Towards Recognition as a Regularizer in Autonomous Driving

Dissertation
der Mathematisch-Naturwissenschaftlichen Fakultät
der Eberhard Karls Universität Tübingen
zur Erlangung des Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)

vorgelegt von
Aseem Behl
aus Haryana, Indien

Tübingen
2021

Gedruckt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Eberhard Karls Universität Tübingen.

Tag der mündlichen Qualifikation:	26.07.2021
Dekan:	Prof. Dr. Thilo Stehle
1. Berichterstatter:	Prof. Dr.-Ing. Andreas Geiger
2. Berichterstatterin:	Prof. Dr. Zeynep Akata

Abstract

Autonomous driving promises great potential for social and economic benefits. While autonomous driving remains an unsolved problem, recent advances in computer vision have enabled considerable progress in development of autonomous vehicles. For instance, recognition methods such as object detection, semantic and instance semantic segmentation affords the self driving vehicles with the precise understanding of their surroundings which is critical to safe autonomous driving. Furthermore, with the advent of deep learning, machine recognition methods have reached human-like performance. Therefore, in this thesis, we propose different methods to exploit semantic cues from state-of-the-art recognition methods to improve performance of other tasks required to solve autonomous driving. To this end, in this thesis, we examine the effectiveness of recognition as a regularizer in two prominent problems in autonomous driving, namely, scene flow estimation and end-to-end learned autonomous driving.

Besides recognizing traffic participants and predicting their position, an autonomous car needs to precisely predict their 3D position in the future for tasks like navigation and planning. Scene flow is a prominent representation for such motion estimation where a 3D position and 3D motion vector is associated with every observed surface point in the scene. However, existing methods for 3D scene flow estimation often fail in the presence of large displacement or local ambiguities, e.g., at texture-less or reflective surfaces which are omnipresent in dynamic road scenes. Therefore, first, we address the problem of large displacements or local ambiguities by exploiting recognition cues such as semantic grouping and fine-grained geometric recognition to regularize scene flow estimation. We compute these cues using CNNs trained on a newly annotated dataset of stereo images and integrate them into a CRF-based model for robust 3D scene flow estimation. We also investigate the importance of recognition granularity, from coarse 2D bounding box estimates over 2D instance segmentations to fine-grained 3D object part predictions. From this study, we show that regularization from recognition cues significantly boosts the scene flow accuracy, in particular in challenging foreground regions. Secondly, we conclude that the instance segmentation cue is by far strongest, in our setting. Alongside, we demonstrate the effectiveness of our method on challenging scene flow benchmarks.

In contrast to cameras, laser scanners provide a 360 degree field of view with just one sensor, are generally unaffected by lighting conditions, and do not suffer from the quadratic error behavior of stereo cameras. Therefore, second, in this work, we extend the idea of semantic grouping as a regularizer for 3D motion to 3D point cloud measurements from LIDAR sensor observations. We achieve this goal by jointly predicting 3D scene flow as well as the 3D bounding box and semantically grouping the motion vectors using 3D object detections. In order to semantically group the motion vectors using 3D object detections, we need to predict pointwise rigid motion. We show that the traditional global representation of rigid body motion prohibits inference by CNNs, and propose a translation equivariant representation to circumvent this problem and amenable to CNN learning. For training our deep network, we augment real scans from with virtual objects, realistically modeling occlusions and simulating sensor noise. A thorough comparison with classic and learning-based techniques highlights the robustness of the proposed approach.

It is well known that recognition cues such as semantic segmentation can be used as an effective intermediate representation to regularize learning end-to-end learned driving policies. However, the task of street scene semantic segmentation requires expensive annotations. Furthermore, segmentation algorithms are often trained irrespective of the actual driving task, using auxiliary image-space loss functions which are not guaranteed to maximize driving metrics such as safety or distance traveled per intervention. Therefore, third, in this work, we analyze several recognition-based intermediate representations for end-to-end learned driving policies. We seek to quantify the impact of reducing segmentation annotation costs on learned behavior cloning agents. We systematically study the trade-off between annotation efficiency and driving performance, i.e., the types of classes labeled, the number of image samples used to learn the visual abstraction model, and their granularity (e.g., object masks vs. 2D bounding boxes). Our analysis uncovers several practical insights into how segmentation-based visual abstractions can be exploited in a more label efficient manner. Surprisingly, we find that state-of-the-art driving performance can be achieved with orders of magnitude reduction in annotation cost.

Kurzfassung

Autonomes Fahren verspricht ein großes Potenzial für soziale und wirtschaftliche Vorteile. Während das autonome Fahren nach wie vor ein ungelöstes Problem darstellt, haben die jüngsten Fortschritte in der Computer Vision erhebliche Fortschritte bei der Entwicklung autonomer Fahrzeuge ermöglicht. Zum Beispiel ermöglichen Erkennungsmethoden wie Objekterkennung, semantische und instanzsemantische Segmentierung den selbstfahrenden Fahrzeugen ein präzises Verständnis ihrer Umgebung, was für sicheres autonomes Fahren entscheidend ist. Darüber hinaus haben maschinelle Erkennungsmethoden mit dem Aufkommen von Deep Learning eine menschenähnliche Leistung erreicht. Daher schlagen wir in dieser Arbeit verschiedene Methoden vor, um semantische Hinweise von modernsten Erkennungsmethoden zu nutzen, um die Leistung anderer Aufgaben zu verbessern, die zur Lösung des autonomen Fahrens erforderlich sind. Zu diesem Zweck untersuchen wir in dieser Arbeit die Effektivität der Erkennung als Regularisierer in zwei prominenten Problemen des autonomen Fahrens, nämlich der Schätzung des Szenenflusses und dem Ende-zu-Ende-gelernten autonomen Fahren.

Neben der Erkennung und Vorhersage der Position von Verkehrsteilnehmern muss ein autonomes Auto für Aufgaben wie Navigation und Planung auch deren 3D-Position in der Zukunft präzise vorhersagen. Der Szenenfluss ist eine prominente Darstellung für eine solche Bewegungsschätzung, bei der jedem beobachteten Oberflächenpunkt in der Szene eine 3D-Position und ein 3D-Bewegungsvektor zugeordnet ist. Bestehende Methoden zur 3D-Szenenflussschätzung versagen jedoch oft bei großen Verschiebungen oder lokalen Mehrdeutigkeiten, z. B. bei texturlosen oder reflektierenden Oberflächen, die in dynamischen Straßenszenen allgegenwärtig sind. Daher adressieren wir zunächst das Problem großer Verschiebungen oder lokaler Mehrdeutigkeiten, indem wir Erkennungsmerkmale wie semantische Gruppierung und feinkörnige geometrische Erkennung zur Regularisierung der Szenenflussschätzung nutzen. Wir berechnen diese Hinweise mit CNNs, die auf einem neu annotierten Datensatz von Stereobildern trainiert wurden, und integrieren sie in ein CRF-basiertes Modell für eine robuste 3D-Szenenflussschätzung. Wir untersuchen auch die Bedeutung der Erkennungsgranularität, von groben 2D-Bounding-Box-Schätzungen über 2D-Instanzsegmentierungen bis hin zu feinkörnigen 3D-Objektteilvorhersagen. Aus dieser Studie geht hervor, dass die Regularisierung von Erkennungsmerkmalen die Genauigkeit des Szenenflusses signifikant erhöht, insbesondere in schwierigen Vordergrundregionen. Zweitens kommen wir zu dem Schluss, dass der Instanzsegmentierungs-Cue in unserer Umgebung bei weitem am stärksten ist. Daneben demonstrieren wir die Effektivität unserer Methode in anspruchsvollen Scene-Flow-Benchmarks.

Im Gegensatz zu Kameras bieten Laserscanner ein 360-Grad-Sichtfeld mit nur einem Sensor, sind im Allgemeinen unbeeinflusst von den Lichtverhältnissen und leiden nicht unter dem quadratischen Fehlerverhalten von Stereokameras. Daher erweitern wir in dieser Arbeit zweitens die Idee der semantischen Gruppierung als Regularisierer für 3D-Bewegungen auf 3D-Punktwolkenmessungen aus LIDAR-Sensorbeobachtungen. Wir erreichen dieses Ziel durch die gemeinsame Vorhersage des 3D-Szenenflusses sowie der 3D-Bounding Box und die semantische Gruppierung der Bewegungsvektoren unter Verwendung von 3D-Objektdetektionen. Um die Bewegungsvektoren mithilfe von 3D-Objekterkennungen semantisch zu gruppieren, müssen wir die punktweise starre Bewegung vorhersagen. Wir zeigen, dass die traditionelle globale Repräsentation der Starrkörperbewegung die Inferenz durch CNNs verbietet, und schlagen eine translationsäquivariante Repräsentation vor, um dieses Problem zu umgehen und für das CNN-Lernen geeignet zu sein. Um unser tiefes Netzwerk zu trainieren, erweitern wir reale Scans mit virtuellen Objekten, modellieren realistisch Verdeckungen und simulieren Sensorrauschen. Ein gründlicher Vergleich mit klassischen und lernbasierten Techniken unterstreicht die Robustheit des vorgeschlagenen Ansatzes.

Es ist bekannt, dass Erkennungshinweise wie die semantische Segmentierung als effektive Zwischenrepräsentation verwendet werden können, um gelernte Ende-zu-Ende-Fahrstrategien zu regularisieren. Die Aufgabe der semantischen Segmentierung von Straßenszenen erfordert jedoch teure Annotationen. Darüber hinaus werden Segmentierungsalgorithmen oft unabhängig von der eigentlichen Fahraufgabe trainiert, indem Hilfsverlustfunktionen verwendet werden, die nicht garantieren, dass sie Fahrmetriken wie Sicherheit oder zurückgelegte Strecke pro Eingriff maximieren. Daher analysieren wir in dieser Arbeit drittens mehrere erkennungsbasierte Zwischendarstellungen für Ende-zu-Ende-gelernte Fahrstrategien. Wir versuchen, die Auswirkungen einer Reduzierung der Segmentierungsannotationskosten auf Agenten, die gelerntes Verhalten klonen, zu quantifizieren. Wir untersuchen systematisch den Kompromiss zwischen Annotationseffizienz und Fahrleistung, d.h. Art der gelabelten Klassen, die Anzahl der Bildproben, die zum Lernen des visuellen Abstraktionsmodells verwendet werden, und deren Granularität (z.b., Objektmasken vs. 2D Bounding Boxes). Unsere Analyse deckt mehrere praktische Erkenntnisse darüber auf, wie segmentierungsbasierte visuelle Abstraktionen auf eine effizientere Weise genutzt werden können. Überraschenderweise stellen wir fest, dass die modernste Fahrleistung mit einer Größenordnung weniger Annotationskosten erreicht werden kann.

Acknowledgments

First and foremost, I would like to express my sincere gratitude to my advisor Prof. Dr-ing. Andreas Geiger. During my Ph.D., Andreas provided great support, motivation and guidance in all of my research. I feel fortunate to have worked in the world-class research environment Andreas has created at the Max-Planck-Institute. I will always be thankful to him for being so patient and for sharing his immense knowledge and experience. In addition, I would like to thank the rest of my committee: Prof. Dr. Zenyap Akata, Prof. Dr. Carsten Rother and Prof. Dr. rer. nat. Andreas Schilling for taking the time to evaluate my work. It is an honor to have you in my thesis committee.

I am very grateful to all the kind, friendly and talented people from the Autonomous Vision Group. Special thanks goes to Lars Mescheder, Joel Janai, Yiyi Liao, Despoina Paschalidou, Michael Niemeyer, Michael Ochsle, Benjamin Coors, Eshed Ohn-Bar, Carolin Schmitt, Simon Donne, Seungjun Nah, Katja Schwarz, Aditya Prakash, Kashyap Chitta. I will always cherish the moments spent together during lab retreats, barbeques and conferences. I am also grateful to the amazing staff at the Max-Planck-Institute who are always there to help: Kerstin McGaughey, Diana Rebmann, Melanie Feldhofer, Nicole Overbaugh, Telintor Ntounis, Joan Piles, and Jojumon Kavalan. I learned a lot working and collaborating with Joel Janai, Fatma Guney, Despoina Paschalidou, Simon Donne, Kashyap Chitta, Aditya Prakash, Eshed Ohn-Bar on many exciting projects during my Ph.D..

