7253. It was funded by a CIFRE fellowship, that was granted from Renault SAS and the ANRT (Agence Nationale de la Recherche et de la Technologie).

1.3 Organization and contributions of the thesis

The rest of this manuscript is organized as follows:

Chapter 2 presents briefly the current state-of-the-art algorithms that are used to solve the subset of perception and motion prediction tasks that we study in this thesis, namely: semantic segmentation, depth estimation and imitation learning for ego-motion prediction. For scene understanding and depth estimation, we focus on monocular camera based approaches, while for motion prediction, we focus on ego-motion prediction approaches as it is more related to our work.

Chapter 3 covers our work on enhancing semantic segmentation with a spatial context information. We explore 3 different ways of incorporating depth and cartographic information into an encoder-decoder semantic segmentation CNN. We have chosen to work on simulated data because no real-world dataset

with both semantic segmentation and cartographic annotation was available then. Our experiments have shown that the approach with the best results consists in adding additional encoders that take the spatial context as input but this approach is limited by the fact that precise depth and cartographic information need to be available at any moment which is an expensive option.

Chapter 4 presents a new weighing scheme for lightweight semantic segmentation neural networks. The most commonly adopted weighing scheme is based on the frequency on the classes, with the rarest classes having the higher weights. However, we argue that this weighing scheme does not necessarily suit autonomous driving as a rare object located far from the ego-vehicle is not more important than a more common one located just in front. For this reason, we propose a disparity-weighing scheme that gives more importance to closer objects, which makes sense for autonomous vehicles as closer objects are more prone to cause a collision. To avoid an additional annotation burden, we propose to estimate disparity maps with an off-the-shelf self-supervised CNN.

Chapter 5 also pertains to scene understanding with a monocular camera, but with the challenge of outputting a semantic mask in bird-eye-view instead of the usual camera-view space. Given that state-of-the-art semantic segmentation networks are fully convolutional encoder-decoders, a geometric transform is necessary to preserve the receptive field when the output mask is not in the same space as the input. Our choice of transform is the homography which also poses a new challenge. Indeed, homographies are planar transforms so 3D objects such as vehicles that lie above the ground plane and cannot be successfully warped to the bird-eye-view. To circumvent this issue, we propose footprint segmentation that consist in segmenting only the footprint of 3D objects in order to respect the planar world assumption implied by the homography transform.

Chapter 6 builds on the previous chapter to propose a holistic end-to-end trajectory planning network. It takes monocular images as input, outputs intermediate semantic masks in bird-eye-view, using the previously introduced footprint segmentation, and finally plans the trajectory of the ego-vehicle

based on these semantic masks. Previous monocular based works have introduced end-to-end driving networks that have intermediate representation in camera-view, whereas we leverage our footprint segmentation method to obtain an intermediate representation in the 3D space. This is arguably preferable because distances in 3D are invariant to the distance to the sensor. In a traditional modular self-driving vehicle pipeline, the different perception outputs are merged in a 3D common representation and then digested by the planning building block which we mimic in our approach but in an end-to-end fashion. This intermediate representation also provides interpretable results that can help understand the driving decisions.

Chapter 7 concludes this thesis, summarizes its contributions and opens perspectives for future works.

This thesis is based on the following scientific publications:

- Loukkal, A., Grandvalet, Y., Frémont, V. & Li, Y. Improving semantic segmentation in urban scenes with a cartographic information. In 2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV) (pp. 400-406). IEEE.
- Loukkal, A., Grandvalet, Y. & Li, Y. Disparity weighted loss for semantic segmentation of driving scenes. In 2019 Intelligent Transportation Systems Conference (ITSC) (pp. 3427-3432). IEEE.
- Loukkal, A., Grandvalet, Y., Drummond, T. & Li, Y. Driving Among Flatmobiles: Bird-Eye-View Occupancy Grids From a Monocular Camera for Holistic Trajectory Planning. In 2021 Winter Conference on Applications of Computer Vision (WACV) (pp. 51-60). CVF/IEEE.
- Loukkal, A., Grandvalet, Y. & Drummond, T. A mixture of view experts approach to bird-eye-view semantic maps from a monocular camera. Under submission to Pattern Recognition Journal.

Chapter 2

Background and related work

2.1 Introduction

In this chapter, we present how robotic perception and ego-motion prediction evolved with the resurgence of neural networks in 2012 when Krizhevsky et al. (2012b) won the ImageNet challenge with Convolutional Neural Networks (CNN). Neural networks have become an essential component of all modern autonomous robots and more specifically autonomous ground vehicles. This has been made possible by improved computing resources, the availability of task-specific datasets and more expressive neural network architectures.

We first present the datasets that allowed to scale up machine learning-based approaches for real-world applications. Second, we present the architecture designs of modern semantic segmentation neural networks, as this task is paramount for scene understanding. Then, we briefly present state-of-the-art networks for another important task, which is depth estimation from monocular cameras. Finally, we present how neural networks can go beyond perception and be used for ego-motion forecasting by imitating expert demonstrations.

TABLE 2.1: Perception datasets for autonomous driving. BB refers to Bounding Boxes, SS to Semantic Segmentation, and DD for Dense Depth.

Dataset	RGB	Stereo	LiDAR	# frames	BB	SS	DD	HD map
Camvid (Brostow et al., 2008)	Yes	No	No	701	No	Yes	No	No
KITTI (Geiger et al., 2012)	Yes	Yes	Yes	15K	3D	No	Yes	No
Cityscapes (Cordts et al., 2016)	Yes	Yes	No	5K	No	Yes	Yes	No
Mapillary (Neuhold et al., 2017)	Yes	No	No	25K	No	Yes	No	No
BDD100K (Yu et al., 2020)	Yes	No	No	100K[1]	2D	Yes	No	No
Apolloscape (Huang et al., 2018)	Yes	Yes	Yes	147K	3D	Yes	Yes	No
A2D2 (Geyer et al., 2020)	Yes	No	Yes	41K[2]	3D	Yes	No	No
Waymo (Sun et al., 2020)	Yes	No	Yes	230K	3D	No	No	No
Lyft L5 (Kesten et al., 2019)	Yes	No	Yes	46K	3D	No	No	Yes
Argoverse (Chang et al., 2019)	Yes	Yes	Yes	22K	3D	No	No	Yes
Nuscenes (Caesar et al., 2019)	Yes	No	Yes	40K	3D	No	No	Yes

[1] 100K with bounding boxes and 10K with semantic segmentation

[2] 41K with semantic segmentation and 12K with bounding boxes

2.2 Autonomous driving perception datasets

Machine learning algorithms are data-driven approaches and their performance depend heavily on the availability of high quality and large datasets. In this section we present the main perception datasets for autonomous driving. An overview of the available datasets is provided in Table 2.1.

High Definition (HD) maps are an essential component of the works developed in this thesis so we first present the datasets that do not contain this information, and then we present those that do.

2.2.1 Datasets with no HD maps

Among the first semantic segmentation datasets for autonomous driving, CamVid (Brostow et al., 2008) is a road scene dataset composed of 5 video sequences captured with a 960×720 resolution camera mounted on a car giving a total of 701 frames. The images were manually annotated according to 32 labels. However, given its very limited size, this dataset does not allow to train neural networks that can generalize well to new environments.

Cityscapes (Cordts et al., 2016) is a more recent urban street scene semantic segmentation dataset that comprises 5000 finely annotated images, 20000 coarsely annotated images (with the 5000 finely annotated images coarsely annotated in order to enable research on densifying coarse labels). Images are annotated according to 30 classes regrouped in 8 meta-classes: flat surfaces, humans, vehicles, constructions, objects, nature, sky, and void. Images are pixel-level annotated with instance level annotation available for the "car" and "person" classes. Data was captured in 50 cities during several months, during daytime and good weather conditions. This dataset has been essential in the development of state-of-the-art semantic segmentation networks and the Cityscapes benchmarks are still relevant for the evaluation of these networks.

KITTI (Geiger et al., 2012) is a larger dataset comprising 6h of driving with 3D bounding box annotations for 7481 training images and 7518 test images. It contains data captured with various sensors such as high resolution RGB and gray-scale stereo cameras, and a LiDAR. The KITTI dataset helped greatly in the development of LiDAR-based perception approaches such as road detection or object detection as it was among the first large datasets to provide LiDAR point clouds along with camera images. Semantic segmentation labels in the camera space are not originally provided but several researchers in the field of computer vision have provided small sets (few hundred samples) of semantic segmentation ground truth data. Semantic segmentation annotation is particularly challenging to obtain as it requires to label every pixel in an image which makes large semantic segmentation datasets quite valuable.

One of the first datasets to provide a large collection of semantic segmentation annotations is Mapillary Vistas (Neuhold et al., 2017) with 25000 annotated high-resolution image, 100 object categories and a high variability in weather conditions and capturing times. The BDD100K (Yu et al., 2020) dataset followed with 100K raw driving video sequences with more than 100 million images. It provides 100K ground-truth 2D bounding-boxes annotations and 10K ground-truth semantic segmentation mask annotations. The largest to date semantic segmentation dataset is Apolloscapes (Huang et al., 2018). Acquired with 4 cameras and 1 LiDAR in Chinese cities, providing semantic segmentation annotation as depth maps regarding the static background for a

staggering number of 146 997 frames. More recently, acquired with 6 cameras and 5 LiDARs in 3 German cities, the Audi Autonomous Driving Dataset (A2D2) (Geyer et al., 2020) provides 41 277 frames with semantic segmentation annotation with 12 497 frames also having 3D bounding box annotations for objects located in the field of view of the front camera. The Waymo open dataset (Sun et al., 2020) is another large-scale perception dataset acquired with 5 LiDARs and 5 cameras and containing 230K frames with 3D bounding boxes.

2.2.2 Datasets with available HD maps

HD maps are a key component in the autonomous driving pipeline and are vastly used in modern self-driving cars projects. NuScenes (Caesar et al., 2019) was a pioneer dataset that comprises highly accurate HD map rasters. The dataset records the measurements of a complete suite of sensors: 6 cameras, 32-channels LIDAR, long-range radars. The whole dataset comprises 40 000 annotated frames in different regions of Boston and Singapore. Each driving sequence lasts around 20s and images are acquired at a framerate of 2Hz. The two most important feature of nuScenes when compared to previous datasets are the joint availability of (i) 3D bounding boxes, (ii) 11 Bird-Eye-View semantic layers provided as binary semantic masks, where each pixel corresponds to 0.1×0.1 square meters.

Lyft level 5 perception dataset (Kesten et al., 2019) is very similar to nuScenes and can even be explored with the same devkit. It contains data from 7 cameras, 3 LiDARs and provides 46 000 3D annotated frames as well as a 7-layers HD Bird-Eye-View semantic map.

The Argoverse 3D tracking dataset (Chang et al., 2019) also provides rasterized HD map annotations along with 22K frames annotated with 3D bounding boxes. This dataset stands out from the others by being the only one to provide stereo front-facing cameras images in addition to LiDAR points clouds and surrounding cameras images. Having access to stereo cameras allows to

obtain a disparity map and opens up more opportunities for purely monocular approaches.

2.3 Autonomous driving simulators

Simulators are an alternative option for training autonomous driving models as they allow to evaluate them in scenarios that are dangerous in the real world. In this section we briefly present open-source simulators that provide the necessary inputs and labels for training perception models. Even though open-source simulators are not as realistic and diverse as industrial level simulators, they are still suitable to evaluate or pre-train neural networks. Carla simulator (Dosovitskiy et al., 2017a) is built over Unreal Engine 4 with a Python client API and offers a complete suite of sensors with virtual RGB, depth and semantic segmentation cameras along with 3D bounding boxes annotations and virtual LiDAR point clouds. Several towns, weathers, vehicles and traffic densities are available to evaluate deep neural networks on a wide set of scenarios. LGSVL (Rong et al., 2020) and AirSim (Shah et al., 2018) also provide a complete set of perception inputs and labels with AirSim being able to simulate both ground vehicles and drones.

2.4 Semantic segmentation with CNNs

The success of deep CNNs for image classification (Krizhevsky et al., 2012a) has encouraged researchers to explore the effectiveness of these networks for dense predictions tasks like semantic segmentation or depth estimation. Semantic segmentation has been drawing a lot of attention from the computer vision and autonomous driving communities for many years because in addition to detecting key elements in the scene, it adds semantic information to the global scene understanding problem. Semantic segmentation is usually applied in the Camera Projective View (CPV) and annotation is also supplied in the same space for most datasets. However, for applications like autonomous

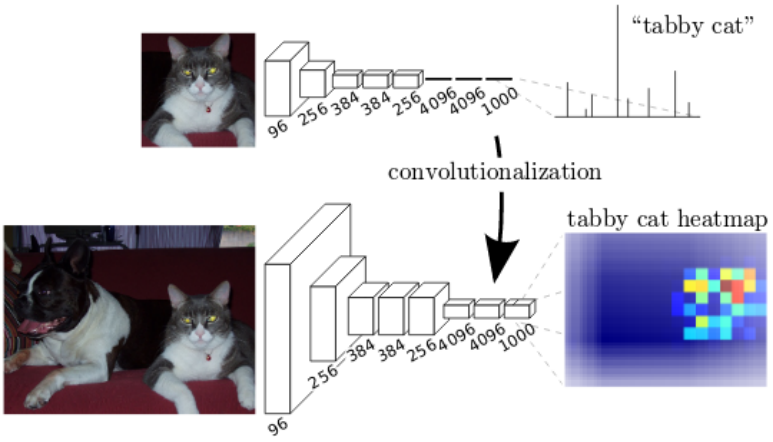


FIGURE 2.1: Fully convolutional networks replace fully connected layers with convolutional ones hence preserving the spatial information. Image originally by Shelhamer et al. (2017).

vehicles, the perception module processes data coming from sensors that operate in different geometric spaces and then fuses the extracted information in a unified view, generally Bird-Eye-View (BEV). Semantic segmentation masks are then usually projected to BEV using the LiDAR depth information but recent works have tackled the problem of outputting directly BEV semantic masks from monocular images.

2.4.1 Semantic segmentation in camera view

2.4.1.1 Evolution of the architectures

The current state of the art approaches in semantic segmentation take advantage of Fully Convolutional neural Networks (FCN) (Shelhamer et al., 2017) keeping the spatial information, that is usually lost in regular CNNs, by avoiding fully connected layers, see Figure 2.1. These networks usually come in two parts: the encoder or backbone network extracts features from the input image and the decoder up-samples the encoded feature maps to match the size of the input image (the most naive decoder being bilinear sampling). SegNet (Badrinarayanan et al., 2017a) was among the first fully convolutional encoder-decoder architectures.

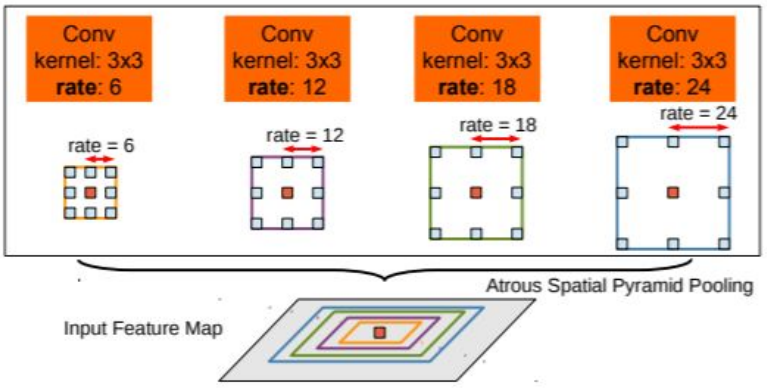


FIGURE 2.2: Atrous Spatial Pyramid Pooling uses dilated convolutions with increasing dilation rates to extract multi-scale contextual information. Image originally by Chen et al. (2018a)

Skip connections between the input and output of convolution layers were introduced as part of the ResNet architecture by He et al. (2016) helping the gradient flow in deep architectures and are often used in the backbone network.

Enlarging the receptive field was key in the success of modern semantic segmentation as it allows to capture a larger contextual information. DeepLab (Chen et al., 2015b) takes advantage of Atrous or dilated convolution to enlarge the receptive field without any additional computational burden. In Yu and Koltun (2016), a series of dilated convolutions with increasing dilation rates are aggregated to obtain multi-scale contextual information. DeepLab V2 (Chen et al., 2018a) also extracts multi-scale contextual information using Atrous Spatial Pyramid Pooling (ASPP), see Figure 2.2. Unlike Yu and Koltun (2016) which operates in serial, the ASPP applies the dilated convolutions with different dilation rates in parallel, hence obtaining a pyramid of multiscale features from the same input.

Pyramid Scene Parsing (PSPNet) (Zhao et al., 2017) introduced the pyramid pooling module to extract richer contextual information as well. This module applies spatial pooling at different scales, upsamples then concatenates the obtained pyramid of features. DeepLab v3 enhances the ASPP with image-level features (Liu et al., 2015, Zhao et al., 2017) and batch normalization (Ioffe and

Szegedy, 2015). DeepLab V3+ combines DeepLab V3 encoder with a convolutional decoder improving the boundaries delineation. (Takikawa et al., 2019) introduced a two-streams architecture with Gated-SCNN: the first stream is referred to as regular stream as it is similar to commonly used network; the second stream takes as input the gradient of the input image and features extracted by the first convolutional layer of the regular stream and and outputs semantic boundaries thanks to gated layers that filter information that is irrelevant for the boundary tasks; ASPP is used to fuse the output of the two streams.

Attention mechanisms are widely used for various computer vision tasks and have been successfully adopted in semantic segmentation networks (Chen et al., 2016, Li et al., 2018a). At the time of writing, state-of-the-art performance has been achieved by a hierarchical multi-scale attention network (Tao et al., 2020) that learns relative attention masks between adjacent scales instead of learning a mask for each scale.

2.4.1.2 Conditional random fields as a post-processing step

Conditional Random Fields (CRF) are graphical models that were commonly used for semantic segmentation. Their energy function is composed of unary potentials and pairwise potentials on neighbors (pixels or blocks of pixels). Originally, the main limitation of this model was its inability to capture long range dependencies, for pixels that are in different regions of the image. Region based approaches that incorporate hierarchical connectivity and higher-order potentials tried to improve on the original CRFs but lacked accuracy due to the unsupervised segmentation that produces regions. Fully connected CRFs are more expressive models, which present the advantage to have pairwise potentials between all pixels in an image, but the difficulty of inference hindered their use. A mean field algorithm was proposed to learn efficiently fully connected CRFs (Krähenbühl and Koltun, 2011). These fully connected CRFs were the state of the art approach in semantic segmentation but with the resurgence and the jump in performance allowed by neural networks, CRFs have then been successfully applied as a post processing or refinement step in

CNNs (Chen et al., 2015b) in order to improve boundaries delineation for segmentation tasks. Joint training procedures are further introduced in (Schwing and Urtasun, 2015) and (Lin et al., 2016). Proposed in (Zheng et al., 2015), this approach includes the CRF in an end-to-end CNN-CRF where the Mean Field approximate inference is formulated as a recurrent neural network and the Mean Field algorithm steps reformulated as convolution operations. More recently, Teichmann and Cipolla (2019) consider a locality assumption where the pair-wise potential of two pixels separated by a Manhattan distance inferior to a certain filter size is considered null which allows to reformulate the inference as convolution operations. Using CRFs does not seem to be a reasonable option at the time of writing given that CRFs slow down training and inference and state-of-the-art segmentation networks have achieved a performance that would hardly be improved using CRFs. An example of that is the DeepLab architectures that first used CRFs as a post-processing step before abandoning it in further iterations.

2.4.1.3 Lightweight CNNs for low-computing power platforms

Another important aspect for CNNs, especially for mobile robotics, is their resource consumption. Paszke et al. (2016) introduced ENET, an efficient CNN for semantic segmentation that is 18 times faster than SegNet (Badrinarayanan et al., 2017b), while obtaining better results on the mean Intersection over Union (mIoU) metric. Real-time computation is achieved with early downsampling that heavily reduces the size of the input. A consequence of the aggressive downsampling is the loss of information that incurs in much lower performance than state of the art networks. Introduced in (Romera et al., 2018), ERFNET is another efficient network based on the skip connections (He et al., 2016) and 1D convolutions to reduce resource usage. ERFNET is twice as slow as ENET but achieves much better mIoU. Image Cascade Network (ICNet) (Zhao et al., 2018), as its name suggests, uses a cascade of different resolutions images as input, with most of the network's weights processing the low resolution input hence reducing the computation burden. BiSeNet (Yu et al., 2018) is composed of two parts or "paths" that increase

the without degrading too much the accuracy: the spatial path that is a shallow network that outputs a large size feature map; the context path that is lightweight network with global average pooling at its end to extract rich contextual information. DFANet (Li et al., 2019) achieves a better speed/accuracy trade-off by aggregating sub-networks and aggregating corresponding stages between the sub-networks. More recently, Multiple Spatial Fusion Network (MSFNet) (Si et al., 2019) uses spatial aware pooling that enlarges the receptive field while maintaining the spatial information and class boundary supervision in the form of another decoder branch supervised by the boundary of the ground truth masks. In SwiftNetRN-18, a regular encoder-decoder architecture with lateral-skip connections and spatial pyramid pooling, the authors advocate for the use of ImageNet-pretraining to benefit from the regularization effect of transfer learning and achieve state-of-the-art performance on Cityscapes (Brostow et al., 2009). Lightweight networks used to present the sole advantage of being fast and were not usable for real applications but recent advances have dramatically increased their performance.

2.4.2 Semantic segmentation in Bird-Eye-View

Occupancy grid maps (OGM) (Elfes, 2013) represent the spatial environment of a robot and reflect its occupancy as a fine-grained metric grid. These grid maps have been widely used to model the environment of indoor and outdoor mobile robots as well as automotive systems. Once acquired, they can be used for various tasks such as path planning. OGMs can be acquired with range sensors like LiDAR or RADAR, but also from RGB-D cameras (Himstedt and Maehle, 2017), stereo cameras (Li and Ruichek, 2014), or from the fusion of multiple sensors (Oh and Kang, 2016). For example, semantic OGMs are predicted from a LiDAR and a monocular camera thanks to deep learning and Bayesian filtering in (Erkent et al., 2018).

As OGMs are usually binary masks indicating the presence of a static or dynamic object, Bird-Eye-View (BEV) semantic maps and OGMs will be used interchangeably. In this section, we will focus on approaches that take only RGB images as input. As OGMs are in the BEV space, two different

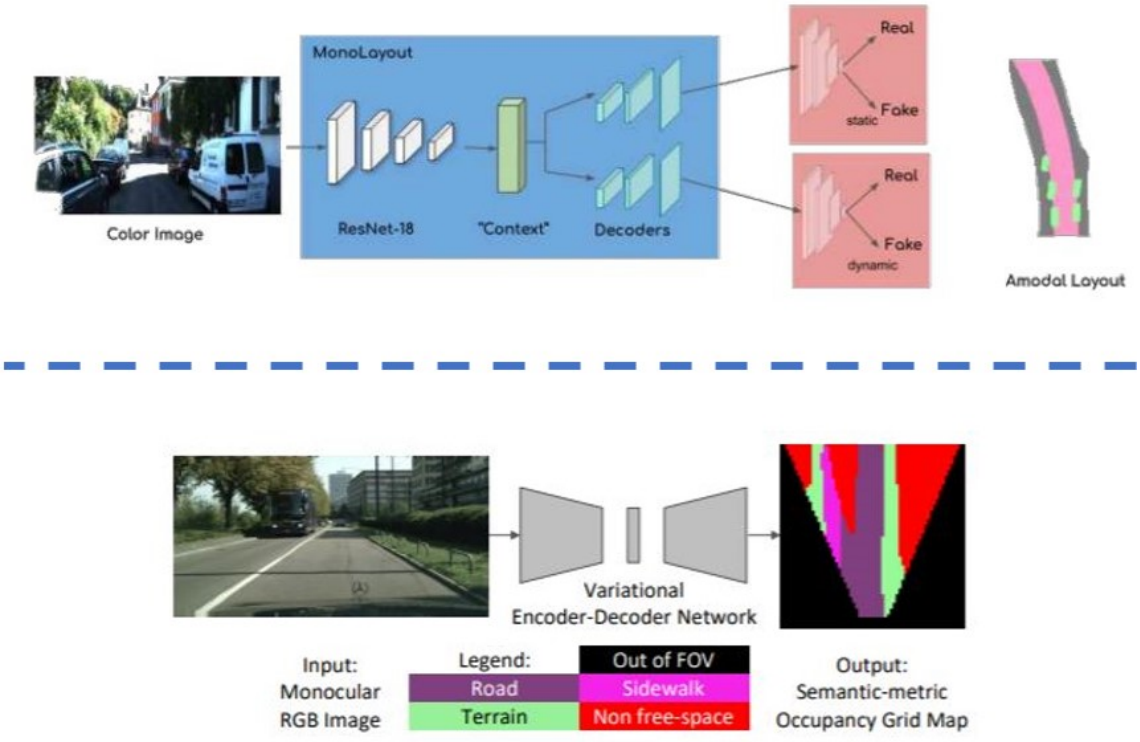


FIGURE 2.3: Two examples of purely-data driven approaches for obtaining BEV semantic maps from a monocular image. Monolayout (Mani et al., 2020) (top image) resizes the input to match the output OGM shape and uses an encoder-decoder network and Lu et al. (2018) (bottom image) encodes the image in a vector then reshapes this vector to match the desired OGM shape.

strategies can be adopted to output them from a Camera Projective View (CPV) space: a fully data-driven approach that ignores the geometric prior information contained in the camera's parameters or an approach that takes these parameters into consideration, see respectively Figure 2.3 and Figure 2.4.

2.4.2.1 Purely-data driven approaches

The papers presented is this section adopt a purely data-driven approach to the transformation from CPV to BEV. Taking only monocular images as input, Variational Encoder-Decoder (Lu et al., 2018) aims to predict semantic BEV maps. The first component of this architecture is an encoder network that extracts feature maps from the input image. These feature maps are

flattened to obtain a latent space vector that is then reshaped to the desired BEV aspect ratio and fed to a decoder network. The learning processes is supervised by two losses: a semantic segmentation loss on the output of the decoder and a KL-divergence loss between the latent space vector and the normal distribution N(0,1). This approach leverages weak ground-truth semantic BEV masks obtained by warping CPV semantic segmentation masks using depth un-projection with the associated depth maps.

