A Master’s Thesis in Mechatronics Engineering

Predicting Depth from Semantic Segmentation using Game Engine Dataset

by

Mohammad Amin Kashi

Supervisor

Hamid D. Taghirad

Summer 2020
لا يوجد نص يمكن قراءته بشكل طبيعي من الصورة المقدمة.
# تاییدیه هیات داوران

**بسمه تعالی**

**تاسیس 1307 دانشگاه صنعتی خواجه نصیرالدین طوسی**

هیأت داوران پس از مطالعه پایان‌نامه و شرکت در جلسه دفاع از پایان‌نامه تهیه شده تحت عنوان:

**Predicting Depth**

بیش بینی عمق از تقسیم‌بندی معناپذیر با استفاده از داده‌های موتور بازی سازی (from Semantic Segmentation using Game Engine Dataset)

توسط آقای محمد امین کاشی، صحت و کفایت تحقیق انجام شده را برای اخذ درجه کارشناسی ارشد رشته مکاترونیک در تاریخ 1399/6/24 مورد تایید قرار می‌گیرد.

| شماره | نشانی | امضای داوران | تاریخ |
|-------|-------|---------------|-------|
| 1     | استاد راهنما اول |  | 29/10/1387 |
| 2     | استاد راهنما دوم |  |  |
| 3     | استاد مشاور |  |  |
| 4     | ممتحن داخلی |  |  |
| 5     | ممتحن خارجی |  |  |
| 6     | نماینده تخصصی |  |  |

9/10/1387-4216
دانشجوی دانشگاه صنعتی خواجه نصیرالدین طوسی

آرایه‌نامه دانشجو

شماره: 

تاریخ: 

امضاء دانشجو:

خ: 

تاریخ: 

شیب بینی عمق از تقسیم‌بندی معنی‌پذیر با استفاده از داده‌های موتور باری سازی (Predicting Depth from Semantic Segmentation using Game Engine Dataset)

این انجمن محمدرضا کمالی دانشجوی کارشناسی ارشد رشته مکانیک دانشکده برق دانشگاه صنعتی خواجه نصیرالدین طوسی گواهی می‌نمایم که تحقیقات ارائه‌شده در پایان‌نامه با عنوان "Predicting Depth from Semantic Segmentation using Game Engine Dataset" با راهنمایی استاد محترم جناب آقای دکتر حمید رضا تقي راد، توسط شخص انجنیاب انجام شده و صحبت و استاد مطالعه نگارش شده در این پایان‌نامه مورد تأیید می‌باشد، و در مورد استفاده از کار دیگر محققان به مرجع مورد استفاده اشاره شده است. بعلاوه گواهی می‌نمایم که مطالعه مورد در پایان‌نامه تا کنون برای دریافت هیچ نوع مدرک یا امتیازی توسعه انجنیاب یا هر دیگری در هیچ جا ارائه‌شده است و در تدوین متن پایان‌نامه چارچوب (فرمت) مصوب دانشگاه را بطور کامل رعایت کرده‌ام.

امضاء دانشجو:

تاریخ: 

1399/7/20

۲۹/۱۰/۱۳۸۷-۴۲۱۷
باسمه تعالی

شماره: 
تاریخ: 
عنوان فرم
حق طبع و نشر و مالکیت نتایج

این جانب محمد امین کاشی دانشجوی دوره کارشناسی ارشد مهندسی مکاترونیک دانشگاه برق دانشگاه صنعتی نصیرالدین طوسی

خواهان نصیرالدین طوسی گوگاهی می‌نامیم که تحقیقات ارائه شده در این پایان نامه توسط شخص اینجانب انجام شده و صحت و اصلت نگارش شده مورد تایید می‌باشد و در موارد استفاده از کار دیگر محققان به مرجع مورد استفاده اشاره شده است. به علاوه گوگاهی می‌نامیم که مطالب مندرج در پایان نامه تاکنون برای اهداف هیچ نوع مدرک یا امتیازی توسط اینجانب یا فرد دیگری در هیچ چارچوب مصوب دانشگاه را پرداخته نشده است.

کامل رعایت کردم.

کلیه حقوق مادی و معنی این اثر فقط متعلق به دانشگاه صنعتی خواهان نصرالدین طوسی می‌باشد و بدون اجازه کننی دانشگاه قابل و اگرگذاری و بهره برداری نیست. همچنین استفاده از اطلاعات و نتایج موجود در پایان نامه بدون ذکر مرجع مجاز نمی‌باشد.

امضا دانشجو: 
تاریخ: 1399/7/20
Acknowledgements

I would like to express my gratitude to my supervisor Prof. Hamid D. Taghirad for the useful comments, remarks, and engagement through the learning process of this master thesis. I would also like to thank Advanced Robotics and Automated System (ARAS) team members for their comments on my work. Furthermore, I would like to thank my friends, specially Erfan Khaniki, for their encouragement during the research process.

Finally, I must express my very profound gratitude to my parents, my brother, and my sister for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them. Thank you.

Tehran, September 26, 2020
Mohammad Amin Kashi
Abstract

Depth perception is fundamental for robots to understand the surrounding environment. As the view of cognitive neuroscience, visual depth perception methods are divided into three categories, namely binocular, active, and pictorial. The first two categories have been studied for decades in detail. However, research for the exploration of the third category is still in its infancy and has got momentum by the advent of deep learning methods in recent years. In cognitive neuroscience, it is known that pictorial depth perception mechanisms are dependent on the perception of seen objects. Inspired by this fact, in this thesis, we investigated the relation of perception of objects and depth estimation convolutional neural networks. For this purpose, we developed new network structures based on a simple depth estimation network that only used a single image at its input. Our proposed structures use both an image and a semantic label of the image as their input. We used semantic labels as the output of object perception. The obtained results of performance comparison between the developed network and original network showed that our novel structures can improve the performance of depth estimation by 52% of relative error of distance in the examined cases. Most of the experimental studies were carried out on synthetic datasets that were generated by game engines to isolate the performance comparison from the effect of inaccurate depth and semantic labels of non-synthetic datasets. It is shown that particular synthetic datasets may be used for training of depth networks in cases that an appropriate dataset is not available. Furthermore, we showed that in these cases, usage of semantic labels improves the robustness of the network against domain shift from synthetic training data to non-synthetic test data.
Keywords: Depth estimation, pictorial depth cues, semantic segmentation, synthetic dataset, deep convolutional neural network.


# Contents

List of Figures iii

List of Tables v

Symbols and Acronyms vi

1 Introduction 1

1.1 Preface ....................................................... 1

1.2 Taxonomy of Depth Estimation Algorithms ..................... 2

1.2.1 Binocular Depth Cues ....................................... 2

1.2.2 Active Depth Cues .......................................... 2

1.2.3 Pictorial Depth Cues ........................................ 3

1.3 Obtaining Pictorial Cues ...................................... 4

1.4 Research Goals, Methods, and Results ........................... 5

1.5 Datasets ..................................................... 7

1.5.1 Dataset Adaptation and Domain Adaptation .................. 10

2 Depth from Semantic Labels 12

2.1 Methods .................................................... 12

2.1.1 Models ................................................... 12

2.1.2 Performance Evaluation and Training Details ................. 22

2.1.3 Datasets ................................................ 25

2.2 Results .................................................... 25

3 Synthetic Data instead of Non-Synthetic Data 33
## List of Figures

1.1 Inaccurate or false segmentation by a human ........................................... 9  
1.2 Snowy point cloud collected by LIDAR ...................................................... 10  
1.3 Relation of AFOV with focal length and sensor size ................................. 11  
2.1 Up-sampling block .............................................................................. 13  
2.2 FCRN architecture ............................................................................. 14  
2.3 M0-M2 Network Architectures ............................................................... 16  
2.4 Semantically masked image .................................................................. 17  
2.5 M3-M4 Network Architectures ............................................................... 18  
2.6 M5-M7 Network Architecture ................................................................ 20  
2.7 M18-M21 Network Architecture ............................................................ 23  
2.8 Performance comparison of SIDE DNNs ............................................... 26  
2.9 Performance comparison of SSIDE DNNs .............................................. 27  
2.10 Performance comparison of M0, M20, M18, and M21 ............................ 28  
2.11 Accuracy heat map of M0 and M21 ....................................................... 32  
3.1 U-Net architecture ............................................................................... 34  
3.2 Intersection over Union .......................................................................... 35  
3.3 Performance of U-Net trained on Cityscapes ......................................... 37  
3.4 Performance of U-Net trained on MAPILLARY ...................................... 38  
3.5 Performance of U-Net trained on SYNTHIA-SF ..................................... 39  
3.6 Performance of U-Net trained on VIPER .............................................. 40  
3.7 Performance of different U-Net networks on CITYSCAPES ................... 41  
3.8 Performance of different U-Net networks on MAPILLARY ..................... 42
4.1 Depth heat map of KITTI dataset ........................................... 47
4.2 Depth heat map of Lyft Level 5 dataset ................................. 47
4.3 Depth heat map of Virtual KITTI 2 dataset ............................. 47
4.4 Depth heat map of PreSIL dataset ........................................ 48
4.5 Depth heat map of SYNTHIA-SF dataset ............................... 49
4.6 Examples of generated semantic labels for KITTI dataset .......... 50
4.7 Performance Comparison on the Lyft Level 5 dataset ............... 51
4.8 Performance Comparison on the KITTI dataset ....................... 52

A.1 Different encodings for semantic segments ............................ 57

C.1 Depth estimation samples of M18 and M20 .......................... 60
C.2 Perception sample of Cityscapes U-Net on Cityscapes .......... 61
C.3 Perception sample of Cityscapes U-Net on Mapillary ............ 61
C.4 Perception sample of Cityscapes U-Net on VIPER ................. 62
C.5 Perception sample of Cityscapes U-Net on SYNTHIA-SF ........ 62
C.6 Perception sample of Mapillary U-Net on Cityscapes .......... 63
C.7 Perception sample of Mapillary U-Net on Mapillary ............ 63
C.8 Perception sample of Mapillary U-Net on VIPER ................. 64
C.9 Perception sample of Mapillary U-Net on SYNTHIA-SF ......... 64
C.10 Perception sample of SYNTHIA-SF U-Net on Cityscapes ....... 65
C.11 Perception sample of SYNTHIA-SF U-Net on Mapillary ....... 65
C.12 Perception sample of SYNTHIA-SF U-Net on VIPER ........... 66
C.13 Perception sample of SYNTHIA-SF U-Net on SYNTHIA-SF .... 66
C.14 Perception sample of VIPER U-Net on Cityscapes .............. 67
C.15 Perception sample of VIPER U-Net on Mapillary .............. 67
C.16 Perception sample of VIPER U-Net on VIPER ................. 68
C.17 Perception sample of VIPER U-Net on SYNTHIA-SF ............ 68
List of Tables

2.1 Performance comparison of SSIDE and SIDE models . . . . . . . . . . . . . . 29
2.2 Relative superiority of the M21 and the M18 over the M0 and the M20 29
4.1 Unification of AFOV between datasets . . . . . . . . . . . . . . . . . . . . 46
B.1 Common semantic labels set . . . . . . . . . . . . . . . . . . . . . . . . . . 59
Symbols and Acronyms

AFOV  angular field of view.
CNN  convolutional neural networks.
CRF  conditional random field.
FCRN  fully convolutional residual network.
GPS  global positioning system.
IMU  inertial measurement unit.
IoU  intersection over union.
LIDAR  light detection and ranging.
MAPE  mean absolute percentage error.
MRF  markov random field.
MSPE  mean square percentage error.
RADAR  radio detection and ranging.
ReLU  rectified linear unit.
RMSE  root mean square error.
RMSElog  root mean square error in log space.
SfM  structure from motion.
SIDE  single image depth estimation.
SIlog  scale-invariant error.
SONAR  sound navigation and ranging.
SSIDE  semantic single image depth estimation.
Chapter 1

Introduction

1.1 Preface

Depth perception is fundamental for robots to understand the surrounding environment. This perception guides robots to know where the objects in their environment are. Various sensors including sound navigation and ranging (SONAR), light detection and ranging (LIDAR), radio detection and ranging (RADAR), and camera can be used to accommodate a robot with depth perception. Among these sensors, cameras are the cheapest and widely available in various locations such as factories, streets, shopping centers and systems such as mobile phones, cars, drones, surveillance camera systems. Furthermore, cameras are used for different purposes such as security, traffic monitoring, mapping, retail analysis, and industrial automation. Therefore, images of cameras may be widely used for depth estimation in different applications.

Due to its importance, depth estimation from images has received a large amount of attention in the literature. These researches are inspired by studying the mechanisms underlying human vision, as the whole computer vision is inspired by the human vision. Therefore, proposed algorithms may be distinguished into the same categories as that of the depth perception mechanisms of human vision.
1.2 Taxonomy of Depth Estimation Algorithms

In cognitive neuroscience, different sources of visual information about distance and depth are called depth cues and are divided into three kinds. The first and second kinds are called binocular cues and active cues, respectively. The third kind is called pictorial cues, which are used for single image depth estimation (SIDE)\(^1\). Using this classification, mechanisms of depth perception may be classified by their source of information.\(^2\)

1.2.1 Binocular Depth Cues

Binocular cues include image disparity and orientation of optical sensors. Since optical sensors are fixed relative to each other in a stereo camera, the later cue has not been investigated in computer vision. In contrast, the binocular disparity has been extremely studied (for reviews of works done from the mid-70’s up to 2020 refer to [3]–[6]). Since the relation of disparity map and depth map is known, in many of the proposed algorithms only estimation of disparity map is addressed.