Finally, I am grateful to my parents, my brother and my wife, for their life long love and support.

Contents

List of Figures

List of Tables

1 Introduction

Autonomous driving promises great potential social and economic benefits such as reducing the cost of mobility, frequency of accidents, parking requirements, traffic congestion. Since the first successful demonstrations in the 1980s [DM92; DG88; Tho+88], considerable advances have been made towards development of autonomous cars specially in the last few years with the emergence of deep learning. However, autonomous driving outside of a constrained setting remains a challenging unsolved problem.

1.1 Modular approaches for Autonomous Driving

Traditionally, most industrial autonomous driving projects employ a modular approach for developing autonomous vehicles. This involves leveraging computer vision techniques to infer vital properties of the scene such as detecting location and motion of other vehicles, estimating derivable region. The output from the computer vision modules are then fed to a path planning and vehicle control module in order to generate the vehicle controls. The advantage of such approach is that each module can be developed in parallel by independent engineering teams. Moreover, the intermediate outputs of each module allow for better explainability of the driving decision policy.

1.2 Recognition in Autonomous Driving

For intelligent systems such as self-driving cars, the precise understanding of their surroundings is critical. Therefore, recognition is a key step in the modular pipeline for autonomous driving. In order to avoid collisions and crashes, it is necessary that the autonomous vehicle recognizes and localizes other vehicles, pedestrians and obstacles in its surroundings. It is also necessary for the autonomous vehicle to follow the traffic rules and regulation. Recognition which is the task of identifying a semantic object in an image enables the self driving car to recognize speed limit signs, traffic lights and road markings. The recognition problem has been approached using a variety of input modalities. Video cameras are the cheapest and most commonly used type of sensors for recognition. Active sensors that emit signals and observe their reflection, like laser scanners can provide range information which is helpful for detecting an object and localizing it in 3D. However, laser scanners often have a smaller resolution compared with video cameras. Depending on the weather conditions, time of day, or material properties, it can be problematic to rely on a single type of sensor alone. Cameras and laser scanners are affected by reflective or transparent surfaces, therefore, the combination of information from different sensors via sensor fusion allows for the robust integration of this complementary information. Human drivers also use recognition to safely

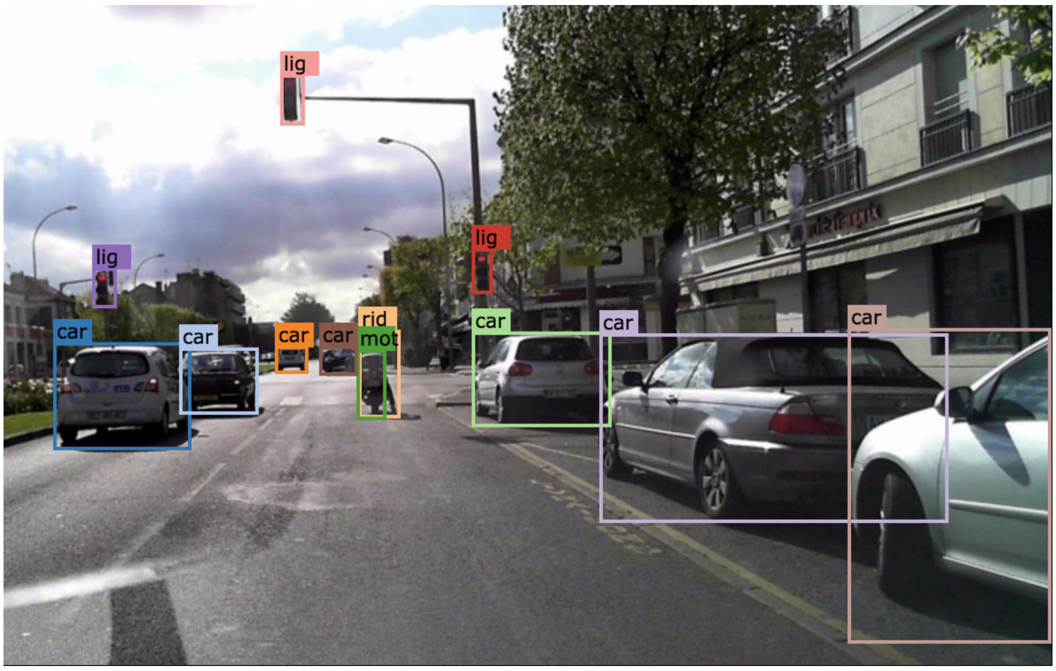


Figure 1.1: Object Detection for Autnomous Driving. In object detection, we are interested in finding all objects of certain classes in an image. These detections are usually represented with bounding boxes. Reproduced from [Yu+19].

drive a vehicle and can recognize and localize other cars, pedestrians and other semantic categories with high accuracy in diverse weather conditions and in presence of high amount of occlusions, shadows or reflections. While recognition is a relatively easy task for humans, machine recognition is a challenging problem because of diversity in intra-class appearance, scale, illumination of the scene and viewpoint of the sensor recording the observations. In addition, semantic objects of interest could appear with non-rigid deformations and can be occluded, making the problem more challenging. The following recognition tasks have significant applications in autonomous driving and are therefore most relevant to this work.

- **Object Detection:** Object detection is the task of identifying and localizing multiple semantic objects present in the scene. Each detected object is defined by a bounding box and semantic label as shown in Fig. 1.1.

- **Semantic Segmentation:** Semantic segmentation is a fundamental problem in computer vision and an intermediate goal towards solving higher-level tasks such as scene understanding. The goal of semantic segmentation is to assign each pixel in the image a label from a predefined set of categories. The task is illustrated in Fig. 1.2 using an example from the Cityscapes dataset[1] by Cordts et al. [Cor+16]. Segmentation of images into semantic regions that are typically found in street scenes, such as cars,

[1]https://www.cityscapes-dataset.com/

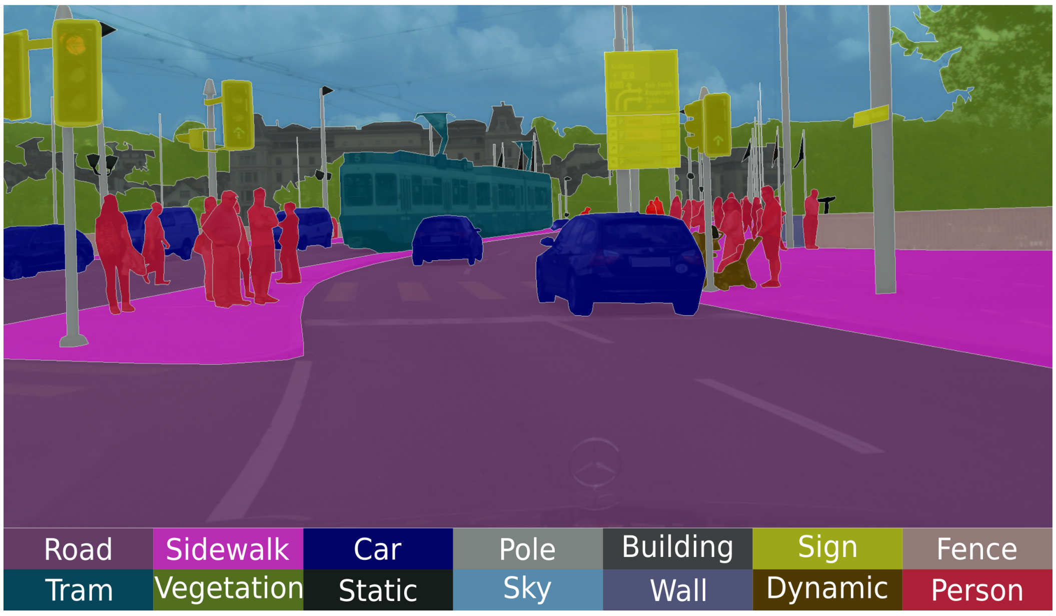


Figure 1.2: Semantic Segmentation for Autonomous Driving. *In semantic segmentation, the goal is to assign a semantic class label to each pixel in the image. Reproduced from [Cor+16].*

pedestrians, or road allows for a comprehensive understanding of the surrounding which is essential to autonomous navigation.

- **Instance Semantic Segmentation:** The goal of semantic instance segmentation is to simultaneously detect, segment and classify every individual object in an image. Unlike semantic segmentation, a solution to this task provides information about the position, semantics, shape, and count of individual objects.

1.3 Scene Flow in Autonomous Driving

Besides recognizing traffic participants and identifying their 3D locations, an autonomous car needs to precisely predict their 3D position in the future for tasks like navigation and planning. Notably, a critical input to modular autonomous driving approaches are such predictions about the future positions of other traffic actors. To this end, the autonomous vehicle requires knowledge about the 3D motion of other agents in the scene.

1.3.1 What is Scene Flow?

Scene flow is the most generic representation of this 3D motion introduced by [Ved+99; Ved+05]. It associates a 3D position and 3D motion vector with every observed surface point in the scene. Scene flow is most commonly estimated from two consecutive in time stereo image pairs from a calibrated stereo camera rig [HD07; Wed+08]. As illustrated in Fig. 1.3,

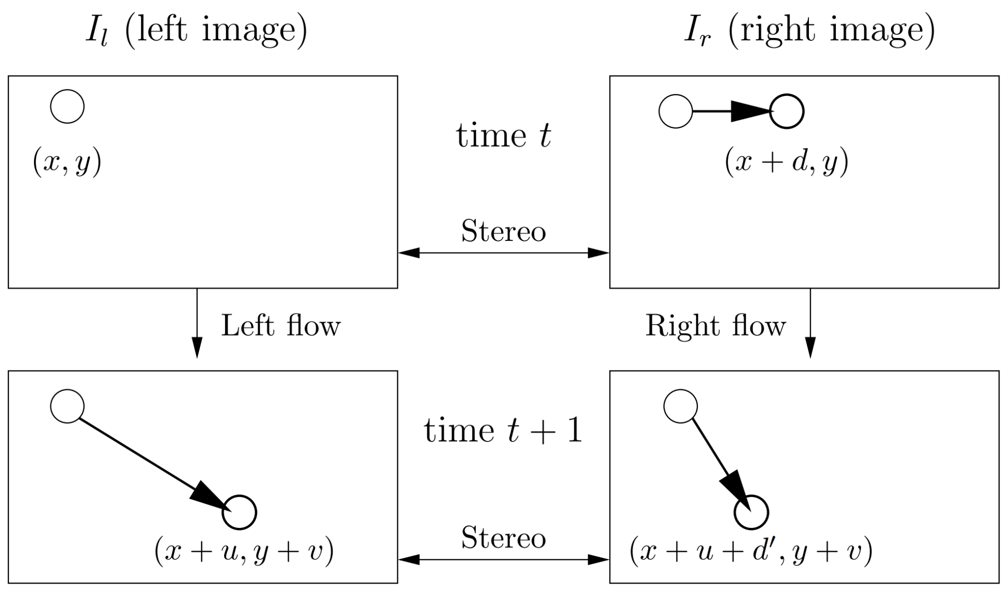


Figure 1.3: Scene Flow Estimation from Stereo Sequences. *The minimal setup for image-based scene flow estimation is given by two consecutive stereo image pairs. Reproduced from [HD07].*

stereo image based methods parametrize scene flow in the image space by employing a 2D representation of scene flow. In such a parameterization, scene flow is represented by optical flow which is the projection of the 3D motion onto the image plane of a camera and disparity maps for the two time points to compute 3D scene flow. The estimated disparity between the left and right camera at the first time frame allows for 3D reconstruction of every surface pixel in the image. The estimated optical flow can be then combined with estimated disparity at the next time frame to generate the second 3D reconstruction. The scene flow at every surface point is the difference between the two reconstructions. Thus, scene flow generalizes optical flow to 3D, or equally, dense stereo to dynamic scenes.