The geometric prior knowledge are also ignored in the View Parsing Network (VPN) (Pan et al., 2020) that transforms feature maps from the camera view to Bird-Eye View using a module named the view transformer module. The feature maps in camera-view are flattened while the channel dimension remains the same. An MLP is then applied to the flattened features keeping the same vector size for the output with the idea to learn the dependencies between each pixel position in the camera view with all pixel positions in BEV. The output vector is then reshaped to match the size of the feature map before applying the transformer module, hence supposing that the camera-view map and the BEV one have the same shape ratio which is not necessarily true.

Lu et al. (2018) flattens the features obtained in the camera projective space to obtain a latent space vector then reshapes it in the BEV grids' dimensions, in a process that does not preserve spatial information. VPN also flattens the CPV feature maps and applies a MLP to the obtained vector which also does not preserve spatial information.

MonoLayout (Mani et al., 2020) introduces another encoder-decoder network for BEV semantic maps prediction. An encoder network extracts features from the camera images and feeds its output feature maps to two decoders: one that outputs a semantic mask for the dynamic objects in the scene such as vehicles or pedestrians, and another decoder that outputs a semantic mask for the static elements of the scene such as the road or the sidewalk. The learning of these masks is supervised by segmentation loss coupled with an adversarial loss that helps the network hallucinate occluded regions of the image. Monolayout reshapes the input image into the BEV OGMs shape and then applies an encoder-decoder architecture. Spatial information is partially

preserved, but the receptive fields in input space (CPV) and output space (BEV) are not homogeneous, and no geometric transformation is applied. Here again weak ground-truth is obtained by projecting semantic segmentation masks (ground-truth or obtained with off-the-shelf network) to BEV using the associated depth maps (obtained from LiDAR or from off-the-self network).

FISHING-NET (Hendy et al., 2020) aggregates the features obtained with three different networks operating on RGB images, BEV-LiDAR images, and BEV-RADAR images and finally predicts semantic maps in the BEV space. A MLP based transform inspired by (Pan et al., 2020) is leveraged to project to the BEV space the feature maps obtained with the network that operates on the RGB images, hence making these features homogeneous with those obtained with the LIDAR and the RADAR.

2.4.2.2 Geometry driven approaches

The Orthographic Feature Transform (OFT) network (Roddick et al., 2019), whose original purpose is to output 3D bounding boxes, consists in projecting the features extracted in camera view to an orthographic space. To do so, voxel-based features are generated by accumulating camera-view features; then these features are collapsed along the vertical dimension to obtain features in an orthographic plane. In addition to bounding boxes coordinates and dimensions, this network also outputs a confidence map in BEV which can be assimilated to an occupancy grid map.

Pyramid Occupancy network (PyrOccNet) (Roddick and Cipolla, 2020) introduces a new dense transformer layer that converts features of shape $H \times W \times C$ in the perspective image space to features of shape $X \times Z \times C$ in an orthographic BEV space. The first step consists in a 1D convolution layer that encodes each column of the perspective image space features (along the H axis) to a fixed length feature vector and a second convolution layer decodes this vector along the depth axis Z. The second step resamples features obtained in the first step to Cartesian coordinates using the camera intrinsic matrix. A backbone network extracts feature from the camera images and a

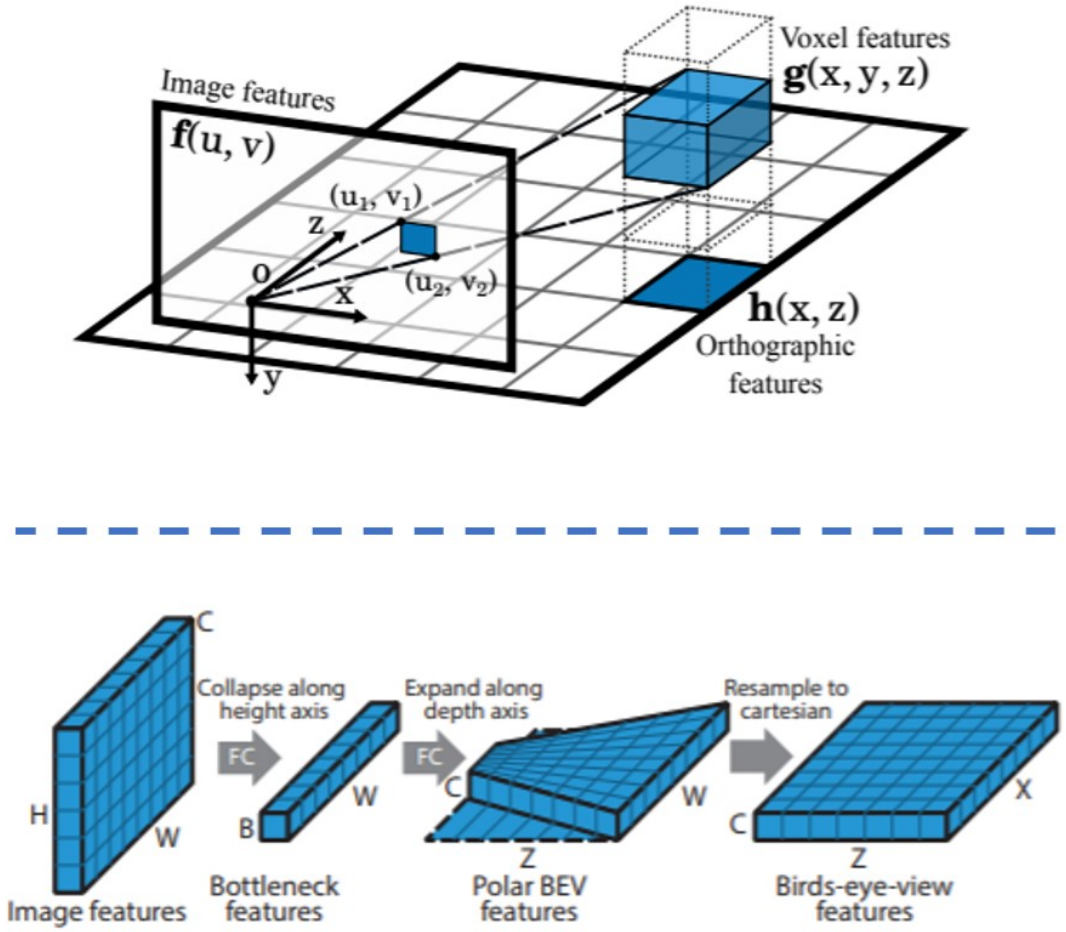


FIGURE 2.4: Two examples of approaches that leverage geometric insights to obtain BEV semantic maps from a monocular camera. Images originally by Roddick et al. (2019) (top) and Roddick and Cipolla (2020) (bottom).

Feature Pyramid Network (FPN) (Lin et al., 2017) extracts multiscale features. Each of the feature map in the pyramid is processed by an individual transformer layer on a subset of depth values. The final BEV feature map is obtained by concatenating the outputs of these transformer layers along the depth axis.

Lift, Splat and Shoot (LSS) (Philion and Fidler, 2020) also predicts BEV semantic maps from monocular cameras in two steps referred to as "lift" and "splat". The camera image is first processed by a neural network that extract feature maps. For each pixel position of these feature maps in the perspective camera space, a distribution over discretized depth values is predicted and

then used to "lift" the feature maps into a 3D point cloud using the camera intrinsic and extrinsic matrices. This 3D point cloud is then converted to a point-pillars like voxel grid (Lang et al., 2019). Each point is assigned to its closest "pillar" and the Z axis is collapsed to obtain a BEV tensor in the "splat" step.

Footprints (Watson et al., 2020) predicts the footprints of dynamic and static objects and both visible and occluded ground surfaces. An encoder-decoder network predicts 4 single channel output masks for the visible ground segmentation, the occluded ground segmentation, the depth map of visible pixels and the depth map of the occluded ground surface. Off-the-shelf segmentation and depth estimation networks are used to obtain weak ground truth for the the visible ground segmentation and visible depth maps of a sequences of stereo camera frames. A frame in the sequence is selected as target frame and the other frames are used as source frames. The visible ground segmentation and visible depth maps of each of the source frames are used to obtain the 3D coordinates of the traversable pixels which are then back projected in the target frame, the camera poses being obtained with ORB-SLAM2 (Mur-Artal and Tardós, 2017). The back projected noisy depth maps and semantic masks of all source frames are aggregated and filtered to obtain a single more robust weak ground truth corresponding to the target frame. The obtained weak ground truth segmentation masks and depth maps are used as training signal for the occluded ground segmentation and depth map. Moving objects are identified by comparing the optical flow with the induced optical flow in the target frame and ignored during training.

Schulter et al. (2018) focused on predicting a semantic road layout for visible and occluded ground surfaces. The trick introduced in the paper to "look around objects" consists in hallucinating the regions of the image occluded by foreground objects (vehicles, pedestrians, etc.) in the semantic segmentation space instead of the RGB space. First an off-the-shelf semantic segmentation network is used to obtain labels. The pixels corresponding to foreground objects are masked in the RGB image and the masked RGB image and the mask itself are fed to an encoder-decoder CNN that predicts the semantic segmentation mask and the depth of all pixels in the image, occluded pixels

included. As it would be very costly to manually annotate the semantics and depth behind foreground object pixels (occluded pixels), these pixels are ignored during training and blocks of visible pixels are masked during training and the model is optimized to hallucinate these regions of the image where ground-truth is available, with the objective of being able to do the same on occluded regions during test time. The semantic mask is projected to the 3D space using the associated depth map and the camera intrinsic matrix obtaining a BEV road layout. The depth and semantics being noisy, the BEV road layout is refined using another CNN (trained separately) and simulated data and/or maps obtained with Open Street Map. Simulated data are leveraged with an adversarial loss that learns to discriminate BEV layout produced by the refinement CNN and the layouts from the simulator. The OSM maps are used as a reconstruction training signal. The authors also briefly propose to include foreground objects by using off-the-shelf 2D object detectors to obtain bounding boxes, project them onto the BEV semantic road layout before the refinement step and use the same strategy that couples an adversarial loss with simulated data. Only two qualitative examples are provided as evidence for the BEV foreground objects segmentation which is not enough to evaluate the performance of the network for this task.

Wang et al. (2019b) also outputs road layouts but in a parametric form and ignores the dynamic objects in the scene like the vehicles and the pedestrians. First, given an RGB image of the scene, an off-the-shelf network precomputes a semantic segmentation mask that is projected to BEV using ground-truth depth obtained with LiDAR. The obtained BEV mask is fed as input to a CNN with a final fully connected layer that encodes the features in a vector. A similar architecture with shared weights operates on simulated BEV masks and an adversarial loss forces domain-agnostic predictions for the features vector. An MLP architecture digests the feature vectors and outputs the layout parameters for both simulated and real data. The parameters are learned with a supervised loss, balanced between real data simulated ones, and refined with a CRF. This approach cannot be used for detecting dynamic objects and using only information about the road layout is not enough for trajectory planning as other agents' positions is of paramount importance.

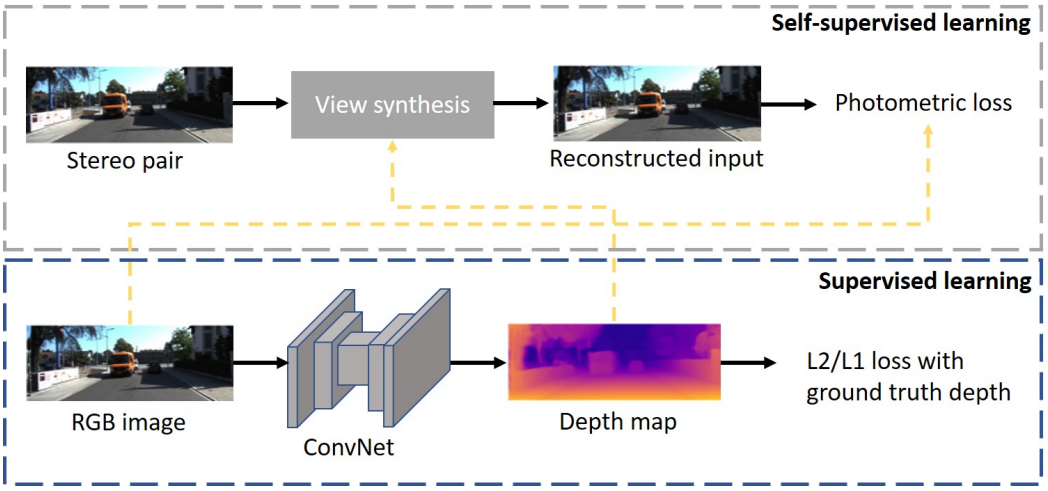


FIGURE 2.5: Self-supervision strategies for generating disparity maps without ground-truth depth. The training signal is either a stereo pair of images (Godard et al., 2019) or a sequence of monocular images with pose estimation (Garg et al., 2016).

2.5 Monocular depth estimation with CNNs

Depth maps can be acquired with a stereo setup but the acquired maps are sparse and contain few measurements at far distance. Traditional monocular depth estimation methods required handcrafted features such as low-level segmentation or contours. Supervised CNNs can be used to estimate dense depth maps from raw pixels but require a large amount of labeled data which is very costly to acquire especially for outdoor environments. Epipolar geometry have then been leveraged to circumvent the need for a large amount of labelled data and depth maps would be learned using self-supervised learning, see Figure 2.5.

2.5.1 Supervised depth estimation

Among the first works to use CNN for monocular depth estimation, Eigen and Fergus (2015) introduced a two-scale convolutional network that outputs depth maps, surface normals and semantic masks. CRFs were also used for depth prediction networks to refine the output map (Bo Li et al., 2015). Depth

prediction is usually treated as a regression problem but recent convolutional networks have successfully changed the problem to a classification one (Cao et al., 2018) or quantized ordinal regression (Fu et al., 2018). Residual learning has been successfully used for image classification (He et al., 2016) which inspired the work of Laina et al. (2016) that takes advantage of residual connections to build a deeper and more accurate network. The reverse Huber loss or BerHu loss for depth supervision have also been introduced in this work and improves the results when compared to regular L2 loss. Networks trained on a certain dataset obtained with a certain camera to estimate depth tend to not generalize well when applied on images shot from another camera. To cope with this issue, Facil et al. (2019) have introduced a new type of convolution referred to as Cam-Convs that takes the camera intrinsics into account by concatenating them the feature maps. Achieving state-of-the-art performance on the KITTI eigen-split benchmark, BTS (Lee et al., 2019) is an encoder-decoder network that takes advantage of recent advances in semantic segmentation and leverages ASPP for depth prediction and and uses a new local planar guidance layer. During the decoding phase and based on a local planar assumption, this new layer defines an explicit relation between each internal output and the final output depth map. Efforts have also been made to make the depth estimation networks more suitable for mobile platform applications with limited computation resources. FastDepth (Wofk, Diana and Ma, Fangchang and Yang, Tien-Ju and Karaman, Sertac and Sze, Vivienne, 2019) is a lightweight network with mobilenet (Howard et al., 2017) as the encoder, separable depth-wise convolution in the decoder and a network pruning strategy.

2.5.2 Self-supervised depth estimation

Building on Xie et al. (2016), epipolar geometry constraints were leveraged in Monodepth (Godard et al., 2017) to estimate disparity maps as an intermediate output in an image reconstruction network. It consists in a CNN that takes the left image as input, estimates the disparity map and performs a reconstruction of the right image using the predicted disparity and a bilinear

sampler. Two disparity maps are produced, left to right and right to left, and a left-right consistency term is added to the SSIM (Zhou Wang et al., 2004) reconstruction loss function. A similar approach is adopted by Garg et al. (2016) with the difference that the image reconstruction is not done end-to-end making the optimisation of the network's parameters more challenging. While these networks require only monocular images for testing, they still rely on stereo pairs of images during training. Among the first works to alleviate this burden, Zhou et al. (2017) introduced a self-supervised depth estimation network trained on monocular videos. Instead of reconstructing the target frame from the second view of a stereo pair, here the network reconstructs it from the previous and next frame in the video sequence. In a stereo training setup, the relative pose between the source and target image is known which is not the case in the monocular case. A separate pose estimation network is then trained to estimate the poses between the consecutive frames and constrains the depth estimation network. The view synthesis formulation implies that there is no dynamic objects in the scene and no occlusion/disocclusion between the source and target frames. To enforce these conditions, another network is trained to output an explainability mask that indicates the pixels where the view reconstruction should be operated. Monodepth2 (Godard et al., 2019) introduced a binary mask that filters out the pixels that do not change appearance between consecutive frames. They also introduce a loss that deals with the occlusions/disocclusions in the sequence of consecutive frames by taking the minimum instead of the average of the reprojection errors between the target frame and the previous/next frames which matches objects that are visible in both views. Training on monocular videos allows to estimate the depth and pose up to a scale and require a ground-truth LiDAR to scale the predicted disparity maps. More recently, Guizilini et al. (2020) introduced a new loss that considers the velocity of the camera during training which allows to output scale-aware depth maps.

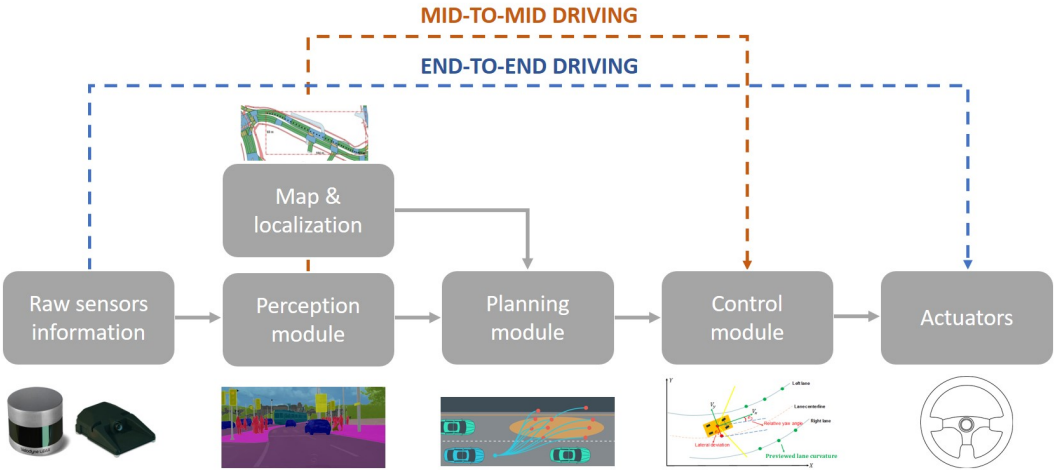


FIGURE 2.6: The autonomous driving pipeline is generally divided in different modular components. End-to-end and mid-to-mid driving approaches are popular alternative approaches that propose to bypass all or some of the modules and replace them with a single neural network.

2.6 Driving with imitation learning

Autonomous robots systems are usually composed of carefully designed modules or building blocks between the raw sensors data and the final decisions provided to the actuators. These modules can either be data-driven algorithms like deep neural network or hand-engineered rule-based algorithms. This pipeline design has the benefit of being highly interpretable and modular but can also lead to an accumulation of errors along the pipeline and requires substantial human effort. In this section, we present approaches that replace several or all modules using imitation learning and neural networks, Figure 2.6. Imitation learning consists in reproducing a desired behavior based on expert demonstrations and can be divided in two main branches: behavior cloning and Inverse Reinforcement Learning.

2.6.1 Behavior cloning

Behavior cloning is a branch of imitation learning that consists in learning a policy that reproduces the desired behavior by learning a direct mapping from states to actions.

2.6.1.1 Direct perception end-to-end driving

End-to-end driving neural networks match the behavior cloning definition and are a fully data-driven approach that takes raw sensor data as input and outputs steering wheel angles for example. Autonomous Land Vehicle In a Neural Network (ALVINN) (Pomerleau, 1988) was a pioneering work on end-to-end driving that used behavior cloning and a neural network to accomplish lane following. More recently, Bojarski et al. (2016) introduced an end-to-end driving network that maps camera images to steering commands. A similar approach was adopted by Codevilla et al. (2017) with the difference that the steering commands were conditioned with high-level commands, e.g., turn left. Also adopting conditional imitation learning, Xiao et al. (2019) used multi-modal inputs and explored different fusion schemes. These direct-perception approaches are promising but lack interpretability and pose a safety problem. An inspectable intermediate representation would alleviate the security problem and provide an understanding of the outputs of the network.

2.6.1.2 Mediated-perception end-to-end driving

Another body of work incorporates an intermediate representation in the form of affordances (Chen et al., 2015a, Sauer et al., 2018), attention maps (Kim and Canny, 2017) or semantic segmentation masks (Li et al., 2018b, Mueller et al., 2018). These approaches address the main flaw of the end-to-end approach by making the results interpretable by a human examinator. Instead of learning an intermediate representation and learning motion from this representation, Xu et al. (2016) learned to forecast motion from a sequence of input images and learned the semantic segmentation of these input images as a side task invoking privileged learning. However, all these camera-based solutions reason in camera view whereas it seems to be more suitable to forecast motion in BEV where the size of objects does not depend on their position in the image. Alleviating the burden of manually designed planning cost functions (Paden et al., 2016), Zeng et al. (2019) introduced an end-to-end motion planner that takes a 3D point cloud and a HD map as input and predicts an intermediate interpretable representation in the form of 3D detections and their predicted

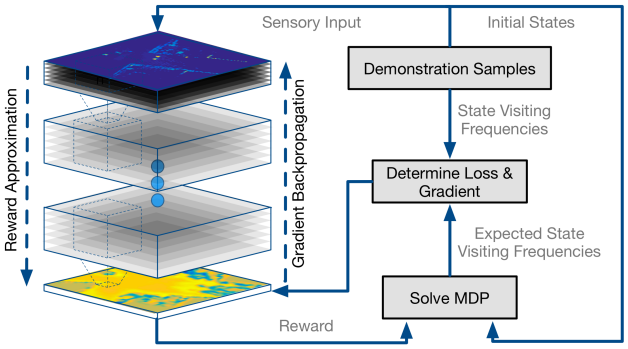


FIGURE 2.7: Inverse reinforcement learning can be used for trajectory planning where the planning cost map is replaced by the inferred reward map. Illustration originally from (Wulfmeier et al., 2016).

trajectories. Also leveraging 3D detections as intermediate representation for a driving policy, Wang et al. (2019) used an off-the-shelf network to predict 2D bounding boxes from a monocular image then trains a network to reproject the 2D detections into a 3D space.

2.6.1.3 Mid-to-end driving

ChauffeurNet (Bansal et al., 2019) adopts a mid-to-mid driving model that takes as input the BEV output of a perception module as input to predict a trajectory. Using a mid-level representation as input allows to augment the training data with synthetic worst case scenarios hence improving the performance of the network in a real-world scenario. Mid-to-mid driving is also adopted by the privileged learner of Chen et al. (2019), and by Srikanth et al. (2019) whose BEV semantic masks are computed using off-the-shelf networks and then used as inputs for a recurrent convolutional network to predict the trajectories of other agents in the scene. A mid-level representation is also adopted in other works that predict the motion of other traffic agents (Cui et al., 2019, Djuric et al., 2020).

2.6.2 Inverse reinforcement learning

A Markov decision process (MDP) is defined by $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathcal{T}, \gamma, \mathcal{R}\}$ where $\mathcal{S}$ is the set of possible states, $\mathcal{A}$ the set of actions, $\mathcal{T}$ the transitions, γ a discount factor and $\mathcal{R}$ the reward function. Reinforcement Learning consists in finding the optimal policy by mixing a strategy of exploration and exploitation while querying a reward function and having a feedback signal from the environment. When instead of having access to the reward function, a set of expert demonstrations are available, Inverse Reinforcement Learning consists in trying to recover the reward function from these demonstrations. Initially developed in the paradigm of maximum entropy (Ziebart et al., 2008), the deep implementation of inverse reinforcement learning was introduced by Wulfmeier et al. (2016) for path planning where the planning cost function corresponds to the inferred reward map outputted by a deep neural network, see Figure 2.7. Similar approaches were adopted by Zhang et al. (2018) and Deo and Trivedi (2020).

2.7 Conclusion

The technological breakthroughs made possible by deep convolutional neural networks have revolutionized computer vision, in doing so, the perception systems of mobile robotic platforms. Self-driving vehicles have specifically benefited from this revolution as their perception capabilities have reached a level of performance that can support the high expectations of the market. Specifically, the accuracy of semantic segmentation has been significantly improved thanks to deeper fully convolutional networks and methods to enlarge the spatial context used for processing the input images and extracts semantic information from them. This has also been made possible by the availability of public annotated datasets that ave grown steadily over the years, allowing a more thorough evaluation of the neural networks developed. Efforts have also been made to bridge the gap between camera-based approaches and LiDAR-based ones on 3D perception tasks such as 3D object detection and more recently semantic segmentation in Bird-Eye-View (BEV).

Early monocular approaches to scene understanding in BEV suffered from the lack of annotated data in BEV, but the release of datasets with raster images of HD maps and 3D bounding boxes has enabled the development of promising approaches. However, it should be noted that at the time of writing, there is not public benchmark for this task, making it difficult to compare performance. Monocular depth estimation is another 3D perception task that has greatly benefited from the advent of deep learning. Supervised learning provides impressive results, but ground truth remains relatively quite expensive to acquire. To alleviate this burden, self-supervised learning approaches have been successfully used, first using epipolar geometry and stereo image pairs as the learning signal and then extended to use only monocular image sequences.

Deep learning not only benefited perception, but also provided fertile ground for other robotic fields such as motion prediction and planning. As a result, end-to-end driving networks have also been explored, but the lack of interpretability has hampered their use. This has led to approaches that adopt the end-to-end driving paradigm but with an intermediate semantic representation of the world. Although these approaches are still reserved for research demonstrators, the availability of increasingly realistic driving simulators facilitates the evaluation of the driving capabilities of these networks.

Chapter 3

Improving semantic segmentation in urban scenes using cartographic and depth information

3.1 Introduction

Urban scenes are very challenging environments for autonomous driving because of their complex dynamics. However, the location of certain objects in these scenes is quite predictable: for example it is impossible to have a building in the middle of a road and cars are more likely to be found on the road rather than the sidewalk. The intuition of using location information in the form of a high definition digital map in perception modules seems reasonable. In this chapter, we argue that adding spatial context in the form of cartographic and depth information to a semantic segmentation neural network can improve its accuracy for some important class of objects like vehicles, pedestrians or traffic signs.