1.2.2 Active Depth Cues

Active cues include accommodation information and motion parallax information. Several works proposed algorithms that use accommodation information to estimate the depth of a scene (e.g. [7]–[9]). For this purpose, a sequence of images is captured from the scene, while the amount of accommodation is changed. Relative to accommodation information, motion parallax information has a more active research area (for reviews of related works refer to [10]–[13]). Depth estimation algorithms related to motion parallax are called structure from motion (SfM). Motion parallax is a special case of parallax\(^3\) in which different views are obtained by motion at different

---

\(^1\)This classification is borrowed from [1]. In another classification depth cues are divided into three kinds of oculomotor cues, monocular cues, and binocular cues [2].

\(^2\)In some papers, the term *monocular depth estimation* was used to describe the estimation of depth from pictorial cues. This designation is ambiguous because *monocular depth estimation* also applies to depth estimation from active cues. Therefore, only the term SIDE is considered in this thesis.

\(^3\)Difference or change in the apparent position or direction of an object as seen from two different points [14]
moments. Another example of parallax occurs when a binocular vision system is used, while different views are obtained by two eyes or cameras at the same moment.

1.2.3 Pictorial Depth Cues

Between the three kinds of depth cues, the third kind contains the highest number of cues. Pictorial cues are:

- Occlusion: Nearby objects occlude farther objects.
- Relative position: By knowing the relative position of an object to another object that its depth is known, depth estimation of the first object becomes possible.
- Size of objects: Image size of a specific object varies with distance.
- Density of textures’ gradient: It varies with distance.
- Image quality: Contrast of an image decreases with distance because of light diffusion by particles in the air.

Until the last decade, few studies investigated the depth estimation using pictorial cues. With the widespread use of deep learning methods, more and more researches have been focused on pictorial cues.

Some works tried to estimate the depth from a specific cue in special situations. Aslantas proposed a numerical model that estimates the depth of an object, provided the defocused image and the sharp image of the object are available [15]. The parameters of the model should be recalculated for different cameras. In [16] the relative position cue is used to estimate the distance of a vehicle on a road. The proposed method first detects the contact point of the vehicle on the road, then by assuming a planar road and a known camera height and using geometry rules the distance of the contact point is calculated and is reported as the distance of the vehicle. Another representative research on this area is the use of a haze removal method as a tool for estimating the depth of a scene [17].
By imposing the burden of extracting the pictorial cues on convolutional neural networks (CNN), the majority of works are focused to directly estimate the depth. Eigen et al. proposed the first successful CNN to estimate depth from a single image [18]. They designed a two-stage architecture; the first stage makes a coarse and global prediction, while the second stage enhances this prediction locally. They also used a scale-invariant loss function to train the proposed architecture regardless of scale. This loss function becomes a standard metric in performance evaluation of depth estimation, e.g. in KITTI depth benchmark [19]. The later works tried to propose better models in terms of performance by using novel architectures [20]–[23], new cost functions [24]–[27], or multi-task architectures [28]–[36].

1.3 Obtaining Pictorial Cues

Although various works addressed the problem of estimating depth from pictorial cues for SIDE, not enough attention has been paid to the relation of pictorial cues and depth estimation. In most of the proposed methods, a CNN is responsible for extracting the depth cues implicitly and estimating depth from those cues; however, most of the pictorial cues are high-level abstract concepts and logically need a separate CNN to extract them with high accuracy. For example, the recognition of the occlusion between two objects already needs the recognition of the two objects, as the determination of the position of one object relative to another object needs the recognition of the two objects. In a similar way, the size determination of an object already needs the recognition of that object, and the determination of density of a texture needs that the texture be detected first.

As mentioned above a common prerequisite for obtaining pictorial cues is to detect and locate different objects and textures in an image; a task that is somewhat similar to semantic segmentation. For the same reason the most of the proposed multi-task CNNs (or conditional random field (CRF)) were designed to estimate depth and simultaneously segment the input image semantically [28]–[36]. Simultaneous learning enables these models to better extract pictorial cues and estimate depth. However, it is plausible that a large aspect of the information, that semantic
segments could have delivered to the model, is not used. As stated earlier, semantic
segmentation is a prerequisite for obtaining pictorial depth cues; therefore, high-
level abstract concepts can not be obtained from the semantic segments in these
multi-task models, because the semantic segments are an output of the model and
can not be processed anymore.

In some other works, semantic segments were used in other various methods.
Schneider et al. used the edges of semantic segments as a guide for up-sampling of
sparse depth data of LIDAR [37]. Guizilini et al. used a pre-designed and pre-trained
segmentation model in parallel with a pre-designed depth estimation model instead
of designing a multi-task CNN. During training and test of the depth model, the
segmentation model injects low-level and high-level features from different layers of
its network to the corresponding layers of the depth model [38]. Liu et al. proposed
the first depth estimation model that uses predicted semantic segments as its input.
They used an markov random field (MRF) for semantic segmentation and an MRF
for predicting depth from both images and semantic segments [39]. Furthermore,
Jafari et al. analyzed the mutual impact of depth and semantic segments in refining
each other. They designed a CNN that uses an estimated depth map and a semantic
segment map as its inputs and jointly refines them [40].

1.4 Research Goals, Methods, and Results

With the assumption that semantic segmentation is a prerequisite for obtaining
pictorial depth cues, this thesis investigates how the use of semantic segments as
a raw data, like an input image, can potentially improve the performance of a
SIDE CNN, and if there is an improvement, how significant it is. This thesis also
investigates the effect of the use of semantic segments on the robustness of a SIDE
model against domain shift.

Experimental Study on Estimation of Depth from Semantic Labels:
For the first goal of the thesis, a SIDE CNN is chosen to be empowered by semantic
segments as its input, to drive an semantic single image depth estimation (SSIDE)
CNNs. After the selection of a SIDE CNN, several similar networks are developed from it. These networks use semantic segments along with an image as inputs, in contrast to the original network that only uses an image as the input. Furthermore, the structures of the derived networks have some differences from the first network. The differences are formed to facilitate the utilization of extra data in semantic segments.

For the evaluation of performances, the driven SSIDE CNNs are compared with the original SIDE CNN on estimating depth on specific datasets. In fact, all of the experiment is carried out on a synthetic dataset to eliminate the altering effect of noisy and sparse depth labels and inaccurate semantic segments of non-synthetic datasets. At the end, some parts of the experiment are repeated on two other datasets, one synthetic and another non-synthetic, as described in section 2.1. The results show that the SSIDE CNNs outperform the SIDE CNN by 18 percent of the absolute relative error in the examined situations (For complete results see section 2.2).

**Experimental Study on Robustness of SSIDE against Domain Shift:**
For the second objective of this thesis, an SSIDE model and a SIDE model are trained on a synthetic dataset (a combination of multiple synthetic datasets), and two other SIDE models are trained on two different non-synthetic datasets. Note that all SIDE models have the same architecture. All models are tested on the test-set of the two non-synthetic datasets, as described in section 4.1. Results show that the SSIDE CNN performs like the two SIDE models that were trained on non-synthetic datasets, while the SIDE model that was trained on the synthetic dataset has a poor performance relative to others. In other words, use of semantic segments improves the robustness of the model against domain shift (For complete results see section 4.2.).

**Experimental Study on Use of Synthetic Data instead of Non-Synthetic Data:** An SSIDE model needs both an image and semantic segments to estimate depth. From here onwards, models that only use an image as their input for depth estimation are referred to as SIDE models, and models that use an image and a semantic label as their inputs for depth estimation are referred to as SSIDE models.
depth. The used datasets in the first experimental study include all needed data: image, semantic segments, and depth. The non-synthetic datasets in the final experimental study do not have semantic labels. To provide semantic segments for these datasets, one way is to train a segmentation model on a synthetic dataset and generate segmentation for these datasets. To ensure that synthetic datasets are capable of training a segmentation model to generate accurate semantic segments for non-synthetic datasets, a supplementary experimental study was carried out before the final experimental study. In this experimental study, multiple segmentation models with the same architecture are trained on different synthetic and non-synthetic datasets. Then all models are tested on non-synthetic datasets as described in section 3.1. The results show that some of synthetic datasets can be more informative than some of non-synthetic datasets, i.e., the model that was trained on a synthetic dataset performed better than a model that was trained on a non-synthetic dataset (For complete results see section 3.2.).

1.5 Datasets

Due to the abundance and diversity of self-driving cars’ datasets, this group of datasets were mainly used in this thesis. These datasets were captured with different sensors (RGB camera, LIDAR, RADAR, inertial measurement unit (IMU), and global positioning system (GPS)) and contain different data (RGB image, depth data, semantic segments, instance segments, ego velocity, ego position, optical flow) and were captured in different weather conditions and different cities. These datasets are available in non-synthetic types and synthetic types as well. However, in the first experimental study, the non-synthetic NYU Depth Dataset V2 [41] was used too.

For two reason, synthetic datasets were used in this these:

- the available non-synthetic datasets does not have enough or doesn’t have any semantic labels at all (except NYU Depth Dataset V2).
• depth maps and semantic labels in non-synthetic datasets are not accurate and may alter the results of the experiments.

Note that you may find depth maps and semantic labels in a dataset, while they are not associated with a specific sample\textsuperscript{5}. For example KITTI benchmark \cite{19} has both semantic segmentation subset and depth subset, while they are linked to different samples. Only some of the synthetic datasets have these data synced at the same sample time.

Semantic labels in non-synthetic datasets are generated manually or automatically by AI-based software under human supervision, mainly because they are not perfect. However, generated semantic segments by a human or supervised by a human are not immune from error and suffer from inaccurate or false segmentation (Figure 1.1).

Depth data in non-synthetic datasets are usually generated by LIDAR sensors for outdoor scenes and by RGB-D cameras for indoor scenes. If a LIDAR is used for depth generation, “the sparse point clouds generated can be corrupted by measurement noise due to the internal noise of the sensor and environmental factors such as reflectivity, lighting, and precipitation” \cite{44}. The reflectivity of glasses and glossy surfaces also causes many 3D points to be missed \cite{45}. Downfall of rain and snow blocks laser beams and create noise in the point cloud (Figure 1.2). Another drawback of LIDAR is its sparsity of points which increases with distance. Generated depth labels by RGB-D cameras share the same drawbacks in addition to a limited range of view.

The existing issues of depth data and semantic labels in non-synthetic datasets do not exist in synthetic datasets, provided they are designed properly. It can be mentioned that synthetic data are completely correct. But this perfection can be considered superiority only when CNNs are capable of predicting high details and impurities in datasets cause them to malfunction.

\textsuperscript{5}By the definition of NUSCENES dataset: \cite{42} “A sample is data collected at (approximately) the same timestamp as part of a single LIDAR sweep” \cite{43}.
Figure 1.1: Inaccurate or false segmentation by a human. In these three consecutive frames, different labels are assigned to an object (inside the orange box), and inaccurate labels are assigned to another object such that the label exceeds the covered area by the object (inside the red box). The labels are from KITTI dataset.
1.5.1 Dataset Adaptation and Domain Adaptation

Since every dataset has its characteristics and distribution of data, in this thesis multiple datasets from both synthetic and non-synthetic types are examined. In this way the hypothesis of this thesis could be examined more accurately and independent of the impact of datasets’ characteristics. Moreover, the difference in the distribution of data in these datasets exposes the used model to the domain shift. In other words, the used models are challenged to adapt to different domains.\(^6\) Recently, the problem of domain adaption from synthetic data to real data has attracted lots of attention and has been investigated in several works [47]–[55]. Furthermore, this approach introduces a difficulty as well, namely dataset adaptation. Because of some specifics of datasets, they can not be used directly, and some adjustments should be applied to jointly use different datasets.

One necessary adjustment is unifying semantic segmentation policies. Almost every dataset has chosen a different policy in the classification of objects for semantic labels. The only way of unifying datasets is to merge subclasses to superclasses.

Another necessary adjustment is the unifying of the angular field of view (AFOV) of all frames (including RGB images, semantic labels, and depth maps). No two

---

\(^6\)Transfer learning and domain adaptation refer to the situation where what has been learned in one setting (i.e., distribution P1) is exploited to improve generalization in another setting (say distribution P2). In transfer learning, the learner must perform two or more different tasks, but we assume that many of the factors that explain the variations in P1 are relevant to the variations that need to be captured for learning P2. In the related case of domain adaptation, the task (and the optimal input-to-output mapping) remains the same between each setting, but the input distribution is slightly different.’ [46]
datasets have the same AFOV. In photography when you zoom in to magnify the scene, AFOV reduces. Therefore, different AFOV leads to a different perception of distances. According to optic laws, the value of AFOV is a function of the camera’s focal length and size of the camera’s sensor Figure 1.3.