More formally, consider left and right image sequences from a stereo camera setup $I_l(x,y,t), I_r(x,y,t)$ where parameters (x,y) denote pixel coordinates and t is time. Scene flow can be defined by computing the corresponding point for every pixel (x,y) in the left reference image at time t in the other three frames. Let optical flow (u,v) denote the motion of the pixel (x,y) in the left image at time t upto time $t+1$. Let (d,d') denote the disparity between the left camera and right camera image at time t and $t+1$ respectively. Given the disparity (d,d') at time $(t,t+1)$ respectively, the corresponding point in the right camera image at time t is $(x+d,y)$. Given the optical flow (u,v), the corresponding point in the left camera image at time $t+1$ is $(x+u,y+v)$. Finally, the corresponding pixel to the pixel (x,y) in I_l is $(x+u+d',y+v)$. After computing the correspondences, the 3D location of the surface pixel in both time frames can be computed and hence fully describes the 3D motion of that surface point.

1.3.2 Computing Scene Flow

Classical methods for computing scene flow minimize an energy function consisting of a data term and a smoothness term. The data term is designed to model the consistency of visual features for matching points across frames. However, data term is not enough to obtain a unique solution because the scene flow estimation problem is ill-posed. Therefore, additional regularization constraints are required to obtain a unique solution. Typically, a smoothness term is added to the energy function to regularize the scene flow estimation by encouraging coherent structure and motion.

While humans are able perform this task effortlessly by integrating depth and motion cues from observations over time, existing scene flow methods rely heavily on local features (e.g., Census [ZW94] and Daisy [TLF10]) for computing the data term and for initialization. Therefore, scene flow estimation shares some of the challenges with stereo and optical flow estimation, such as matching ambiguities in weakly textured regions, large motions and varying illumination described below.

Large Displacements. As vehicles move at high velocity, matching based on local features can lead to large errors. As shown in Fig. 1.4 (top), in presence of large displacement, local visual features can lead to matching of the front wheel with the real wheel.

Varying illumination. The brightness consistency assumption is often violated in dynamic road scenes. It can occur because of changing illumination conditions because of sudden movement of clouds. In addition, shadows and external illumination from light sources from other vehicles and buildings can pose problems.

Textureless Regions. Textureless and specular regions such as windows of vehicles make matching based on local visual features challenging and therefore can lead to large errors in estimation.

As a consequence of the above listed challenges, even state-of-the-art methods fail in the presence of very large displacements or local ambiguities from, e.g., textureless or reflective surfaces. However, these challenges are omnipresent in outdoor road scenes, which is the application we focus on in this thesis.

1.4 Recognition as a Regularizer for Scene Flow from Stereo Images

First, in this work, we propose an approach to address the problem with large displacements or local ambiguities in scene flow estimation. Towards this goal, we propose regularization of scene flow estimation with recognition cues such as semantic grouping and fine-grained geometric recognition. In addition, we investigate the importance of recognition granularity, from coarse 2D bounding box estimates over 2D instance segmentations to fine-grained 3D object part predictions. We will formulate the output of the recognition task as a new constraint, besides standard constraints such as local appearance and 3D motion-smoothness

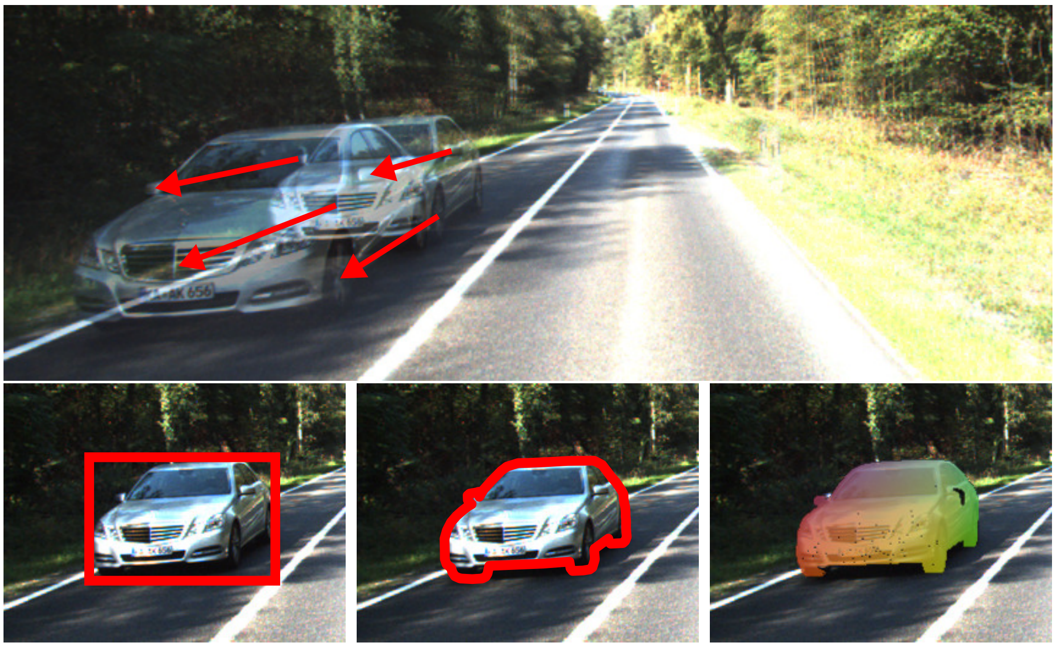


Figure 1.4: **Motivation for Recognition as a Regularizer for Scene Flow.** *Top: Two consecutive frames (overlaid) from the KITTI 2015 scene flow dataset. Large displacements and specular surfaces are challenging for current scene flow estimation algorithms. Bottom: Recognition can provide powerful geometric cues to help with this problem. In this work, we investigate: 2D bounding boxes, 2D instance segmentations and 3D object coordinates.*

within an image, in a CRF-based framework. For semantic grouping, we consider two scenarios: i) a bounding box around the visible part of each semantic instance, and ii) a pixel-wise segmentation of the visible part of each semantic instance. While the latter output provides a more detailed mask of the object, and hence may be more beneficial for the scene flow task, it is also a harder task to solve, and hence may contain segmentation errors. While these cues do not provide additional geometric evidence, they constrain the space of possible rigid body motions: pixels which are grouped together are likely to move as a single rigid object in the case of vehicles. We integrate both cues into the scene flow task by enforcing consistency between neighboring views (either in time or space). In short, a pixel within an instance region (bounding box or segment) in one frame should be mapped to an instance in the other frame. Furthermore, all pixels within an instance should move as one rigid entity. The 3D object coordinates specify the relative 3D location of an object's surface point with respect to the object's local coordinate frame as shown in Fig. 1.4 (bottom-right) with illustrative colors. Thus, they provide a detailed geometric cue for matching pixels in neighbouring frames, even in the presence of large displacements. The fine-grained geometric recognition and the semantic grouping have, certainly, a different trade-off between modeling-power versus prediction accuracy. If the recognition task was to be solved perfectly well, the continuous object part would be the strongest geometric cue, followed by the segmentation mask and the bounding box. On the other hand, as we will see experimentally, the continuous object parts are most challenging to predict precisely, followed by the segmentation mask and the bounding box. Given this trade-off, one of the key questions addressed in this work is: Which level of recognition is most beneficial for the scene flow task, when combining different cues, i.e. local appearance, 3D motion-smoothness and recognition-based geometric constraint, in a CRF-based framework?

1.5 Deep Learning for Scene Flow

Image-based scene flow methods have rarely made it into robotics applications. The reason for this is that robotics application requires real-time estimation of scene flow. However, most leading techniques take several minutes or hours to predict scene flow for each frame due to the inefficient optimization step. Deep learning methods employing convolutional network provide the promise of fast estimation for real time applications of scene flow methods which is required in autonomous driving vehicle. However, very few approaches for CNN based scene flow estimation have been proposed so far. SceneFlowNet [May+16] is one of the notable exceptions, which concatenates features from FlowNet [Dos+15] and DispNet [May+16] for image-based scene flow estimation.

1.6 Scene Flow Estimation from 3D Point Clouds

Despite significant progress in image-based 3D scene flow estimation, the performance of such approaches has not yet reached the fidelity required by many applications. The reason for this is that stereo-based scene flow methods suffer from a fundamental flaw, the "curse of two-view geometry": it can be shown that the depth error grows quadratically with the distance to the observer [Len+11]. This causes problems for the large baselines and object depths often found in self-driving cars. Consequently, most modern self-driving car platforms rely on LIDAR technology for 3D geometry perception. In contrast to cameras, laser scanners provide a 360 degree field of view with just one sensor, are generally unaffected by lighting conditions, and do not suffer from the quadratic error behavior of stereo cameras. Therefore, there have been several methods proposed recently for estimating 3D scene flow from pairs of unstructured 3D point clouds. Dewan et al. [Dew+16b] propose a 3D scene flow approach where local SHOT descriptors [TSS10] are associated via a CRF that incorporates local smoothness and rigidity assumptions. However, local shape representations such as SHOT often fail in the presence of noisy or ambiguous inputs.

1.7 Recognition as a Regularizer for Scene Flow from 3D point clouds

Second, in this work, we propose to estimate 3D motion from such unstructured point clouds using a deep neural end-to-end trainable model that is able to learn local and global statistical relationships directly from 3D point cloud data. Moreover, we extend the idea of semantic grouping as a regularizer for 3D motion to 3D point cloud measurements from LIDAR sensor observations. We achieve this goal by jointly predicting 3D scene flow as well as the 3D bounding box and semantically grouping the motion vectors using 3D object detections. In a single forward pass, our model jointly predicts 3D scene flow as well as the 3D bounding box and rigid body motion of objects in the scene. In order to semantically group the motion vectors using 3D object detections, we need to predict pointwise rigid motion. We show that the traditional global representation of rigid body motion prohibits inference by CNNs, and propose a translation equivariant representation to circumvent this problem and amenable to CNN learning. Furthermore for training our deep network, we generate large diverse training datasets by augmenting real scans from KITTI with virtual objects, realistically modeling occlusions and simulating sensor noise. A thorough comparison with classic and learning-based techniques highlights the robustness of the proposed approach.