In order for a neural network to take advantage of some cartographic or depth information, a dataset containing semantic segmentation labels with

their corresponding cartographic information is necessary. In publicly available datasets for semantic segmentation, GPS data is available but we lack accurate cartographic information. For this reason, we chose to work on synthetic data produced by CARLA autonomous driving simulator (Dosovitskiy et al., 2017b) which provides a complete platform with semantic segmentation, depth, LIDAR and an accurate map of a virtual city. For each frame obtained from the simulator, we extract the part of the map that corresponds to a hundred meters ahead of the vehicle. The communication and computation overheads for dealing with such a limited area remain light while providing an almost exhaustive labeling of the road segments visible in the camera frame. These extracted portions of the map are presented in bird-eye-view . We apply an inverse perspective mapping to project the map portion in the camera plane, in order to ease matching with camera frames.

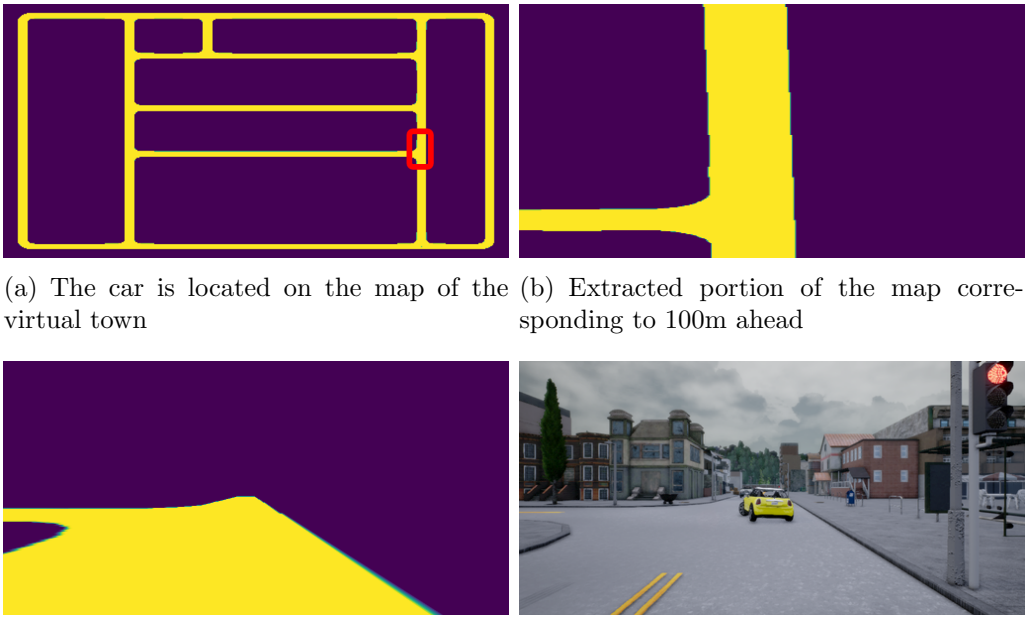
Incorporating a spatial context information into a semantic segmentation CNN can be done in several ways, either as an input or as a learning signal. We propose three neural networks designs for injecting cartographic and depth information in a Convolutional Neural Network (CNN):

- As a smoothness cost in a post-processing algorithm on top of the CNN
- As an additional learning signal along with the semantic segmentation ground-truth
- As an additional input to the CNN along with a monocular image

The three methods are evaluated and compared with a state-of-the-art CNN with respect to pixel-wise accuracy, mean intersection over union and intersection over union of some important classes.

3.2 Synthetic dataset

Real world semantic segmentation datasets are very expensive to annotate and large-scale public datasets do not provide precise cartographic information.



(a) The car is located on the map of the virtual town

(b) Extracted portion of the map corresponding to 100m ahead

(c) The map portion is projected in the camera plane

(d) Camera image

FIGURE 3.1: Inverse perspective mapping pipeline

Therefore, simulation was the designated solution for our problem. Among the available autonomous driving simulators (Best et al., 2017, Dosovitskiy et al., 2017b, Shah et al., 2018), we chose CARLA as image, depth, semantic and cartographic map are easy to access. The map of the virtual town is provided as a PNG image encoded in three layers with one layer giving information about roads, another one about intersections and the last one about lanes. The car can be positioned in the 2D image map through its world location, using the transformation (provided by the authors) from world coordinates to pixel coordinates in the 2D image map. All the methods investigated in this chapter rely on the fact that the cartographic map is projected in the camera plane. So first, for each camera frame extracted from the simulator, using the coordinates of the vehicle, a portion of the image map corresponding to one hundred meters ahead of the vehicle is extracted. Then, using the transformation between the bird-eye-view plane and the camera plane, the points of the bird's-eye view map are projected in the camera plane with inverse perspective mapping, see Figure 3.1. The transformation between the two planes is the following:

$$p \sim K[R|T]w \ , \tag{3.1}$$

where K is the intrinsic matrix of the camera, R the rotation matrix, T the translation matrix, $p = (u, v, 1)^T$ the coordinates in the camera plane and $w = (x, y, 1)^T$ the coordinates in the bird-eye-view plane. This projected map is homogeneous to a segmentation label. We assign labels to road, intersection and "other" pixels. Alignment of the projected map and the road in images is not perfect because of the vehicle dynamics that change the rotation and translation matrices that define the inverse perspective mapping. This could be compensated using the vehicle's inertial sensors but we chose not to do it to be more realistic.

3.3 Proposed methods

Our approach is based on the idea that the cartographic image is considered as a prior to inject in a CNN. Here we use Deeplab V2 (Chen et al., 2018a), a state-of-the-art network for pixel-wise semantic segmentation. We have explored three ways of incorporating the cartographic information in the network:

- The first method consists in adding the ground truth cartographic and depth maps as additional entries in the pairwise potentials of the CRF-RNN layer (Zheng et al., 2015) on top of a CNN (see Figure 3.2).
- The second method is similar to the previous one regarding the CRF except that the depth and cartographic maps that are used for the CRF are predicted by a multi-task network instead of being the ground truth (see Figure 3.3).
- The last method is a CNN with three encoder streams: image, depth and cartographic maps. The features extracted from the three streams are fused by element-wise operations and fed to a decoder that outputs a semantic segmentation map (see Figure 3.4).

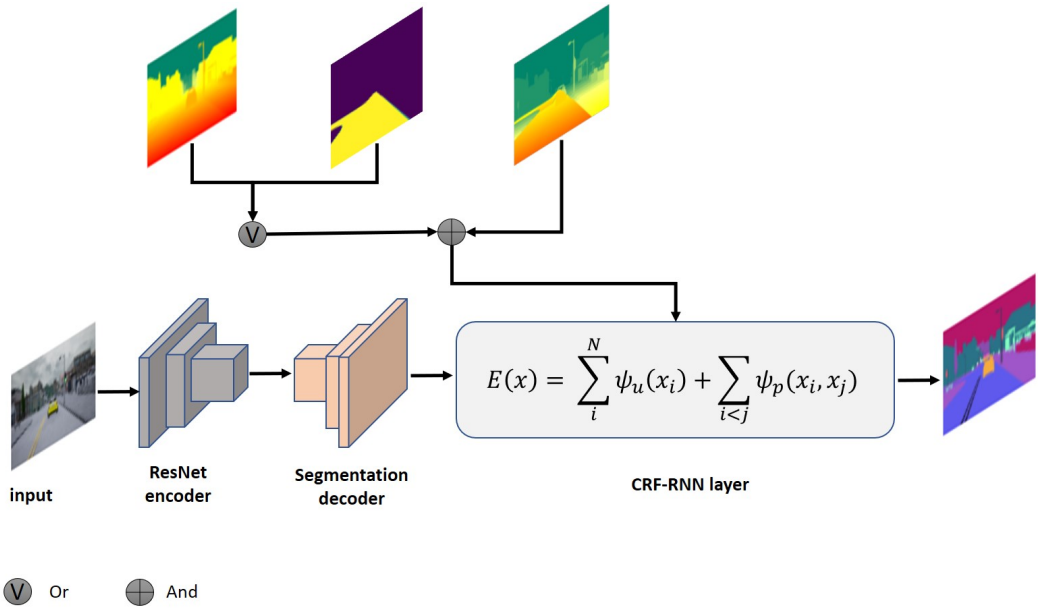


FIGURE 3.2: Method 1: Different combinations of pairwise potentials for improving semantic segmentation

3.3.1 Deeplab with CRF-RNN layer

We design our network using Deeplab V2 as our basis. This network takes advantage of dilated convolution to enlarge the size of the feature maps and the receptive field without increasing the number of parameters and a trous spatial pyramid pooling to aggregate multi scale information. We add an additional CRF-RNN layer at the output of the network to allow end-to-end training of the CRF with the CNN. The Gibbs energy of the fully connected CRF is the following:

$$E(x) = \sum_{i=1}^{N} \psi_u(x_i) + \sum_{i<j} \psi_p(x_i, x_j) \ . \tag{3.2}$$

The unary potential $\psi_u(x_i)$ is computed for each pixel by the CNN that produces a distribution over the label assignment x_i. The pairwise potentials are

a linear combination of Gaussian kernels:

$$\psi_p(x_i, x_j) = \mu_p(x_i, x_j) \underbrace{\sum_{m=1}^{K} w^{(m)} k^{(m)}(f_i, f_j)}_{k(f_i, f_j)} \ , \tag{3.3}$$

where $k^{(m)}$ is a Gaussian kernel, $w^{(m)}$ a weight, f_i a feature vector and μ a compatibility function that can be given by a simple Potts model $\mu_p(x_i, x_j) = [x_i \neq x_j]$.

The original paper on fully connected CRFs (Krähenbühl and Koltun, 2011) defines the following pairwise potentials that have been successfully applied since:

$$k(f_i, f_j) = w^{(1)} \underbrace{\exp(-\frac{|p_i - p_j|^2}{2\theta_\alpha^2} - \frac{|c_i - c_j|^2}{2\theta_\beta^2})}_{k^{(1)}(f_i, f_j)} + w^{(2)} \underbrace{\exp(-\frac{|p_i - p_j|^2}{2\theta_\gamma^2})}_{k^{(2)}(f_i, f_j)} \tag{3.4}$$

where c_i and p_i are respectively the vector of RGB values for pixel i and its position in the image, and $k^{(1)}$ and $k^{(2)}$ are the two kernels respectively measuring similarity with regard to appearance (pixels values and positions) and smoothness (pixels positions only).

We explore here different combinations of potentials. We consider adding the cartographic map and the depth map as separate appearance kernels and we also evaluate adding a kernel based on the aggregation of both depth and cartographic maps that we call focus map:

$$k^{focus}(f_i, f_j) = \exp(-\frac{|p_i - p_j|^2}{2\theta_\alpha^2} - \frac{|F_i - F_j|^2}{2\theta_\beta^2}) \ , \tag{3.5}$$

where:

$$F = M \circ D^{\circ -1} \ , \tag{3.6}$$

and F is the obtained 2D focus map, D the 2D depth map, M the 2D cartographic map (analogous to a segmentation map with label 3 for intersections, label 2 for the road and label 1 everywhere else) and $^{\circ -1}$ and $\circ$ respectively

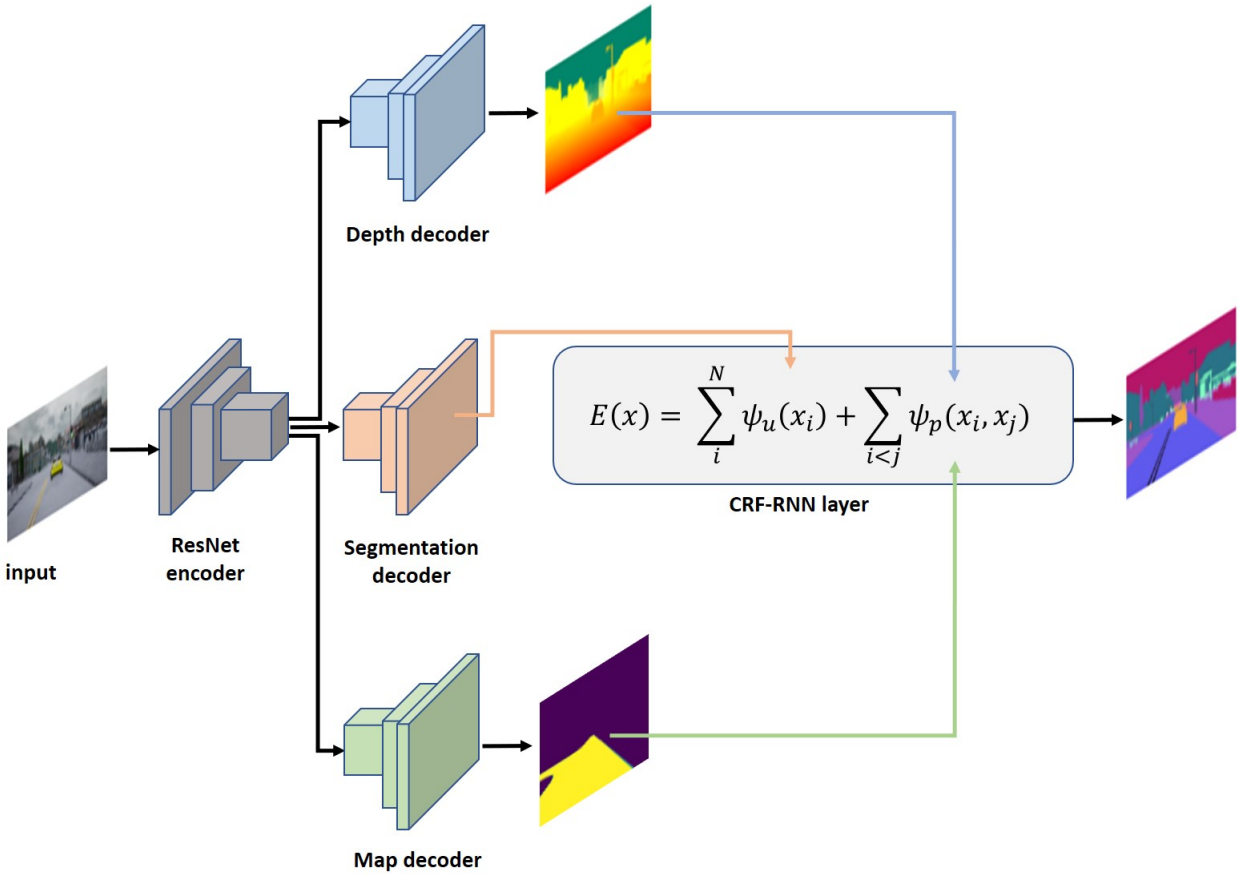


FIGURE 3.3: Method 2: Multi-task network for semantic segmentation, cartographic map and depth. The output of the semantic segmentation branch is used as a unary potential in the CRF layer and depth and map outputs are used as pairwise potentials in the CRF.

the Hadamard inverse and the Hadamard product. The weights $w^{(m)}$ of the different Gaussian kernels are learned end to end through back-propagation of the gradients in the network.

3.3.2 Multi-task network

To design the multi-task network, we built it on Deeplab to which we add two decoder branches, one for the depth estimation and one for the map estimation. The Deeplab network outputs feature maps eight times smaller than the original input image, so bilinear sampling is used for up-sampling. The map estimation being considered as another segmentation task, the decoder we use

is similar to the one used in the original Deeplab. For the depth estimation task, we added three up-convolution layers to resize the output to the input size and obtain a smoother depth estimation. To weight the different losses, we did manual tuning of the weights. The best results were obtained with the following configuration:

$$Loss_{total} = 10 * Loss_S + 0.1 * Loss_D + Loss_M \ , \tag{3.7}$$

where $Loss_S$, $Loss_D$ and $Loss_M$ are respectively the loss functions for semantic segmentation, depth estimation and map segmentation. This particular weighting was obtained empirically. The loss function for the semantic segmentation and cartographic map estimation tasks is the cross entropy loss. For the depth estimation task, we have chosen the BerHu loss which was shown to yield better results than the L1 loss (Laina et al., 2016). The reverse Huber loss is defined as:

$$B(x) = \begin{cases} |x| & if\ |x| \leq c \\ \frac{x^2+c^2}{2c} & if\ |x| > c \end{cases} \ . \tag{3.8}$$

Following Laina et al. (2016), for every gradient descent step where we compute $B(y - \hat{y})$, c is defined as $c = \frac{1}{5}\max_i(|\hat{y}_i - y_i|)$, where y_i and $\hat{y}_i$ are respectively the ground truth depth and predicted depth.

We add the CRF-RNN layer at the end of the semantic segmentation branch. The output of the depth estimation and cartographic map estimation branches are then used as additional pairwise potentials in the CRF, as in the (3.3).

3.3.3 Multi-encoder streams network

In this section, we use a neural network with multiple encoder streams. We build on Deeplab to which we add two encoder streams, one for the depth and another one for the map. The feature maps extracted from the three streams are fused to obtain a single feature map that is processed by the semantic segmentation decoder. The aggregation strategy that achieved the best results first uses element-wise multiplication of the image and map features, whose result is added, still element-wise, to the depth features.

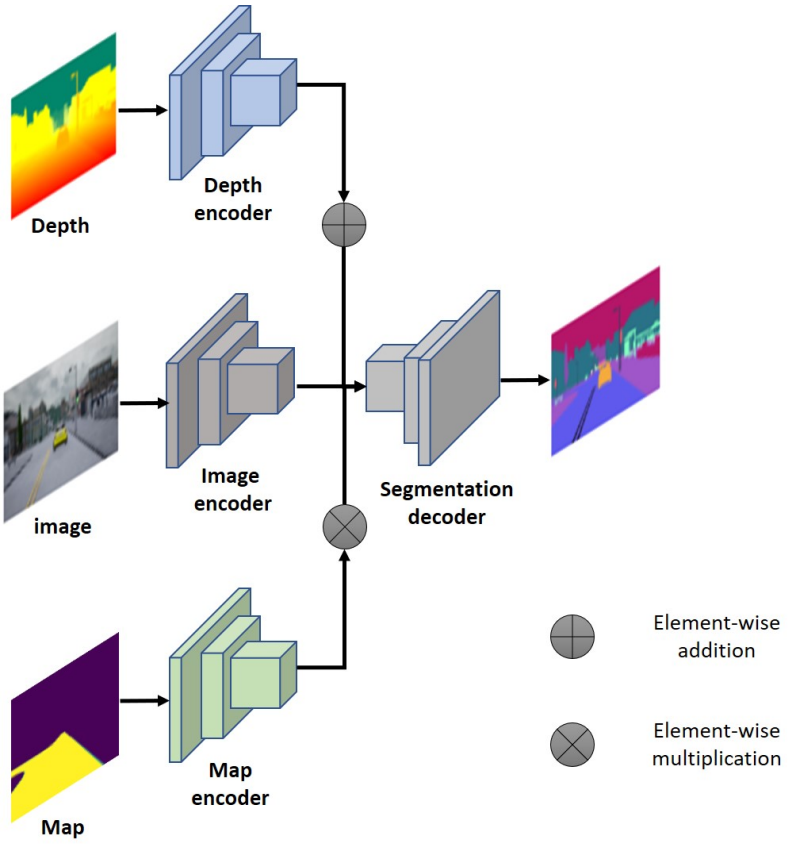


FIGURE 3.4: Method 3: Multi-encoder stream network. The features extracted from the depth and map encoders are fused with the features extracted from the RGB image, by element-wise multiplication for the map, and element-wise addition for the depth.

3.4 Experiments

We run the experiments on a set of two NVIDIA Titan X. We use ADAM optimizer (Kingma and Ba, 2014) with an initial learning rate of 10^{-4} and exponential decay. We compare the three networks with the original Deeplab without CRF. We compare with different CRF pairwise potentials and encoder streams to highlight the impact of the cartographic map. All networks are fine-tuned from the pretrained solution fitted on Cityscapes and Pascal VOC. All the networks containing the CRF-RNN layer were trained with a batch size of one because of implementation limits of the custom CRF-RNN layer, other networks were trained with a batch size of three, due to limited computing resources. In this work, Tensorflow has been used to design and train the

CNNs. For the CRF part of the code, the keras/tensorflow code given by Zheng et al. (2015) was used. The metric used to evaluate the performance of the network is the IoU, intersection over union, defined as:

$$IoU = \frac{TP}{TP + FP + FN} \ ,$$

where TP, FP and FN are respectively the true positive, false positive, and false negative pixel counts on the set of test images. This metric is evaluated for each class and the mean over all classes is computed as a general summary.

3.4.1 Dataset

We generated 4700 labeled images using the CARLA autonomous driving simulator. We ran 10 episodes of 470 frames each, each episode starting from a different position in the virtual world. We used virtual town number one for all our experiments. The number of vehicles in the world was fixed to 140 and the number of pedestrians to 120. For each episode, the weather was randomly selected among 4 possibilities: Clear noon, cloudy noon, clear morning, cloudy morning. During the simulation, the vehicle was controlled by the autopilot included in CARLA simulator code. The dataset was split in a training set composed of 4230 images and a test set of 470 images (one episode).

3.4.2 Deeplab with CRF-RNN layer

We have first fine-tuned a pretrained Deeplab V2 network on our dataset for 10 epochs. We train a Deeplab network with an additional CRF-RNN layer and compare the results with those of the regular Deeplab network that was fine-tuned on the training data. We tested different additional pairwise potentials. The results are shown in Table 3.1. We observe that the CRF with the cartographic map and depth map as separate additional potentials has the best results regarding the fences and traffic signs IoU improving the IoU by respectively 2.2% and 3.3% compared to the regular Deeplab and the focus

TABLE 3.1: IoUs (in %) of the first method with CRF processing, with different kernels, and compared to the raw Deeplab V2 without CRF processing. Best results are shown in bold. All networks are trained on 10 epochs.

	Deeplab	Deeplab + CRF			
		pos	pos + depth	pos + depth + map	pos + focus
mIoU	50.2	**52.4**	48.2	50.5	51.1
Pixel accuracy	88.3	**88.8**	86.5	87.8	88.6
Vehicle IoU	86.3	87.0	86.3	86.5	**87.2**
Pedestrians IoU	18.0	20.7	16.6	19.8	**21.9**
Traffic sign IoU	57.0	60.2	57.2	**60.3**	60.0
Fences IoU	37.1	38.7	34.7	**39.3**	**39.3**
Poles IoU	29.7	**30.2**	28.0	30.1	**30.2**
Road IoU	89.7	**90.7**	88.6	89.2	89.7
Road lines IoU	22.6	**39.4**	35.2	36.5	24.9

map additional potential has the best results for the vehicles, pedestrians and poles IoU improving the IoU by respectively 0.9%, 3.9% and 0.5%. This gives a hint on the fact that adding the cartographic information in the CRF adds an information on the location of some important classes like pedestrians, vehicles or traffic signs that are more likely to be found respectively on or outside the road. The Deeplab network with CRF and no additional potential has the best overall performance regarding the classes of interest and achieves the best mIoU and pixel wise accuracy and the best road and road lines IoUs.

3.4.3 Multi-task model

For the multi-task model, we have used the pretrained weights for the encoder and trained the network for ten epochs first. Once the network has learned to predict depth and cartographic map, the outputs of these two branches have been injected in a CRF-RNN layer at the end of the semantic segmentation branch and the network was trained for ten more epochs. Therefore, results with the different CRF pairwise additional potentials are compared with the regular Deeplab and the multi-task network without CRF both trained during 20 epochs. The results are shown in Table 3.2. The focus map, which is obtained by fusing the output of the depth and map branches of the network,

TABLE 3.2: IoUs (in %) of the second method. The multi-task networks, without CRF (trained on 20 epochs), the multi-task with CRFs (first trained on 10 epochs without CRF then on another 10 epochs with CRF), are compared to raw Deeplab V2. Best results are shown in bold

	Deeplab	Multi-Task network				
		raw	Multi-Task + CRF			
			pos	pos + depth	pos + depth + map	pos + focus
mIoU	52.1	52.3	53.2	52.4	51.9	**53.3**
Pixel accuracy	**88.8**	**88.8**	88.5	88.2	88.4	88.7
Vehicle IoU	**87.4**	87.3	87.2	87.0	87.1	86.9
Pedestrians IoU	**20.8**	18.9	19.0	20.0	19.3	18.9
Traffic sign IoU	61.7	63.3	62.2	**64.2**	63.4	63.5
Fences IoU	**39.4**	38.4	37.7	38.5	38.2	38.5
Poles IoU	**30.6**	30.1	30.4	29.7	30.0	30.2
Road IoU	89.9	89.9	**90.8**	89.5	89.8	89.3
Road lines IoU	22.4	23.4	**39.0**	29.6	22.5	35.4

TABLE 3.3: IoUs (in %) of the third method. The Multi-encoder networks evaluated in this table were trained on 10 epochs. Best results are shown in bold

	Deeplab	Multi-encoder networks			
		Image + depth	Image + focus	Image × map	Image × map + depth
mIoU	50.2	**51.6**	50.7	49.6	50.7
Pixel accuracy	88.3	88.5	88.6	88.55	**88.6**
Vehicle IoU	86.3	87.5	87.5	87.6	**87.9**
Pedestrians IoU	18.0	19.7	20.9	20.6	**22.0**
Traffic sign IoU	57.0	63.5	62.8	62.6	**66.0**
Fences IoU	37.1	**40.4**	39.8	38.3	39.9
Poles IoU	29.7	30.2	30.1	30.1	**30.6**
Road IoU	89.7	89.8	89.93	**90.0**	**90.0**
Road lines IoU	22.6	23.1	21.7	22.1	**23.5**

results in the highest mean IoU, outperforming the regular Deeplab by 1.2%. However, regarding the classes of interest, the regular Deeplab has the highest IoUs. When comparing the multi-task network with and without CRF, we don't observe improvements in the main classes of interest.