According to Equation 1.1 adjusting the AFOV is possible through changing the focal length or sensor size. However, changing the focal length of the camera or the sensor size is not possible after taking a photo. However, since changing the image size is the same as changing the sensor size, one can crop the image to decrease sensor size. Therefore, AFOV can only be decreased.

\[
AFOV[^\circ] = 2 \times \tan^{-1}\left(\frac{h}{2f}\right)
\]

Figure 1.3: Relation of AFOV with focal length and sensor size. Yellow line is the sensor.
Chapter 2

Depth from Semantic Labels

This chapter describes an experimental study that analyzes the performance of several SIDE and SSIDE CNN architectures. This study aims to investigate the probable advantages of SSIDE over SIDE.

2.1 Methods

For the purpose of the study, one approach is to derive an SSIDE CNN from an existing SIDE CNN and compare their performances. The fully convolutional residual network (FCRN) model [20] was chosen as the baseline SIDE model. The FCRN has a straightforward architecture to implement which can be easily manipulated. The details of FCRN architecture and its derived architectures are described in the subsection 2.1.1, while subsection 2.1.2 elaborates on the used comparison study. The used datasets and data preparations are described in subsection 2.1.3.

2.1.1 Models

The FCRN model has an encoder-decoder architecture. This architecture is depicted in Figure 2.2. Its encoder is built upon ResNet-50 [56]. The encoder progressively decreases the size of the input data stream, allowing upper layers to extract high-level global features. The encoder consists of the residual units of the ResNet architecture. As it is seen in Figure 2.2, two kinds of residual units exist (purple units with
projection connection and green units with skip connection). Their main difference is that those residual units with convolution in their skip connection reduces the size of the data stream but the other type preserve the size of the data stream. Laina et al. used their proposed up-sampling blocks in the decoder part to yield a depth map using extracted high-level features [20]. These blocks have borrowed the idea of residual units in that they have two branches that are added together at the output and one branch plays the role of a skip connection (the lower branch in Figure 2.1). Each branch consists of four convolutions that their outputs are interleaved together. The $I$ sign in Figure 2.1 represents the process of interleaving. Overall, the up-sampling blocks increase the size of the data stream and finally generate the depth map.

Figure 2.1: Up-sampling block [20]

By selecting the baseline structure, gradually several structures have been driven from it, while derivations continued until the desirable result at the output was achieved. The structures of derived models have a lot in common with the original FCRN. Hence, only explanation of differences is necessary. However, full schematics of structures are depicted here for better understanding and reproducibility. Most of the symbols in these schematics are the same as those used in [20] for the sake of
Figure 2.2: FCRN architecture [20]
comparability. Due to being able to distinguish different models, each one is named by a number following $M$ character (as an abbreviation of the term Model). Also, note that the order of presentation of models is chronological:

**M0 Network**

This network has the same architecture as the FCRN, except in its output that has an additional up-sample block. The architecture of this network is depicted in Figure 2.3. This difference enables the M0 to produce a depth map with an equivalent resolution to the resolution of the input image. The performance results of the M0 are used as a representation for the performance of SIDE CNNs.

**M1 Network**

The difference between the M1 and the M0 lies on the inputs of the networks. In addition to RGB image, semantic label is present in the input of the M1. RGB image and semantic label are fed to the network after being concatenated. The architecture of this network is depicted in Figure 2.3. This structure is probably the simplest derivative of the FCRN architecture that integrates RGB image and semantic label to estimate depth map.

**M2 Network**

In the M2 network, the decoder is divided into two parts: a common part and a semantic part that has $n$ similar branches, while $n$ denotes the number of all semantic segment classes existing in the dataset. These branches are named *semantic decoders* and the common part is named *common decoder*. The output of the common decoder is fed to all semantic decoders. Then outputs of semantic decoders are concatenated, yielding a semantic depth output and are added up, yielding a complete depth map. Hence, this architecture has two outputs. This design was intended to permit the network separately and semantically learn the depth of each different semantic class. The architecture of this network is depicted in Figure 2.3.
Figure 2.3: M0-M2 Network Architectures
M3 Network

The M3 architecture was designed to address a probable drawback of the M2. Despite the designed decoder of M2 with multiple branches for a semantical depth prediction, there are only two outputs, and there are only two gradients used in the process of training, and the gradients are common to all branches (of semantic decoders). It seemed that semantic decoders of the M2 would not learn depth prediction exclusively. According to this issue, the matrix of semantic output with a depth of $n$ was split to $n$ matrices. The architecture of this network is depicted in Figure 2.5.

M4 Network

The goal of designing the M4 network was to improve the focus of semantic decoders on the probable bindings between RGB images and semantic segments. To achieve this goal, the semantic decoders were expanded and their shares of up-sample blocks were increased. The architecture of this network is depicted in Figure 2.5. Furthermore, a new concept was injected into the decoders by masking RGB images with semantic labels and concatenating the results with the mainstream of data at the early stages of the semantic decoders. By this means, a semantic decoder that is responsible for depth prediction of the semantic class of for example *car*, receives a semantically masked image like that shown in Figure 2.4c. Thus, semantic decoders were expected both to help the network to estimate depth maps more accurately and to help the network retrieve the depth map dimensions more precisely.
Figure 2.5: M3-M4 Network Architectures
M5 Network

The M5 architecture follows the same goal as the M4 network, but in a bit different way. In the design of the M5 network, the share of the semantic decoders was decreased and the share of the common part was increased to improve the speed of the network, both in learning and inference. In addition to feeding semantic segmented images to the semantic decoders, their outputs are masked semantically to ensure each semantic decoder exclusively estimates the depth of the assigned class to the semantic decoder. The architecture of this network is depicted in Figure 2.6.

M6 Network

The M6 network has a similar architecture to that of the M0 (Figure 2.6), while the difference is that the M6 uses edges of the input image instead of the raw image. At the input of the network, a Sobel filter extracts the edges of each channel of an RGB image. The network uses the summation of three edge channels. The purpose of building this network is to evaluate the impact of image’s edges on the perception of semantic segments. The edges of semantic segments are (almost, unless it is not visually recognizable) a subset of edges of the image. Hence, edges of semantic labels are helpful in depth estimation if image’s edges are helpful in depth estimation. Furthermore, if the performance of M6 is similar to the performance of the best network that uses semantic labels, it indicates that the quality of the depth estimation is increased because of the information in the edges of the image, not because of the information in the semantic segments.

M7 Network

The M7 is a combination of the M1 and the M6. Semantic segments layers are multiplied by the image edges form the input of the network, and the result can be named semantic edges. The architecture of this network is depicted in Figure 2.6. If M6 is not capable of improving depth estimation because of the absence of semantic segments, then the M7 may be immune to this defect. If M1 is capable of improving depth estimation, then M7 in addition to M1 capability may have robustness against
variations of color in different semantic categories and domain shift, because it uses edges of images.
M8-M17 Networks

From M8 to M17 networks a series of models that only use an image or semantic segments as their inputs are developed. From initial models to final models, progressively the size of structures was reduced such that the minimal size of the FCRN architecture that is capable of depth estimation be found. Evolution of these models leads to the design of M18, therefore, examination of their structures and their performance in the results section (section 2.2) are not necessary.

M18 Network

The M18 network is a slim version of M7. During the process of reducing network size, from the M8 to the 17, the capability of slim versions of the FCRN architecture in estimating depth was improved. The architecture of this network is depicted in Figure 2.7.

M19 Network

The M19 is a slim version of the M6 and a counterpart of M18 that only uses image edges as its input. The architecture of this network is depicted in Figure 2.7. It was examined only for comparison purpose. A comparison of M18 with M6 may not be fair because of the significant difference in their structures.

M20 Network

Same as the M19, the M20 is a slim version of M0 and a counterpart of M18 that only uses image as its input. The architecture of this network is depicted in Figure 2.7. It was examined only for comparison purpose. A comparison of M18 with M0 may not be fair because of the significant difference in their structures.

M21 Network

Same as the M19 and the M20, the M21 is a slim version of M1 and a counterpart of M18. The architecture of this network is depicted in Figure 2.7. The difference
between M21 and M18 is that M18 uses image’s edges and semantic segments while the M21 uses raw image and semantic segments.

2.1.2 Performance Evaluation and Training Details

To compare the designed networks, all of them were trained on 75 percent of the SYNTHIA-SF dataset [58] and tested on the remaining of the dataset. The number of all semantic segment classes \(n\) existing in the SYNTHIA-SF dataset is 19. The batch size for training of all networks was three \(^1\). Since the networks have different structures and different inputs, using same initial weight values is meaningless and impossible, hence no identical seed \(^2\) was used.

After each epoch of training, networks were tested on the test subset of the SYNTHIA-SF. At least all networks were trained for 50 epochs, so as at least 50 times each network was tested, and at least 50 times metrics were calculated for each network. The performances of networks were compared both by visualizing the trend of metrics during 50 epochs and by comparing mean value for each metric from epoch 20 to epoch 50.

Metrics and Loss Function

Various metrics and loss functions have been used in works reported on SIDE. The following list contains a number of these metrics; however, it is not meant to be an exhaustive list.

- mean absolute percentage error (MAPE)

\[
\frac{100}{n} \sum \left( \frac{|y - y^*|}{y^*} \right)
\]  

(2.1)

- mean square percentage error (MSPE)

\[
\frac{100}{n} \sum \left( \frac{(y - y^*)^2}{y^*} \right)
\]  

(2.2)

\(^1\)The maximum possible value for the available hardware

\(^2\)A random seed is a number (or vector) used to initialize a pseudorandom number generator. [59]
Figure 2.7: M18-M21 Network Architecture
• root mean square error (RMSE)

\[ \text{RMSE} = \sqrt{\frac{1}{n} \sum (y - y^*)^2} \]  

(2.3)

• root mean square error in log space (RMSElog)

\[ \text{RMSElog} = \sqrt{\frac{1}{n} \sum (\log y - \log y^*)^2} \]  

(2.4)

• log10

\[ \log10 = \frac{1}{n} \sum |\log_{10} y - \log_{10} y^*| \]  

(2.5)

• Delta Thresholds \((\delta_i)\)

\[ \delta_i = \frac{|\{y| \max(y, y^*) < 1.25\}|}{|\{y\}|} \]  

(2.6)

• scale-invariant error (SILog)

\[ D(y, y^*) = \frac{1}{n^2} \sum_{i,j} ((\log y_i - \log y_j) - (\log y_i^* - \log y_j^*))^2 \] 

(2.7)

\[ D(y, y^*) = \frac{1}{n} \sum d_i^2 - \frac{1}{n^2} \sum d_i d_j = \frac{1}{n} \sum d_i^2 - \frac{1}{n^2} \left( \sum d_i \right)^2 \]  

(2.8)

These metrics may be used directly or indirectly as a loss function. Since examining the impact of different loss functions on depth estimation was not the purpose of this thesis, MAPE as a typical metric, was chosen as the loss function. This metric is simple but calculates the error meaningfully. Because of its relative nature, errors in near distances are more important than errors in far distances. Therefore, the network is trained to be more accurate in near distances, as required for practical implementations. Furthermore, other metrics were calculated for detailed evaluation during the tests.
2.1.3 Datasets

First, this experimental study was performed on the SYNTHIA-SF dataset. Later, to make sure that the results were general, some parts of the experimental study were repeated on the Virtual KITTI dataset [60] and NYU depth dataset (This dataset contains 894 semantic classes, but only 21 classes were used). For all used datasets, the following data preparations were performed:

- Images values were normalized to [0, 1] range.
- Semantic segments were converted from RGB mappings to one-hot vectors.
- Images and Semantic segments were resized to 640 × 480.
- Depth data dimension were converted to meters.

2.2 Results

The comparison of performances is based on the MAPE metric on the test set of SYNTHIA-SF dataset. First, performances of SIDE and SSIDE models are compared separately in two graphs of Figure 2.8 and Figure 2.9, respectively. Then performances of two selected models from SIDE group (the best model and the original model) are compared with two selected models from SIDE group (two of the best models) in Figure 2.10.

Figure 2.8 shows the performances of the SIDE models during test and train. As it can be seen M0’s graph and the M6’s graph are twisted to each other. The same is true for the M19 and the M20. The M6 and the M19 are counterparts for the M0 and the M20, respectively. The M6 and the M19 instead of using a raw input image, pass it through the Sobel filter and use its edges. These observation imply that the usage of the image edges or the usage of the raw image, results in a fairly similar performance. Figure 2.8 also shows that the M20 and the M19 performed better than the M0 and the M6, in general.