1.8 End-to-end Learned approaches for Autonomous Driving

A major drawback of modular approaches for autonomous driving is the fact that solving of human-designed intermediate sub-problems may not be optimal for the driving task performance. Moreover, each sub-problem module is developed in isolation optimizing the problem specific loss. For instance, estimating accurate motion for far away objects in the scene may not be important for safe autonomous driving. However, the independently developed motion planning algorithm lacks information about the importance weights of other cars in the scene and therefore assigns equal weight to all objects. Thus, the system is wasting computational resources on irrelevant tasks leading to overall sub-optimal driving performance. An alternative to modular pipelines is end-to-end learning-based models which try to learn a policy, i.e. a function from observations to actions using a generic model such as a deep neural network. The network parameters of end-to-end driving methods are typically learned via behavior cloning by replicating the behavior of a teacher using expert demonstrations or using reinforcement learning to explore the environment by trial and error. Behavior cloning approaches learn to map sensor observations, such as RGB images, to desired driving behavior by learning to clone the behavior of an expert. Thus, these approaches fall into the category of supervised learning techniques. Most commonly, a deep neural network is employed to represent the mapping from observations to expert actions. In the 1980s, Pomerleau [Pom88] propose ALVINN, the first demonstration of imitation learning for self-driving vehicles using a small fully connected neural network. Few decades later, Bojarski et al. [Boj+16] propose a deeper end-to-end deep convolutional neural network for lane following, illustrated in Fig. 1.5, that maps images from the front-facing camera of a car to steering angles, given expert data. Codevilla et al. [Cod+18] extend this approach for lane changes or turns by proposing a conditional imitation learning framework to learn a driving policy for steering and throttle control from a high-level navigational input in addition to the observations from the camera. The high-level navigational input represents the driver's intention, such as the direction to take at the next intersection, which cannot be recovered from sensory input alone. However, these methods for end-to-end learned driving

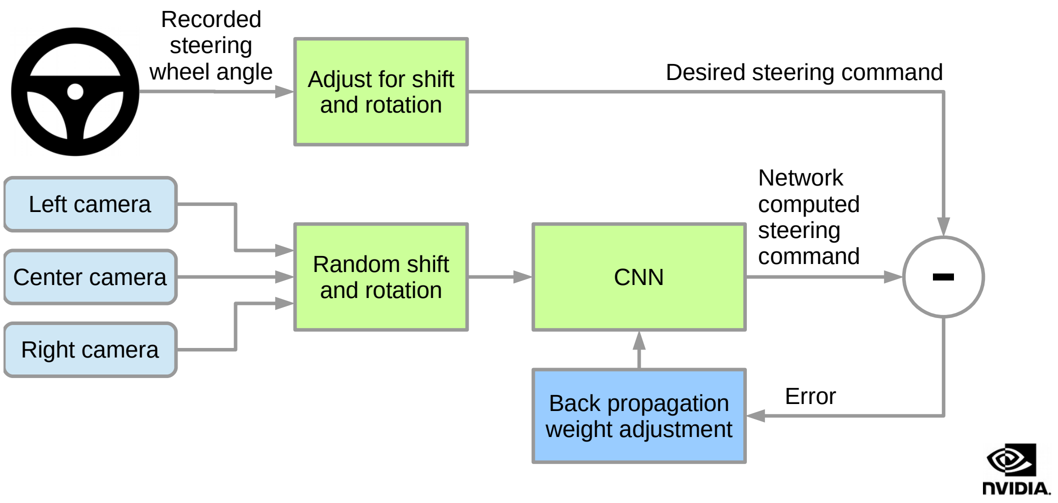

Recorded steering wheel angle
Adjust for shift and rotation
Desired steering command
Left camera
Center camera
Right camera
Random shift and rotation
CNN
Network computed steering command
-
Back propagation weight adjustment
Error
NVIDIA.

Figure 1.5: End-to-end Learning for Lane Following. *A block diagram of an end-to-end model for lane following proposed by [Boj+16]. Conditioned on the image, a CNN estimates a steering command which is compared to the expert command for tuning the CNN weights in order to bring the CNN output closer to the desired output. Reproduced from [Boj+16]*

models working directly with raw image sensor observations fail to generalize to different visual conditions like change in weather or city neighborhood.

1.9 Recognition as a Regularizer for End-to-end Learned Driving

Therefore, there has been a surge in interest on using recognition cues as *visual priors* for learning end-to-end control policies with improved performance, generalization and sample efficiency [ZKK19; Mou+19; Sax+19]. Instead of learning to act directly from image observations, a visual prior is enforced by feeding a visual representation such as semantic segmentation as an input to a policy learning algorithm, e.g., [Sax+19; SS16; Mül+18].

However, the task of street scene semantic segmentation requires expensive annotations. Besides, it is unclear which semantic classes are relevant to the driving task and to which granularity they should be labeled. Furthermore, segmentation algorithms are often trained irrespective of the actual driving task, using auxiliary image-space loss functions which are not guaranteed to maximize driving metrics such as safety or distance traveled per intervention.

Therefore, last, in this work, we seek to quantify the impact of reducing segmentation annotation costs on learned behavior cloning agents. We analyze several segmentation-based intermediate representations. We use these *visual abstractions*, which are compact recognition-based representations of the scene with fewer classes, coarser annotation, and learned with little supervision, to systematically study the trade-off between annotation efficiency and driving performance, i.e., the types of classes labeled, the number of image

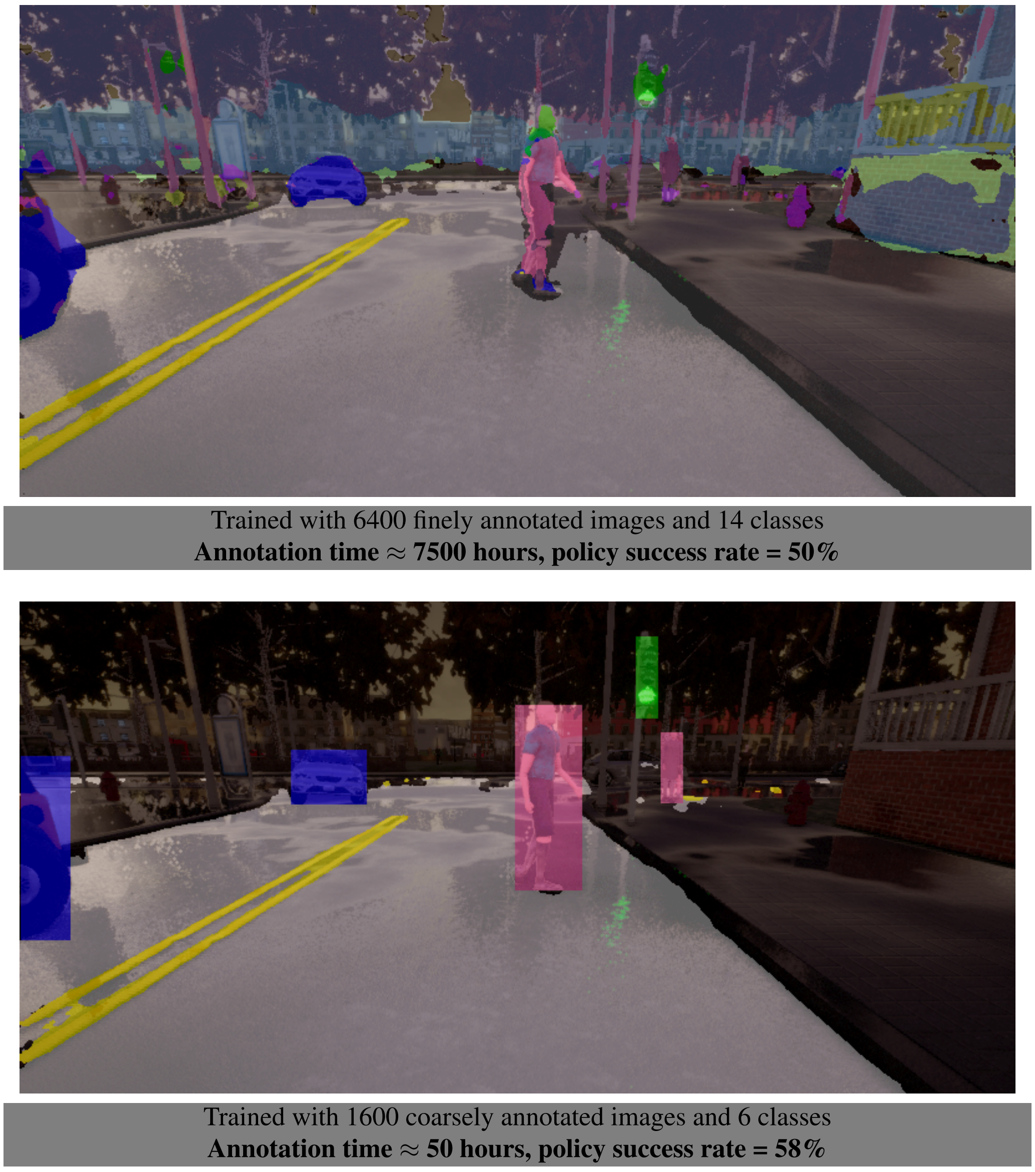

Trained with 6400 finely annotated images and 14 classes
Annotation time ≈ 7500 hours, policy success rate = 50%
Trained with 1600 coarsely annotated images and 6 classes
Annotation time ≈ 50 hours, policy success rate = 58%

Figure 1.6: Label efficient visual abstractions for learning driving policies. *To address issues with obtaining time-consuming annotations, we analyze image-based representations that are both* ***efficient*** *in terms of annotation cost (e.g., bounding boxes), and* ***effective*** *when used as intermediate representations for learning a robust driving policy. Considering six coarse safety-critical semantic categories and combining non-salient classes (e.g., sidewalk and building) into a single class can significantly reduce annotation cost while at the same time resulting in more robust driving performance.*

samples used to learn the visual abstraction model, and their granularity (e.g., object masks vs. 2D bounding boxes). Surprisingly, we find that certain visual abstractions learned with only a fraction of the original labeling cost can still perform as well or better when used as inputs for training behavior cloning policies (see Fig. 1.6). Beyond label efficiency, we find several additional training benefits when leveraging visual abstractions, such as a significant reduction in the variance of the learned policy when compared to state-of-the-art end-to-end driving models.

1.10 Contributions

The contributions of this thesis can be summarized as:

- We address the problem with large displacements or local ambiguities by regularization of scene flow estimation with recognition cues:
  - A new 3D scene flow method, leveraging recognition-based geometric cues to achieve state-of-the-art performance on the challenging KITTI 2015 scene flow benchmark, at the time of submission.
  - A detailed ablation study of the importance of recognition granularity, from coarse 2D bounding boxes over 2D instance segmentations to fine-grained 3D object part predictions, within our scene flow framework.
  - High-quality, instance-level annotations of 400 stereo images from the KITTI 2015 benchmark [MG15].
- Second, we extend the idea of semantic grouping as a regularizer for 3D motion to 3D point cloud measurements from LIDAR sensor observations:
  - We present *PointFlowNet*, an end-to-end trainable model for joint 3D scene flow and rigid motion prediction and 3D object detection from pairs of unstructured LIDAR 3D point clouds data, as captured from a (self-driving) car.
  - We show that a global representation is not suitable for rigid motion prediction, and propose a local translation-equivariant representation to mitigate this problem.
  - We augment the KITTI dataset with virtual cars, taking into account occlusions and simulating sensor noise, to provide more (realistic) training data.
  - We demonstrate that our approach compares favorably to the state-of-the-art.
- Third, we systemically study which properties of a recognition prior are useful for end-to-end learning of autonomous driving policies:
  - Given the same amount of training data, we empirically show that using classes less relevant to the driving policy can lead to degraded performance. We find that only few of the commonly used classes are directly relevant for the driving task.
  - We demonstrate that despite requiring only a few hundred annotated images in addition to the expert driving demonstrations, training a behavior cloning policy

with visual abstractions can significantly outperform methods which learn to drive from raw images, as well as existing state-of-the-art methods that require a prohibitive amount of supervision.

– We further show that our visual abstractions lead to a large variance reduction when varying the training seed which has been identified as a challenging problem in imitation learning [Cod+19].

1.11 Thesis Overview

We start with chapter 2, where we provide a short survey of different methods in machine recognition, namely, object detection and semantic segmentation. We then briefly discuss methods of scene flow estimation and end-to-end learned autonomous driving most related to our work. In chapter 3, we address the problem of large displacements or local ambiguities in scene flow estimation by exploiting recognition cues such as semantic grouping and fine-grained geometric recognition to regularize scene flow estimation in a CRF-based model. In chapter 4, we propose a novel CNN to extend the idea of semantic grouping as a regularizer for 3D motion estimation from LIDAR sensor observations. In chapter 5, we analyze several recognition-based cues and propose label-efficient visual representations to improve the generalization of end-to-end learned autonomous driving policies. Finally, we conclude our work in chapter 6 and provide an outlook for future research directions.

2 Related Work

2.1 Object Detection

Classical object detection systems usually consist of multiple steps that are applied consecutively to solve the object detection task. A classical detection pipeline usually comprises the following steps: preprocessing, region of interest extraction (ROI), object classification, and verification or refinement. In the preprocessing step, tasks such as exposure and gain adjustment, as well as camera calibration and image rectification, are usually performed.