3.4.4 Multi-encoder model

For this approach, we have used the pretrained weights for the image branch and trained from scratch the depth and map branches. We have tested different aggregation strategies. The best results were obtained by multiplying the image features by the cartographic map features and then adding the depth features. The results are shown in Table 3.3. This method achieves the best results among the three tested ones and, in only ten epochs, it has better results for the important classes than the CRF, regular Deeplab and multi-task approaches in respectively ten, twenty and twenty epochs. The multi encoder network where we multiply by the map and add the depth improves the IoU of the pedestrians, vehicles and traffic signs classes by respectively 4%, 1.6% and 9%. Multiplying by the features of the cartographic map insures a stronger relationship with the features of the image and enforces the relation between the classes and their location in the map.

3.4.5 Discussion

This chapter shows some evidence that the predictability of some objects' location in a road scene can help improving the semantic segmentation and presents encouraging results regarding the use of a cartographic information as an image in CNNs. We hope that this line of works will encourage the release of publicly available real world datasets with synchronous semantic segmentation labels and precise cartographic information. The maps would need to have a high precision at centimeter-level. This kind of map is already popular for autonomous driving and is called High Definition (HD) map. In this chapter, the only pieces of information conveyed by the map are road boundaries and intersections. HD maps contain richer information, which can potentially improve the accuracy of the segmentation further, especially road lines IoU. Whatever format these HD maps come in, it would be possible to rasterize them to use them as proposed in this work.

3.5 Conclusion

Location provides important information about a vehicle's surroundings. In this work, we investigated how to inject location information into a CNN pipeline, considering it as another image input into the network. We studied the possibility of adding the map image as a pairwise potential in a CRF, as an additional task in a multi-task model, and as an additional encoding branch. After comparison with the regular Deeplab network, we conclude that adding a map-based pairwise potential in the CRF can slightly improve the intersection over union of important classes but the regular CRF still has a better overall performance. Training a multi-task network with a cartographic map prediction branch fails to improve the performance regarding the most important classes. Finally, adding map information as an additional encoding branch of a segmentation network provides the best results, significantly improving the intersection over union of important classes such as vehicles, pedestrians or traffic signs.

Chapter 4

Disparity weighted loss for semantic segmentation of driving scenes

4.1 Introduction

Semantic segmentation has been drawing a lot of attention from computer vision and autonomous driving communities for many years because in addition to detecting key elements in the scene, it adds semantic information to the global scene understanding problem. Convolutional neural networks (CNN) have achieved impressive semantic segmentation results and replaced the classic computer vision methods. However, state-of-the-art CNNs rely on a very important number of parameters and this comes with the price of an important resource consumption that hinders their usability for real-time mobile robotics applications. Indeed, this computational burden makes these networks less convenient for an autonomous mobile robot where multiple cameras are often needed and resources and data bandwidth are limited. Recent works have focused on designing very efficient lightweight networks that can run with a sufficient frame rate on modern GPUs dedicated to autonomous driving. These networks have fewer parameters than their state-of-the-art counterparts and the direct consequence is an important drop in performance.

One of the issues in driving scene datasets is the imbalance between the labeled classes: pixel-wise, critical classes like pedestrians or cyclists are under-represented compared to the sky or buildings. In order to compensate this imbalance, the loss function of the CNNs is often weighted according to the frequency of the classes, under-represented classes having the biggest weights. However, class imbalance is not the only issue. Finely segmenting an object located far from the ego vehicle does not seem to be a necessary asset for an autonomous pilot system. A coarse segmentation or a bounding box in this situation could be sufficient, whereas having access to a fine semantic segmentation of close objects can be useful to have a better free space estimation.

In this chapter, a new weighting scheme, depicted in Figure 4.1, is proposed. Based on the assumption that objects that are close to the vehicle are more important than those located far away, this weighting scheme gives more weight to close objects in the training objective. This is accomplished by pixel-wise multiplying the cross-entropy loss by the disparity map for each training image. This weighting is applied to two lightweight networks, with different efficiency/performance trade-offs, that were designed for real-time autonomous driving. These networks are trained on CamVid and Cityscapes datasets and the disparity maps are obtained with an off-the-shelf unsupervised depth estimation network. Our weighting scheme does not increase the number of parameters of the network nor implies any additional manual labeling. It is evaluated on both the regular mean intersection over union (mIoU) and a close-range mIoU. Compared to the standard frequency weighting scheme, this new loss weighting improves the mIoU and the IoU of pertinent classes for autonomous driving, especially at close range.

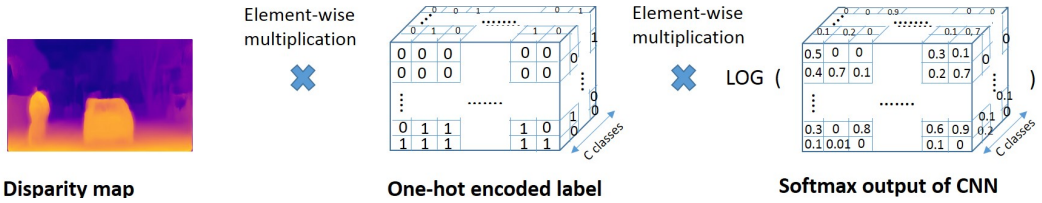
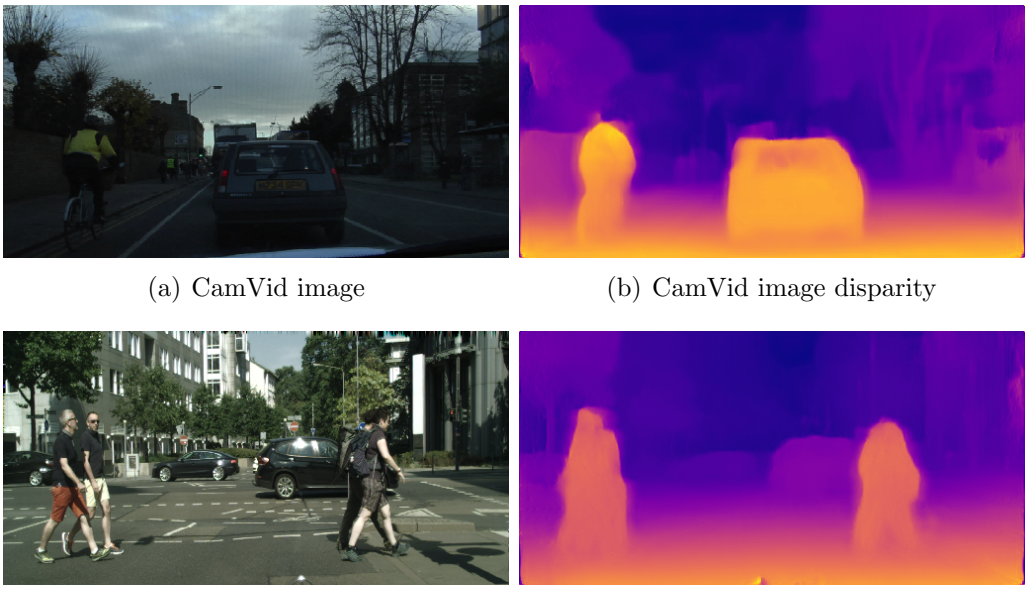


FIGURE 4.1: Disparity weighting: Each pixel in the cross-entropy loss function is weighted by its value in the disparity map.

4.2 Disparity weighting for semantic segmentation

4.2.1 Acquiring disparity maps



(a) CamVid image (b) CamVid image disparity

(c) Cityscapes image (d) Cityscapes image disparity

FIGURE 4.2: Disparity estimation with an off-the-shelf unsupervised CNN

Disparity maps can be acquired with a stereo setup but the acquired maps are sparse and contain few measurements at far distance. Given enough depth labels, supervised CNNs (Laina et al., 2016) can also be used to estimate dense depth maps but require a large amount of labeled data. Godard et al. (2017) have leveraged epipolar geometry constraints to estimate disparity maps as an intermediate output in an image reconstruction network. It consists in taking the left image as input, estimating the disparity map and performing a reconstruction of the right image using the disparity and a bi-linear sampler. Two disparity maps are produced, left to right and right to left, and a left-right consistency term is added to the reconstruction loss function. This CNN is not suitable for an application where a precise depth information is required but for our new disparity-based weighting scheme, only a magnitude estimation is

required. The disparity maps are precomputed for the whole dataset before training, thus not requiring any additional labelling efforts.

4.2.2 Loss weighting

The usual cross entropy loss is the following:

$$-\sum_{i=1}^{h}\sum_{j=1}^{w}\sum_{k=1}^{c} t_{ijk}\log(o_{ijk}) \ ,$$

where h, w are the dimensions of input images, c is the number of classes, and o and t are the (h, w, c) tensors corresponding respectively to the softmax output of the CNN and to the one-hot encoded segmentation labels. To remedy class imbalance, the widely used technique is median class frequency balancing (Eigen and Fergus, 2015), that consists in weighting each pixel of class k by $\alpha_k = \text{median}(\{p_1, ..., p_c\})/p_k$ where p_k is the proportion of pixels of class k in the dataset. Another weighting, also based on the frequency of the classes in the dataset, is:

$$\alpha_k = 1/\log(\beta + p_k) \ , \tag{4.1}$$

where β is a hyper-parameter (Paszke et al., 2016). This weighting is also used in Romera et al. (2018). The former weighting is referred to as FW and compared to disparity weighting in the experiments section. The disparity weighted cross-entropy is straightforward:

$$-\sum_{i=1}^{h}\sum_{j=1}^{w} d_{ij}\sum_{k=1}^{c} t_{ijk}\log(o_{ijk}) \ ,$$

where d is the (h, w) matrix of disparity map. Frequency based weightings put more weight on classes that are globally underrepresented in terms of surface. These weightings are not especially relevant for autonomous pilot systems, in the sense that a big truck located far from the vehicle may not be as important as a pedestrian close to the vehicle. Disparity weighting encourages the network to focus on the close-range objects.

TABLE 4.1: Class proportions (in %) in the Cityscapes train set. Most represented classes are shown in bold.

road	sidewalk	building	wall	fence	pole	traffic_light	traffic_sign	vegetation	terrain
25.3	5.8	**25.3**	0.7	1.0	1.4	0.2	0.6	**17.6**	1.1
sky	person	rider	car	truck	bus	train	motorcycle	bicycle	unlabeled
4.4	1.3	0.1	7.6	0.3	0.3	0.3	0.1	0.5	6.1

TABLE 4.2: Class proportions (in %) in the CamVid train set.

building	sky	road	tree	car	pavement
29.1	21.1	17.0	12.1	7.0	4.4
unlabeled	traffic sign	fence	pole	pedestrian	bicycle
4.2	1.5	1.4	1.2	0.8	0.3

4.3 Experiments

The disparity weighting is tested on the Pytorch implementations of ENET and ERFNET. These networks are trained using a single NVIDIA TITAN X with ADAM optimizer, momentum 0.9, a batch size of 4, initial learning rate of $5e^{-4}$ and weight decay of $2e^{-4}$. Pretrained weights on Cityscapes dataset are used for all networks.

These networks are trained on two semantic segmentation datasets, CamVid and Cityscapes. CamVid contains 367 training samples and 233 validation samples. Cityscapes contains 2975 training samples and 500 validation samples. For both datasets, the additional disparity labels are obtained with an unsupervised disparity CNN trained on Cityscapes. Tables 4.1 and 4.2 report the raw proportions of pixels belonging to each class.

The evaluation metric for each class is the intersection over union IoU defined as following:

$$IoU = \frac{TP}{TP + FP + FN} \ ,$$

where TP, FP and FN are respectively the true positive, false positive, and false negative pixel counts on the set of test images. The mIoU is the mean of the individual classes IoUs. A close-range IoU is introduced: it consists in

filtering both the ground truth segmentation and the predicted segmentation with regard to the disparity. All pixels with a disparity value inferior to a defined threshold are ignored. The unsupervised depth estimation network outputs disparity values that are normalised by the image width: the depth values on each pixel can be retrieved by rescaling acccording to the image width. To obtain the value of the depth in meter, we use this relation relation between B, f and d respectively the baseline, the focal length of the camera and the disparity: $Depth = \frac{B \cdot f}{d}$. We arbitrarily consider a depth of 30 meters as being our close-range limit and use the corresponding disparity value as our threshold to filter the pixels. For each dataset and each efficient network, three weighting schemes are compared: No Weighting, Frequency Weighting and Disparity Weighting respectively referred to as NW, FW and DW. Here, frequency weighting corresponds to the weighting introduced in Paszke et al. (2016), see equ. 4.1, with $\beta = 1.02$.

The values in the result tables are averaged on 3 runs for each configuration.

4.3.1 Results on CamVid

Results for CamVid are presented in Table 4.3.

ENET ENET network with DW outperforms its FW counterpart on the mIoU by 1.1% but more importantly on important classes like pedestrians, cyclists, vehicles and traffic signs by respectively 6.4%, 2.7%, 0.4% and 3%. The network trained without any weighting has the worst mIoU and the worst IoU on most of the important classes. Another important effect of the disparity weighting is the decrease in the sky IoU: this class has always the smallest disparity value and has in consequence the smallest weight in the learning loss function.

ERFNET The same effects are observed with ERFNET DW topping the FW by 1.5% on the mIoU and respectively 4.4%, 1.4%, 1.3% and 1.4% on the

TABLE 4.3: Results on CamVid validation set. Best results are shown in bold. All networks are trained on 150 epochs.

	ENET			ERFNET		
	NW	FW	DW (ours)	NW	FW	DW (ours)
mIoU	51.6	52.6	**53.7**	66.6	68.1	**69.6**
Pedestrians	30.2	29.0	**35.4**	55.4	56.4	**60.8**
Cyclist	4.8	20.0	**22.7**	52.8	59.2	**60.6**
Vehicle	65.1	67.2	**67.6**	77.8	79.4	**80.7**
Traffic sign	25.6	25.1	**28.1**	44.6	44.7	**46.1**
Road	89.5	**89.7**	89.5	92.9	**94.2**	93.3
Fence	27.7	28.2	**31.9**	42.0	46.7	**51.4**
Pole	19.1	18.9	**19.6**	35.4	**38.4**	38.0
Building	**76.5**	73.6	75.7	82.3	81.7	**84.2**
Sky	89.1	**89.4**	82.5	**91.8**	91.4	89.4
Pavement	71.8	71.4	**72.2**	80.4	81.9	**82.0**
Tree	**67.8**	66.6	65.5	75.9	76.2	**77.1**

IoUs of pedestrians, cyclists, vehicles and traffic signs. We also observe the decrease on the sky IoU by a lesser margin.

4.3.2 Results on Cityscapes

Results for Cityscapes dataset are presented in Table 4.4. Both ENET and ERFNET networks with disparity weighting improve on the mIoU by respectively 0.5% and 0.3%. However, the disparity weighting does not improve the IoU of all important classes.

ENET We observe the effect of the disparity weighting on the road, sidewalk, bicycle and pedestrians IoUs with respective improvements of 1.7%, 4.7%, 3.5% and 4.2% compared to FW. However FW outperforms DW on important classes like rider, car, bus, train and motorcycle.

TABLE 4.4: Results on Cityscapes validation set. Best results are shown in bold. All networks are trained on 150 epochs.

	ENET			ERFNET		
	NW	FW	DW (ours)	NW	FW	DW (ours)
mIoU	50.4	51.2	**51.7**	66.6	70.5	**70.8**
Road	93.7	93.1	**94.8**	96.5	96.6	**97.1**
Sidewalk	68.7	67.1	**71.8**	80.3	80.6	**82.3**
Building	84.6	83.4	**84.7**	89.9	90.1	**90.6**
Wall	**26.3**	24.0	24.7	44.4	51.5	**52.8**
Fence	29.5	30.8	**33.3**	50.2	54.4	**55.2**
Pole	**39.2**	38.0	**39.2**	57.5	59.1	**59.2**
Traffic light	20.7	**23.9**	20.5	57.1	**60.4**	59.3
Traffic sign	**38.5**	33.8	37.3	68.1	**71.4**	71.2
Vegetation	81.1	85.5	**86.8**	90.7	91.0	**91.1**
Terrain	42.7	37.6	**44.1**	58.4	60.4	**61.9**
Sky	**87.1**	86.2	86.6	**93.3**	91.8	93.2
Pedestrians	56.3	53.6	**57.8**	72.5	75.1	**76.1**
Rider	21.7	**27.3**	25.4	46.9	**53.3**	53.1
Car	86.0	**86.5**	86.4	91.5	92.5	**92.9**
Bus	45.8	**54.2**	51.1	57.1	**74.5**	74.3
Train	23.0	**29.2**	27.8	46.5	58.0	**60.9**
Motorcycle	10.0	**21.6**	13.8	35.5	**44.3**	43.5
Bicycle	53.5	51.5	**55.0**	66.2	69.7	**70.2**

ERFNET DW obtains the best IoUs for road, pedestrians, trains, bicycle, and cars with respective improvements of 0.5%, 1% , 2.9% , 0.5% and 0.4%. Overall, ERFNET achieves better results than ENET when weighted with the disparity map.

Close range evaluation Given that DW puts more weight on closer objects, these two networks are evaluated with the close range IoU to verify their emphasis on close objects, see Table 4.5. ENET DW obtains the best IoU on most of the important classes with the exception of bus, train and motorcycle.

ERFNET obtains also the best IoU results except for motorcycle and traffic light. In close range, both networks perform better with our DW than with other weighting schemes on the regular IoU.

Figures 4.3, 4.4, and 4.5 show the evolution of the IoU with respect to the depth threshold for some important classes. The presented results are for ERFNET trained on Cityscapes for both frequency and disparity weightings (best of the 3 runs for each for each weighting). In this experiment, all pixels with a depth value above the specified thresholds are ignored in the IoU computation. The depth values are rough estimations as the unsupervised network is not precise. In these figures, we first observe that for some classes IoUs improve as the depth threshold decreases and for other classes IoUs increase then decrease when getting too close to the vehicle. This decrease can be explained by the fact that some classes are not very well represented in close range during training: road or cars are represented at all ranges but this is not the case for the bus class for example. We also clearly observe the effect of disparity weighting when getting closer to the ego vehicle. The IoUs with a 60 meters threshold are equivalent for both weightings or slightly better for frequency weighting but when the threshold is at 30 meters or less, disparity weighting has consistently a better IoU (by a significant margin for bus class, 20%).

TABLE 4.5: Close-range results on Cityscapes validation set. Best results are shown in bold. All networks are trained on 150 epochs.

	ENET			ERFNET		
	NW	FW	DW (ours)	NW	FW	DW (ours)
mIoU CR	52.2	52.4	**53.9**	67.6	71.2	**72.5**
Road	95.3	94.1	**96.3**	98.8	98.8	**99.0**
Sidewalk	72.7	70.8	**76.4**	81	81.3	**83.1**
Fence	36.7	39.3	**42.3**	50.1	**54.8**	52.5
Pole	48.7	47.3	**49.5**	66.6	67.2	**68.3**
Traffic light	25.4	23.8	**26.8**	67.0	**69.2**	68.9
Traffic sign	49.1	40.0	**50.1**	75.9	**78.2**	**78.2**
Pedestrians	59.7	59.7	**61.7**	76.6	79.0	**80.3**
Rider	28.9	33.0	**33.2**	48.6	53.6	**54.2**
Car	90.3	89.9	**91.3**	93.7	94.2	**94.7**
Bus	49.0	**52.3**	51.9	61.1	75.3	**77.7**
Train	**25.9**	24.9	19.2	50.9	63.3	**71.6**
Motorcycle	9.7	**26**	11.7	37.8	**47.1**	46.5
Bicycle	61.0	57.6	**63.7**	70.1	73.5	**74.2**

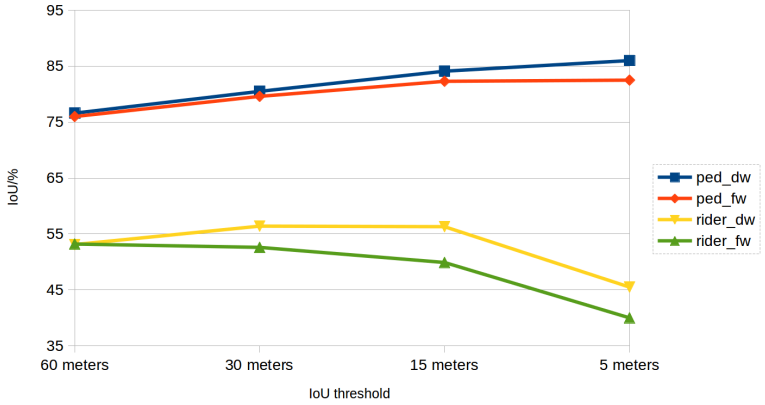


FIGURE 4.3: IoU of pedestrians and riders classes for different depth thresholds. All pixels whose depth is above the threshold are ignored in the IoU computation.

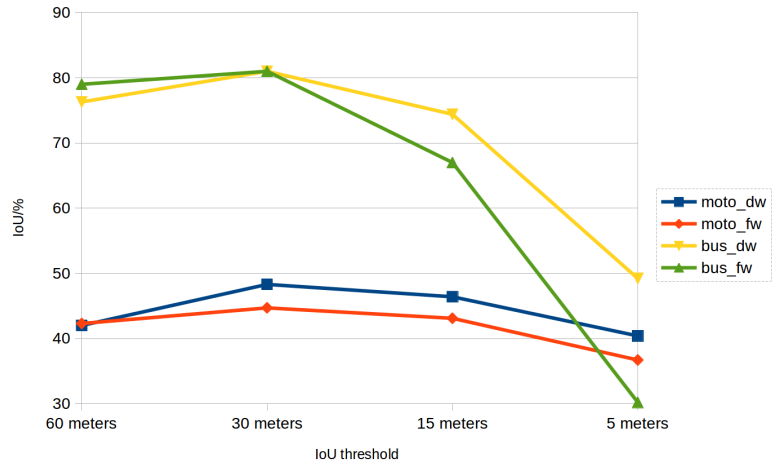


FIGURE 4.4: IoU of motorcycle and bus classes with different depth thresholds. All pixels whose depth is above the threshold are ignored in the IoU computation.

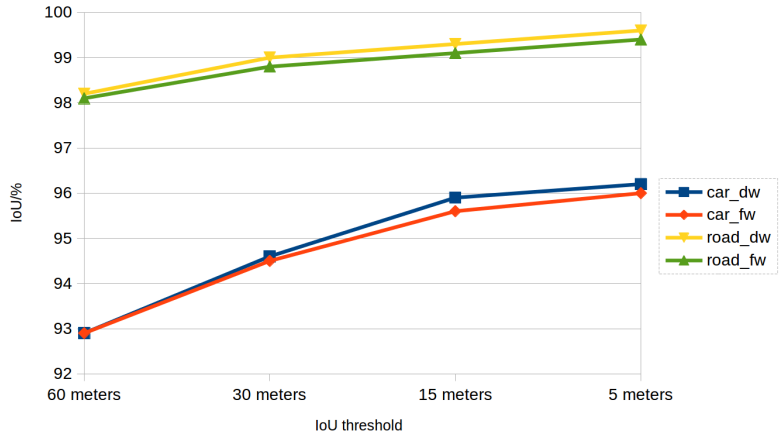


FIGURE 4.5: IoU of road and car classes with different depth thresholds. All pixels whose depth is above the threshold are ignored in the IoU computation.

4.3.3 Qualitative results

Figure 4.6 presents different situations where the close range effect of the disparity weighting is observed. On these figures we observe that DW produces better road segmentation in close range which can avoid dangerous situations like in the sixth image where a portion of sidewalk is detected in the middle of the road. We also observe that DW ensures better pedestrian delineation

and sometimes detects objects that are not detected by FW: on the first, the second and the sixth image the rider in front of the vehicle is detected by FW but not the bicycle he rides. On image 3 and 5 we observe that DW assigns the correct class to the truck and the bus but FW mistakes some part of these objects with other similar classes. A better segmentation in close range avoids dangerous situations where the vehicle could brake because of a false alarm or hit an obstacle that was not detected.

4.3.4 Discussion

Experiments show improving but varying results depending on the dataset, the range of the metric and the chosen network. A first explanation of these variations is given in Tables 4.1 and 4.2. The proportion of pixels corresponding to the class sky in CamVid dataset is roughly 5 times greater than in Cityscapes, which can explain why disparity weighting works better on the former. A large part of the images in CamVid corresponds to the sky and this part is ignored in the loss function which helps the network focus on the other more important classes. The performance increase is not very important because these efficient networks naturally segment better closer objects because close objects are big in camera view. Nevertheless, the disparity weighting still has interesting properties and clearly observable effects on close-range segmentation.