Figure 2.9 shows the performances of the SSIDE models during test and train. As you can see, the performances of the M2, the M3, the M4, and the M5 are weaker
Figure 2.8: Performance comparison of SIDE CNNs
Figure 2.9: Performance comparison of SSIDE CNNs. Performance ranks of the networks are not the same during training and test. This shows some network are more capable in generalization.
Figure 2.10: Performance comparison of M0, M20, M18, and M21
Table 2.1: Performance comparison of SSIDE and SIDE models. This table contains the average of metrics (calculated on the test subset of SYNTHIA-SF dataset during training) between epoch 20 and 50. The last two columns are the number of trainable parameters of each network and runtime for a single prediction. Smaller values are better except for the delta thresholds, trainable parameters, and runtime.

| Model | MAPE   | MSPE   | RMSE   | RMSELog | Log10 | \(\delta_1\) | \(\delta_2\) | \(\delta_3\) | SILog | Trainable params | Runtime [s] |
|-------|--------|--------|--------|----------|-------|-------------|-------------|-------------|-------|----------------|-------------|
| M0    | 8.639  | 1607.413 | 102.129 | 0.407    | 0.114 | 0.924       | 0.962       | 0.973       | 0.181 | 62819073       | 0.120       |
| M1    | 6.852  | 933.107  | 84.719  | 0.331    | 0.088 | 0.939       | 0.974       | 0.983       | 0.113 | 62836481       | 0.172       |
| M2    | 7.836  | 1110.061 | 96.477  | 0.336    | 0.099 | 0.935       | 0.971       | 0.981       | 0.117 | 62392442       | 0.363       |
| M3    | 8.921  | 1185.751 | 97.803  | 0.351    | 0.109 | 0.925       | 0.967       | 0.979       | 0.127 | 62392442       | 0.374       |
| M4    | 9.556  | 1270.288 | 104.149 | 0.382    | 0.121 | 0.916       | 0.963       | 0.975       | 0.148 | 31776122       | 0.638       |
| M5    | 20.994 | 11255.954| 331.330 | 1.600    | 0.492 | 0.749       | 0.786       | 0.795       | 2.589 | 61599898       | 0.501       |
| M6    | 8.877  | 1592.307 | 104.419 | 0.416    | 0.120 | 0.916       | 0.960       | 0.972       | 0.184 | 62812801       | 0.120       |
| M7    | 7.131  | 1107.448 | 93.785  | 0.371    | 0.098 | 0.938       | 0.971       | 0.979       | 0.143 | 62833345       | 0.176       |
| M18   | 6.573  | 991.400  | 87.162  | 0.342    | 0.086 | 0.951       | 0.976       | 0.982       | 0.121 | 41206145       | 0.154       |
| M19   | 8.266  | 1649.762 | 93.452  | 0.373    | 0.099 | 0.937       | 0.967       | 0.976       | 0.144 | 41157313       | 0.106       |
| M20   | 7.823  | 1525.541 | 95.763  | 0.383    | 0.101 | 0.939       | 0.968       | 0.976       | 0.157 | 41149697       | 0.100       |
| M21   | **6.403** | **925.714** | **83.090** | **0.326** | **0.083** | **0.946** | **0.976** | **0.984** | **0.111** | 41215553       | 0.160       |

Table 2.2: Relative superiority of the M21 and the M18 over the M0 and the M20 for different metrics in percent. The two last columns are the ratio of the networks’ sizes and the ratio of the networks’ speeds. The minus sign means retrogression.

|                  | MAPE   | MSPE   | RMSE   | RMSELog | Log10 | \(\delta_1\) | \(\delta_1\) | \(\delta_3\) | SILog | Trainable params | Runtime |
|------------------|--------|--------|--------|----------|-------|-------------|-------------|-------------|-------|----------------|---------|
| m21 relative to m0 | 25.9   | 42.4   | 18.6   | 19.9     | 27.5  | 2.4         | 1.5         | 1.1         | 38.8  | 34.4           | -32.9   |
| m21 relative to m20 | 18.1   | 39.3   | 13.2   | 14.7     | 17.9  | 0.7         | 0.8         | 0.8         | 29.8  | 0.2            | -59.0   |
| m18 relative to m0 | 23.92  | 38.32  | 14.66  | 16.14    | 24.21 | 2.91        | 1.45        | 0.95        | 32.81 | 34.41          | -27.99  |
| m18 relative to m20 | 15.98  | 35.01  | 8.98   | 10.73    | 14.21 | 1.27        | 0.77        | 0.57        | 22.93 | 0.14           | -53.12  |
than the performances of the M1, the M7, the M18, and the M21. The first conclusion from this observation is that it is harder to improve models’ performance by injecting raw information (semantic segments) in higher layers of an SSIDE network (as the M4 and the M5). The second conclusion is that creating different branches for every semantic class at the final layers does not necessarily help to improve the performance of an SSIDE network (as in the M2, the M3, the M4, and the M5). The final conclusion is that the M18 and the M21 (minimized counterparts of the M7 and the M1 respectively) have superior performances.

In Figure 2.10 the original SIDE network (M0), the superior network of the first group (M20), and two superiors of the second group (M18 and M21) are compared. The performance superiority of the M21 and the M18 is clearly seen in this graph.

For a quantitative comparison, averages of all used metrics from epoch 20 to epoch 50 were calculated for all the networks and reported in here. The lower bound of this span was selected because the smooth trend of graphs starts approximately from the 20'th epoch. The higher bound was also selected since some of the networks were only trained up to 50 epochs. These averages are collected and represented in Table 2.1. According to the table data, the M21 had the best performance in eight metrics among the nine metrics considered in this study.

Besides metric values, the runtime of a single prediction and the number of trainable parameters of each network were mentioned in the Table 2.1. The M21 has two million parameters less than the original network (M0); nevertheless, its runtime is 40 ms longer than the runtime of the original network (M0). The reason for this is that the M21 has 19 more layers in its input matrix. This fact significantly increases the calculation time of the first convolutional layer.

Table 2.2 contains the percentage of the relative outperforming of the M21 and the M18 over the M0 and the M20 in each metric. Outperforming vary from 0.7 to 42 percent. Apart from delta thresholds, other metrics improved at least by 8.9 percent. However, delta thresholds improved by less than two percent. These small improvements are the result of the discrete nature of the delta thresholds. A delta threshold only counts the desirable results that lie within a specific threshold and it
does not take into account the difference of the results to the threshold. This results in the fact that varying the threshold may increase or decrease the improvement percentage.

Delta thresholds are not detailed and they are neither usable in practical implementations. In practice, one may want to know the accuracy of the depth estimation in different distances separately. Furthermore, a few thresholds for measuring accuracy (common thresholds for delta threshold metric are $\delta_1$: 1.250m, $\delta_2$: 1.563m, $\delta_3$: 1.953m) are not enough to determine the distribution of error. A histogram for the error distribution may express more information about the accuracy of depth estimation. Combining these two ideas to a heat map diagram Figure 2.11 is the result of calculating error histogram for different ranges. This graph shows how the M21 outperformed the M0 in different ranges. For example, the error distribution in range $[0, 5]$ m for the M21 is more intensive near zero relative to that of the M0.

To study the visual performance comparison for M18 and M20 networks, one sample of depth estimation of M20 and M18 (in SYNTHIA-SF dataset) is depicted in Appendix C. Similar comparisons between M18 and M20 were performed in Virtual KITTI and NYU depth datasets. The M18 outperformed the M20 by 52 percent of MAPE metric in the Virtual KITTI datasets and by 17 percent of MAPE metric in the NYU depth dataset.
Figure 2.11: Accuracy heat map of M0 and M21. [0, 20] and [30, 40]: error distribution for the M21 has a closer median to zero. Also, you can see a peak in the error distribution for M0 around 1.2m in [10, 15] m range and it’s not valid for M21. [20, 30]: the M0 has a better error distribution. [40, 50]: distribution of the error for the M0 seems like noise, however for the M21 this distribution is smooth and has a growing trend toward zero. Note that colors are mapped to percentages in a power norm.
Chapter 3

Synthetic Data instead of Non-Synthetic Data

To ensure that synthetic datasets are capable of training a segmentation model to generate accurate semantic segments for non-synthetic images, an experimental study was carried out. This chapter describes this experimental study.

3.1 Methods

In this experimental study, first, multiple segmentation models with the same architecture are trained on different synthetic and non-synthetic datasets. Second, each model is tested on each dataset to determine how general is the training dataset of the model; if the performance of the model is uniform over all datasets, the training dataset of the model is comprehensive relative to other datasets. Third, performances of all models are compared on each non-synthetic datasets to determine how models that were trained on synthetic datasets perform relative to models that were trained on non-synthetic datasets. Note that this comparison excluded the performance of the model that is not affected by domain shift, but this performance was mentioned as a reference in the comparison. If a model that was trained on a synthetic dataset performs better than a model that was trained on a non-synthetic dataset, the training dataset of the earlier model is assigned as appropriate for the
task mentioned in the goal of this experimental study, i.e., to train a segmentation model to generate semantic segments for non-synthetic images.

### 3.1.1 Semantic Segmentation Model

Lots of semantic segmentation models are available, but for simplicity and ease of implementation, a basic model was used. This model is an improved variant [61] of the U-Net [62] network.

The architecture of the original U-Net network is depicted in Figure 3.1. It has an encoder-decoder architecture. Its encoder consists of repeated blocks of convolutional layers with rectified linear unit (ReLU) activation function, followed by a max-pooling operation. The feature map generated by each block of the encoder is fed to the next block of the encoder as well as the corresponding block in the decoder[62].

---

**Figure 3.1:** “U-Net architecture (example for 32x32 pixels in the lowest resolution). Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations”[62].
The improved variant of the U-Net network has supplementary layers of batch-normalization after each convolution (before activation functions) and each concatenation layer (in decoder blocks). Furthermore, it has zero-padded convolution that makes the network output to have the same resolution as the input of the network. Another difference is the presence of an additional step in the encoder and the decoder.

### 3.1.2 Metric and Loss Function

Intersection over union (IoU) is used as the metric for the evaluation of semantic segmentation CNNs. The IoU is the area of intersection between the predicted segmentation and its ground truth divided by the area of their union (Figure 3.2). Also, IoU is used in the loss function for training semantic segmentation models. The output of a semantic segmentation model consists of a matrix with depth $d$. The $d$ is the number of all classes available in the used dataset. The IoU is calculated for every layer of the output matrix, then an average of all IoU values is calculated that is between zero and one because the IoU metric is between zero and one. The IoU reaches one as predictions improve. Therefore, $1 - IoU_{\text{average}}$ is used as the loss function to have a decreasing loss function as predictions improve.

![Figure 3.2: Intersection over Union](image-url)
3.1.3 Datasets

Two non-synthetic datasets (Cityscapes [63] and Mapillary Vistas [57]) and two synthetic datasets (VIPER [64] and SYNTHIA-SF [58]) are used in this experimental study. The following data preparations were performed:

- Images values were normalized to [0, 1] range.
- Semantic segment labels were converted from RGB mappings to one-hot labels (see Appendix A for a definition of one-hot encoding).
- Images and semantic segment labels were resized to 640 × 480.
- To make datasets compatible in terms of semantic segment classes, only common classes were considered in the experimental study.

3.2 Results

In each of the following four subsections, the performance of a U-Net network on different datasets is examined. If the performance of the network is uniform over all datasets, the training dataset of the network is comprehensive relative to other datasets. In each subsection, the training dataset is fixed. In the last subsection (fifth), performances of differently trained U-Net networks (i.e., they were trained on different datasets) are compared on the non-synthesized datasets to determine to what extent and conditions synthesized datasets may be replaced non-synthesized ones.

3.2.1 Cityscapes as the Training Dataset

First, a U-Net CNN is trained on the Cityscapes dataset information, and then it is tested on all other datasets considered in this study. Results showed that the trained network has some ability to discover semantic segments in the test subset of the Cityscapes dataset. However, the network performance is not satisfying when tested on other datasets. The visual performance results of the U-Net for the test
sets of Cityscapes, Mapillary, VIPER, and SYNTHIA-SF datasets are depicted in Figure C.2 to Figure C.5 respectively (in Appendix C).

For quantitative scrutiny, a comparison of IoU values for each semantic class is performed. The IoU values for different datasets are depicted in Figure 3.3. Each bar represents the accuracy of the network in segmenting a specific category. For example, when the network is tested on Cityscapes dataset, the accuracy is more than 75% for two classes of Road and Sky, and between 75% and 50% for Sidewalk, Building, Vegetation, Car, and Unlabeled categories. The accuracy for other categories is lower than 50% because of their complexity for segmentation or because they appeared less often in training set relative to other categories. However, the important point is that the accuracy of the network in the first database, whatever it is, is half or less in other datasets. Even this observation is true for two classes of Road and Sky, which have the highest accuracy in the Cityscapes dataset. We expected that the network to perform well on other non-synthetic datasets due to their common origin. However, the results showed that the Cityscapes dataset is not suitable enough to prepare a semantic segmentation CNN to operate in different environments.
3.2.2 MAPILLARY as the Training Dataset

Now a U-Net CNN is trained on the MAPILLARY dataset information, and then it is tested on all other datasets in hand. Unlike the previous network, this network has a more uniform performance on different datasets as seen in Figure 3.4. For example, the accuracy of segmentation in *Road*, *Building*, *Sky* categories is uniform over all datasets and is partially uniform over some datasets in *Sidewalk*, *Pole*, *Car*, *Person* categories. Even in some categories, the accuracy of segmentation is higher on datasets other than MAPILLARY, that show how comprehensive the MAPILLARY dataset is relative to other (*Road*, *Sidewalk*, and *Building*). The results also show that this network has some difficulties when predicting on synthesized VIPER datasets. Overall, these results showed that the Mapillary dataset permits the U-Net CNN to generalize its estimation in different environments in a more uniform result. The visual performance results of the U-Net for the test sets of Cityscapes, Mapillary, VIPER, and SYNTHIA-SF datasets are depicted in Appendix C and in Figure C.6 to Figure C.9, respectively.
Figure 3.5: Performance of U-Net trained on SYNTHIA-SF. The network failed in generalizing learned concepts on other datasets.