Regions of interest can be extracted using a sliding window approach, which shifts a window over the image at different scales. As exhaustive search is very expensive, several heuristics have been proposed for reducing the search space. Typically, the number of evaluations is reduced by assuming a certain ratio, size, and position of candidate bounding boxes. Apart from that, image features, stereo, or optical flow can be leveraged for focusing the search on relevant regions. Broggi et al. [Bro+00], for instance, leverage morphological characteristics (size, ratio, and shape), vertical symmetry of human shape, and distance information obtained from stereo for the extraction of relevant ROIs. Selective Search [Uij+13] is an alternative approach to generate regions of interest. Instead of an exhaustive search over the full image domain, selective search exploits a segmentation of the image to extract approximate locations efficiently.

The next step is the processing of candidate image regions from sliding window to verify them and classify objects. The classification of all candidates in an image can be quite costly due to the vast amount of image regions that need to be processed. Therefore, a fast decision is necessary which quickly discards candidates in the background region of the image. Viola, Jones, and Snow [VJS05] combine simple and efficient classifiers, learned using AdaBoost, in a cascade that allows them to quickly discard false candidates while spending more time on promising regions. With the work of Dalal and Triggs [DT05], linear Support Vector Machines (SVMs) in combination with Histogram of Orientation (HOG) features have become popular tools for classification. Benenson et al. [Ben+14] conclude that the number and diversity of features is clearly an important factor for the performance of classifiers since the classification problem becomes easier with higher dimensional representations. Consequently, today, all state-of-the-art object detection systems use convolutional neural networks to learn expressive features in an end-to-end fashion from large datasets [Cai+16; Xia+17; Zhu+16; YCL16; Che+15c; Ren+15; Gir15].

2.1.1 Deep Learning for Detection

All previous methods rely on hand-crafted features that are difficult to design and limited in their representation capabilities. With the renaissance of deep learning [KSH12], con-

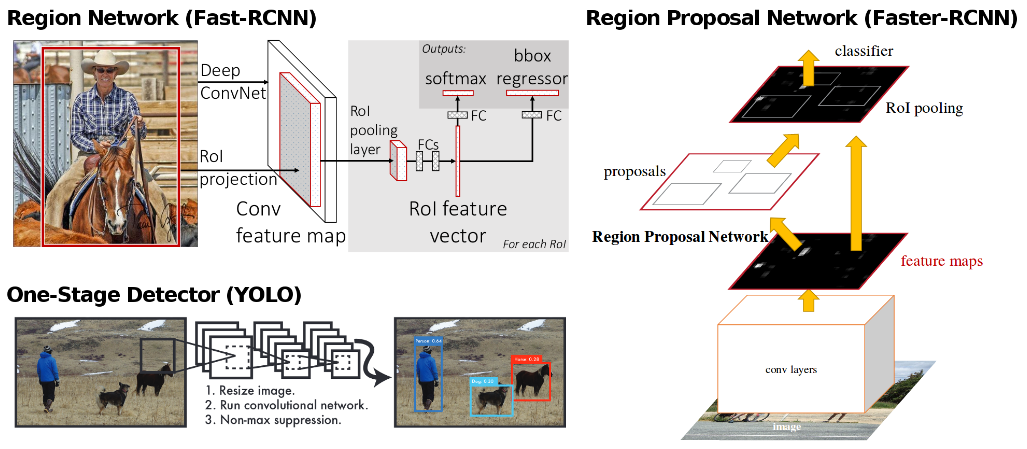


Figure 2.1: Object Detection Networks. *Illustration of the three popular object detection networks. Upper left: Region-based network Fast-RCNN [Gir15] that works on regions. Right: Region proposal network Faster-RCNN [Ren+15] that learn to extract regions. Lower left: One-stage detector YOLO [Red+16] that formulates the detection task as regression problem. Reproduced from [Gir15; Red+16; Ren+17].*

volutional neural networks have been applied to the object detection problem, resulting in significantly increased performance. Examples of the three most popular architectures are illustrated in Fig. 2.1.

Sermanet et al. [Ser+13] introduced CNNs to the pedestrian detection problem by learning the extraction of expressive features in an unsupervised fashion using convolutional sparse auto-encoders. Eventually, they train a classifier in an end-to-end supervised fashion while extracting the features with a sliding window scheme and jointly fine-tuning the auto-encoders. However, they use a shallow network with a small receptive field, which allows precise localization of the objects using a sliding window approach. In contrast, deeper networks with larger receptive fields complicate the precise localization because local information is extracted in earlier layers, while high-level information is represented in deeper layers. Therefore, Girshick et al. [Gir+14] propose R-CNNs to solve the CNN localization problem via a “recognition using regions” paradigm. They generate many region proposals using selective search [Uij+13], extract a fixed-length feature vector for each proposal using a CNN and classify each region with a linear SVM. Region-based CNNs are computationally expensive but several improvements have been proposed to reduce the computational burden [He+14; Gir15]. He et al. [He+14] use spatial pyramid pooling which allows computing a convolutional feature map for the entire image with only one run of the CNN in contrast to R-CNN that needs to be applied on many image regions. Girshick [Gir15] (Fast-RCNN) further improve upon these results by proposing a single-stage training algorithm using a multi-task loss that jointly learns to classify object proposals and refine their spatial locations.

In region-based CNNs, the classical region proposal algorithm remained the primary computational bottleneck and the main factor limiting performance. Therefore, Ren et al.

[Ren+15] (Faster-RCNN) introduced Region Proposal Networks (RPN), which share full-image convolutional features with the detection network and thus do not incur additional computational costs. RPNs are trained end-to-end to generate high-quality region proposals, which are classified using the Fast R-CNN detector [Gir15]. In chapter 5, we employ a Faster-RCNN based detector for generating the 2D detections of objects from outdoor driving scene images.

Eventually, one-stage detectors [Ser+14; Red+16; Liu+16; RF17; Lin+17b] completely got rid of the region proposal step by formulating the object detection task as a regression problem. The first one-stage detector by Sermanet et al. [Ser+14] was a deep convolutional version of the sliding window approach. They extract features with a CNN and apply a classifier network based on AlexNet [KSH12] on the extracted feature maps in a sliding window fashion. Redmon et al. [Red+16] (YOLO) instead suggest to jointly learn spatially separated bounding boxes and class probabilities from the topmost feature maps of a network based on GoogLeNet [Sze+15]. This allows them to achieve real-time performance and eventually YOLO9000 [RF17] to outperform the Region Proposal Networks. Liu et al. [Liu+16] further improve in accuracy and efficiency by incorporating feature maps from different scales and considering a fixed set of bounding boxes.

2.1.2 3D Object Detection from 3D Point Clouds

In contrast to cameras, laser range sensors directly provide accurate 3D information, which simplifies the extraction of object candidates and can be helpful for the classification task as it provides 3D shape information. Li, Zhang, and Xia [LZX16] exploit a fully convolutional neural network for detecting vehicles from range data. They use a 2D representation of the 3D range data analogous to cylindrical images with the channels encoding the 3D location of the points. Given this representation, they simultaneously predict an objectness confidence and bounding box using a single 2D CNN. In contrast, Wang and Posner [WP15] propose an efficient scheme to apply the common 2D sliding window detection approach to 3D data. More specifically, they discretize the space into a 3D voxel grid and exploit the sparse nature of the problem with a voting scheme on top of a linear classifier, which is shown to be equivalent to convolutions on the full 3D point cloud. Engelcke et al. [Eng+17] extend this feature-centric voting scheme by implementing a novel convolutional layer to apply sparse convolutions across the 3D point cloud. Additionally, they encourage sparsity in the intermediate representation using ReLU non-linearities and L_1 penalty. While [WP15; Eng+17] extract hand-crafted features from the voxels, VoxelNet from Zhou and Tuzel [ZT18] learns the features in an end-to-end trainable deep network. They propose a voxel feature encoding layer that learns a unified feature representation for the points of the voxels. Eventually, a region proposal network generates detections from these feature representations. In chapter 4, we use an architecure based on VoxelNet from Zhou and Tuzel [ZT18] for estimating 3D bounding boxes from LIDAR laser scans.

Relying only on laser range data makes the detection task challenging due to the limited density of the laser scans and lack of appearance information. Thus, existing LiDAR-based approaches perform weaker compared to their image-based counterparts on the 2D detection problem of KITTI. However, recently, it has been shown that the fusion of LiDAR and camera information allows reducing the gap and eventually even outperforming state-of-the-art 2D detectors [Che+17b; Ku+18; CVN17; Qi+18; Du+18].

2.2 Semantic Segmentation

Traditionally, the problem was posed as maximum-a-posteriori (MAP) inference in a conditional random field (CRF), defined over pixels [HZC04; VT07; Sho+09] or superpixels [HZR06; KLT09]. Hierarchical [HZC04; KH05; Lad+09; Lad+14] and long-range connectivity as well as higher-order potentials defined on image regions [HZR06; KLT09] have been exploited to compensate for limitations of CRFs with local connections and to model long-range interactions within the image. Krähenbühl and Koltun [KK11] propose a tractable inference algorithm for fully connected CRF models which model pairwise potentials between all pairs of pixels in the image. While previous methods using fully connected CRFs [Rab+07; GRB08; KLT09] could only be applied to smaller image regions due to the computational and memory complexity of these algorithms, [KK11] allows deploying fully connected CRF models at pixel-level.

An alternative to inference in graphical models for the task of semantic segmentation is presented by Munoz, Bagnell, and Hebert [MBH10]. They train a sequence of inference models in a hierarchical procedure that captures context over larger image regions. This allows them to bypass the difficulties of training structured prediction models when exact inference is intractable and yields a very efficient and accurate scene labeling algorithm.

While most previous approaches rely on very simple features such as color, edge and texture information, Shotton et al. [Sho+09] observed that more powerful features have the potential to significantly boost performance. They propose an approach based on a novel feature type called texture-layout filter that exploits the textural appearance of objects, its layout as well as textural context. They combine texture-layout filters with lower-level image features in a CRF to obtain pixel-level segmentations.

2.2.1 Deep Learning for Semantic Segmentation

The success of deep convolutional neural networks for image classification and object detection has sparked interest in leveraging their potential for solving pixel-level tasks, in particular semantic segmentation. The fully convolutional neural network [LSD15] illustrated in Fig. 2.2 is one of the earliest works which applies CNNs to the image segmentation problem. Modern convolutional neural networks for image classification combine multi-scale contextual information by consecutive pooling and sub-sampling layers that lower the resolution. However, semantic segmentation requires multi-scale contextual reasoning together with full-resolution predictions, i.e. dense predictions.

Several methods [Che+15b; YK16; GF16; BKC17] have therefore been proposed to tackle the opposing needs of multi-scale inference and full-resolution outputs. Dilated convolutions [Che+15b; YK16] enlarge the receptive field of neural networks without loss of resolution. The dilated convolution operation corresponds to a regular convolution that skips pixels while applying the filter. This allows for efficient multi-scale reasoning without

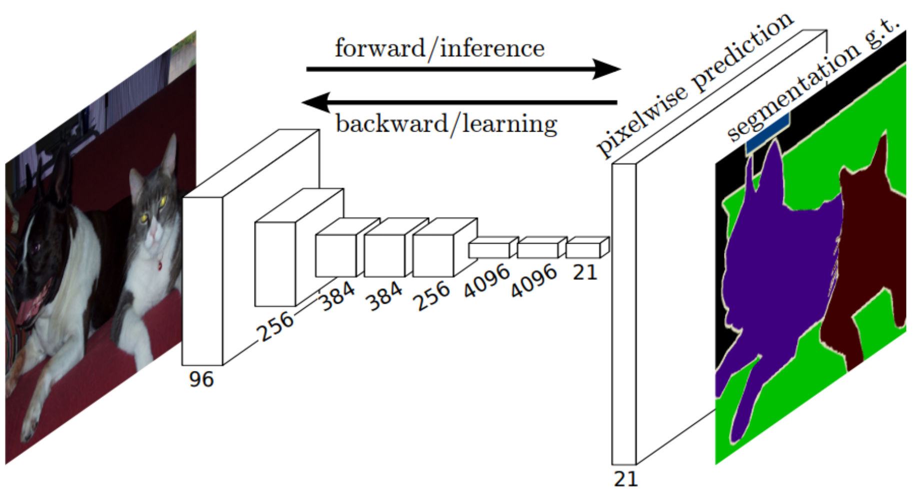


*Figure 2.2: **Convolutional Neural Network.** Fully convolutional neural network for semantic segmentation proposed by Long, Shelhamer, and Darrell [LSD15]. Reproduced from Long, Shelhamer, and Darrell [LSD15].*

increasing the number of model parameters. Chen et al. [Che+18b] extend this idea by using multiple dilated convolutions with different sampling rates in parallel.