4.4 Conclusion

In this chapter, a new loss weighting scheme is introduced for semantic segmentation of driving scene with lightweight CNNs. This weighting consists in multiplying each of the pixels of the training images by their disparity value. The disparity maps for each image are precomputed prior to training with an off-the-shelf unsupervised CNN. Intersection over union metric is improved on CamVid and Cityscapes datasets with better results on the former. This is partly explained by the proportion of classes in both datasets. CamVid

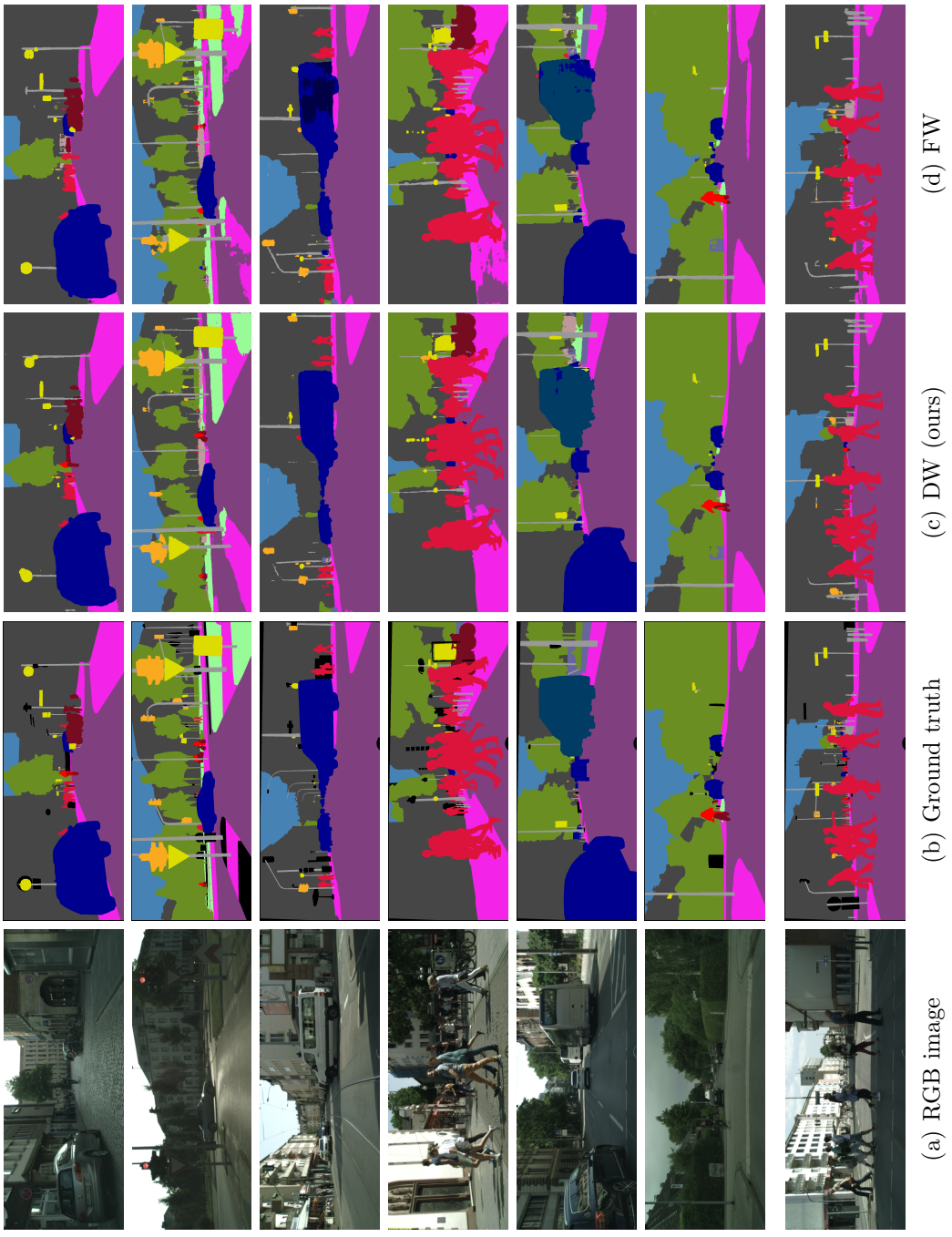


FIGURE 4.6: Qualitative results on Cityscapes with ERFNET: disparity weighing (DW) behaves better than frequency weighing (FW) in close range

sky class being over-represented, it is ignored because of its very low disparity, hence putting more emphasis on other classes. Further investigation on Cityscapes dataset show even more improvements when evaluating with a close range IoU.

Without any additional labelling effort nor computation burden, disparity weighting improves semantic segmentation performance. A similar approach with dense optical flow could be tested with an unsupervised flow estimation network. Instead of focusing on nearby objects, it could do so on moving objects, which would also make sense for autonomous driving.

Chapter 5

FlatMobileNet: Bird-Eye-View semantic masks from a monocular camera

5.1 Introduction

Perception systems in autonomous mobile platforms are composed of modular computer vision tasks such as semantic segmentation, depth estimation or object detection. These different outputs are often merged in a common Bird-Eye-View (BEV) semantic representation of the driving scene to be used as input by the navigation and planning building blocks. The BEV representation space is preferred to the Camera-Projective-View (CPV) space because the scale in this space is homogeneous, in the sense that the size of the represented objects is invariant to their position, in particular to their distance to the sensors. LiDAR has been vastly adopted in autonomous mobile platforms because it is a 3D sensor and it inherently allows to precisely detect objects in the 3D space. However, cameras still have a better resolution, a better framerate, provide colour information while being by far cheaper sensors. Recent approaches have then tackled the challenging problem of estimating this top-down semantic representation of the driving scene from a monocular image only with the most successful ones applying a geometric transform to

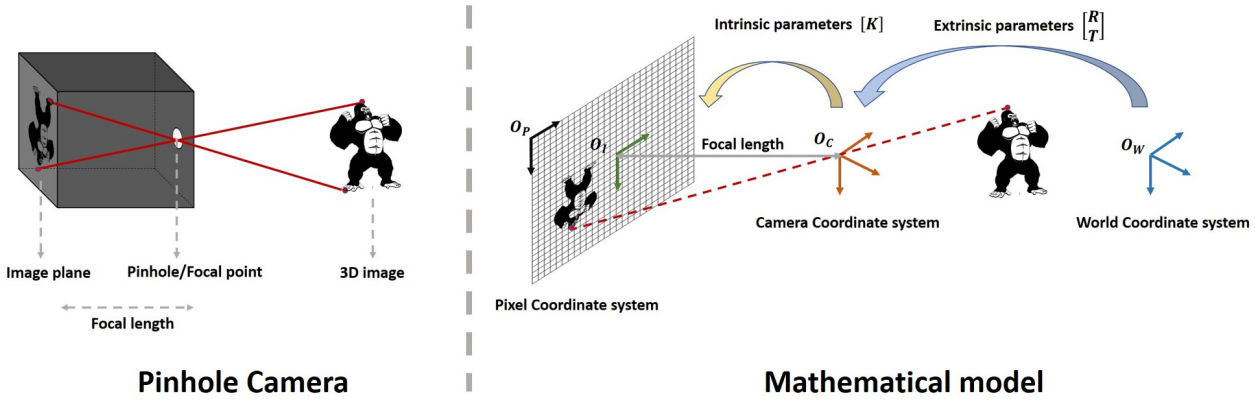


FIGURE 5.1: Camera pinhole model

project the learned features from the camera-projective-view to the bird-eye view and then processing the features in this coordinate system. In this section, we argue that reasoning directly in the camera coordinate system can still be beneficial for close-range BEV scene understanding as closer objects appear bigger hence having more impact on a pixel-wise learning loss. To cope with the inherent poor far-range performance of such an approach, a mixture of experts framework is then adopted to combine the prediction of a short-sighted network that learns in camera-view with one with better far-range performance that learns in the BEV. We demonstrate state-of-the-art BEV semantic segmentation performance on nuScenes dataset and Carla simulator.

5.2 Theoretical framework

Before diving into the details of our approach, we first briefly introduce the camera pinhole framework on which our model is defined. We also give theoretical detail about the homography warping that is a key component in our approach to project feature maps from CPV to BEV.

5.2.1 Camera Pinhole model

A pinhole camera is a simple box with no lens and a small aperture called pinhole. The light rays from the exterior scene go through the pinhole and project an inverted image on the opposite side of the hole, see Figure 5.1 left side. The camera pinhole model is a mathematical formulation of this simple camera that allows to project 3D points in a world coordinate system to 2D points in a pixel coordinate system, see Figure 5.1 right side. This projective transform can be defined by the extrinsic and intrinsic parameters of the camera. The extrinsic parameters refer to the rotation matrix and translation vector of the camera coordinate system with regard to the world coordinate system. Therefore, given a point $(x, y, z, 1)^T$ in the world coordinate system, we can form its pixel coordinates $(u, v, 1)^T$ as follows:

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & u_0 \\ 0 & f & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{5.1}$$

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K \begin{bmatrix} R|T \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{5.2}$$

where:

- K is the 3×3 intrinsic camera matrix.
- f the focal length.
- (u_0, v_0) the principal point at the center of the image plane.
- R is the 3×3 rotation matrix.
- T is the translation vector.

5.2.2 Homographies

A homography is a projective transform that maps points from one plane to another and can be derived from Equ. (5.1):

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & u_0 \\ 0 & f & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} u' \\ v' \\ 0 \\ 1 \end{bmatrix} \tag{5.3}$$

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & u_0 \\ 0 & f & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & t_1 \\ r_{21} & r_{22} & t_2 \\ r_{31} & r_{32} & t_3 \end{bmatrix} \begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} \tag{5.4}$$

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} \tag{5.5}$$

$$P = HP' \tag{5.6}$$

where:

- P and P' are two points on different planes.
- H is an invertible 3×3 matrix formed by the intrinsic and extrinsic parameters that relate the two planes.

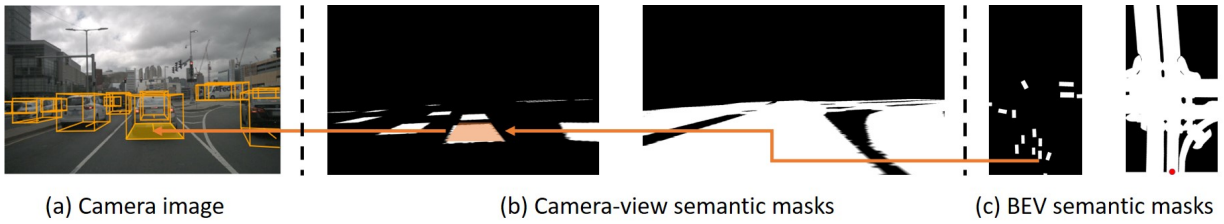


FIGURE 5.2: From camera images to BEV semantic masks. For each camera frame (a), a BEV road layout is extracted from a HD map raster as a binary semantic mask. A vehicle binary mask with the same scale and size is generated from the ground-truth 3D bounding boxes (c), where the red dot indicates the ego vehicle. These two masks are projected in camera view (b) using the homography between the two planes.

5.3 FlatMobile network: footprint segmentation

5.3.1 Data

Our method relies on the availability of 3D bounding box annotations and rasterized HD maps that come in the form of binary semantic masks and inform about the road layout. These two different annotations constitute the learning signal for our monocular BEV semantic segmentation neural network.

5.3.1.1 Ground truth Bird-Eye-View semantic maps

For each of the considered frames, a map portion of the available HD map raster is extracted with the ego vehicle being positioned at the bottom center of this map portion. This map portion is rotated according to the ego vehicle heading angle such that it always faces forward. Given the 3D position of each vehicle in the scene, it is possible to draw its ground-truth bounding boxes on a blank canvas aligned with the map portion, with the correct size and position. These two binary semantic masks, depicted in the right-hand side of Figure 5.2, can also be viewed as occupancy grid maps. These binary semantic masks are then projected to the camera-view using the homography

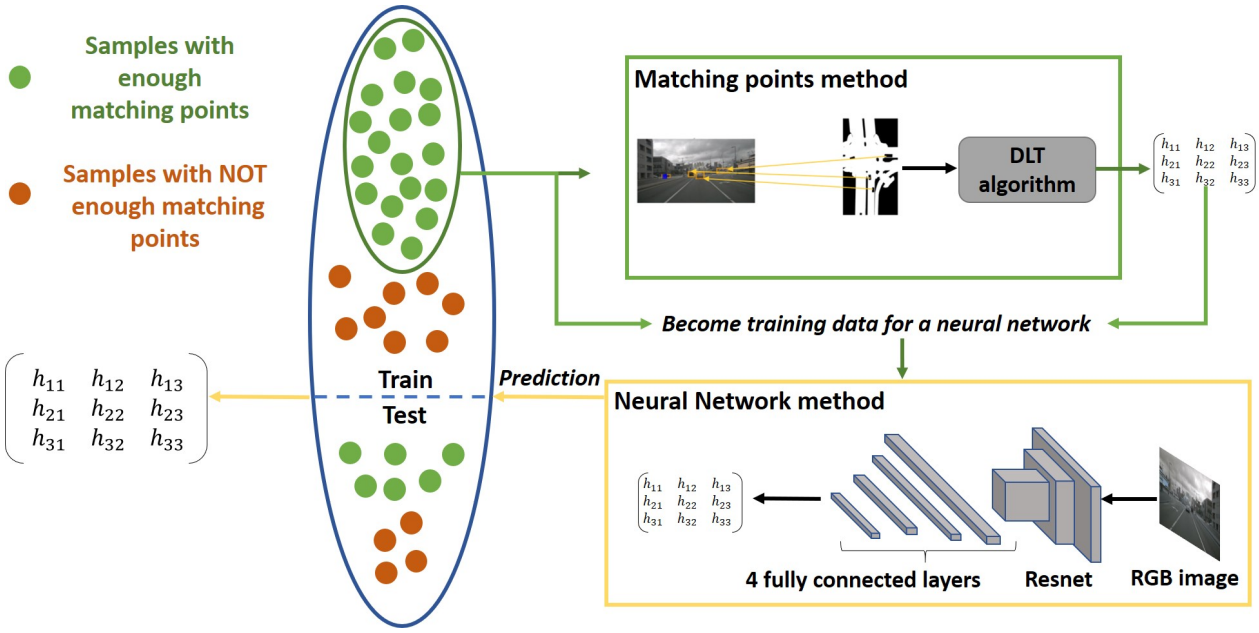


FIGURE 5.3: Homography estimation. For training images with enough matching points, the homography is computed with the matching method. These samples become training data for a neural network that learns to estimate the homography from RGB input images. This trained network is then used to pre-compute the homographies of all the samples in the dataset.

between the HD-map BEV plane and the CPV plane hence obtaining learning signals in both views.

5.3.1.2 Homography estimation

A key input of our end-to-end network is the homography matrix that maps a point in the camera plane $(u, v, 1)^T$ to a point in the HD-map BEV plane $(u', v', 1)^T$ as described in Section 5.2.2. Our framework relies on the availability of this homography matrix at each frame. We take advantage of the annotated 3D bounding boxes to get corresponding points in BEV and the camera planes. The pixel positions of the 3D bounding boxes "ground" face corners in the camera image are matched with their 3D position in the BEV plane and the homography matrix is obtained using the Direct Linear Transformation algorithm (Hartley and Zisserman, 2003). This algorithm requires at least 4 correspondences, and some samples don't contain enough matching

points, so we rely on a neural network to learn a mapping between the camera images and their corresponding homography matrices. This network is fitted on the subset of the training set that contain enough matching points, see Figure 5.3. The homography network is composed of a ResNet-18 encoder and 4 fully connected layers with the fourth layer outputting 9 values corresponding to the elements of the homography matrix. The homography matrix contains 4 rotational terms and 2 translation terms. The difference in magnitude between these terms must be taken into account during training. DeTone et al. (2016) circumvent this issue by predicting 4 matching points instead of the homography matrix elements as there is a one-to-one correspondence between the two representations; we simply rescale the rotational and translation terms with a constant factor such that all the elements of the homography have the same magnitude. The L2 loss function is then used as the training criterion.

5.3.2 Model formulation

The model presented in this chapter generates BEV Occupancy Grid Maps (OGMs) from monocular images. These OGMs' cells can take two states, occupied or free. Two OGMs are considered: one that gives an information about the road layout, and the other one about the vehicles in the observed scene. The considered OGMs being homologous to binary semantic masks, their estimation is formulated as the semantic segmentation of the road layout and the vehicles in the scene.

State of the art semantic segmentation CNNs come in an encoder-decoder configuration where the input and the output of the network have the same aspect ratio and are in the same geometric plane. The ultimate goal of our model is to output BEV semantic masks from a camera image. However, it is not straightforward to directly output BEV masks from a camera plane input as the receptive field of a pixel in the BEV output would not match the region of the image that is responsible for its processing. Hence, the semantic masks are first outputted in the CPV. ResNet-101 (He et al., 2016) is used as the encoder and Deeplab v3+ (Chen et al., 2018b) as the decoder. ResNet-101 encoder extracts rich features maps from the input image and Deeplab

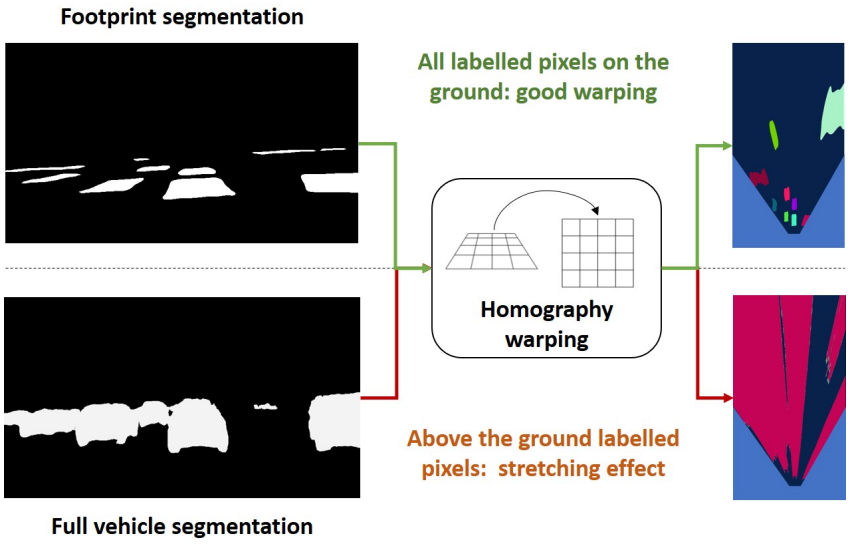


FIGURE 5.4: Footprint segmentation *vs.* object segmentation. Applying a planar homography to a full segmentation mask causes the pixels located above the ground plane to be stretched in BEV. Our footprint segmentation respects the flat world hypothesis implied by the homography hence avoiding the deformation caused by above the ground pixels.

v3+ decoder combines A trou Spatial Pyramid Pooling (He et al., 2014) and a convolutional decoder module to benefit from rich contextual information and better object boundaries delineation. Two decoding heads output the road layout and vehicles semantic masks in CPV. The final step of our model is to warp the semantic masks that were predicted in CPV to BEV using a perspective warping layer (Riba et al., 2018) along with the homography matrix between the CPV plane and the BEV plane.

However, vehicles are 3D objects and projecting regular semantic masks in BEV leads to the "stretching" effect observed in Figure 5.4. To cope with this deformation, we introduce the FlatMobile representation to warp the vehicles from CPV to BEV. It consists in segmenting the footprint of the vehicles, i.e., the "ground" face of their 3D bounding boxes in camera view. By doing so, only pixels located on the ground surface (assumed to be locally flat) are segmented and warped in BEV which respects the planar world hypothesis required by the homography. The homography for each sample is obtained

with the homography estimation network trained separately from the end-to-end network but on samples from the same training set. The output size of the decoding heads is s times smaller than the original camera image so the homography needs to be re-scaled as following:

$$H_s = S \cdot H \cdot S^{-1} \ , \text{ with } S = \begin{bmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 1 & 0 \end{bmatrix} \ , \tag{5.7}$$

where H_s is the re-scaled homography, H the original homography and s is the (fixed) scaling factor accounting for the ratio of image to decoder output size.

The overall architecture of the network is depicted in Figure 5.5.

5.3.3 Cost function and learning

We chose to predict separately a binary OGM for the vehicles and another one for the road layout, since our problem is best formalized as a multi-label classification problem rather than a multiclass classification problem. Indeed, each cell can belong to one of the four classes: {vehicle, drivable, vehicle and drivable, none}. The overall loss of the network, denoted $\mathcal{L}_{perception}$, is defined as the sum of two losses:

$$\mathcal{L}_{perception} = \mathcal{L}(\mathcal{O}_{road}, \mathcal{Y}_{road}) + \mathcal{L}(\mathcal{O}_{vehicle}, \mathcal{Y}_{vehicle}) \ , \tag{5.8}$$

where $\mathcal{L}(\mathcal{O}_{road}, \mathcal{Y}_{road})$ is the loss for the road layout, and $\mathcal{L}(\mathcal{O}_{vehicle}, \mathcal{Y}_{vehicle})$ is the loss for the vehicle occupancy, more precisely:

- $\mathcal{L}$ is the binary cross-entropy loss,
- $\mathcal{O}_{road}$ and $\mathcal{O}_{vehicle}$ the predicted road layout and vehicles masks,
- $\mathcal{Y}_{road}$ and $\mathcal{Y}_{vehicle}$ the ground-truth road layout and vehicles masks.

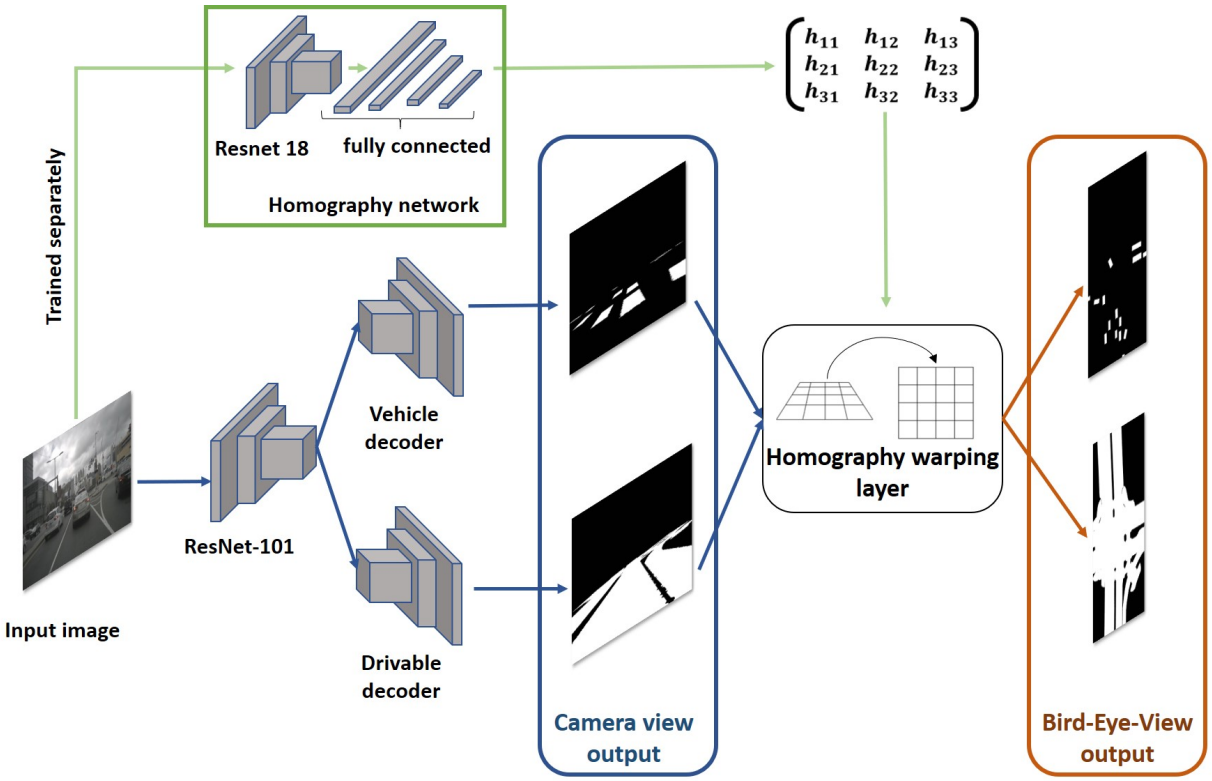


FIGURE 5.5: Overview of FlatMobile network. An encoder-decoder architecture takes a camera image as input and segments the road layout and the vehicles' footprints in the CPV. A homography warping layer along with the homography between the CPV and BEV are then leveraged to warp these masks to BEV.

5.3.4 Mixture Of View Experts (MOVE) model

5.3.4.1 Mixture of experts background

Mixture of experts is a divide-and-conquer strategy introduced by Jacobs et al. (1991), in which the input space is divided into regions which are each governed by an "expert" model that drives the decision in that region. The final decision is taken by combining the decisions made by the experts $f_1, \ldots, f_i, \ldots, f_N$, weighted by the convex combination computed by a gating network. The outputs of the gating network $g_1, \ldots, g_i, \ldots, g_N$ are positive and sum up to 1, so that the final prediction F for the input x is:

$$F(x) = \sum_{i=1}^{N} g_i(x) \cdot f_i(x) \tag{5.9}$$

The experts and the gating function are learned simultaneously. The strategy for combining experts will determine the behavior of the different experts. Comparing the average of the experts' predictions with the target will encourage cooperation while comparing individually each expert with the target will encourage competition:

$$Loss_{coop} = ||y - \sum_{i=1}^{N} g_i(x) \cdot f_i(x)||^2 \tag{5.10}$$

$$Loss_{comp} = \sum_{i=1}^{N} g_i(x) \cdot ||y - f_i(x)||^2 \tag{5.11}$$

Learning with a cooperation loss function will update the weights of each expert network based on the ensemble error rather than the error of each expert which makes the experts learn on the entire problem space. On the contrary, learning with a competition loss makes the experts specialize in their preferred sub-space with the gating network selecting which expert is more "suitable" for a specific input. In our experiments we empirically compare different combinations of cooperation and competition losses and chose the one that gives the best result.