3.2.3 SYNTHIA-SF as the Training Dataset

Next a U-Net CNN is trained on the SYNTHIA-SF dataset, and then it is tested on all other datasets. The accuracy of the network in several categories is zero or near zero which is a sign of absence of these categories in the training set of SYNTHIA-SF dataset. Like the previous result, when the training set was from Cityscapes dataset, we observed that this network is poor to extend its prediction over different other datasets and only performed well on segmentation of sky, as it can be seen in Figure 3.5. In summary, these results showed the SYNTHIA-SF dataset is not appropriate for training a U-Net CNN that is supposed to operate in different environments. The visual performance results of the U-Net for the test sets of Cityscapes, Mapillary, VIPER, and SYNTHIA-SF datasets are depicted in Appendix C and in Figure C.10 to Figure C.13, respectively.

3.2.4 VIPER as the Training Dataset

After a U-Net CNN was trained on the VIPER dataset, it was tested on all datasets. Like the results from the network that was trained on MAPILARY dataset, the ob-
Figure 3.6: Performance of U-Net trained on VIPER. The network succeed in generalizing of some classes (building, sky, vegetation classes) and failed in others.

Tained results for VIPER dataset showed that the network can generalize some of its learned concepts from the training sets to other test sets (Figure 3.5). The network extends well its prediction of building, sky, and vegetation classes according to this graph. Furthermore, the accuracy of segmentation is more uniform in other categories relative to the accuracy of the networks that were trained on the Cityscapes or SYNTHIA-SF dataset. A more detailed comparison of the performance of different networks is elaborated in the next section. The visual performance results of the U-Net for the test sets of Cityscapes, Mapillary, VIPER, and SYNTHIA-SF datasets are depicted in Appendix C and in Figure C.14 to Figure C.17, respectively.

3.2.5 Comparison on Non-Synthesized Datasets

Two different graphs with duplicate data from previous graphs are considered for this comparison. In this comparison, results are related to a common test dataset instead of a common training dataset. This comparison contains IoU scores of all four trained CNNs, while evaluated only on two non-synthesized datasets. Figure 3.7 shows the performance of differently trained networks when tested on Cityscapes.
dataset. The best performance as expected belongs to the CNN that is trained on
the Cityscapes, since by this means there is no domain shift. The surprising fact
is that second-best CNN which is superior in 8 classes is trained on MAPILLARY
dataset. To further explore the characteristics of these datasets, Figure 3.8 shows the
performance of differently trained CNNs when tested on the MAPILLARY dataset.
In this graph, all superior results belong to the CNN that is trained on the MAPIL-
LARY itself as expected, while the second-best performance is on the CNN which is
trained on a synthesized dataset, on VIPER! The worst result belongs to the CNN
which is trained on a non-synthesized dataset, i.e. the Cityscapes.

![IoU - Testing Dataset: CITYSCAPES](image)

**Figure 3.7:** Performance of different U-Net networks on CITYSCAPES

### 3.2.6 Discussion on the Observed Results

In the first and third experiments, we observed that networks are poor in extending
their predictions over different datasets. The network in the first experiment was
trained on Cityscapes dataset. This dataset was captured in different cities of Ger-
many and contains 2975 samples. Therefore, this dataset contains no information
about urban areas of other countries, different types of vegetation, different types of
sun lighting in different latitudes, and any other objects that are subject to change
Figure 3.8: Performance of different U-Net networks on MAPILLARY

due to a change in location. Similar conditions apply to the SYNTHIA-SF dataset, which led to the results of the third experiment. This dataset was captured in a small uniform virtual environment and contains 2224 samples.

On the other hand, in the second experiment, we observed that the network performs well in extending its predictions over different datasets, and somewhat well in the fourth experiment. The MAPILARY datasets, that was used as the reference dataset in the second experiment, was captured by different imaging devices like mobile phones, tablets, action cameras, and professional capturing rigs and in cities of different countries and continents, and contains 25000 samples. The VIPER dataset, that was used as the reference dataset in the fourth experiment, sits between uniform datasets and non-uniform datasets. It was captured in a large virtual environment (about 8.12 square kilometers) that has different cities, jungles, seas, deserts, and mountains, and contains 250000 samples.

Finally, we concluded that the diversity in the content of datasets’ samples is very influential in the performance of segmentation networks and the observed performance differences mainly stem from differences in the diversity of contents in used datasets. Although other factors like the realness of synthesized images or quality
of images in terms of the resolution also affect the performance of the networks, the obtained results showed that these factors are minor here. Therefore, in cases where appropriate non-synthetic datasets (like Mapillary) is not available, one may use a synthetic dataset with diverse contests and large samples (like VIPER).
Chapter 4

Robustness of SSIDE against Domain Shift

This chapter describes the final experimental study which is based on the results of the two previous experimental studies: first, it is assumed that an SSIDE CNN exists that outperforms its SIDE counterpart. Second, it is assumed that semantic segments generated for non-synthetic images by a CNN trained on synthetic datasets are accurate enough. The goal of this experimental study is to investigate the effect of the use of semantic segments on the robustness of a SIDE model against domain shift. In other words, the goal is to determine how much an SSIDE CNN outperforms its SIDE counterpart while both generalize concepts from synthetic data to non-synthetic data.

4.1 Methods

For the goal of this study, multiple models of the M20 architecture are trained on two non-synthetic datasets (KITTI [65] and Lyft Level 5 [66]) and three synthetic datasets (Virtual KITTI 2 [60], PreSIL [67], and SYNTHIA-SF [58]). Several other M20 models are also trained on different combinations of synthetic datasets. Furthermore, multiple models of the M18 and the M21 architecture are trained on three synthetic datasets (Virtual KITTI 2, PreSIL, and SYNTHIA-SF) and on different combinations of synthetic datasets.
All networks are tested on non-synthetic datasets (KITTI and Lyft Level 5) after every 5000 batches during training for 30 times. Then networks’ performances are compared separately on each test dataset (KITTI and Lyft Level 5). Each comparison excluded the performance of the network that the training set of that network and test set, which comparison is based on it, belonged to one dataset.

Since the used non-synthetic datasets have no semantic segments along with the depth data, a U-Net CNN is used to generate semantic segments for each sample. Various combinations of synthetic datasets are tested to train multiple U-Net CNNs. Among them, a combination is selected that outperformed others on the KITTI semantic segmentation dataset (training set of the dataset).

Finally, a statistical analysis is performed on depth data of all used datasets. This supplementary analysis is carried out to determine the differences and similarities of the depth data distribution between all used datasets. In this analysis, the histograms of the depth distribution for each row of depth maps are calculated (on all samples of each dataset). The range size and the bin size of histograms are 100 m and 0.2 m, respectively. The size of depth maps is 1216 × 352, and therefore, the number of rows is 352. For each dataset, instead of drawing 352 histograms, a heat map is generated and reported. The histograms constituted the rows of the heat maps, such that the heat maps have 352 rows and 500 columns (500 = size\_range \div size\_bin = 100/0.2).

**Trainings Details**

The loss function, batch size, optimizer, and learning rate for the training of all depth estimation models were MAPE, two, Adam optimizer [68], and 0.001, respectively.

### 4.1.1 Datasets

To make datasets consistent in terms of depth data, the AFOVs of all samples are reduced to a specific AFOV. It is carried out by cropping margins of samples. After the cropping, all samples are resized to 1216 × 352. Initial and final AFOV of each datasets is mentioned in Table 4.1.
Table 4.1: Unification of AFOV between datasets. *Focal Length* column shows the focal length of the camera used in each dataset. *Original size* column shows the original size of samples in each dataset (images, semantic segments, and depth maps). The AFOVs of all samples were reduced to the AFOV of samples of Lyft level 5 dataset.

| Dataset Name   | Focal Length | Original size | Cropped size | Final AFOV |
|----------------|--------------|----------------|--------------|------------|
|                | width | height | width | height | horizontal | vertical |
| Lyft Level 5   | 880.0 | 1224 | 1024 | 1216 | 352 | 69.28 | 22.62 |
| PreSIL         | 960.0 | 1920 | 1080 | 1328 | 384 | 69.34 | 22.62 |
| SYNSCAPES      | 1590.0 | 1440 | 416  | 1440 | 352 | 48.72 | 14.91 |
| SYNTHIA-SF     | 847.6 | 1920 | 1080 | 1172 | 340 | 69.32 | 22.68 |
| Virtual KITTI  | 725.0 | 1242 | 375  | 1002 | 290 | 69.29 | 22.62 |
| VVIPER         | 1158.0 | 1920 | 1080 | 1602 | 464 | 69.34 | 22.66 |
| KITTI          | 721.5 | 1242 | 375  | 998  | 289 | 69.3 | 22.6 |

Five datasets are used for the training of the U-Net. These five datasets are KITTI (for the test), Virtual KITTI 2, VIPER, Synscapes [53], and SYNTHIA-SF. To have unified semantic categories between these datasets, those subcategories which are not available in all datasets are merged to their higher categories. Among these subcategories, some did not have a higher category, and therefore, they are removed (see Appendix B for details).

### 4.2 Results

#### 4.2.1 Statistical Analysis of Depth Data

Among the models that were trained on synthetic datasets, the best performances belonged to the models that were trained on the Virtual KITTI dataset. The statistical analysis of the depth distribution of the datasets could be used to explain the underlying facts behind this observation. The depth heat maps shows that the distribution of depth in the synthetic Virtual KITTI dataset is more similar to the distribution of depth in the non-synthetic datasets relative to other synthetic datasets. These heat maps are depicted in Figure 4.1 to Figure 4.5. In the left heat map of each figure, the sum of percentages of all points is 100 (globally normalized), while in the right heat maps, the sum of percentages of all points in each row is 100 (normalized per each row).
Figure 4.1: Depth heat map of KITTI dataset.

Figure 4.2: Depth heat map of Lyft Level 5 dataset

Figure 4.3: Depth heat map of Virtual KITTI 2 dataset
These heat maps contain significant amount of information about depth distributions. As an example, consider the heat maps of the KITTI dataset in Figure 4.1, which shows that the used LIDAR had a range limit of 80 m, while this range in Lyft Level 5 dataset is more than 100 m, as it can be seen in Figure 4.2. For this reason, depth labels of used synthetic datasets were limited to 100 m. The existence of large distances in the upper part of the heat maps is a sign of wide streets and existence of large objects like tall buildings and trees in the sampling scene, like in Figure 4.3, Figure 4.4, and Figure 4.5. As a similarity to the non-synthetic datasets, the Virtual KITTI dataset has the lowest density of depth distribution in the upper right corner of its heat map among the heat maps of synthetic datasets (it is better visualized in the graphs in the right). The existence of large distances in the lower part of the heat maps shows that some samples of the datasets were captured on uphill streets or on roads along valleys, like in Figure 4.4, and Figure 4.5. Again, another similarity of Virtual KITTI dataset to non-synthetic datasets is seen here; the density of depth data in lower part of heat maps of Virtual KITTI dataset is concentrated in short distances.

In addition to visual similarity, calculation of the Euclidean distance of these heat maps shows that the distribution of depth in the Virtual KITTI dataset is very similar to the depth distribution in the non-synthetic version of this dataset. Considering these arguments and observations, in the following analysis, the re-
Figure 4.5: Depth heat map of SYNTHIA-SF dataset

results obtained from Virtual KITTI, KITTI, Lyft level 5, PreSIL, and SYNTHIA-SF datasets are reported.

4.2.2 Performance Analysis

For the performance comparison between SIDE architectures and SSIDE architectures, the M20 architecture was selected as the representative of SIDE networks and M18 and M21 architectures were selected as the representatives of SSIDE networks. We selected two architectures of M18 and M21 as the representatives of SSIDE networks because their performances were very close to each other when they were trained and tested on the same dataset. However, we considered the possibility that one architecture may perform better when a domain shift occurs. As mentioned before, the M21 architecture uses RGB images as its inputs and the M18 uses semantic edges as its inputs.

For evaluation of an M18 and an M21 network, which were trained on Virtual KITTI, semantic labels for the test sets of the KITTI and Lyft Level 5 datasets had to be prepared first. After training a U-Net network on different combinations of synthetic datasets, the best performance was achieved when the training set was composed of 10000 samples of VIPER dataset, 10000 samples of SYNSCAPES dataset, 10000 samples of Virtual KITTI dataset, and all samples of SYNTHIA-SF dataset (more than 4000 samples). By using this combination for the training, the U-
Net achieved a mean IoU of 73% on the training set of the KITTI dataset (KITTI segmentation dataset consists of 200 samples.). Some samples of the generated semantic labels are depicted in Figure 4.6.