In contrast, Badrinarayanan, Kendall, and Cipolla [BKC17] propose an encoder-decoder network with skip connections. Each decoder layer maps a low resolution feature map of an encoder (max-pooling) layer to a higher resolution feature map. In particular, the decoder in their model takes advantage of the pooling indices computed in the max-pooling (i.e. downsampling) step of the corresponding encoder to implement the upsampling process. This eliminates the need to learn the upsampling and thus results in a smaller number of parameters. Furthermore, sharper segmentation boundaries can be obtained using this approach. In chapter 3, we use and encoder-decoder network similar to Badrinarayanan, Kendall, and Cipolla [BKC17] to estimate the 3D object coordinates on the surface of every car in the scene.

While activation maps at lower-levels of the CNN hierarchy lack information specific to object categories, they provide information of higher spatial resolution. Ghiasi and Fowlkes [GF16] leverage this assumption and propose to construct a Laplacian pyramid based on a fully convolutional network. Aggregating information at multiple scales allows them to successively refine the boundary reconstructed from lower-resolution layers. They achieve this by using skip connections from higher resolution feature maps and multiplicative confidence gating, penalizing noisy high-resolution outputs in regions where low-resolution predictions have high confidence.

A different way to address the needs of multi-scale inference and full resolution prediction is the combination of CNNs with CRF models. Chen et al. [Che+15b; Che+18b] propose to refine the label map obtained using a convolutional neural network using a fully connected CRF model [KK11]. The CRF allows them to capture fine details based on the raw RGB input which are missing in the CNN output due to the limited spatial accuracy of the CNN model.

Simonyan and Zisserman [SZ15] and Szegedy et al. [Sze+15] have shown that the depth of a CNN is crucial to represent rich features. However, increasing the depth of a network leads to an increase in complexity as well as to saturation and degradation in accuracy. He et al. [He+16] proposed a deep residual learning framework (ResNet) to address this problem. In deep residual networks, each stacked layer learns a residual mapping instead of the original mapping. This facilitates the backpropagation of gradients and thus training and results in higher accuracy in comparison to regular deep networks. Wu, Shen, and Hengel [WSH19] propose a more efficient ResNet architecture by analyzing the effective depth of residual units. They point out that ResNets behave as linear ensembles of shallow networks. Based on this understanding, they design a group of relatively shallow convolutional networks for the task of semantic image segmentation, which performs better. To better incorporate global context information into the pixel-level prediction task, Zhao et al. [Zha+17] propose a pyramid scene parsing network (PSPNet). They apply a pyramid parsing module to the last convolutional layer of a CNN which fuses features of several pyramid scales to combine local and global context information. The resulting representation is fed into a convolution layer to obtain the final per-pixel predictions. Inspired by this work, [Che+17a] revisited the Atrous Spatial Pyramid Pooling (ASPP) [Che+18b] by experimenting with cascading and parallel application of dilated convolutions. This allows them to improve upon their previous work [Che+18b] while achieving comparable results to PSPNet [Zha+17]. In chapter 5, we use a ResNet and Feature Pyramid Network (FPN) backbone [He+16; Lin+17a], with a fully-convolutional segmentation head [LSD15] to generate the semantic segmentation masks for images collected from cameras on the self driving vehicle.

2.2.2 Instance Semantic Segmentation

There exist two major lines of research for the task of semantic instance segmentation: Proposal-based and proposal-free instance segmentation. While proposal-based approaches usually consist of two steps, i.e. proposal extraction and proposal classification, proposal-free methods predict pixel labels directly from the image.

Proposal-based Approaches Proposal-based instance segmentation methods extract class-agnostic proposals which are classified as an instance of a semantic class in order to obtain pixel-level instances. There exist several region proposal methods like Constrained Parametric Min-Cut (CMPC) [CS12], Multiscale Combinatorial Grouping (MCG) [Arb+14], DeepMask [PCD15], and SharpMask [Pin+16] returning generic class-agnostic region proposals which can be directly used as instance segments. Several object detection classifiers were proposed which simultaneously address object detection and semantic segmentation by leveraging region features from instance segments to improve the detection accuracy, i.e. O^2P [Car+12], Simultaneous Detection, and Segmentation (SDS) [Har+14], Convolutional Feature Masking (CFM) [DHS15], HyperColumn [Har+15].

Proposal-based algorithms are slow at inference time due to the computationally expensive proposal generation step. To avoid this bottleneck, Dai, He, and Sun [DHS16] propose

Multi-task Network Cascade (MNC) a fully convolutional network with three stages. They extract box proposals, use shared features to refine these to segments, and finally classify them into semantic categories. The causal relations between the outputs of the stages complicate training of the multi-task cascade. In order to overcome these difficulties, a fully differentiable mask prediction layer is presented to train the whole model in an end-to-end fashion. Box proposals can also induce errors into the proposal-based instance segmentation method due to wrongly scaled or shifted bounding boxes. In order to tackle this problem, Hayder, He, and Salzmann [HHS17] present a shape aware object mask network that predicts a binary mask for each bounding box proposal, potentially extending beyond the box itself. They integrate the object mask network into the Multi-task Network Cascade framework of Dai, He, and Sun [DHS16] by replacing the original mask prediction stage. In chapter 3, we use the Multi-task Network Cascade (MNC) from Dai, He, and Sun [DHS16] to predict the 2D bounding boxes and segmentation masks for all cars present in the scene.

While earlier methods address the detection and segmentation problem with two sub-networks sequentially, recent work [Li+17; He+17; Che+18a] propose to jointly address these problems. All joint formulations use ResNet-like architectures [He+16] for feature extraction. Li et al. [Li+17] propose FCIS, the first fully convolutional neural network for end-to-end instance semantic segmentation. They extend the fully convolutional mask proposal network [Dai+16] by sharing the convolutional representation of the proposals with a detection and segmentation sub-network. In contrast to FCIS, Mask R-CNN [He+17] and MaskLab [Che+18a] both build on Faster R-CNN [Ren+15]. He et al. [He+17] extend Faster R-CNN [Ren+15] by an additional branch for predicting segmentation masks. Chen et al. [Che+18a] combine box predictions from Faster R-CNN with semantic segmentation logits for pixel-wise classification and direction prediction logits estimating the direction towards instance centers. The direction towards instance centers allows them eventually to separate instances from the same class.

Proposal-free Approaches Due to the problem of proposal-based approaches to inherit errors of the proposal generation, a number of alternative methods have been proposed recently. These methods jointly infer the segmentation and the semantic category of individual instances by casting instance segmentation directly as a pixel labeling task.

Several approaches [Zha+15; ZFU16; Uhr+16] show how depth information can be used to identify different object instances. Zhang et al. [Zha+15] and Zhang, Fidler, and Urtasun [ZFU16] train a fully convolutional neural network (FCN) to directly predict pixel-level instance segmentations of densely sampled image patches while the instance ID encodes a depth ordering. They improve the predictions and enforce consistency with a subsequent Markov Random Field. Uhrig et al. [Uhr+16] propose an FCN to jointly predict semantic segmentation as well as depth and an instance-based direction relative to the centroid of each instance. This relative direction cue is then used for clustering pixels into individual instances. However, all [Zha+15; ZFU16; Uhr+16] require ground-truth depth data for training their model.

Instead of relying on depth information, concurrent work [Kir+17; BU17; AT17] present proposal-free approaches based on an initial semantic segmentation. Kirillov et al. [Kir+17]

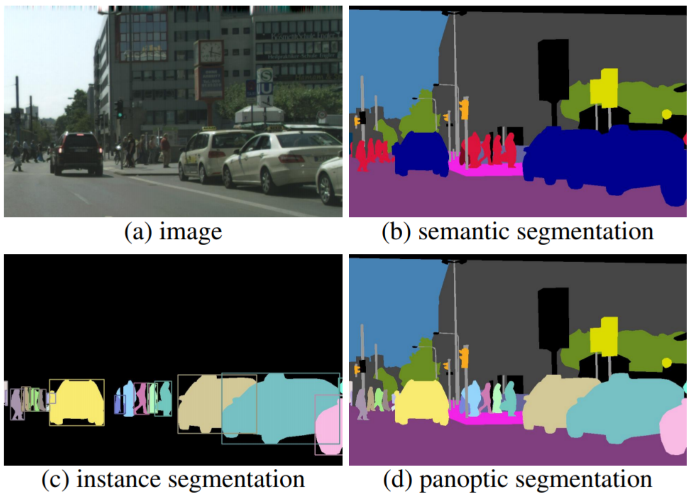

(a) image
(b) semantic segmentation
(c) instance segmentation
(d) panoptic segmentation

Figure 2.3: Panoptic Segmentation. *Difference between semantic (b), instance (c) and panoptic segmentation (d). Reproduced from [Kir+19b]*

combine semantic segmentation and object boundary detection via global reasoning in a multi-cut formulation to infer semantic instance segmentation. Bai and Urtasun [BU17] combine ideas from classical watershed transform with deep learning to create an energy map from an initial semantic segmentation and the input image where the basins correspond to object instances. This allows them to cut at a single energy level for obtaining a pixel-level instance segmentation. Arnab and Torr [AT17] propose to refine an initial semantic segmentation using an instance subnetwork. The initial category-level segmentation is used along cues from the output of an object detector within an unrolled Conditional Random Field [Zhe+15] to predict pixel-level instances.

2.2.3 Panoptic Segmentation

Instance segmentation focuses on instances of objects and usually ignores classes that are not amendable to this task like sky or road as illustrated in Fig. 2.3. In contrast, panoptic segmentation, first introduced by Kirillov et al. [Kir+19b], addresses the dense estimation of a semantic label and instance id. Several approaches [Kir+19b; LAT18; CPN18; Xio+19; Kir+19a] have been proposed to address the problem.

Proposal-free instance segmentation approaches like [AT17; BU17; Kir+17] can be used directly to learn panoptic segmentation. However, ground truth for training them on this problem is very limited. Thus, Li, Arnab, and Torr [LAT18] use [AT17] in a semi-supervised fashion to learn panoptic segmentation. They use interactive foreground extraction (GrapCut) [RKB04], proposal segmentation [Pon+17] and gradient-based localization of classes [Sel+17] to train the network in a semi-supervised fashion.