5.3.4.2 Mixture of experts for OGM estimation

Learning the semantic masks in CPV and BEV can be seen as learning these masks with two experts, with each expert learning both the vehicles and road layout masks. In CPV, closer objects appear bigger hence having more impact on the learning loss so we can expect the CPV expert to have better close-range accuracy. On the other hand, in BEV the size of objects is invariant to their distance to the sensor so we can expect the BEV expert to have a better far-range performance than the CPV one. To make the most of the available ground truth in both CPV and BEV and allow each view expert to specialize in its preferred range, a Mixture Of View Experts (MOVE) approach is adopted, see Figure 5.6. The MOVE architecture is built on the FlatMobile network described in Figure 5.5 with an additional shallower OGM estimation network

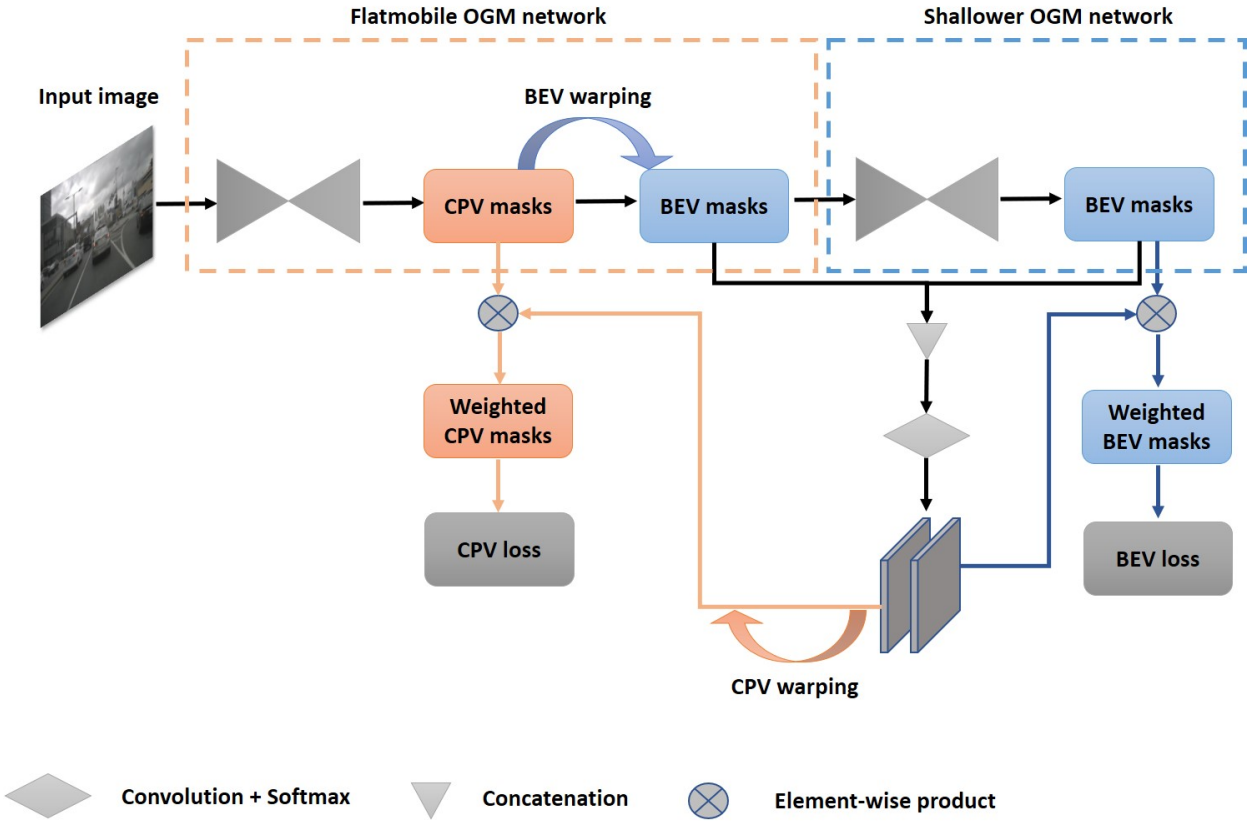


FIGURE 5.6: Mixture Of View Experts (MOVE) network

whose purpose is to learn the BEV masks directly in the BEV. The shallower OGM network is composed a ResNet-18 as encoder and two DeepLab-v3+ decoders for the road layout and vehicles masks prediction. In an end-to-end fashion, it takes the BEV output of the FlatMobile network and produces new masks in BEV. The gating network is composed of a single convolution layer with a softmax activation function. The BEV outputs (both road layout and vehicles) of the FlatMobile network and the shallow OGM network are concatenated along the channel dimension and fed to the gating network that produces a weight map $\mathcal{W}$; the CPV expert is weighted by $\mathcal{W}$ and the BEV expert by $1-\mathcal{W}$. The weight maps are obtained in the BEV space so the CPV weight map has to be warped back to CPV. The competition and cooperation losses for the vehicles and the road layout are defined as:

$$Loss_{comp} = \Psi(\mathcal{W}, H^{-1}) \cdot \mathcal{L}(\mathcal{O}_{cpv}, \mathcal{Y}_{cpv}) + (1 - \mathcal{W}) \cdot \mathcal{L}(\mathcal{O}_{bev}, \mathcal{Y}_{bev}) \qquad (5.12)$$

$$Loss_{coop} = \mathcal{L}(\mathcal{W} \cdot \Psi(\mathcal{O}_{cpv}, H) + (1 - \mathcal{W}) \cdot \mathcal{O}_{bev}, \mathcal{Y}_{bev}) \tag{5.13}$$

where:

- $\mathcal{L}$ is the softmax cross-entropy loss function
- Ψ the homography warping function and H the homography from CPV to BEV
- $\mathcal{O}_{cpv}$ and $\mathcal{O}_{bev}$ the output masks in CPV and BEV
- $\mathcal{Y}_{cpv}$ and $\mathcal{Y}_{bev}$ the ground-truth masks in CPV and BEV

The final loss is empirically defined as the following combination of the competition and cooperation losses:

$$Loss_{final} = 0.8 \cdot Loss_{comp} + 0.2 \cdot Loss_{coop} \tag{5.14}$$

5.3.5 Experimental evaluation

5.3.5.1 Training setup

Real-world experiments are conducted on the nuScenes dataset (Caesar et al., 2019). The dataset is split into 30002 training frames and 4146 testing frames from held-out sequences. The HD map raster and 3D bounding boxes are processed like explained in Section 5.3.1.1 to obtain the ground-truth semantic masks first in BEV. These masks of size $1\,000 \times 550$ (that is, 100m×55m) are then homography warped to obtain the ground-truth semantic masks of size 450×800 in the CPV. The specific size of the semantic masks in BEV was chosen such that after warping the obtained masks roughly match the limits of the camera field-of-view.

The BEV simulated-world experiments are run on the CARLA simulator (Dosovitskiy et al., 2017b) version 0.9.6, as implemented by Chen et al. (2019).

21426 frames are collected at 10 fps for training in Town 1 under 4 training weather conditions with 100 vehicles and 250 pedestrians. Each frame contains a 576×240 RGB camera image, a dense depth map of the same shape, the 3D bounding boxes coordinates of the vehicles in the scene and a road layout raster image. Here again semantic masks of size 600×300 (that is, 60m×30m) are generated in BEV and warped in CPV. 3659 frames are collected in Town 2 under 2 testing weathers for testing.

All networks are trained over 100 epochs with Stochastic Gradient Descent (SGD) and a batch size of 8 for real-world experiments and 14 for simulated data, with learning rate of 10^{-3}.

5.3.5.2 Evaluation baselines

Monocular camera based networks The performance of our FlatMobile network (FMNet) is evaluated against the following other monocular baselines:

- VED (Lu et al., 2018), MonoLayout (Mani et al., 2020): Multi-task training led to poor results so two separate models were trained, one for the vehicles OGM and another for the road layout OGM.
- VPN (Pan et al., 2020): trained in a multi-task way with two decoders, one for the vehicles OGM and another for the road layout OGM.
- OFT-DeepLab v3+ (Roddick et al., 2019): A multi-task DeepLab v3+ is augmented with an OFT module at the end of the network to project the CPV features to BEV. This baseline is similar to our FMNet with the difference that it uses OFT instead of homography warping and it learns directly in the BEV space.

Point cloud based networks LiDAR provides a very accurate depth estimation in the form of 3D point clouds and is almost always preferred to camera-based depth estimation for 3D perception tasks such as 3D object detection. It has been argued by Wang et al. (2019a) that the superior performance of LiDARs is not only due to their inherent 3D nature but also to

the choice of representation adopted in camera-based approaches. Their main idea consists in converting a dense depth map D estimated in CPV to a 3D point cloud such as:

$$z = D(u, v) \tag{5.15}$$

$$x = \frac{(u - C_U) \cdot z}{f} \tag{5.16}$$

$$y = \frac{(u - C_V) \cdot z}{f} \tag{5.17}$$

where z is the depth, x the width, y the height, f the focal length and (C_U, C_V) the coordinates of the optical center.

The obtained point cloud can then be processed with any LiDAR-based approach. The pseudo-LiDAR approach was initially a two-step approach with first a depth estimation network and then a 3D object detection network. This approach was further extended to be fully end-to-end by Qian et al. (2020) that introduced a differentiable soft quantization method to convert the point cloud to a 3D occupation tensor.

Based on these works, we introduce the following new baselines for OGM estimation:

- The available LiDAR point clouds were processed to obtain BEV 3-channels images that encode the distance to the LiDAR, the height and the intensity. A Deeplab v3+ was then trained to take these images as input and with two decoders output the vehicles and road layout OGMs.
- A similar network was trained with Pseudo-LiDAR instead as input (intensity set to 1). Dense depth maps are available in Carla simulator so the Pseudo-LiDAR baseline is trained end-to-end as described in (Qian et al., 2020) for the simulation experiments. For NuScenes, the dense depth maps are not readily available and were obtained using off-the-shelf depth estimation network (Lee et al., 2019) and the Pseudo-LiDAR baseline was obtained in two steps as described in Wang et al. (2019a).

- Similar to FishingNet (Hendy et al., 2020), we design a baseline that fuses the predictions of a network that takes LiDAR BEV snapshots as input and another that takes an image as input and uses VPN as a view transformer module.

5.3.5.3 Evaluation metrics

As the OGM prediction amounts to semantic segmentation, the classical Intersection over Union (IoU) metric is also adopted.

$$IoU = \frac{TP}{TP + FP + FN} \ , \tag{5.18}$$

where TP, FP and FN are respectively the true positive, false positive, and false negative pixel counts on the set of test images. We report the IoU values for the whole semantic masks but also for a close range and far range section of the masks. Close range is defined as the region 50m ahead of the ego vehicle and 10m on each side for real world data, and 30m ahead of the ego vehicle and 10m on each side for simulated data. Far range is defined as the region farther than 50m ahead of the ego vehicle for real world data and 30m ahead for simulated data.

5.3.5.4 Quantitative results

Comparative results with monocular and point cloud based baselines are presented in Tables 5.1 and 5.2. Two versions of our footprint segmentation model, referred to as FlatMobile Network (FMNet), are presented: FMNet which is the model described in Figure 5.5 and FMNet-MOVE the Mixture Of View Experts variation described in Figure 5.6. For each network, results are provided for the road layout and vehicles IoUs at different ranges as explained in Section 5.3.5.3.

Comparison with monocular baselines When comparing the results obtained by the baselines and our networks on real world data in Figure 5.1, the

TABLE 5.1: OGMs evaluation by *IoU* (in %) on real data (Caesar et al., 2019). Full refers to the whole OGMs, close focuses on a region of 50m ahead of the ego vehicle and 10m on each side, far on the region farther than 50m ahead of the ego vehicle.

	IoU static			IoU dynamic		
Range	Full	Close	Far	Full	Close	Far
VED (Lu et al., 2018)	60.2	61.4	55.0	44.7	42.5	44.7
OFT-DeepLab v3+ (Roddick et al., 2019)	65.9	64.4	**66.8**	54.2	52.5	**54.4**
VPN (Pan et al., 2020)	61.8	68.9	56.7	52.9	58.2	49.9
MonoLayout (Mani et al., 2020)	63.5	68.2	57.1	49.9	47.8	49.9
FMNet	66.1	78.0	56.7	54.4	59.7	51.7
FMNet-MOVE	**68.4**	**80.5**	59.7	**58.8**	**66.6**	52.8
Fishing (Hendy et al., 2020)	71.4	82.3	62.1	60.5	68.1	54.2
BEV Pseudo-LiDAR	63.0	72.0	54.7	54.1	58.4	50.3
BEV LiDAR	70.7	82.2	61.4	59.0	67.1	52.4

TABLE 5.2: OGMs evaluation by *IoU* (in %) on simulated data (Dosovitskiy et al., 2017a). Full refers to the whole OGMs, close focuses on a region of 30m ahead of the ego vehicle and 10m on each side, far on the region farther than 30m ahead of the ego vehicle.

	IoU static			IoU dynamic		
Range	Full	Close	Far	Full	Close	Far
VED (Lu et al., 2018)	79.3	81.3	78.6	51.4	48.1	51.0
BEV Pseudo-LiDAR End-to-End	75.8	80.0	73.0	50.9	59.6	50.8
OFT-DeepLab v3+ (Roddick et al., 2019)	85.3	83.8	85.5	53.8	50.1	54.8
VPN (Pan et al., 2020)	82.1	80.2	81.9	50.5	50.2	50.7
MonoLayout (Mani et al., 2020)	83.3	81.1	83.3	51.7	50.5	52.4
FMNet	87.2	89.0	84.3	59.3	65.4	56.8
FMNet-MOVE	90.2	90.0	89.6	62.2	68.9	59.2
Fishing (Hendy et al., 2020)	92.2	91.4	91.1	66.1	69.7	63.3
BEV LiDAR	92.1	91.8	90.1	65.3	69.4	62.5

first observation is the superior performance achieved by the networks that use a geometric information to warp the feature maps from BEV to CPV instead of a purely data-driven approach. FMNet has the best IoU in full range and close range evaluation while our OFT-DeepLab v3+ baseline is second best in full range and close range and obtains the best far range performance among all the tested networks including LiDAR-based ones. FMNet is on par with OFT-DeepLab (Roddick et al., 2019) on the road layout OGM and the vehicles OGM in full range but outperforms it in close-range by 13.6 % for the road layout OGM and 7.2 % for the vehicles OGM. The gain in performance in close range is due to our CPV training criterion that gives a higher relative weight to closer objects. Indeed, since our network learns in camera view, where closer objects appear bigger, it has an incentive to being more accurate in close range. OGMs are often used as input for motion planning algorithms and better performance in close range seems to be appropriate when planning a short-term trajectory. For the same reason, OFT has a better performance in long range: since it predicts directly in BEV, it does not suffer from the increase in error due to the imprecision in warping. The MOVE variation of the FMNet architecture improves the IoU of the road layout OGM in full range by 2.3%, 2.5% in close range and 3% in far-range. It also improves the vehicles OGM IoU by 4.4% in full range, 7.9 % in close range and 1.1% in far range.

Experiments made on simulated data in Figure 5.2 confirm the superiority of the FMNet approach when compared to other monocular baselines with the MOVE variation obtaining the best IoU results at all ranges and on both road layout and vehicles tasks. We also observe that the road layout IoUs for simulated data are much higher for all networks and the difference between tested networks is less important: this is because the Carla environment graphics are very simplistic (only T-junctions) compared to real life images and also because in simulation we consider a smaller BEV portion.

Comparison with point cloud based baselines In the real world or simulation, the BEV LiDAR baseline obtains slightly better results than the

FMNet-MOVE network on nearly all metrics while fusing outputs of a LiDAR-based and monocular network increases the computation burden without a significant improvement in performance. In close-range, the LiDAR baseline is 1.7% better than the MOVE network on the road layout IoU and 0.5% on the vehicles IoU. In far range, it is 1.7% better on the road layout IoU and the MOVE network is better on the vehicles IoU by 0.4 %. This comparison shows that our monocular camera-based network is very competitive with the LiDAR baseline on this task and one could argue that the slightly better performance of the LiDAR-based network would not justify its usage in a real-world application in place of a much cheaper camera-based solution. The Pseudo-LiDAR network has a reasonable performance on NuScenes dataset and is second best for close-range prediction when compared to monocular baselines. In simulation, it has the worst performance of all tested networks. It is important to remind that the network is learned end-to-end in simulation as opposed to real world experiments where the depth is first learned using an off-the-shelf depth estimation network.

5.3.5.5 Qualitative results

Qualitative results for the FMNet network are shown in Figure 5.7. The upper part of the figure shows the output OGMs of the FMNet network in CPV and the lower part shows the same OGMs in Bird-Eye-view after the homography warping. As our network operates only on planar surfaces, warping the vehicle masks from camera view to BEV does not cause the "stretching" that is observed in Figure 5.4 when warping regular semantic masks without the knowledge of depth. Hence, our approach can distinguish two vehicles in front of each other by simply applying the connected components algorithm to the binary masks for example. As indicated by the IoUs results, we observe that our network is more accurate in the lower part of the OGMs, which corresponds to the first meters in front of the camera which is a direct consequence of learning the grids in CPV. For the same reason, we also observe that the grids are not accurate enough in far-range as minor errors on far away objects in CPV are amplified when warping to BEV because of perspective. The FMNet-MOVE variation described in Figure 5.6 combines OGMs predicted

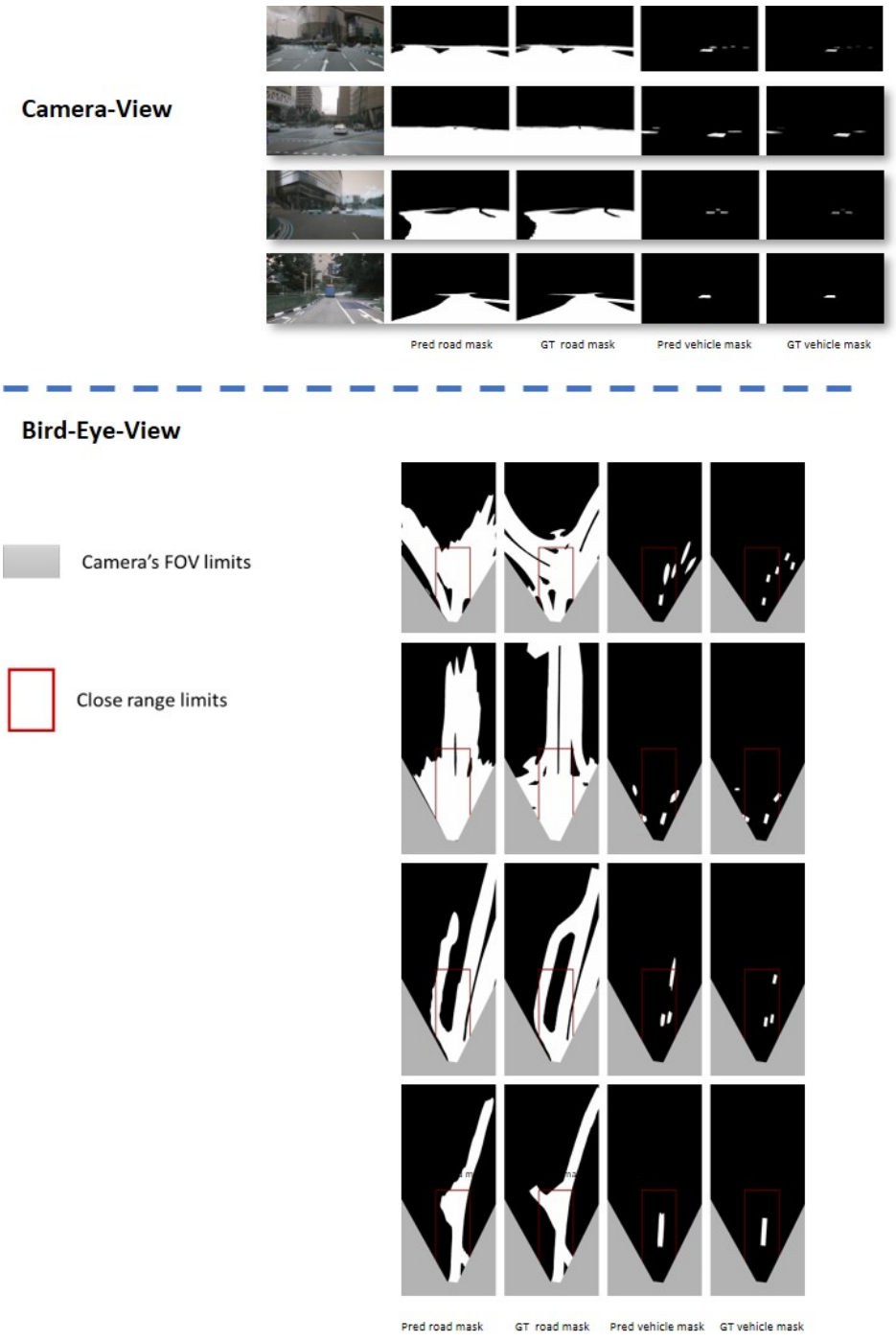


FIGURE 5.7: Qualitative results of our OGM estimation network without mixture of experts. The gray part of the predicted masks corresponds to the limits of the camera's field of view. GT stands for Ground Truth and Pred for predicted.

FIGURE 5.8: Qualitative results of our OGM estimation network with mixture of experts. The gray part of the predicted masks corresponds to the limits of the camera's field of view. GT stands for Ground Truth and Pred for predicted.

with a CPV learning loss by the close-range expert and OGMs predicted with a BEV learning loss by the far-range expert to generate an OGM that benefits from the "best of the two worlds". The MOVE architecture not only improves the far-range performance but also improves the close-range performance as the close-range expert can specialize even more. Sample OGMs obtained with the MOVE network along with the intermediate predictions of the CPV and BEV experts are shown in Figure 5.8. BEV refers to the predictions of the BEV expert, CPV2BEV refers to the prediction of the CPV expert that are then warped to BEV and MOVE refers to the final output of the FMNet-MOVE architecture. We observe that some vehicles are detected by the BEV expert but not by the CPV expert. The same goes for portions of the road layout. We also observe that the predictions of the CPV expert are much finer in close-range than those of the BEV expert: for example multiple close vehicles are predicted as a single "blob" by the BEV expert when the CPV expert is able to distinguish them. The final prediction of the MOVE network is able to take advantage of both experts capabilities and obtain a refined prediction for the full range of the OGMs. The figure also shows the two weighting maps used by the MOVE architecture: we observe that as expected the CPV map focuses on close-range prediction and inversely for the BEV one.

5.4 Conclusion

In this chapter, we introduce a novel method to output bird-eye-view occupancy grid maps from a monocular camera using homography warping. Homography warping being a planar transformation, vehicles are successfully warped to Bird-Eye-View by considering only their footprints. This simple trick allows us to outperform in terms of IoU all other publicly available baselines both on the vehicles and road layout occupancy grid maps especially when considering the close range area ahead of the ego-vehicle. Given that ground-truth is available in both Camera-Projective-View and Bird-Eye-View, we introduce a mixture of views approach that allows us to learn in both views thus further improving the performance of our network.

Chapter 6

Driving among flatmobiles: An application of footprint segmentation

6.1 Introduction

Camera-based end-to-end driving neural networks bring the promise of a low-cost system that maps camera images to driving control commands. These networks are appealing because they replace laborious hand-engineered building blocks but their black-box nature makes them difficult to delve in case of failure. Recent works have shown the importance of using an explicit intermediate representation that has the benefits of increasing both the interpretability and the accuracy of networks' decisions. Nonetheless, these camera-based networks reason in camera view where scale is not homogeneous and hence may not be directly suitable for motion forecasting. In this chapter, we introduce a novel monocular camera-only holistic end-to-end trajectory planning network with a Bird-Eye-View (BEV) intermediate representation that comes in the form of binary Occupancy Grid Maps (OGMs). To ease the prediction of OGMs in BEV from camera images, the previously described footprint segmentation approach is adopted. The gain in performance is demonstrated

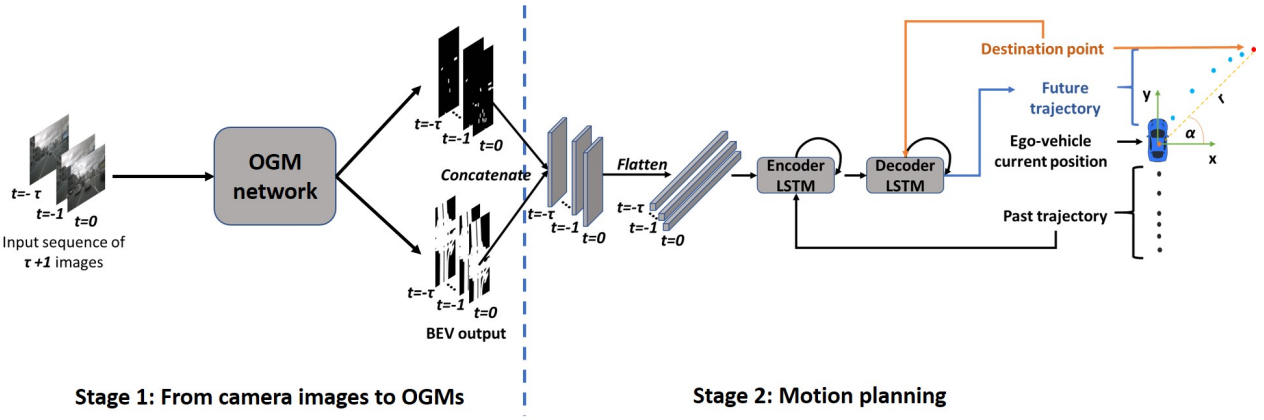


FIGURE 6.1: Two-stage network for end-to-end trajectory planning from a monocular camera with intermediate BEV OGM outputs.

off-line and on-line on respectively the nuScenes dataset (Caesar et al., 2019) and Carla simulator (Dosovitskiy et al., 2017b).

6.2 Encoder-decoder LSTM for trajectory planning

6.2.1 Model formulation

In this section, we introduce a two-stage holistic end-to-end trajectory planning network that takes a sequence of 5 monocular images as input, leverages a BEV mediated perception strategy and plans the trajectory of the ego vehicle knowing its destination position at a 5s horizon, see Figure 6.1. The intermediate representation of this network is generated by the first stage and comes in the form of two binary Occupancy Grid Maps (OGMs) in BEV, one giving the road layout and the other one the occupancy of the vehicles in the scene. As it is not straightforward to directly obtain BEV outputs from a camera plane input while preserving the receptive field, the footprint segmentation FMNet-MOVE described in Figure 5.6 is adopted for the first stage of the network and is applied to each of the 5 input images. The second stage of the network implements mid-to-end trajectory planning. The two sequences of road layout and vehicle BEV masks are concatenated along the channel

dimension at each time step and flattened to obtain a sequence of feature vectors $f_{-\tau}, ..., f_0$. These feature vectors become input to an encoder-decoder LSTM (Cho et al., 2014). The encoder embeds the sequence of feature vectors of length $\tau + 1$ and processes it introducing the following recurrence:

$$\begin{aligned} e_t &= \phi_e(f_t; W_e) \\ a_t &= \phi_a((x_t, y_t); W_a) \\ h_t &= LSTM(h_{t-1}, concat(e_t, a_t); W_l) \ , \end{aligned} \tag{6.1}$$

where (x_t, y_t) is the position of the ego vehicle at time $t \in [-\tau, ..., 0]$, ϕ_e and ϕ_a are the embedding functions, W_e and W_a are the embedding weights, h_t and W_l are respectively the hidden state and the weights of the encoder LSTM.

The final cell state c_0 and hidden state h_0 of the encoder that summarizes the input OGM sequence are fed to the decoder as its initial cell and hidden states. The decoder then recursively outputs the sequence of future positions:

$$\begin{aligned} a'_t &= \phi'_a(s_{t-1}; W'_a) \\ h'_t &= LSTM(h'_{t-1}, a'_t; W'_l) \\ s_t &= \phi_s(concat(h'_t, (r, \alpha)); W_s) \ , \end{aligned} \tag{6.2}$$

where ϕ'_a and ϕ_s are the embedding functions, W'_a and W_s are the embedding weights, (c'_t, h'_t) and W'_l are respectively the cell/hidden states and the weights of the decoder LSTM, s_t the estimated future position at time $t \in [1, ..., T]$ and (r, α) the polar coordinates of the destination point. The prediction of the future positions $s_{1:T}$ is expressed in Cartesian coordinates, whereas the destination point is expressed in polar coordinates, for encouraging solutions that depart from a simple direct interpolation, thereby enhancing the impact of different design choices.