In the first sample, minor errors are seen in Figure 4.6b. Among them, the notable error is the semantic label for the bicycle. In the second sample, some gross errors exist as seen in Figure 4.6d. For example, a large area of the street is labeled as *Car*, which its reason is probably the saturated colors in the image that the network never experienced before in training sets. In the middle of the generated segmentation map, another large area of the street is labeled as *Pole*.

These observation lead us to two important conclusions. First, a comprehensive synthesized dataset along with all kinds of diversities in its samples should contain samples with synthesized artifacts like image distortion, color saturation, and motion blur. This will enable the network to perform better when it faces with artifacts in non-synthetic data. Second, the accuracy of the generated semantic labels may lead the SSIDE networks to perform worse than what is expected. Therefore, a minimum accuracy requirement of semantic labels must always be met to have a reliable output from SSIDE networks.

To summarize the comparison study of SSIDE models performances, we find that unlike the results obtained in the first experimental study (section 2.2), the performance of the M18 model is better than the performance of the M21 model by 13 percent with respect to MAPE. It indicates that the architecture of M18 is more
Figure 4.7: Performance Comparison on the Lyft Level 5 dataset

robust against domain shift relative to the architecture of the M21. In other words, the use of semantic edges\(^1\) is more effective in the robustness of an SSIDE model compared to that on the cases when simply concatenation of semantic labels and raw image is used. Therefore, in the upcoming observation reported in this section, we only consider the performance of M18.

In the performance comparison of SIDE models that were trained on KITTI, Lyft Level 5, and Virtual KITTI, the M20 network trained on Virtual KITTI has the worst performance (red graphs in Figure 4.7 and Figure 4.8). On the other hand, the best performance is obtained when the training and test dataset were the same, as expected since there is no domain shift in these cases (green graph in Figure 4.7 and purple graph in Figure 4.8).

The most important observation in comparison of SIDE models is that the domain shift cause a significant reduction in accuracy when the model is trained on a synthetic dataset compared to when it is trained on a non-synthetic dataset (the red graph compared to purple graph in Figure 4.7 and the red graph compared to green graph in Figure 4.8).

\(^1\)See the definition of semantic edge in subsection 2.1.1
However, the M18 model, as the representative of SSIDE networks, performed well compared to the M20 model that was trained on the synthetic Virtual KITTI dataset (blue graph compared to red graph in Figure 4.7 and Figure 4.8). The M18 model outperforms M20 model by 15% of MAPE metric in the KITTI dataset and by 30% of MAPE metric in the Lyft Level 5 dataset. Furthermore, the performance of the M18 network is very close to the performance of the M20 model that is trained on KITTI when the test dataset is from Lyft Level 5 and close to the performance of the M20 model that is trained on the Lyft Level 5 when test dataset is from KITTI. In summary, we conclude that the use of semantic labels in single image depth estimation improves the depth estimation and make the model robust against domain shift while SIDE models are more sensitive to domain shift.
Chapter 5

Conclusions and Future Work

5.1 Conclusions

In this thesis, we investigated how the use of semantic labels as raw data, like an input image, can potentially improve the performance of a SIDE network. We also investigated the effect of the use of semantic labels on the robustness of a SIDE model against domain shift.

In the first experimental study (section 2.2), we experimented and evaluated various methods to reinforce a pre-designed SIDE network with the knowledge of the input image’s semantic label. The results showed that by concatenation of the raw image and its semantic label and feeding them both to a well structured network can significantly improve the performance of the depth estimation. From this observations, it is concluded that the use of semantic labels can improve the performance of a SIDE network. This result and conclusion are aligned with the assumptions that semantic segments are a prerequisite for the perception of pictorial cues, and furthermore, noting that pictorial cues could be effective in the improvement of a SIDE network performance. This conclusion also agrees with the various works that reported on the improvement in the depth estimation by different use of semantic segmentation or semantic segments [28]–[40].

It is further concluded that the use of a semantic label is most effective when it is fed to a SIDE network through the input layer of the network and not through
its higher levels. This conclusion theoretically implies that multi-task models, that try to simultaneously estimate the depth and segment the image, are not efficient. This is mainly because this kind of model generates the semantic segments in the last layers.

In the second experimental study, we examined different synthetic datasets and compared them to some non-synthetic datasets in terms of training a segmentation model. The obtained results showed that if a synthetic dataset is more diverse in its content relative to a non-synthetic dataset, then it is superior in training a segmentation network. Therefore, we concluded that regardless of the origin of the dataset (whether it is synthetic or non-synthetic), the more diverse a dataset is in its content, the better performance for the network is obtained. This conclusion lead us to generate semantic labels for non-synthetic images by means of synthetically trained models in the last experimental study.

Finally, in the last experimental study, an SSIDE model outperformed its SIDE counterpart, while both suffered from domain shift from synthetic data to non-synthetic data. The SSIDE model performed like other SIDE models that were trained on non-synthetic datasets. From these results, we concluded that the use of semantic segments increases the robustness of the model against domain shifts. Therefore, for the task of depth estimation, if an appropriate synthetic dataset related to the environment of the task would be available, without loosing too much accuracy one can use an SSIDE model along with the synthetic datasets to achieve suitable performance.

Unlike other works, in this thesis by isolating the experiments against altering parameters the following conclusions are obtained.

• Use of semantic segments may significantly improve the accuracy of the depth estimation through well structured deep neural networks.

• The performance of a depth estimation network improves only when semantic segments are fed to it from the input layer.
• The use of semantic segments improves the robustness of a depth estimation model against domain shifts.

• The robustness of the model is maximized when it uses semantic edges.

These conclusions are derived by the use of ground truths instead of inaccurate predictions for semantic segments. Furthermore the synthetic datasets are analyzed to distinguish the accurate and more reliable synthetic dataset from inaccurate and noisy non-synthetic ones.

Considering all the conclusions reported in this thesis, leads to a practical application; a method that could possibly improve the performance of a SIDE model. In this method a SIDE network could be converted to an SSIDE network only by feeding the semantic edges of the input image.

5.2 Future Work

In this thesis, we used a simple SIDE as the base network to develop our models. However, this base network is not very powerful and the developed SSIDE networks do not have competitive performance compared to recent state-of-the-art methods. Therefore, a demanded future work could be the examination of the proposed method in this thesis to convert recently proposed SIDE networks to SSIDE networks.

In previous section we mentioned that multi-task methods that both estimate a depth map and a semantic label are not efficient. Another interesting future research could be to develop a multi-task network based on the M0 architecture, and to compare it with other developed SSIDE networks.

In the second experimental study, we observed that a synthetic dataset led to better performance compared to a non-synthetic dataset that had lower diversity in its content. In this experimental study, we only used synthetic datasets that seemed visually with high quality or more realistic, compared to other available synthetic datasets. This leads to the following questions: What would happen if we used low-quality synthetic datasets? Or, if two synthetic and non-synthetic datasets with
the same diversity exist, what qualities should the synthetic dataset have to lead to a better performance of a network? These questions could be answered in future work.

In the last experimental study we used generated semantic labels for non-synthetic datasets to evaluate an SSIDE network. Definitely, the quality of generated semantic labels affected the performance of the network. However, its extent is unknown to us. A probable future work related to this issue could be to investigate the relation of the accuracy of semantic labels to the performance of SSIDE networks.
Appendix A

Semantic Segmentation

Semantic segmentation of an image is to assign a semantic class to every pixel of the image. Usually every semantic class is encoded with an RGB color, so that a semantic segmented image can be visualized in a color image and stored in an image-type format (Figure A.1).

![RGB image](image1)
![RGB mapping of semantic segments](image2)
![one-hot encoding of road class](image3)
![one-hot encoding of person class](image4)

Figure A.1: Different encodings for semantic segments. An example of an RGB image with its corresponding semantic labels from [53]

But this type of encoding is not useful for training a CNN, because semantic segmentation is a classification task which needs an array of binary numbers as the ground truth for each pixel; 1 shows the existence of a class and 0 shows the absence of a class. Therefore
an RGB mapping of semantic segments can be converted to a one-hot labels, i.e., a group of binary images (Figure A.1).
### Appendix B

## Common Semantic Labels Set

Table B.1: Common semantic labels set. Each column shows the available semantic classes in a dataset by mentioning its RGB code. The last column shows a set of unified semantic classes. In the last column an arrow shows that the class of the row is merged to the class that arrow points to.

| Class name | VIPER       | SYNTHIA-SE | Synscapes | Virtual KITTI | KITTI | Common set |
|------------|-------------|------------|-----------|---------------|-------|-------------|
| 1 Fence    | [190, 153, 153] | [190, 153, 153] | [190, 153, 153] | [190, 153, 153] | -     | → Unlabeled |
| 2 Guard Rail | -          | -          | [180, 153, 153] | [180, 153, 153] | -     | → building  |
| 3 Wall     | -          | [102, 102, 156] | [102, 102, 156] | [102, 102, 156] | -     | → Road      |
| 4 Parking  | -          | -          | [250, 170, 160] | [250, 170, 160] | -     | → Road      |
| 5 Rail Track | [230, 150, 140] | -          | [230, 150, 140] | [230, 150, 140] | -     | → Road      |
| 6 Road     | [128, 64, 128] | [128, 64, 128] | [128, 64, 128] | [128, 64, 128] | [128, 64, 128] | -     |
| 7 Sidewalk | [244, 35, 232] | [244, 35, 232] | [244, 35, 232] | [244, 35, 232] | -     | → Road      |
| 8 Bridge   | -          | -          | [150, 100, 100] | [150, 100, 100] | -     | → building  |
| 9 Building | [70, 70, 70] | [70, 70, 70] | [70, 70, 70] | [70, 70, 70] | [70, 70, 70] | -     |
| 10 Tunnel  | -          | -          | [150, 120, 90] | -              | [150, 120, 90] | → building  |
| 11 Person  | [220, 20, 60] | [220, 20, 60] | [220, 20, 60] | [220, 20, 60] | [220, 20, 60] | -     |
| 12 Bicyclist | [255, 0, 0] | [255, 0, 0] | [255, 0, 0] | [255, 0, 0] | [255, 0, 0] | → Person |
| 13 Lane Marking - General | -          | -          | -              | [157, 234, 50] | -     | → Road      |
| 14 Sky     | [70, 130, 180] | [70, 130, 180] | [70, 130, 180] | [70, 130, 180] | [70, 130, 180] | -     |
| 15 Terrain | [152, 251, 152] | [152, 251, 152] | [152, 251, 152] | [152, 251, 152] | [152, 251, 152] | -     |
| 16 Vegetation | [35, 142, 35] | [107, 142, 35] | [107, 142, 35] | [90, 240, 0] | [107, 142, 35] | -     |
| 17 Pole    | [153, 153, 153] | [153, 153, 153] | [153, 153, 153] | [153, 153, 153] | [153, 153, 153] | -     |
| 18 Traffic Light | [250, 170, 30] | [250, 170, 30] | [250, 170, 30] | [250, 200, 0] | [250, 170, 30] | → Pole |
| 19 Traffic Sign (Front) | [220, 220, 0] | [220, 220, 0] | [220, 220, 0] | [255, 255, 0] | [220, 220, 0] | → Pole |
| 20 Trash Can | [81, 0, 81] | -          | -              | -              | -     | → Unlabeled |
| 21 Bicycle | [119, 11, 32] | [119, 11, 32] | [119, 11, 32] | -              | [119, 11, 32] | -     |
| 22 Boat    | [50, 0, 90] | -          | -              | -              | -     | → Unlabeled |
| 23 Bus     | [60, 60, 100] | [60, 60, 100] | [60, 60, 100] | [60, 60, 100] | -     | → Car       |
| 24 Car     | [0, 0, 142] | [0, 0, 142] | [0, 0, 142] | [255, 127, 80] | [0, 0, 142] | -     |
| 25 Caravan | [0, 0, 90] | -          | [0, 0, 90] | -              | [0, 0, 90] | → Car       |
| 26 Motorcycle | [0, 0, 230] | [0, 0, 230] | [0, 0, 230] | -              | [0, 0, 230] | → Bicycle   |
| 27 On Rails | [80, 80, 100] | [80, 80, 100] | [80, 80, 100] | [80, 80, 100] | -     | → Car       |
| 28 Trailer | -          | -          | [0, 110] | -              | [0, 110] | → Car       |
| 29 Truck   | [0, 0, 70] | [0, 0, 70] | [0, 0, 70] | [160, 60, 60] | [0, 0, 70] | → Car       |
| 30 Unlabeled | [0, 0, 0] | [0, 0, 0] | [0, 0, 0] | [0, 0, 0] | [0, 0, 0] | -     |
| 31 ground  | -          | [81, 0, 81] | [81, 0, 81] | -              | [81, 0, 81] | → Unlabeled |
| 32 dynamic | [111, 74, 0] | -          | [111, 74, 0] | -              | [111, 74, 0] | → Unlabeled |
| 33 plane   | [0, 100, 100] | -          | -              | -              | -     | → Unlabeled |
| 34 trash   | [81, 0, 21] | -          | -              | -              | -     | → Unlabeled |
| 35 chair   | [103, 153, 153] | -          | -              | -              | -     | → Unlabeled |
| 36 firehydrant | [173, 153, 153] | -          | -              | -              | -     | → Unlabeled |
| 37 mobilebarrier | [189, 180, 100] | -          | -              | -              | -     | → Unlabeled |
| 38 billboard | [150, 20, 20] | -          | -              | -              | -     | → Unlabeled |
| 39 tree    | [87, 182, 35] | -          | -              | [0, 199, 0] | -     | → Unlabeled |
| 40 Misc    | -          | -          | [80, 80, 80] | -              | -     | → Unlabeled |
Appendix C

Experiments Results

C.1 Supplementary Data for First Experiment

Figure C.1: Depth estimation samples of M18 and M20

C.2 Supplementary Data for Second Experiment
Figure C.2: Perception sample of Cityscapes U-Net on Cityscapes. The top right image is the input RGB image, the other two top images are ground truth and prediction in a single semantic class and two bottom images are overall ground truth and prediction.