In contrast, several approaches [CPN18; Xio+19; Kir+19a] address panoptic segmentation with a joint semantic and instance segmentation formulation based on Mask R-CNN

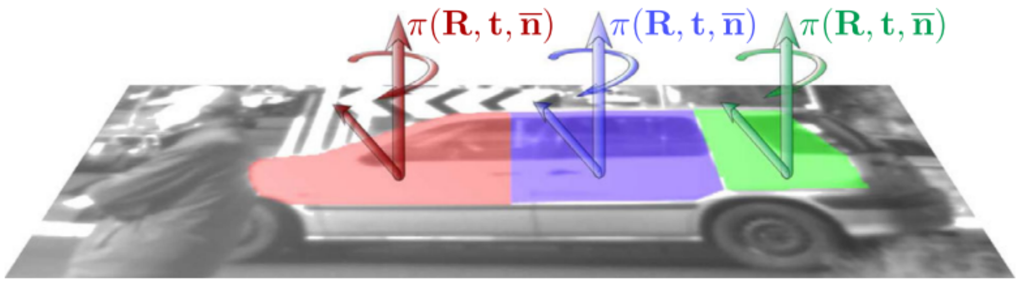

$\pi(\mathbf{R},\mathbf{t},\overline{\mathbf{n}})$
$\pi(\mathbf{R},\mathbf{t},\overline{\mathbf{n}})$
$\pi(\mathbf{R},\mathbf{t},\overline{\mathbf{n}})$

Figure 2.4: Piecewise Rigidity. *In Vogel, Schindler, and Roth [VSR15] the scene is modeled as a collection of rigidly moving planar segments. Reproduced from [VSR15].*

[He+17]. Costea, Petrovai, and Nedevschi [CPN18] propose to fuse object detections, semantic, and instance segmentation. They use semantic segmentation to distinguish between fore- and background regions. While the semantic class of background regions is directly obtained from the semantic segmentation, they use object detection, instance, and semantic segmentation to determine the class of foreground regions. In contrast, concurrent work [Xio+19; Kir+19a] extends Mask R-CNN by an additional semantic segmentation branch removing the requirement of a heuristic fusion. Xiong et al. [Xio+19] combine a semantic segmentation network based on deformable convolutions with Mask R-CNN. They predict dense class labels by applying a softmax on concatenated channels of the semantic and instance segmentation networks. In contrast, Kirillov et al. [Kir+19a] train the semantic and instance segmentation networks simultaneously without concatenating the channels. They apply non-maximum suppression [Kir+19b] to avoid overlapping instances.

2.3 Scene Flow Estimation from Stereo Images

Following the work of Vedula et al. [Ved+99; Ved+05] several approaches formulate 3D scene flow estimation as a variational optimization problem where optimization proceeds in a coarse-to-fine manner, and local regularizers are leveraged to encourage spatial smoothness of depth and motion [BMK13; HD07; PKF07; Val+10; Wed+11; VSR11]. Wedel et al. [Wed+08; Wed+11] propose a variational framework by decoupling the motion estimation from the disparity estimation while maintaining stereo constraints. Starting from a precomputed disparity map at each time step, optical flow for the reference frame and disparity for the other view are estimated. The motivation for this decoupling is mainly computational efficiency by choosing the optimal technique for each task. In addition, Wedel et al. [Wed+11] propose a solution for varying lighting conditions based on residual images and provide an uncertainty measure which they showed to be useful for object segmentation. Rabe et al. [Rab+10] integrate a Kalman filter to the decoupling approach for temporal smoothness and robustness.

Unfortunately, coarse-to-fine variational optimization suffers when displacements are large. Similar to stereo and optical flow, prior assumptions about the geometry and motion

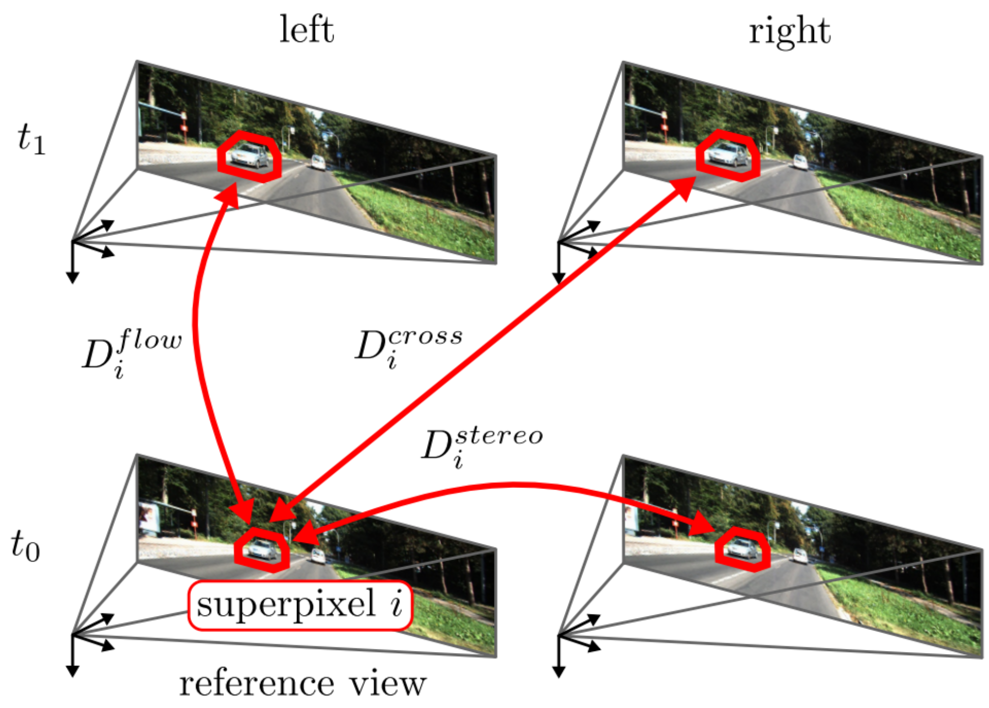


Figure 2.5: Object Scene Flow. *Menze and Geiger [MG15] propose decomposition of the scene into a small number of independently moving objects and the background. By conditioning on a superpixelization, they jointly estimate this decomposition as well as the rigid motion of the objects. Reproduced from [MG15].*

can be exploited to better handle the challenges of the scene flow problem. Thus, slanted-plane models [YMU13; YMU14] have recently been introduced [VSR13; VRS14; MG15; MHG15b; Lv+16] which gain their robustness by decomposing the scene into a collection of rigidly moving planes and exploiting discrete-continuous optimization techniques. Vogel, Schindler, and Roth [VSR15] and Lv et al. [Lv+16] represent the dynamic scene as a collection of rigidly moving planar regions, as shown in Fig. 2.4. Vogel, Schindler, and Roth [VSR15] jointly recover this segmentation while inferring the shape and motion parameters of each region. They use a discrete optimization framework and incorporate occlusion reasoning as well as other scene priors in the form of spatial regularization of geometry, motion, and segmentation. In addition, they reason over multiple frames by constraining the segmentation to remain stable over a temporal window. Their experiments show that their view-consistent multi-frame approach significantly improves accuracy for challenging scenarios. Using the same representation, Lv et al. [Lv+16] focus on an efficient solution to the problem. They assume a fixed superpixel segmentation and perform optimization in the continuous domain for faster inference. Starting from an initialization based on Deep Matching [Wei+13], they independently refine the geometry and motion of the scene, and finally perform a global non-linear refinement using the Levenberg-Marquardt algorithm.

As visualized in Fig. 2.5 Menze and Geiger [MG15] also follow a slanted plane approach, but in addition to previous methods [VSR15; Lv+16], they model the decomposition of the scene into a small number of independently moving objects and the background. By conditioning on a superpixelization, they jointly estimate this decomposition as well as the rigid motion of the objects and the plane parameters of each superpixel in a discrete-continuous Conditional Random Field (CRF). Compared to [VSR15; Lv+16], they leverage a more compact representation, by implicitly regularizing over larger distances. They also present a new scene flow dataset by annotating dynamic scenes from the KITTI raw data collection using detailed 3D CAD models. Menze, Heipke, and Geiger [MHG15b] propose an extension of this model where the pose and 3D shape of the objects are inferred in addition to the rigid motion and segmentation. In particular, they incorporate a deformable 3D active shape model of vehicles into the scene flow approach.

With the advent of challenging real-world benchmarks, such as the KITTI 2012 [GLU12] and KITTI 2015 [MG15], great progress has been made in this area. While these methods have demonstrated impressive performance on the challenging KITTI benchmark [GLU12; MG15], they fail to establish correspondences in textureless, reflective or fast-moving regions due to violations and ambiguities of the data term and weak prior assumptions. In chapter 3, we propose to use recognition cues to improve the performance of scene flow methods from stereo images in the presence of such challenges.

Deep Learning for Motion Estimation While several learning-based approaches for stereo [Ken+17; ŽL16; Lia+17] and optical flow [Sun+18; Dos+15; Ilg+17] have been proposed in literature, there is little prior work on learning scene flow estimation. A notable exception is SceneFlowNet [May+16], which concatenates features from FlowNet [Dos+15] and DispNet [May+16] for *image-based* scene flow estimation. In contrast, in chapter 4 we propose a novel end-to-end trainable approach for scene flow estimation from unstructured *3D point clouds*.

2.3.1 Scene Flow from RGB-D Sequences

When per-pixel depth information is available, two consecutive RGB-D frames are sufficient for estimating 3D scene flow. Initially, the image-based variational scene flow approach was extended to RGB-D inputs [Wed+08; HRF13; Qui+14]. Franke et al. [Fra+05] instead proposed to track KLT feature correspondences using a set of Kalman filters. Exploiting PatchMatch optimization on spherical 3D patches, Hornacek et al. [HFR14] recover a dense field of 3D rigid body motions. However, while structured light scanning techniques (e.g., Kinect) are able to capture indoor environments, dense RGB-D sequences are hard to acquire in outdoor scenarios like ours. Furthermore, structured light sensors suffer from the same depth error characteristics as stereo techniques.

2.3.2 Scene Flow from 3D Point Clouds

In the robotics community, motion estimation from 3D point clouds has so far been addressed primarily with classical techniques. Several works [DON11; Ush+17; Tan+14] extend occupancy maps to dynamic scenes by representing moving objects via particles which are updated using particle filters [DON11; Tan+14] or EM [Ush+17]. Others tackle the problem as 3D detection and tracking using mean shift [Asv+16], RANSAC [Dew+16a], ICP [MS13], CRFs [VRT10] or Bayesian networks [Hel+16]. In contrast, Dewan et al. [Dew+16b] propose a 3D scene flow approach where local SHOT descriptors [TSS10] are associated via a CRF that incorporates local smoothness and rigidity assumptions. While impressive results have been achieved, all the aforementioned approaches require significant engineering and manual model specification. In addition, local shape representations such as SHOT [TSS10] often fail in the presence of noisy or ambiguous inputs. In contrast, in chapter 4 we address the scene flow problem using a generic end-to-end trainable model which is able to learn local and global statistical relationships directly from data. Accordingly, our experiments show that our model compares favorably to the aforementioned classical approaches.

2.4 Semantic Priors as Geometric Constraints

Several works have considered semantic information for stereo or optical flow estimation. Güney et al. [GG15] presented a model for stereo estimation where 3D object hypotheses from a CAD model database are jointly estimated with the disparity map. Hur et al. [HR16] proposed a model for joint optical flow and semantic segmentation. In particular, they classify the scene into static and dynamic regions and introduce a constraint which measures how well the homography of a superpixel meets the epipolar constraint. Sevilla-Lara et al. [Sev+16] used semantic segmentation to identify object instances and combine per-object layered flow predictions [Sun+13] with DiscreteFlow [MHG15a]. Bai et al. [Bai+16] proposed a model which first identifies car instances and then predicts each rigidly moving component in the scene with a separate epipolar flow constraint.

While existing methods leverage recognition to aid either reconstruction or motion estimation, in chapter 3 we consider recognition for the 3D scene flow problem, i.e., the combination of the two. In contrast to flow-only methods [Bai+16] that need to search for correspondences along epipolar lines, our method exploits the semantic cues as well as the geometry which allows us to estimate the rigid motion between frames more robustly and leads to significantly improved results compared to all baselines. Furthermore, we are (to the best of our knowledge) the first to investigate the impact of recognition granularity on the task, ranging from coarse 2D boxes to fine grained 3D object coordinates predictions.