6.2.2 Cost function and learning

At decoding (i.e., planning) time, the LSTM predicts the distribution of the future position of the ego vehicle at each time-step, as developed in (6.2). Similar to Alahi et al. (2016), the output s_t of the encoder-decoder LSTM

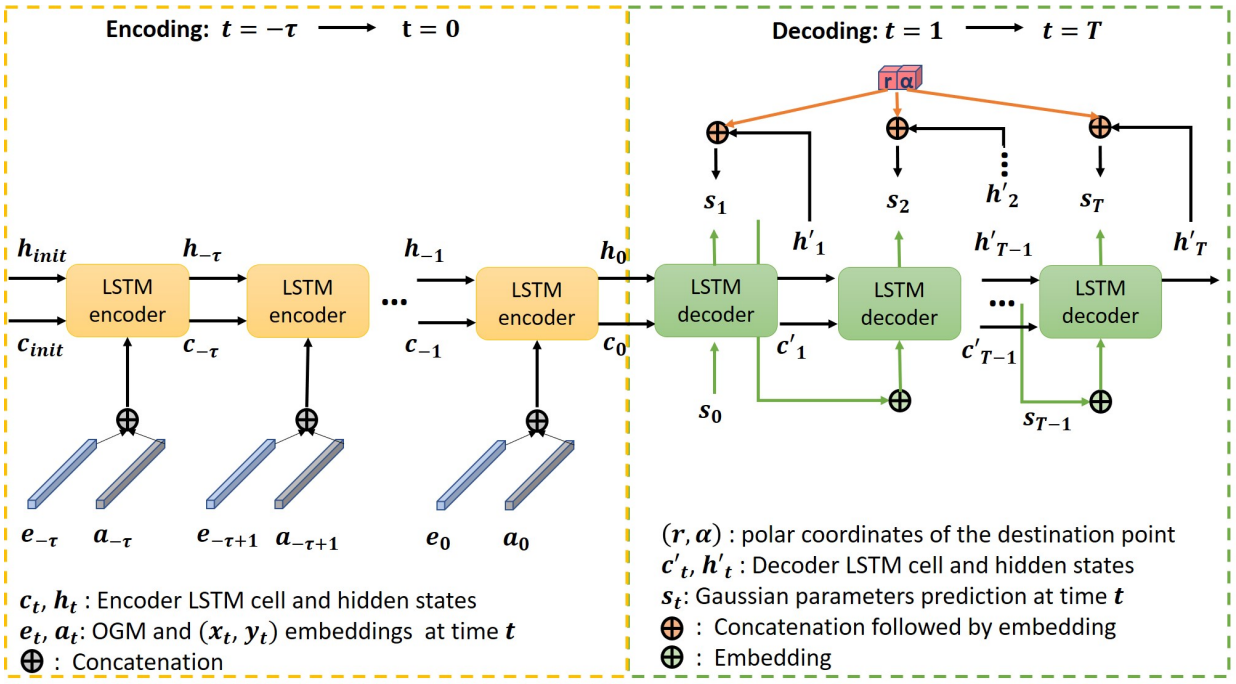


FIGURE 6.2: Encoder-decoder LSTM for trajectory planning.

module predicts the parameters of a bi-variate Gaussian distribution characterized by its mean $\mu_t = (\mu_t^x, \mu_t^y)$ and its covariance matrix parameterized by the standard deviations $\sigma_t = (\sigma_t^x, \sigma_t^y)$ and the correlation ρ_t.

The predicted position of the ego vehicle at time t is given by $(x_t, y_t) \sim \mathcal{N}(\mu_t, \sigma_t, \rho_t)$. The parameters of the encoder-decoder LSTM module are learned by minimizing the negative log-likelihood of the Gaussian distribution:

$$\mathcal{L}_{motion} = -\sum_{t=1}^{T} \log(\mathbb{P}(x_t, y_t | \mu_t, \sigma_t, \rho_t)) \quad . \tag{6.3}$$

For the holistic end-to-end network, the final loss function is a linear combination of the perception loss and the motion loss:

$$\mathcal{L}_{total} = \alpha \, \mathcal{L}_{perception} + \mathcal{L}_{motion} \quad , \tag{6.4}$$

where $\mathcal{L}_{perception}$ is defined in (5.8) and α is empirically set to 0.1.

6.3 Experimental evaluation

6.3.1 Evaluation metrics

The Average Displacement Error (ADE) corresponds to the average Euclidean distance between the predicted trajectory and the ground-truth one: $ADE = \frac{1}{NT}\sum_{i=1}^{N}\sum_{t=1}^{T}\left\|\widehat{Z}_i^t - Z_i^t\right\|_2$, where N is the number of samples, T is the number of prediction timesteps, Z_i^t are the ith ground-truth coordinates at time step t and $\widehat{Z}_i^t$ are their predictions.

6.3.2 Evaluation baselines

The performance of the holistic two-stage end-to-end model with intermediate OGM outputs is compared against the following baseline models:

- LSTM E-D: Same architecture than the LSTM encoder-decoder described in Figure 6.2, except that it takes only the past trajectory as input.
- End-to-end: This model takes a sequence of camera images as input, encodes it with a ResNet-101, flattens the feature maps and feeds the obtained sequence of feature vectors to the same encoder-decoder LSTM described in Figure 6.2. This model is comparable to direct perception approaches (see, e.g. Bojarski et al., 2016, Codevilla et al., 2017, Pomerleau, 1988), with the difference that it takes a sequence of images as input and leverages a LSTM model to process this temporal information.
- Mid-to-end: Similar to the end-to-end model, except that it takes as input a sequence of concatenated *ground-truth* drivable area and vehicle semantic masks. This model is comparable to mid-to-mid approaches (Bansal et al., 2019, Srikanth et al., 2019) or the privileged learner of Chen et al. (2019).

6.3.3 Training setup

All networks are trained using SGD with a batch size of 10 for 200 epochs. All results are obtained with a momentum of 0.9 and a learning rate of 10^{-3}. After removing the frames that do not have 5s of history and 5s of future (10 previous and 10 future frames), nuScenes dataset (used for off-line evaluation) is split in 13982 training samples and 3167 testing samples from held-out sequences.

6.3.4 Off-line experiments

6.3.4.1 Quantitative results

The results for motion planning are presented in Table 6.1. The $L1$ norm of lateral and longitudinal displacements are presented in addition to the Average Displacement error (ADE) metric. The two-stage holistic network shows improvements on the three metrics over its regular end-to-end counterpart, with an improved accuracy in both directions with more important impact on the lateral direction and the performance discrepancy increasing over time. Overall, the regular end-to-end network has the worst performance among the tested networks, which confirms the intuition that the intermediate BEV representation is an asset for motion forecasting. The mid-to-end network outperforms the end-to-end network but falls behind the holistic network on all metrics. This was not expected as the mid-to-end network takes ground-truth masks as input. A possible explanation for this surprising fact is that, our network learning the OGMs from camera images in an end-to-end fashion, it may benefit from pieces of information contained in the natural images which help disambiguate some driving scenarios. In other frameworks, accurate smooth estimates of discrete quantities have already been shown to be more efficiently processed than the ground truths themselves (Hinton et al., 2015). Our continuous masks, which are quite accurate in close range, may thus convey more information for fitting the second stage of the network in critical areas. The importance of having access to a scene context is highlighted by the inferior results obtained by the LSTM E-D model.

TABLE 6.1: Average displacement errors and $L1$ norm (in meters) for lateral and longitudinal displacements. Our model is referred to as Driving Among Flatmobiles (DAF). The mid-to-end and end-to-end are respectively referred to as MTE and ETE. NP (No Past) refers to models that do not take the past trajectory as input; CV (Camera View) refers to a model where the predicted OGMs are not warped in BEV. Best results are shown in bold.

	ADE			$L1$ longitudinal			$L1$ lateral		
	0.5s	2.5s	4.5s	0.5s	2.5s	4.5s	0.5s	2.5s	4.5s
LSTM	1.78	2.38	3.57	1.69	2.07	3.10	0.07	0.30	0.68
MTE	0.47	0.90	1.11	0.47	0.79	**0.74**	**0.01**	0.18	0.49
ETE	0.49	0.92	1.17	0.46	0.80	0.78	0.02	0.22	0.54
DAF	**0.40**	**0.86**	**1.02**	**0.40**	**0.77**	**0.74**	**0.01**	**0.17**	**0.40**
DAF CV	0.48	0.92	1.13	0.45	0.81	0.77	0.02	0.20	0.51
MTE NP	**0.39**	0.96	1.11	**0.39**	**0.77**	**0.72**	**0.01**	**0.19**	0.51
ETE NP	0.43	0.96	1.27	0.43	0.80	0.76	0.02	0.24	0.68
DAF NP	0.46	**0.90**	**1.10**	0.46	0.79	0.73	**0.01**	**0.19**	**0.48**

6.3.4.2 Ablation study

We show the importance of some components of the approach introduced here with the degraded results that are presented in Table 6.1. Removing the past trajectory input shows the usefulness of BEV when there is no prior about the past motion. An important gap in performance is observed between the holistic network that leverages BEV information to forecast motion and the end-to-end network, with up to 20cm difference in long-term lateral forecast, but the two networks have similar performance on the other metrics. The performance of the mid-to-mid network is similar to that of our network in the lateral direction and is slightly better in the longitudinal direction, especially in short-term prediction. The importance of BEV is also highlighted by removing the perspective warping and feeding directly the Camera-View (CV) features as input to the second stage. The BEV version of the holistic network achieves a better overall performance than its CV counterpart with an average lateral gain of 11cm at a horizon of 4.5s.

6.3.4.3 Qualitative results

Qualitative results of our network on nuScenes dataset (on offline trajectory planning) are provided in Figure 6.3. We observe that our approach handles different road layouts and different times of the day. As expected, situations where the vehicle is turning are more challenging to predict on the lateral direction and are most responsible for the error in this direction. We also observe that in most cases the longitudinal error increases with time, which is also expected as the trajectory prediction network has only access to the past scene representations and recursively outputs the future positions which can lead to an accumulation of errors over time. The results presented in Figure 6.4 show examples of situations where our model fails to accurately predict the future trajectory. For example, in scenarios where the vehicle is moving rapidly on highways, our model often fails to accurately predict the trajectory and outputs future positions at a much slower pace, showing that our network is biased towards urban situations where the vehicle is moving more slowly, see the top image. The bottom image shows another recurring scenario where our model fails on the long-term prediction when the vehicle is turning and predicts a turn in the opposite direction. We also attribute this weakness to a bias induced by the learning samples, augmenting the dataset with randomly rotated coordinates and masks could solve this issue (Bansal et al., 2019).

6.3.5 On-line experiments

6.3.5.1 Quantitative results

Having a better performance on off-line metrics is not necessarily correlated with proper driving (Codevilla et al., 2018). The on-line performance of our model is evaluated in testing town 2 on the Carla simulator NoCrash benchmark. This benchmark consists of 3 driving scenarios with a varying number of vehicles and pedestrians in the simulated town: empty (no traffic), regular traffic and dense traffic. The vehicle drives 25 predefined routes with different starting points and an episode is counted as successful if the vehicle reaches its goal within a certain time limit without colliding with a static or

FIGURE 6.3: Qualitative results of our holistic trajectory planning network on nuScenes; blue dots are for ground truth, red dots for prediction.

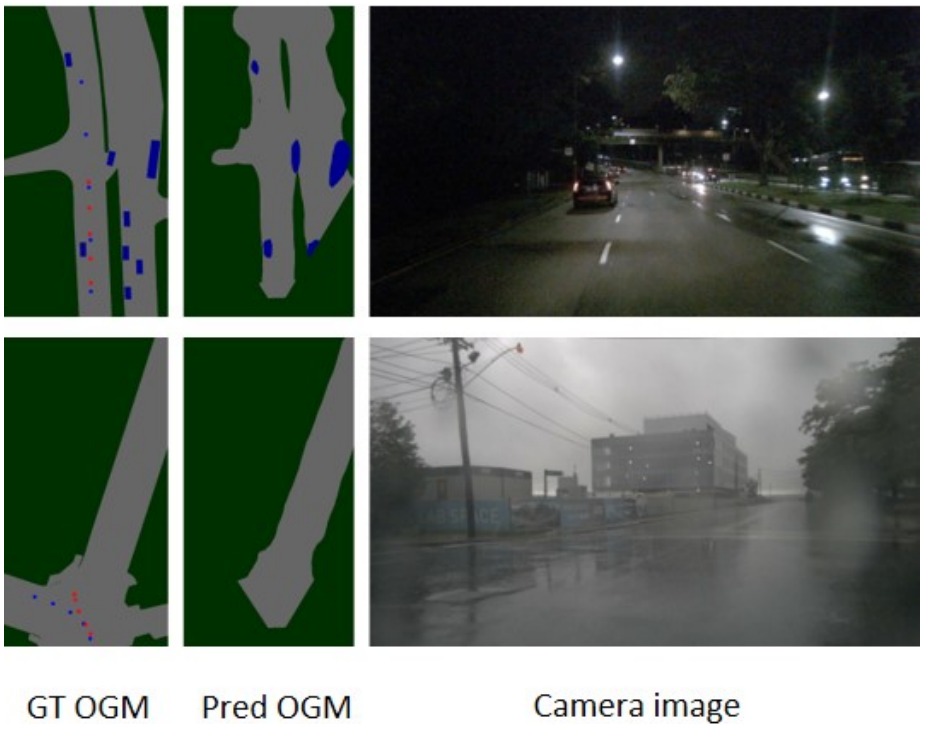


FIGURE 6.4: Two examples where our model fails to plan the trajectory on nuScenes; blue dots are for ground truth, red dots for prediction.

a dynamic object. The training weathers are "Clear noon", "Clear noon after rain", "Heavy raining noon", and "Clear sunset" and the test-only weathers are "After rain sunset" and "Soft raining sunset".

Our DAF network and the baselines are adapted to the task and a one-hot encoded high-level command ("follow lane", "turn left", "turn right", "go straight") is provided instead of a goal position. Similar to Codevilla et al. (2017), the high-level command guides the vehicle through predefined routes. We also modify the trajectory part of the networks and adopt the architecture and loss function of Chen et al. (2019) relying on a single image used as input. Instead of the encoder-decoder LSTM version that takes several images as input, the modified DAF takes only a monocular image as input, predicts the intermediate BEV semantic maps and finally outputs heatmaps for each predicted time-step. The heatmaps are then converted to waypoints using a soft-argmax function. The set of waypoints is then converted to driving commands using a low-level controller as described in Chen et al. (2019). The

TABLE 6.2: Carla NoCrash benchmark success rates in closed-loop. Mid-to-mid relies on ground-truth OGMs.

Weather	Traffic	End-to-End	Mid-to-Mid	Ours
Seen	Empty	43	91	89
	Regular	30	80	76
	Dense	11	43	39
Unseen	Empty	37	88	64
	Regular	23	85	55
	Dense	8	44	32

mid-to-mid baseline input is also changed to a 7 channels grid map similar to Chen et al. (2019) containing information about the road layout, the vehicles, the pedestrians and the traffic lights. Pedestrians in Carla simulator have a very erratic behavior as they cross frequently the road. This required access to pedestrian information in the mid-to-mid network. Information about traffic lights is also necessary for navigation, because the model is fed with a high-level command instead of a goal position. We use the same heatmap-based trajectory prediction sub-network for the MTE and ETE baselines. For the ETE network, the heatmaps and the waypoints are predicted in the camera space and then the waypoints are projected to 3D knowing the intrinsic parameters of the camera.

Table 6.2 displays the results. Our model significantly outperforms the end-to-end baseline in all conditions, and it is at par with the mid-to-mid baseline in weather conditions seen during training. The mid-to-mid baseline performance is not affected by weather conditions that were unseen during training as it relies on ground truth OGMs instead of OGMs inferred from camera images.

6.3.5.2 Qualitative results

Figure 6.5 displays qualitative results of our network in Carla simulator, with the intermediate OGMs ground-truth and predicted masks in both camera-view and bird-eye-view along with the input camera image. It also shows the ground-truth and predicted trajectories on top of the ground-truth OGM mask. A video of our network driving in closed-loop in Carla simulator can

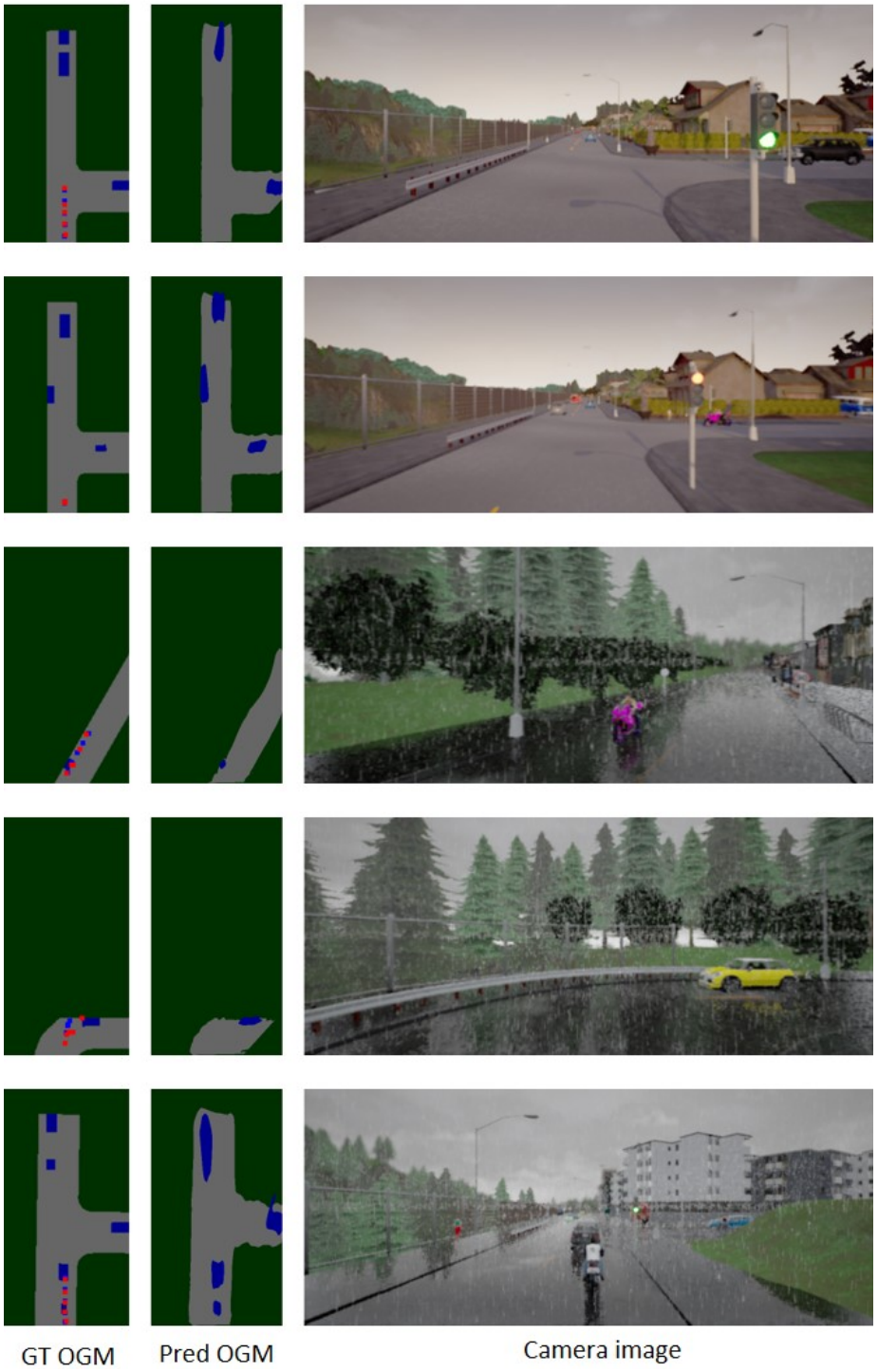


FIGURE 6.5: Qualitative results of our holistic trajectory planning network on Carla simulator; blue dots are for ground truth, red dots for prediction.

be found on the following link: Closed loop driving in Carla simulator. This video shows 4 different scenarios of the vehicle driving in: train town with train weather, train town with test weather, test town with train weather and finally test town with test weather. Although the intermediate representation of our network does not inform about pedestrians, our model manages to stop in front of pedestrians and avoids collisions. Looking at the intermediate representation, we see that pedestrians are detected as vehicles when they are in the drivable area. The vehicles class in the simulator includes cars and trucks, but also cyclists that look very much like pedestrians. This observation highlights the importance of having an intermediate representation to delve the decisions made by the model.

6.4 Conclusion

In this chapter, a novel camera-based holistic end-to-end motion planning network is introduced. It takes as input a sequence of camera images and first leverages the Driving Among Flatmobiles approach to obtain bird-eye-view occupancy grid maps. These intermediate grid maps are then used to plan the ego-vehicle trajectory in an end-to-end fashion. The performance of our end-to-end trajectory planning approach is evaluated both off-line and on-line and compared to its regular end-to-end counterpart, our method is better at imitation driving and provides interpretable intermediate results. The benefit of embedding computer vision tools for the transformation from the camera view to the bird-eye-view intermediate representation is also highlighted in our ablation study. End-to-end driving may not be reliable enough to be used as a primary solution in autonomous vehicles, but such a cheap camera-based component could prove useful for redundancy in a more complex system.

Chapter 7

Conclusion

In this chapter, we first summarize the contributions of this thesis that cover scene understanding in both the camera-view and the Bird-Eye-View. In the second part, we discuss the possible research axis that can sprout from the work done in this thesis.

7.1 Contributions

7.1.1 Leveraging spatial context to enhance scene understanding in the camera-view space

The first approaches developed during this thesis focused on injecting a cartographic information and more broadly a spatial context into a semantic segmentation convolutional neural network (CNN). We have first explored 3 different ways of injecting a cartographic and depth information into these CNNs: as a smoothness cost in a post-processing algorithm, as an auxiliary task in a multi-task network and as additional input to a multi-encoder network. Our experiments have shown that the first two methods did not improve significantly the performance whereas the third one improved the semantic segmentation accuracy with the cost of increased computational burden and annotation efforts.

In semantic segmentation or object detection, learning losses are often weighed according to the frequency of classes in the dataset with the less-represented classes having the highest weights. However, weighing classes according to their frequency in the dataset may not be the most appropriate way for autonomous driving as an object from an under-represented class located at a 100m from the ego-vehicle is arguably less important than an object from an over-represented class located 3m in front as closer objects are presumably the ones that are more susceptible to cause a collision. Still with the objective of leveraging spatial context to improve the accuracy of semantic segmentation networks, we have introduced a new disparity-based learning loss weighing scheme. This pixel-wise disparity weighting scheme multiplies each pixel in the loss function by its disparity value hence giving more importance to closer-objects. We verify the effect of this weighting on two different lightweight network architectures on two datasets and observed an overall improvement with a even more improved performance in close range. Given that the disparity maps are obtained with an off-the-shelf self-supervised network, our method does not increase the annotation burden.

7.1.2 Bird-Eye-View scene understanding from a monocular camera with an application to end-to-end trajectory planning

The objective of this body of work was to show another example of how cameras can be used for 3D perception. Even though LiDAR-based semantic segmentation is getting more and more traction, annotated LiDAR datasets are still scarce when compared to camera-based ones. Therefore, we have designed a network that takes camera images as input and outputs semantic masks in Bird-Eye-View (BEV). This task poses several challenges with the first one being how to preserve the receptive field of the CNN between the input and the output of the network when they are not in the same geometric space. Usually, semantic masks obtained in the camera-view are projected to BEV using inverse perspective mapping which is the homography transform that relates BEV plane and the camera plane. However, homographies

are planar transforms and warping 3D objects like vehicles that lie above the ground plane causes a stretching effect that makes these warped masks non-usable. Our network processes the cameras images, extracts feature maps in the camera-view space and then uses a homography warping layer to obtain the masks in BEV. To avoid the stretching effect mentioned earlier, we introduce footprint segmentation that consists in segmenting the footprint of the vehicles. This simple yet effective trick allows us to outperform all the works in the literature and even achieve a performance on-par with a LiDAR baseline that we have designed. A consequence of learning in camera-view and warping to bird-eye-view is that our network has a superior performance in close-range but suffers in far-range. To deal with the poor far-range performance, we improve our network with what we call a mixture of view experts which is a variation of our footprint segmentation network that learns in both camera-view and bird-eye-view and leverages a mixture of experts approach.

These BEV semantic masks are comparable to Occupancy Grid Maps (OGMs) which are usually used for trajectory planning in traditional self-driving pipelines. Being able to output these masks from a camera input opened up the possibility to have a holistic end-to-end trajectory planning that has these OGMs as an intermediate representation which allows allows to interpret the network's decisions. We first evaluate the driving capabilities of our network offline and show that our network has superior performance when compared to a regular end-to-end network. However having a better off-line performance does not necessarily translate to better driving in a closed-loop setting so we evaluate the on-line driving capabilities of our network on Carla simulator (Dosovitskiy et al., 2017a) and show that our network still outperforms its regular end-to-end counterpart and has even a similar performance to a baseline network that takes ground truth OGMs as input.

7.2 Perspectives

We believe that the following avenues of research can take advantage of the works developed during this thesis:

- Being able to output accurate semantic masks in the bird-eye-view from a monocular camera can be used for several perception or motion estimation tasks. In our experiments we present a baseline based on OFTNet (Roddick et al., 2019) whose original purpose is to output 3D bounding-boxes. A semantic map similar to our Vehicles mask is used in OFTNet as a confidence map (x, y) which indicates the probability of an object having a 3D bounding-box centered at (x, y, z). OFTNet does not use the cartographic information in its confidence map and we have shown in Chapter 3 that it contains a valuable information to locate vehicles. A possible monocular 3D object detection architecture would be our footprint segmentation network augmented with the top-down network as described in OFTNet taking as input the concatenated road layout and vehicles semantic masks.