Figure C.3: Perception sample of Cityscapes U-Net on Mapillary. The top right image is the input RGB image, the other two top images are ground truth and prediction in a single semantic class and two bottom images are overall ground truth and prediction.
Figure C.4: Perception sample of Cityscapes U-Net on VIPER. The top right image is the input RGB image, the other two top images are ground truth and prediction in a single semantic class and two bottom images are overall ground truth and prediction.

Figure C.5: Perception sample of Cityscapes U-Net on SYNTHIA-SF. The top right image is the input RGB image, the other two top images are ground truth and prediction in a single semantic class and two bottom images are overall ground truth and prediction.
Figure C.6: Perception sample of Mapillary U-Net on *Cityscapes*. The top right image is the input RGB image, the other two top images are ground truth and prediction in a single semantic class and two bottom images are overall ground truth and prediction.

Figure C.7: Perception sample of Mapillary U-Net on *Mapillary*. The top right image is the input RGB image, the other two top images are ground truth and prediction in a single semantic class and two bottom images are overall ground truth and prediction.
Figure C.8: Perception sample of Mapillary U-Net on VIPER. The top right image is the input RGB image, the other two top images are ground truth and prediction in a single semantic class and two bottom images are overall ground truth and prediction.

Figure C.9: Perception sample of Mapillary U-Net on SYNTHIA-SF. The top right image is the input RGB image, the other two top images are ground truth and prediction in a single semantic class and two bottom images are overall ground truth and prediction.
Figure C.10: Perception sample of SYNTHIA-SF U-Net on Cityscapes. The top right image is the input RGB image, the other two top images are ground truth and prediction in a single semantic class and two bottom images are overall ground truth and prediction.

Figure C.11: Perception sample of SYNTHIA-SF U-Net on Mapillary. The top right image is the input RGB image, the other two top images are ground truth and prediction in a single semantic class and two bottom images are overall ground truth and prediction.
Figure C.12: Perception sample of SYNTHIA-SF U-Net on VIPER. The top right image is the input RGB image, the other two top images are ground truth and prediction in a single semantic class and two bottom images are overall ground truth and prediction.

Figure C.13: Perception sample of SYNTHIA-SF U-Net on SYNTHIA-SF. The top right image is the input RGB image, the other two top images are ground truth and prediction in a single semantic class and two bottom images are overall ground truth and prediction.
Figure C.14: Perception sample of VIPER U-Net on Cityscapes. The top right image is the input RGB image, the other two top images are ground truth and prediction in a single semantic class and two bottom images are overall ground truth and prediction.

Figure C.15: Perception sample of VIPER U-Net on Mapillary. The top right image is the input RGB image, the other two top images are ground truth and prediction in a single semantic class and two bottom images are overall ground truth and prediction.
Figure C.16: Perception sample of VIPER U-Net on VIPER. The top right image is the input RGB image, the other two top images are ground truth and prediction in a single semantic class and two bottom images are overall ground truth and prediction.

Figure C.17: Perception sample of VIPER U-Net on SYNTHIA-SF. The top right image is the input RGB image, the other two top images are ground truth and prediction in a single semantic class and two bottom images are overall ground truth and prediction.
Bibliography

[1] E. Brenner and J. B. J. Smeets, “Depth perception,” in Stevens’ Handbook of Experimental Psychology and Cognitive Neuroscience, American Cancer Society, 2018, pp. 1–30, ISBN: 978-1-119-17017-4. DOI: 10.1002/9781119170174.epcn209. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119170174.epcn209 (visited on 12/09/2019) (cit. on p. 2).

[2] E. B. Goldstein and J. Brockmole, Sensation and Perception. Cengage Learning, Jan. 19, 2016, 480 pp., Google-Books-ID: x5d4CgAAQBAJ, ISBN: 978-1-305-88832-6 (cit. on p. 2).

[3] S. T. Barnard and M. A. Fischler, “Computational stereo,” ACM Comput. Surv., vol. 14, no. 4, pp. 553–572, Dec. 1, 1982, ISSN: 0360-0300. DOI: 10.1145/356893.356896. [Online]. Available: https://doi.org/10.1145/356893.356896 (visited on 06/28/2020) (cit. on p. 2).

[4] U. Dhond and J. Aggarwal, “Structure from stereo-a review,” IEEE Transactions on Systems, Man, and Cybernetics, vol. 19, no. 6, pp. 1489–1510, Nov. 1989, Conference Name: IEEE Transactions on Systems, Man, and Cybernetics, ISSN: 2168-2909. DOI: 10.1109/21.44067 (cit. on p. 2).

[5] D. Scharstein and R. Szeliski, “A taxonomy and evaluation of dense two-frame stereo correspondence algorithms,” International Journal of Computer Vision, vol. 47, no. 1, pp. 7–42, Apr. 1, 2002, ISSN: 1573-1405. DOI: 10.1023/A:1014573219977. [Online]. Available: https://doi.org/10.1023/A:1014573219977 (visited on 06/28/2020) (cit. on p. 2).
[6] K. Zhou, X. Meng, and B. Cheng. (Mar. 23, 2020). “Review of stereo matching algorithms based on deep learning,” Computational Intelligence and Neuroscience. ISSN: 1687-5265 Library Catalog: www.hindawi.com Pages: e8562323 Publisher: Hindawi Volume: 2020, [Online]. Available: https://www.hindawi.com/journals/cin/2020/8562323/ (visited on 06/28/2020) (cit. on p. 2).

[7] H. Kim, C. Richardt, and C. Theobalt, “Video depth-from-defocus,” in 2016 Fourth International Conference on 3D Vision (3DV), Oct. 2016, pp. 370–379. DOI: 10.1109/3DV.2016.46 (cit. on p. 2).

[8] C. Hazirbas, S. G. Soyer, M. C. Staab, L. Leal-Taixé, and D. Cremers, “Deep depth from focus,” arXiv:1704.01085 [cs], Oct. 28, 2018. arXiv: 1704.01085. [Online]. Available: http://arxiv.org/abs/1704.01085 (visited on 06/28/2020) (cit. on p. 2).

[9] M. Maximov, K. Galim, and L. Leal-Taixé, “Focus on defocus: Bridging the synthetic to real domain gap for depth estimation,” arXiv:2005.09623 [cs], May 19, 2020. arXiv: 2005.09623. [Online]. Available: http://arxiv.org/abs/2005.09623 (visited on 06/28/2020) (cit. on p. 2).

[10] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, “Bundle adjustment — a modern synthesis,” in Vision Algorithms: Theory and Practice, B. Triggs, A. Zisserman, and R. Szeliski, Eds., ser. Lecture Notes in Computer Science, Berlin, Heidelberg: Springer, 2000, pp. 298–372, isbn: 978-3-540-44480-0. doi: 10.1007/3-540-44480-7_21 (cit. on p. 2).

[11] J. Oliensis, “A critique of structure-from-motion algorithms,” Computer Vision and Image Understanding, vol. 80, no. 2, pp. 172–214, Nov. 1, 2000, ISSN: 1077-3142. DOI: 10.1006/cviu.2000.0869. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1077314200908695 (visited on 06/29/2020) (cit. on p. 2).

[12] R. Hartley and A. Zisserman, Multiple View Geometry in Computer Vision. Cambridge University Press, 2003, 676 pp., Google-Books-ID: si3R3Pfa98QC, isbn: 978-0-521-54051-3 (cit. on p. 2).
[13] O. Özyesil, V. Voroninski, R. Basri, and A. Singer, “A survey on structure from motion,” Acta Numer., 2017. DOI: 10.1017/S096249291700006X (cit. on p. 2).

[14] Parallax, n. In OED Online, Oxford University Press. [Online]. Available: https://www.oed.com/view/Entry/137461 (visited on 06/29/2020) (cit. on p. 2).

[15] V. Aslantas, “A depth estimation algorithm with a single image,” Opt Express, vol. 15, no. 8, pp. 5024–5029, Apr. 16, 2007, ISSN: 1094-4087. DOI: 10.1364/OE.15.005024 (cit. on p. 3).

[16] A. S. Joglekar, D. Joshi, R. Khemani, S. Nair, and S. Sahare, “Depth estimation using monocular camera,” 2011 (cit. on p. 3).

[17] K. Wang, E. Dunn, J. Tighe, and J.-M. Frahm, “Combining semantic scene priors and haze removal for single image depth estimation,” in IEEE Winter Conference on Applications of Computer Vision, ISSN: 1550-5790, Mar. 2014, pp. 800–807. DOI: 10.1109/WACV.2014.6836021 (cit. on p. 3).

[18] D. Eigen, C. Puhrsch, and R. Fergus, “Depth map prediction from a single image using a multi-scale deep network,” arXiv:1406.2283 [cs], Jun. 9, 2014. arXiv: 1406.2283. [Online]. Available: http://arxiv.org/abs/1406.2283 (visited on 11/02/2019) (cit. on p. 4).

[19] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, “Vision meets robotics: The KITTI dataset,” Int. J. Rob. Res., vol. 32, no. 11, pp. 1231–1237, Sep. 1, 2013, ISSN: 0278-3649. DOI: 10.1177/0278364913491297. [Online]. Available: https://doi.org/10.1177/0278364913491297 (visited on 03/09/2020) (cit. on pp. 4, 8).

[20] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab, “Deeper depth prediction with fully convolutional residual networks,” in 2016 Fourth International Conference on 3D Vision (3DV), Oct. 2016, pp. 239–248. DOI: 10.1109/3DV.2016.32 (cit. on pp. 4, 12–14).
[21] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao, “Deep ordinal regression network for monocular depth estimation,” presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 2002–2011. [Online]. Available: http://openaccess.thecvf.com/content_cvpr_2018/html/Fu_Deep_Ordinal_Regression_CVPR_2018_paper.html (visited on 12/14/2019) (cit. on p. 4).

[22] H. Ren, M. El-khamy, and J. Lee, “Deep robust single image depth estimation neural network using scene understanding,” presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2019, pp. 37–45. [Online]. Available: http://openaccess.thecvf.com/content.CVPRW.2019/html/Deep_Vision_Workshop/Ren_Deep_Robust_Single_Image_Depth_Estimation_Neural_Network_Using_Scene.CVPRW.2019_paper.html (visited on 05/02/2020) (cit. on p. 4).

[23] J. H. Lee, M.-K. Han, D. W. Ko, and I. H. Suh, “From big to small: Multi-scale local planar guidance for monocular depth estimation,” arXiv:1907.10326 [cs], Aug. 26, 2019. arXiv:1907.10326. [Online]. Available: http://arxiv.org/abs/1907.10326 (visited on 11/06/2019) (cit. on p. 4).

[24] W. Yin, Y. Liu, C. Shen, and Y. Yan, “Enforcing geometric constraints of virtual normal for depth prediction,” presented at the Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 5684–5693. [Online]. Available: http://openaccess.thecvf.com/content_ICCV_2019/html/Yin_Enforcing_Geometric_Constraints_of_Virtual_Normal_for_Depth_Prediction_ICCV_2019_paper.html (visited on 05/02/2020) (cit. on p. 4).

[25] R. Diaz and A. Marathe, “Soft labels for ordinal regression,” presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 4738–4747. [Online]. Available: http://openaccess.thecvf.com/content_CVPR_2019/html/Diaz_Soft_Labels_for_Ordinal_Regression_CVPR_2019_paper.html (visited on 05/02/2020) (cit. on p. 4).
[26] B. Li, Y. Dai, and M. He, “Monocular depth estimation with hierarchical fusion of dilated CNNs and soft-weighted-sum inference,” Pattern Recognition, vol. 83, pp. 328–339, Nov. 1, 2018, ISSN: 0031-3203. DOI: 10.1016/j.patcog.2018.05.029. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0031320318302097 (visited on 03/16/2020) (cit. on p. 4).

[27] L. Liebel and M. Körner, “MultiDepth: Single-image depth estimation via multi-task regression and classification,” in 2019 IEEE Intelligent Transportation Systems Conference (ITSC), Oct. 2019, pp. 1440–1447. DOI: 10.1109/ITSC.2019.8917177 (cit. on p. 4).

[28] P. Wang, X. Shen, Z. Lin, S. Cohen, B. Price, and A. Yuille, “Towards unified depth and semantic prediction from a single image,” in 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), ISSN: 1063-6919, Jun. 2015, pp. 2800–2809. DOI: 10.1109/CVPR.2015.7298897 (cit. on pp. 4, 53).

[29] R. Ji, L. Cao, and Y. Wang, “Joint depth and semantic inference from a single image via elastic conditional random field,” Pattern Recognition, Compositional Models and Structured Learning for Visual Recognition, vol. 59, pp. 268–281, Nov. 1, 2016, ISSN: 0031-3203. DOI: 10.1016/j.patcog.2016.03.016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0031320316001114 (visited on 06/18/2020) (cit. on pp. 4, 53).

[30] J. Jiao, Y. Cao, Y. Song, and R. Lau, “Look deeper into depth: Monocular depth estimation with semantic booster and attention-driven loss,” in Computer Vision – ECCV 2018, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds., ser. Lecture Notes in Computer Science, Cham: Springer International Publishing, 2018, pp. 55–71, ISBN: 978-3-030-01267-0. DOI: 10.1007/978-3-030-01267-0_4 (cit. on pp. 4, 53).

[31] M. Ochs, A. Kretz, and R. Mester, “SDNet: Semantically guided depth estimation network,” in Pattern Recognition, G. A. Fink, S. Frintrop, and X. Jiang,
[32] P. Zama Ramirez, M. Poggi, F. Tosi, S. Mattoccia, and L. Di Stefano, “Geometry meets semantics for semi-supervised monocular depth estimation,” in Computer Vision – ACCV 2018, C. V. Jawahar, H. Li, G. Mori, and K. Schindler, Eds., ser. Lecture Notes in Computer Science, Cham: Springer International Publishing, 2019, pp. 298–313, ISBN: 978-3-030-20893-6. DOI: 10.1007/978-3-030-20893-6_19 (cit. on pp. 4, 53).

[33] Z. Zhang, Z. Cui, C. Xu, Y. Yan, N. Sebe, and J. Yang, “Pattern-affinitive propagation across depth, surface normal and semantic segmentation,” presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 4106–4115. [Online]. Available: http://openaccess.thecvf.com/content_CVPR_2019/html/Zhang_Pattern-Affinitive_Propagation_Across_Depth_Surface_Normal_and_Semantic_Segmentation_CVPR_2019_paper.html (visited on 05/02/2020) (cit. on pp. 4, 53).

[34] X. Lin, D. Sánchez-Escobedo, J. R. Casas, and M. Pardàs, “Depth estimation and semantic segmentation from a single RGB image using a hybrid convolutional neural network,” Sensors (Basel), vol. 19, no. 8, Apr. 15, 2019, ISSN: 1424-8220. DOI: 10.3390/s19081795 (cit. on pp. 4, 53).

[35] P.-Y. Chen, A. H. Liu, Y.-C. Liu, and Y.-C. F. Wang, “Towards scene understanding: Unsupervised monocular depth estimation with semantic-aware representation,” in 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), ISSN: 2575-7075, Jun. 2019, pp. 2619–2627. DOI: 10.1109/CVPR.2019.00273 (cit. on pp. 4, 53).

[36] D.-h. Kwak and S.-h. Lee, “A novel method for estimating monocular depth using cycle GAN and segmentation,” Sensors, vol. 20, no. 9, p. 2567, Jan. 2020, Number: 9 Publisher: Multidisciplinary Digital Publishing Institute. DOI:
[37] N. Schneider, L. Schneider, P. Pinggera, U. Franke, M. Pollefeys, and C. Stiller, “Semantically guided depth upsampling,” in Pattern Recognition, Springer, Cham, Sep. 12, 2016, pp. 37–48. doi: 10.1007/978-3-319-45886-1_4. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-45886-1_4 (visited on 12/16/2019) (cit. on pp. 5, 53).

[38] V. C. Guizilini, R. Hou, J. Li, R. Ambrus, and A. Gaidon, “Semantically-guided representation learning for self-supervised monocular depth,” ICLR 2020, 2020 (cit. on pp. 5, 53).

[39] B. Liu, S. Gould, and D. Koller, “Single image depth estimation from predicted semantic labels,” in 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, ISSN: 1063-6919, Jun. 2010, pp. 1253–1260. doi: 10.1109/CVPR.2010.5539823 (cit. on pp. 5, 53).

[40] O. H. Jafari, O. Groth, A. Kirillov, M. Y. Yang, and C. Rother, “Analyzing modular CNN architectures for joint depth prediction and semantic segmentation,” in 2017 IEEE International Conference on Robotics and Automation (ICRA), May 2017, pp. 4620–4627. doi: 10.1109/ICRA.2017.7989537 (cit. on pp. 5, 53).

[41] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, “Indoor segmentation and support inference from RGBD images,” in Computer Vision – ECCV 2012, A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid, Eds., ser. Lecture Notes in Computer Science, Berlin, Heidelberg: Springer, 2012, pp. 746–760, isbn: 978-3-642-33715-4. doi: 10.1007/978-3-642-33715-4_54 (cit. on p. 7).

[42] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, “nuScenes: A multimodal dataset for autonomous driving,” arXiv:1903.11027 [cs, stat], Jan. 8, 2020. arXiv: 1903.
[43] (). “nuScenes dataset - data format,” [Online]. Available: https://www.nuscenes.org/data-format (visited on 05/12/2020) (cit. on p. 8).

[44] N. Charron, S. Phillips, and S. L. Waslander, “De-noising of lidar point clouds corrupted by snowfall,” in 2018 15th Conference on Computer and Robot Vision (CRV), ISSN: null, May 2018, pp. 254–261. DOI: 10.1109/CRV.2018.00043 (cit. on pp. 8, 10).

[45] P. Wang, X. Huang, X. Cheng, D. Zhou, Q. Geng, and R. Yang, “The ApolloScape open dataset for autonomous driving and its application,” IEEE Transactions on Pattern Analysis and Machine Intelligence, pp. 1–1, 2019, Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence, issn: 1939-3539. DOI: 10.1109/TPAMI.2019.2926463 (cit. on p. 8).

[46] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. MIT Press, 2016 (cit. on p. 10).

[47] B. Sun and K. Saenko, “From virtual to reality: Fast adaptation of virtual object detectors to real domains,” in BMVC, 2014. DOI: 10.5244/C.28.82 (cit. on p. 10).

[48] D. Vázquez, A. M. López, J. Marín, D. Ponsa, and D. Gerónimo, “Virtual and real world adaptation for pedestrian detection,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 36, no. 4, pp. 797–809, Apr. 2014, Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence, issn: 1939-3539. DOI: 10.1109/TPAMI.2013.163 (cit. on p. 10).

[49] J. Xu, S. Ramos, D. Vázquez, and A. M. López, “Domain adaptation of deformable part-based models,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 36, no. 12, pp. 2367–2380, Dec. 2014, Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence, issn: 1939-3539. DOI: 10.1109/TPAMI.2014.2327973 (cit. on p. 10).
[50] P. P. Busto, J. Liebelt, and J. Gall, “Adaptation of synthetic data for coarse-to-fine viewpoint refinement,” in *BMVC*, 2015. DOI: 10.5244/C.29.14 (cit. on p. 10).

[51] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez, “The SYNTHIA dataset: A large collection of synthetic images for semantic segmentation of urban scenes,” presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 3234–3243. [Online]. Available: https://www.cv-foundation.org/openaccess/content_cvpr_2016/html/Ros_The_SYNTHIA_Dataset_CVPR_2016_paper.html (visited on 03/09/2020) (cit. on p. 10).

[52] J. Xu, S. Ramos, D. Vázquez, and A. M. López, “Hierarchical adaptive structural SVM for domain adaptation,” *Int J Comput Vis*, vol. 119, no. 2, pp. 159–178, Sep. 1, 2016, issn: 1573-1405. DOI: 10.1007/s11263-016-0885-6. [Online]. Available: https://doi.org/10.1007/s11263-016-0885-6 (visited on 09/24/2020) (cit. on p. 10).

[53] M. Wrenninge and J. Unger, “Synscapes: A photorealistic synthetic dataset for street scene parsing,” *arXiv:1810.08705 [cs]*, Oct. 19, 2018. arXiv: 1810.08705. [Online]. Available: http://arxiv.org/abs/1810.08705 (visited on 03/31/2020) (cit. on pp. 10, 46, 57).

[54] Y. Luo, L. Zheng, T. Guan, J. Yu, and Y. Yang, “Taking a closer look at domain shift: Category-level adversaries for semantics consistent domain adaptation,” *arXiv:1809.09478 [cs]*, Apr. 1, 2019. arXiv: 1809.09478. [Online]. Available: http://arxiv.org/abs/1809.09478 (visited on 05/03/2020) (cit. on p. 10).

[55] S. I. Nikolenko, “Synthetic data for deep learning,” *arXiv:1909.11512 [cs]*, Sep. 25, 2019. arXiv: 1909.11512. [Online]. Available: http://arxiv.org/abs/1909.11512 (visited on 04/08/2020) (cit. on p. 10).

[56] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition*
(CVPR), ISSN: 1063-6919, Jun. 2016, pp. 770–778. DOI: 10.1109/CVPR.2016.
90 (cit. on p. 12).

[57] G. Neuhold, T. Ollmann, S. Rota Bulò, and P. Kontschieder, “The mapillary vistas dataset for semantic understanding of street scenes,” in *International Conference on Computer Vision (ICCV)*, 2017. [Online]. Available: https://www.mapillary.com/dataset/vistas (cit. on pp. 17, 36).

[58] D. Hernandez-Juarez, L. Schneider, A. Espinosa, D. Vázquez, A. M. López, U. Franke, M. Pollefeys, and J. C. Moure, “Slanted stixels: Representing san francisco’s steepest streets,” *arXiv:1707.05397 [cs]*, Jul. 17, 2017. arXiv: 1707.05397. [Online]. Available: http://arxiv.org/abs/1707.05397 (visited on 11/01/2019) (cit. on pp. 22, 36, 44).

[59] Random seed, in *Wikipedia*, Page Version ID: 933429432, Dec. 31, 2019. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Random_seed&oldid=933429432 (visited on 03/27/2020) (cit. on p. 22).

[60] Y. Cabon, N. Murray, and M. Humenberger, “Virtual KITTI 2,” *arXiv:2001.10773 [cs, eess]*, Jan. 29, 2020. arXiv: 2001.10773. [Online]. Available: http://arxiv.org/abs/2001.10773 (visited on 04/03/2020) (cit. on pp. 25, 44).

[61] (). “Google colaboratory.” Library Catalog: colab.research.google.com, [Online]. Available: https://colab.research.google.com/github/MarkDaoust/models/blob/segmentation_blogpost/samples/outreach/blogs/segmentation_blogpost/imageSegmentation.ipynb#scrollTo=iPV7RMA9TjPC (visited on 05/29/2020) (cit. on p. 34).

[62] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, Springer, Cham, Oct. 5, 2015, pp. 234–241. DOI: 10.1007/978-3-319-24574-4_28. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-24574-4_28 (visited on 12/18/2019) (cit. on p. 34).
[63] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, “The cityscapes dataset for semantic urban scene understanding,” presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 3213–3223. [Online]. Available: https://www.cv-foundation.org/openaccess/content_cvpr_2016/html/Cordts_The_Cityscapes_Dataset_CVPR_2016_paper.html (visited on 01/06/2020) (cit. on p. 36).

[64] S. R. Richter, Z. Hayder, and V. Koltun, “Playing for benchmarks,” in 2017 IEEE International Conference on Computer Vision (ICCV), Oct. 2017, pp. 2232–2241. DOI: 10.1109/ICCV.2017.243. [Online]. Available: http://arxiv.org/abs/1709.07322 (cit. on p. 36).

[65] J. Uhrig, N. Schneider, L. Schneider, U. Franke, T. Brox, and A. Geiger, “Sparsity invariant CNNs,” in 2017 International Conference on 3D Vision (3DV), ISSN: 2475-7888, Oct. 2017, pp. 11–20. DOI: 10.1109/3DV.2017.00012 (cit. on p. 44).

[66] (). “Lyft level 5 AV dataset 2019,” Lyft Level 5. Library Catalog: level5.lyft.com, [Online]. Available: https://level5.lyft.com/dataset/ (visited on 03/11/2020) (cit. on p. 44).

[67] B. Hurl, K. Czarnecki, and S. Waslander, “Precise synthetic image and LiDAR (PreSIL) dataset for autonomous vehicle perception,” in 2019 IEEE Intelligent Vehicles Symposium (IV), ISSN: 2642-7214, Jun. 2019, pp. 2522–2529. DOI: 10.1109/IVS.2019.8813809 (cit. on p. 44).

[68] D. P. Kingma and J. Ba. (2015). “Adam: A method for stochastic optimization,” undefined. Library Catalog: www.semanticscholar.org, [Online]. Available: https://arxiv.org/pdf/1412.6980.pdf (visited on 07/10/2020) (cit. on p. 45).