- Visual Odometry (VO) is an important image-based approach for mobile robotics as it allows to localize the robot in its environment and is often used jointly with localization sensors such as GPS and IMU. Traditional pipelines for VO imply several hand-engineered atomic tasks such as feature matching, motion estimation or scale estimation. Similar to many other computers tasks, deep neural networks have been able to eliminate hand-engineering and achieve impressive results in VO. DeepVO (Wang et al., 2017) was first to introduce a monocular camera-based CNN that uses convolutional layers to extract features from a sequence of images and a recurrent neural network to process the sequence of features maps and output the poses in the real world scale. Similar to trajectory planning, VO is a sequential motion estimation task so we could leverage a similar CNN/RNN combination and use our footprint segmentation approach to output intermediate BEV semantic maps. We could expect that having these BEV masks between the CNN and the RNN part of

the network would improve the accuracy of the prediction as it did for the trajectory planning. Indeed the objects size in these BEV semantic masks does not vary with the distance to the camera as they do not suffer from the perspective effect and thus may be better candidate input for the motion estimation RNN.

- An important asset to plan the future ego-motion of a self-driving vehicle is the predicted behavior of the other road users. Our current trajectory planning lacks this asset which could hinder its performance in a real-world application. Usually the trajectory of each road user needs to be tracked and predicted but with these intermediate BEV semantic masks an alternative approach can be adopted: it is possible to predict the position of all road actors by predicting the future semantic masks using ConvLSTM (Shi et al., 2015) layers. Previous works have explored this research direction (Mohajerin and Rohani, 2019, Schreiber et al., 2019) but to the best of our knowledge none of these works do it end-to-end from a monocular camera. Being able to predict the future configuration of the scene can be used in our current trajectory planning network to penalise the predicted positions according to the other vehicles and the road layout, similar to the collision and on-road losses in ChauffeurNet (Bansal et al., 2019).

- In our work on trajectory planning, we have only explored behavior cloning but Inverse Reinforcement Learning (IRL) also allows to imitate an expert's demonstrations for this task (Abbeel et al., 2008, Deo and Trivedi, 2020, Wulfmeier et al., 2016, Ziebart et al., 2009). Instead of learning a direct mapping between the environment state and the agent's actions by supervised learning, IRL learns the unknown reward function that explains the expert's behavior and optimizes a policy based on this reward. According to these previous works, IRL allows for a better generalization to new environments as it is a more parsimonious description of the expert's behavior (Abbeel and Ng, 2004), better conforms to the scene configuration and also has a better long-horizon performance. The deep IRL approaches introduced previously take as input a BEV representation of the environment so we believe that our monocular approach

to estimate BEV semantic masks can be re-purposed for an end-to-end IRL trajectory planning network. One could even argue that estimating the BEV semantic masks beforehand could ease the reward function estimation as it would already provide the navigable space.

Bibliography

Abbeel, P., Dolgov, D., Ng, A. Y., and Thrun, S. (2008). Apprenticeship learning for motion planning with application to parking lot navigation. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1083–1090.

Abbeel, P. and Ng, A. Y. (2004). Apprenticeship learning via inverse reinforcement learning. In *ICML '04*, page 1, New York, NY, USA. Association for Computing Machinery.

Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Fei-Fei, L., and Savarese, S. (2016). Social lstm: Human trajectory prediction in crowded spaces. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Bacha, A., Bauman, C., Faruque, R., Fleming, M., Terwelp, C., Reinholtz, C., Hong, D. W., Wicks, A., Alberi, T., Anderson, D., Cacciola, S., Currier, P., Dalton, A., Farmer, J., Hurdus, J. W., Kimmel, S., King, P., Taylor, A., Covern, D. V., and Webster, M. (2008). Odin: Team victortango's entry in the darpa urban challenge. *J. Field Robotics*, 25:467–492.

Badrinarayanan, V., Kendall, A., and Cipolla, R. (2017a). Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Badrinarayanan, V., Kendall, A., and Cipolla, R. (2017b). Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(12):2481–2495.

Bansal, M., Krizhevsky, A., and Ogale, A. S. (2019). Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst. In *Robotics: Science and Systems XV, University of Freiburg, Freiburg im Breisgau, Germany, June 22-26, 2019.*

Best, A., Narang, S., Barber, D., and Manocha, D. (2017). Autonovi: Autonomous vehicle planning with dynamic maneuvers and traffic constraints. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 2629–2636.

Bo Li, Chunhua Shen, Yuchao Dai, van den Hengel, A., and Mingyi He (2015). Depth and surface normal estimation from monocular images using regression on deep features and hierarchical crfs. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1119–1127.

Bojarski, M., Testa, D. D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L. D., Monfort, M., Muller, U., Zhang, J., Zhang, X., Zhao, J., and Zieba, K. (2016). End to end learning for self-driving cars. *CoRR*, abs/1604.07316.

Brostow, G. J., Fauqueur, J., and Cipolla, R. (2008). Semantic object classes in video: A high-definition ground truth database. *Pattern Recognition Letters*, xx(x):xx–xx.

Brostow, G. J., Fauqueur, J., and Cipolla, R. (2009). Semantic object classes in video: A high-definition ground truth database. *Pattern Recognit. Lett.*, 30(2):88–97.

Buehler, M., Iagnemma, K., and Singh, S. (2009). *The DARPA Urban Challenge: Autonomous Vehicles in City Traffic.* Springer Publishing Company, Incorporated, 1st edition.

Caesar, H., Bankiti, V., Lang, A. H., Vora, S., Liong, V. E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., and Beijbom, O. (2019). nuScenes: A multimodal dataset for autonomous driving. *arXiv preprint arXiv:1903.11027.*

Cao, Y., Wu, Z., and Shen, C. (2018). Estimating depth from monocular images as classification using deep fully convolutional residual networks.

IEEE Transactions on Circuits and Systems for Video Technology, 28:3174–3182.

Chang, M.-F., Lambert, J. W., Sangkloy, P., Singh, J., Bak, S., Hartnett, A., Wang, D., Carr, P., Lucey, S., Ramanan, D., and Hays, J. (2019). Argoverse: 3d tracking and forecasting with rich maps. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Chen, C., Seff, A., Kornhauser, A. L., and Xiao, J. (2015a). Deepdriving: Learning affordance for direct perception in autonomous driving. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 2722–2730.

Chen, D., Zhou, B., Koltun, V., and Krähenbühl, P. (2019). Learning by cheating. In *Conference on Robot Learning (CoRL)*.

Chen, L., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. (2018a). Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(4):834–848.

Chen, L., Zhu, Y., Papandreou, G., Schroff, F., and Adam, H. (2018b). Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part VII*, pages 833–851.

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. (2015b). Semantic image segmentation with deep convolutional nets and fully connected crfs. In *ICLR*.

Chen, L.-C., Yang, Y., Wang, J., Xu, W., and Yuille, A. (2016). Attention to scale: Scale-aware semantic image segmentation. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3640–3649.

Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of*

the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Codevilla, F., López, A., Koltun, V., and Dosovitskiy, A. (2018). On offline evaluation of vision-based driving models. In *ECCV*.

Codevilla, F., Miiller, M., López, A., Koltun, V., and Dosovitskiy, A. (2017). End-to-end driving via conditional imitation learning. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–9.

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., and Schiele, B. (2016). The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Cui, H., Radosavljevic, V., Chou, F., Lin, T., Nguyen, T., Huang, T., Schneider, J., and Djuric, N. (2019). Multimodal trajectory predictions for autonomous driving using deep convolutional networks. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 2090–2096.

Deo, N. and Trivedi, M. (2020). Trajectory forecasts in unknown environments conditioned on grid-based plans. *ArXiv*, abs/2001.00735.

DeTone, D., Malisiewicz, T., and Rabinovich, A. (2016). Deep image homography estimation. *CoRR*, abs/1606.03798.

Djuric, N., Radosavljevic, V., Cui, H., Nguyen, T., Chou, F., Lin, T., Singh, N., and Schneider, J. (2020). Uncertainty-aware short-term motion prediction of traffic actors for autonomous driving. In *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 2084–2093.

Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., and Koltun, V. (2017a). CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16.

Dosovitskiy, A., Ros, G., Codevilla, F., López, A., and Koltun, V. (2017b). CARLA: an open urban driving simulator. In *1st Annual Conference on Robot Learning (CoRL 2017)*, pages 1–16.

Eigen, D. and Fergus, R. (2015). Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2650–2658.

Eigen, D. and Fergus, R. (2015). Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *International Conference on Computer Vision (ICCV)*, pages 2650–2658.

Elfes, A. (2013). Occupancy grids: A stochastic spatial representation for active robot perception. *CoRR*, abs/1304.1098.

Erkent, Ö., Wolf, C., Laugier, C., González, D. S., and Romero-Cano, V. (2018). Semantic grid estimation with a hybrid bayesian and deep neural network approach. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2018, Madrid, Spain, October 1-5, 2018*, pages 888–895.

Facil, J. M., Ummenhofer, B., Zhou, H., Montesano, L., Brox, T., and Civera, J. (2019). Cam-convs: Camera-aware multi-scale convolutions for single-view depth. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11818–11827.

Fu, H., Gong, M., Wang, C., Batmanghelich, K., and Tao, D. (2018). Deep Ordinal Regression Network for Monocular Depth Estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Garg, R., Kumar, B. V., Carneiro, G., and Reid, I. (2016). Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *European Conference on Computer Vision*, pages 740–756. Springer.

Geiger, A., Lenz, P., and Urtasun, R. (2012). Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Geyer, J., Kassahun, Y., Mahmudi, M., Ricou, X., Durgesh, R., Chung, A. S., Hauswald, L., Pham, V. H., Mühlegg, M., Dorn, S., Fernandez, T., Jänicke,

M., Mirashi, S., Savani, C., Sturm, M., Vorobiov, O., Oelker, M., Garreis, S., and Schuberth, P. (2020). A2d2: Audi autonomous driving dataset.

Godard, C., Mac Aodha, O., and Brostow, G. J. (2017). Unsupervised monocular depth estimation with left-right consistency. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6602–6611.

Godard, C., Mac Aodha, O., Firman, M., and Brostow, G. J. (2019). Digging into self-supervised monocular depth prediction. In *The International Conference on Computer Vision (ICCV)*.

Guizilini, V., Ambrus, R., Pillai, S., Raventos, A., and Gaidon, A. (2020). 3d packing for self-supervised monocular depth estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Hartley, R. and Zisserman, A. (2003). *Multiple View Geometry in Computer Vision*. Cambridge University Press, New York, NY, USA, 2 edition.

He, K., Zhang, X., Ren, S., and Sun, J. (2014). Spatial pyramid pooling in deep convolutional networks for visual recognition. In *13th European Conference on Computer Vision (ECCV)*, pages 346–361.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

Hendy, N., Sloan, C., Tian, F., Duan, P., Charchut, N., Xie, Y., Wang, C., and Philbin, J. (2020). Fishing net: Future inference of semantic heatmaps in grids. *ArXiv*, abs/2006.09917.

Himstedt, M. and Maehle, E. (2017). Online semantic mapping of logistic environments using rgb-d cameras. *International Journal of Advanced Robotic Systems*, 14:172988141772078.

Hinton, G. E., Vinyals, O., and Dean, J. (2015). Distilling the knowledge in a neural network. *ArXiv*, abs/1503.02531.

Howard, A., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., and Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *ArXiv*, abs/1704.04861.

Huang, X., Cheng, X., Geng, Q., Cao, B., Zhou, D., Wang, P., Lin, Y., and Yang, R. (2018). The ApolloScape dataset for autonomous driving. In *Conference on Computer Vision and Pattern Recognition Workshops (CVPR Workshops)*, pages 954–960.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Bach, F. and Blei, D., editors, *International Conference on Machine Learning*, pages 448–456.

Jacobs, R. A., Jordan, M. I., Nowlan, S. J., and Hinton, G. E. (1991). Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87.

Kesten, R., Usman, M., Houston, J., Pandya, T., Nadhamuni, K., Ferreira, A., Yuan, M., Low, B., Jain, A., Ondruska, P., Omari, S., Shah, S., Kulkarni, A., Kazakova, A., Tao, C., Platinsky, L., Jiang, W., and Shet, V. (2019). Lyft level 5 perception dataset 2020. https://level5.lyft.com/dataset/.

Kim, J. and Canny, J. F. (2017). Interpretable learning for self-driving cars by visualizing causal attention. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 2961–2969.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Krähenbühl, P. and Koltun, V. (2011). Efficient inference in fully connected crfs with gaussian edge potentials. In *Advances in Neural Information Processing Systems 24 (NIPS 2011)*, pages 109–117.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012a). Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25 (NIPS 2012)*, pages 1106–1114.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012b). Imagenet classification with deep convolutional neural networks. In Pereira, F., Burges, C. J. C., Bottou, L., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems*, volume 25, pages 1097–1105. Curran Associates, Inc.

Laina, I., Rupprecht, C., Belagiannis, V., Tombari, F., and Navab, N. (2016). Deeper depth prediction with fully convolutional residual networks. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 239–248.

Laina, I., Rupprecht, C., Belagiannis, V., Tombari, F., and Navab, N. (2016). Deeper depth prediction with fully convolutional residual networks. In *Fourth International Conference on 3D Vision (3DV)*, pages 239–248.

Lang, A. H., Vora, S., Caesar, H., Zhou, L., Yang, J., and Beijbom, O. (2019). Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Lee, J. H., Han, M.-K., Ko, D. W., and Suh, I. H. (2019). From big to small: Multi-scale local planar guidance for monocular depth estimation. *arXiv preprint arXiv:1907.10326*.

Li, H., Xiong, P., An, J., and Wang, L. (2018a). Pyramid attention network for semantic segmentation. In *BMVC*.

Li, H., Xiong, P., Fan, H., and Sun, J. (2019). Dfanet: Deep feature aggregation for real-time semantic segmentation. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9514–9523.

Li, Y. and Ruichek, Y. (2014). Occupancy grid mapping in urban environments from a moving on-board stereo-vision system. *Sensors*, 14(6):10454–10478.

Li, Z., Motoyoshi, T., Sasaki, K., Ogata, T., and Sugano, S. (2018b). Rethinking self-driving: Multi-task knowledge for better generalization and accident explanation ability. *ArXiv*, abs/1809.11100.

Lin, G., Shen, C., van dan Hengel, A., and Reid, I. (2016). Efficient piecewise training of deep structured models for semantic segmentation. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3194–3203.

Lin, T., Dollár, P., Girshick, R., He, K., Hariharan, B., and Belongie, S. (2017). Feature pyramid networks for object detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 936–944.

Liu, W., Rabinovich, A., and Berg, A. (2015). Parsenet: Looking wider to see better. *arXiv preprint arXiv:1506.04579*.

Lu, C., van de Molengraft, M. J. G., and Dubbelman, G. (2018). Monocular semantic occupancy grid mapping with convolutional variational encoder–decoder networks. *IEEE Robotics and Automation Letters*, 4:445–452.

Mani, K., Daga, S., Garg, S., Narasimhan, S. S., Krishna, M., and Jatavallabhula, K. M. (2020). Monolayout: Amodal scene layout from a single image. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 1689–1697.

Mohajerin, N. and Rohani, M. (2019). Multi-step prediction of occupancy grid maps with recurrent neural networks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10592–10600.

Montemerlo, M., Becker, J., Bhat, S., Dahlkamp, H., Dolgov, D., Ettinger, S. M., Hähnel, D., Hilden, T., Hoffmann, G., Huhnke, B., Johnston, D., Klumpp, S., Langer, D., Levandowski, A., Levinson, J., Marcil, J., Orenstein, D., Paefgen, J., Penny, I., Petrovskaya, A., Pflueger, M., Stanek, G., Stavens, D., Vogt, A., and Thrun, S. (2008). Junior: The stanford entry in the urban challenge. *Journal of Field Robotics*, 25:569–597.

Mueller, M., Dosovitskiy, A., Ghanem, B., and Koltun, V. (2018). Driving policy transfer via modularity and abstraction. In *2nd Annual Conference on Robot Learning, CoRL 2018, Zürich, Switzerland, 29-31 October 2018, Proceedings*, pages 1–15.

Mur-Artal, R. and Tardós, J. D. (2017). Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Transactions on Robotics*, 33(5):1255–1262.

Neuhold, G., Ollmann, T., Rota Bulò, S., and Kontschieder, P. (2017). The mapillary vistas dataset for semantic understanding of street scenes. In *International Conference on Computer Vision (ICCV)*.

Oh, S.-I. and Kang, H.-B. (2016). Fast occupancy grid filtering using grid cell clusters from lidar and stereo vision sensor data. *IEEE Sensors Journal*, 16:7258–7266.

Paden, B., Cáp, M., Yong, S. Z., Yershov, D. S., and Frazzoli, E. (2016). A survey of motion planning and control techniques for self-driving urban vehicles. *IEEE Transactions on Intelligent Vehicles*, 1:33–55.

Pan, B., Sun, J., Leung, H. Y. T., Andonian, A., and Zhou, B. (2020). Cross-view semantic segmentation for sensing surroundings. *IEEE Robotics and Automation Letters*, 5:4867–4873.

Paszke, A., Chaurasia, A., Kim, S., and Culurciello, E. (2016). Enet: A deep neural network architecture for real-time semantic segmentation. *arXiv preprint arXiv:1606.02147*.

Philion, J. and Fidler, S. (2020). Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. In *Proceedings of the European Conference on Computer Vision*.

Pomerleau, D. (1988). Alvinn: An autonomous land vehicle in a neural network. In *NIPS*.

Qian, R., Garg, D., Wang, Y., You, Y., Belongie, S., Hariharan, B., Campbell, M., Weinberger, K. Q., and Chao, W. L. (2020). End-to-end pseudo-lidar

for image-based 3d object detection. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5880–5889.

Riba, E., Fathollahi, M., Chaney, W., Rublee, E., and Bradski, G. (2018). torchgeometry: when pytorch meets geometry.

Roddick, T. and Cipolla, R. (2020). Predicting semantic map representations from images using pyramid occupancy networks. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11135–11144.

Roddick, T., Kendall, A., and Cipolla, R. (2019). Orthographic feature transform for monocular 3d object detection. *British Machine Vision Conference*.

Romera, E., Alvarez, J. M., Bergasa, L. M., and Arroyo, R. (2018). Erfnet: Efficient residual factorized convnet for real-time semantic segmentation. *IEEE Trans. Intelligent Transportation Systems*, 19(1):263–272.

Rong, G., Shin, B. H., Tabatabaee, H., Lu, Q., Lemke, S., Možeiko, M., Boise, E., Uhm, G., Gerow, M., Mehta, S., Agafonov, E., Kim, T. H., Sterner, E., Ushiroda, K., Reyes, M., Zelenkovsky, D., and Kim, S. (2020). Lgsvl simulator: A high fidelity simulator for autonomous driving.

Sauer, A., Savinov, N., and Geiger, A. (2018). Conditional affordance learning for driving in urban environments. In *CoRL*.

Schreiber, M., Hörmann, S., and Dietmayer, K. (2019). Long-term occupancy grid prediction using recurrent neural networks. *2019 International Conference on Robotics and Automation (ICRA)*, pages 9299–9305.

Schulter, S., Zhai, M., Jacobs, N., and Chandraker, M. (2018). Learning to look around objects for top-view representations of outdoor scenes. In *Computer Vision – ECCV 2018*, pages 815–831, Cham. Springer International Publishing.

Schwing, A. G. and Urtasun, R. (2015). Fully connected deep structured networks. *ArXiv*, abs/1503.02351.

Shah, S., Dey, D., Lovett, C., and Kapoor, A. (2018). Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and service robotics*, pages 621–635. Springer.

Shelhamer, E., Long, J., and Darrell, T. (2017). Fully convolutional networks for semantic segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(4):640–651.

Shi, X., Chen, Z., Wang, H., Yeung, D., Wong, W., and Woo, W. (2015). Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 802–810.

Si, H., Zhang, Z., Lv, F., Yu, G., and Lu, F. (2019). Real-time semantic segmentation via multiply spatial fusion network.

Singh, S. (2015). Critical reasons for crashes investigated in the national motor vehicle crash causation survey. Technical Report DOT HS 812 115, National Highway Traffic Safety Administration, Washington, D.C., USA.

Srikanth, S., Ansari, J. A., Ram, R. K., Sharma, S., Murthy, J. K., and Krishna, K. M. (2019). Infer: Intermediate representations for future prediction. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 942–949.

Sun, P., Kretzschmar, H., Dotiwalla, X., Chouard, A., Patnaik, V., Tsui, P., Guo, J., Zhou, Y., Chai, Y., Caine, B., Vasudevan, V., Han, W., Ngiam, J., Zhao, H., Timofeev, A., Ettinger, S., Krivokon, M., Gao, A., Joshi, A., Zhang, Y., Shlens, J., Chen, Z.-F., and Anguelov, D. (2020). Scalability in perception for autonomous driving: Waymo open dataset. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2443–2451.

Takikawa, T., Acuna, D., Jampani, V., and Fidler, S. (2019). Gated-scnn: Gated shape cnns for semantic segmentation. *ICCV*.

Tao, A., Sapra, K., and Catanzaro, B. (2020). Hierarchical multi-scale attention for semantic segmentation. *ArXiv*, abs/2005.10821.

Teichmann, M. and Cipolla, R. (2019). Convolutional crfs for semantic segmentation. In *30th British Machine Vision Conference 2019, BMVC 2019, Cardiff, UK, September 9-12, 2019*, page 142. BMVA Press.

Ulmer, B. (1994). Vita ii-active collision avoidance in real traffic. In *Proceedings of the Intelligent Vehicles '94 Symposium*, pages 1–6.

Urmson, C., Anhalt, J., Bagnell, J. A. D., Baker, C. R., Bittner, R. E., Dolan, J. M., Duggins, D., Ferguson, D., Galatali, T., Geyer, H., Gittleman, M., Harbaugh, S., Hebert, M., Howard, T., Kelly, A., Kohanbash, D., Likhachev, M., Miller, N., Peterson, K., Rajkumar, R., Rybski, P., Salesky, B., Scherer, S., Seo, Y.-W., Simmons, R., Singh, S., Snider, J. M., Stentz, A. T., Whittaker, W. R. L., and Ziglar, J. (2007). Tartan racing: A multimodal approach to the darpa urban challenge. Technical report, Carnegie Mellon University, Pittsburgh, PA.

Wang, D., Devin, C., Cai, Q., Krähenbühl, P., and Darrell, T. (2019). Monocular plan view networks for autonomous driving. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2876–2883.

Wang, S., Clark, R., Wen, H., and Trigoni, N. (2017). Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2043–2050.

Wang, Y., Chao, W.-L., Garg, D., Hariharan, B., Campbell, M., and Weinberger, K. (2019a). Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving. In *CVPR*.

Wang, Z., Liu, B., Schulter, S., and Chandraker, M. (2019b). A parametric top-view representation of complex road scenes. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10317–10325.

Watson, J., Firman, M., Monszpart, A., and Brostow, G. J. (2020). Footprints and free space from a single color image. In *Computer Vision and Pattern Recognition (CVPR)*.

Wofk, Diana and Ma, Fangchang and Yang, Tien-Ju and Karaman, Sertac and Sze, Vivienne (2019). FastDepth: Fast Monocular Depth Estimation on Embedded Systems. In *IEEE International Conference on Robotics and Automation (ICRA)*.

Wulfmeier, M., Wang, D. Z., and Posner, I. (2016). Watch this: Scalable cost-function learning for path planning in urban environments. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2089–2095.

Xiao, Y., Codevilla, F., Gurram, A., Urfalioglu, O., and López, A. M. (2019). Multimodal end-to-end autonomous driving. *CoRR*, abs/1906.03199.

Xie, J., Girshick, R., and Farhadi, A. (2016). Deep3d: Fully automatic 2d-to-3d video conversion with deep convolutional neural networks. In Leibe, B., Matas, J., Sebe, N., and Welling, M., editors, *Computer Vision – ECCV 2016*, pages 842–857. Springer International Publishing.

Xu, H., Gao, Y., Yu, F., and Darrell, T. (2016). End-to-end learning of driving models from large-scale video datasets. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3530–3538.

Yu, C., Wang, J., Peng, C., Gao, C., Yu, G., and Sang, N. (2018). Bisenet: Bilateral segmentation network for real-time semantic segmentation. In *ECCV*.

Yu, F., Chen, H., Wang, X., Xian, W., Chen, Y., Liu, F., Madhavan, V., and Darrell, T. (2020). Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yu, F. and Koltun, V. (2016). Multi-scale context aggregation by dilated convolutions. In *ICLR*.

Zeng, W., Luo, W., Suo, S., Sadat, A., Yang, B., Casas, S., and Urtasun, R. (2019). End-to-end interpretable neural motion planner. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zhang, Y., Wang, W., Bonatti, R., Maturana, D., and Scherer, S. (2018). Integrating kinematics and environment context into deep inverse reinforcement learning for predicting off-road vehicle trajectories. In *CoRL*.

Zhao, H., Qi, X., Shen, X., Shi, J., and Jia, J. (2018). ICNet for real-time semantic segmentation on high-resolution images. In *ECCV*.

Zhao, H., Shi, J., Qi, X., Wang, X., and Jia, J. (2017). Pyramid scene parsing network. In *CVPR*.

Zheng, S., Jayasumana, S., Romera-Paredes, B., Vineet, V., Su, Z., Du, D., Huang, C., and Torr, P. H. S. (2015). Conditional random fields as recurrent neural networks. In *International Conference on Computer Vision (ICCV)*, pages 1529–1537.

Zhou, T., Brown, M., Snavely, N., and Lowe, D. G. (2017). Unsupervised learning of depth and ego-motion from video. In *CVPR*.

Zhou Wang, Bovik, A. C., Sheikh, H. R., and Simoncelli, E. P. (2004). Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612.

Ziebart, B. D., Maas, A., Bagnell, J. A., and Dey, A. K. (2008). Maximum entropy inverse reinforcement learning. In *Proc. AAAI*, pages 1433–1438.

Ziebart, B. D., Ratliff, N. D., Gallagher, G., Mertz, C., Peterson, K. M., Bagnell, J., Hebert, M., Dey, A. K., and Srinivasa, S. (2009). Planning-based prediction for pedestrians. *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3931–3936.