Continuous 3D object parts, also known as 3D object coordinates, have so far mainly been leveraged in the context of 3D pose estimation [Tay+12; Bra+14; Bra+16; Mic+15], camera re-localization [Val+15] and model-based tracking [Kru+14]. The typical approach is to train a random forest for predicting instance-specific object probabilities and 3D object coordinates with respect to a fixed local coordinate system. These predictions are used to fit a 3D model of the known object, resulting in the 3D object pose. In chapter 3, we also explore the possibility of using object coordinates as a continuous labeling for the surface points of the object.

2.5 End-to-End Learning for Autonomous Driving

Behavior cloning approaches learn to map sensor observations to desired driving behavior through supervised learning. Behavior cloning for driving has historical roots [Pom88] as well as recent successes [Boj+16; Zen+19; Pra+20; Ohn+20]. Bojarski et al. [Boj+16] propose an end-to-end CNN for lane following that maps images from the front facing camera of a car to steering angles, given expert data. Xu et al. [Xu+17] propose an alternative approach and exploit large scale online datasets from uncalibrated sources to learn a driving model. Specifically, they formulate autonomous driving as a future ego-motion prediction problem. They claim that predicting ego-motion instead of vehicle control allows their approach to generalize better to new platforms. Their deep learning architecture combines FCNs and LSTMs, and learns to predict the motion path given the current state of the agent.

A challenge with behavior cloning approaches is that the training data is collected using an off-policy expert teacher, i.e. the training data is collected by rolling out the expert policy, which is different from the policy being learned. As collecting expert demonstrations for all possible situations is not practical, the training trajectories do not cover all possible states. At test time, the rollout of the behavior cloning policy thus causes it to move to a different distribution of states compared to the one it was trained on. Due to this covariate shift between the training and test time trajectories, the behavior cloning agent's errors compound when drifting away from the expert demonstrations. In other words, the vehicle is likely to encounter new situations it has not been trained for and therefore acts wrongly. In contrast, in on-policy rollout, training data is collected using the current policy being learned. Ross and Bagnell [RB] propose DAgger to alleviate covariate shift by iteratively collecting corrective expert actions for the states visited by rolling out the currently learned driving policy. The driving policy parameters are then trained using the data collected on-policy. However, doing on-policy rollouts with an imperfect policy has the disadvantage of drifting and potentially reaching dangerous states, thus requiring a simulator for safe training. Laskey et al. [Las+17] claim to provide a safer way of generating training data using expert policy with small amounts of noise injected to approximate the errors of on-policy rollout. They achieve this by iterating between learning a noise model that minimizes the covariate shift and generating data for training the behavior cloning agent.

Besides the drifting problem during test time, behavior cloning-based driving systems have other limitations. Sensor input alone is often not sufficient to uniquely infer control. Consider intersections, for example, where multiple possible actions are valid (left, right, straight). Without conditioning on the goal, all three options are acceptable. Thus, some of the behavior cloning agents, such as the one by Bojarski et al. [Boj+16], require human intervention for lane changes or turns. Conditional Imitation Learning (CIL) extends this framework by incorporating high-level navigational commands into the decision making process [Cod+18], as illustrated in Fig. 2.6. The high-level navigational input represents the driver's intention, such as the direction to take at the next intersection, which cannot be recovered from sensory input alone. Chen et al. [Che+19] show that imitation learning can be simplified by decomposing it into two stages. They first train an agent that has access to privileged information. This privileged agent cheats by observing the ground-truth layout of the environment and the positions of all traffic participants. In the second stage, the privileged agent acts as a teacher that trains a purely vision-based sensorimotor agent. The resulting sensorimotor agent does not have access to any privileged information and does not cheat. They demonstrate that this approach substantially outperforms the state of the art on the CARLA benchmark and the recent NoCrash benchmark, attaining the best performance to date.

Approaches based on reinforcement learning (RL) learn to drive by training an agent that tries to maximize a user defined reward which the agent receives while interacting with the environment. In the autonomous driving application, the reward is defined by specifying the driving agent's preferences and goals. Dosovitskiy et al. [Dos+17] propose a reinforcement learning method that trains a deep network based on a reward function provided by the CARLA simulator which combines speed, distance traveled towards the goal, collision damage, overlap with sidewalk and overlap with the opposite lane. For training the agent, they use the asynchronous advantage actor-critic (A3C) algorithm [KT99] which uses the value function learned by the critic to update the actor's policy. Dosovitskiy et al. [Dos+17] observe that the RL agent performs significantly worse compared to a behavior cloning agent trained using conditional imitation learning [Dos+17] despite the fact that the RL agent was trained on a significantly larger set of visual observations. Recently, Kendall et al. [Ken+18] showed first promise in learning to drive in the real-world using a reinforcement learning agent. They use the deep deterministic policy gradients algorithm for training the RL agent and define the reward as the distance traveled by the vehicle without the safety driver taking control.

The aforementioned methods are trained using model-free reinforcement learning. The disadvantage of model-free methods is that they are often data inefficient and require a large number of interactions with the environment. In contrast, model-based reinforcement learning approaches learn a model of the environment dynamics from observational data and then exploit this model for training a driving policy. Model-based methods have been shown to significantly reduce the number of environment interactions required to learn an effective policy. However, model-based methods also typically require an interactive environment as a dynamics model trained on a fixed set of demonstrations may make incorrect predictions outside the training domain. The interactive training environment is however not practical in the real-world where such interactions are expensive and dangerous. To alleviate this problem, Henaff, Canziani, and LeCun [HCL19] propose to train a model-based policy which is encouraged to produce actions which the forward dynamics model is confident about. They achieve this by training the policy network to minimize an uncertainty cost which represents the mismatch between the states it induces and the states in the trained data. However, RL approaches are inefficient to train and require a simulator or non-practical trial and error runs in a real environment as well as careful design of the reward function. Therefore, several methods have been proposed to combine the strengths of both approaches. Liang et al. [Lia+18] propose an approach to alleviate the low exploration efficiency of RL for large action space. They achieve this by constraining the policy search space by initializing the weights of the policy network of an RL algorithm by a network trained to clone the expert behavior. They observe significant improvements on the CARLA benchmark over agents trained using RL from scratch. Li et al. [Li+18] propose an approach that learns to clone only the best behaviors of several sub-optimal teachers. They estimate the best

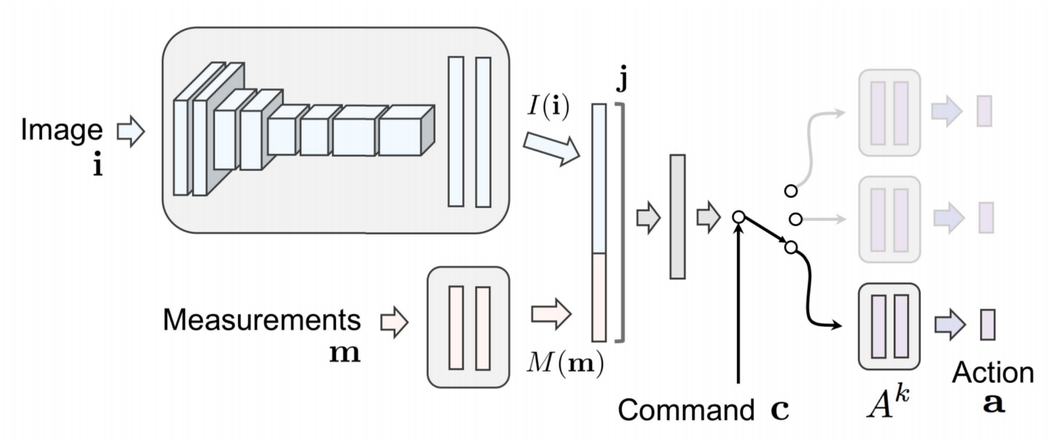


Figure 2.6: Goal-conditional Behavior Cloning. *Architecture of goal-conditional end-to-end behavior cloning for autonomous driving proposed by Codevilla et al. [Cod+18]. The goal command acts as a switch that selects between specialized sub-policies that correspond to different commands such as lane following, turning left or turning right. Reproduced from [Cod+18]*

teacher by estimating the value function of each sub-optimal teacher. The sub-optimal teachers are defined using several simple controllers over the planner output. Therefore, they do not require expert teachers for labeling data and allow for better exploration compared to learning from a single expert teacher. In addition, learning from multiple sub-optimal teachers leads to faster training compared to pure RL agents as exploration only happens from feasible states. The requirement to specify the reward function limits the practical use of Reinforcement Learning. An accurate specification of the reward requires tedious and computationally inefficient hyper-parameter tuning. Sharifzadeh et al. [Sha+16] propose to learn the unknown reward function of the driving behavior from expert demonstrations by applying Inverse Reinforcement Learning (IRL). In contrast to behavior cloning approaches that directly learn the observation-control mapping in a supervised fashion, Inverse Reinforcement Learning approaches claim to offer better generalization by learning a reward function that explains the expert behavior.

Codevilla et al. [Cod+19] present an analysis of several limitations of End-to-End Learning for Autonomous Driving. In particular, they observe that driving performance drops significantly in new environments and weather conditions. They also observe drastic variance in performance caused by model initialization and data sampling during training. In chapter 5 we address these issues with semantic input representations while maintaining low labeling costs.

2.6 Semantic Priors for Improving Generalization

Recent papers have shown the effectiveness of using mid-level visual priors to improve the generalization of visuomotor policies [Wan+19b; ZKK19; Sax+19; Mou+19; Mül+18].

Object detection, semantic segmentation and instance segmentation have been shown to help significantly for generalization in navigation tasks [Wan+19b; ZKK19]. Müller et al. [Mül+18] train a policy in the CARLA simulator with a binary road segmentation as the perception input, demonstrating that learning a policy independent of the perception and low-level control eases the transfer of learned lane-keeping behavior for empty roads from simulation to a real toy car. Similarly, Wang et al. [Wan+19a] infer the depth and poses of the objects present in the scene from front-facing camera images and project the objects into an overhead view. They train a behavior cloning agent over the concatenation of front-facing and overhead images and observe improved performance over an agent trained only on front-facing images. More recently, Zhao et al. [Zha+19] studied the significance of using intermediate representations pursued in computer vision research such as depth, segmentation, optical flow for improving several sensorimotor tasks such as urban driving. They observed that an agent that takes as input one or more of these intermediate representations along with the image learns significantly better sensorimotor control than an agent which uses just the raw image as input. They observed significant improvements even when the intermediate representations were noisy predictions by a simple deep network. Toromanoff et al. [TWM20] show how to effectively incorporate knowledge from segmentation labels into a reinforcement learning trained policy. These existing studies either compare different visual priors or focus on improving policies by choosing a specific visual prior, regardless of the annotation costs. Nonetheless, knowing these costs is extremely valuable from a practitioner's perspective. In chapter 5, we are interested in identifying and evaluating label efficient representations in terms of the performance and variance of the learned policies.

Compact Semantic Representations for Driving Instead of using a comparably higher-dimensional visual prior such as pixel-level segmentation, Chen et al. [Che+15a] present an approach which estimates a small number of human interpretable, pre-defined affordance measures such as the angle of the car relative to the road and the distance to other cars. These predicted affordances are then mapped to actions using a rule-based controller, enabling autonomous driving in the TORCS racing car simulator [Wym+15]. The advantage of such compact intrediate representations is that the failure cases are more interpretable compared to traditional behavior cloning approaches. Similarly, Sauer et al. [SSG18] estimate several affordances from sensor inputs to drive in the CARLA simulator. In contrast to [Che+15a], they consider the more challenging scenario of urban driving where the agent needs to avoid collision with obstacles on the road and navigate junctions with multiple possible driving directions. They achieve this by expanding the set of affordances to be more applicable to urban driving. Similar to the aforementioned methods, Mehta, Subramanian, and Subramanian [MSS18] also propose to use intermediate visual affordances such as "distance to intersection", and action primitives such as "slow down" as input to the driving policy network. However, in contrast to the aforementioned works, they predict visual affordances and action primitives as an auxiliary task to the driving control task. They claim that predicting representations which are crucial for the driving decision allow the policy network to learn superior internal representations leading to more efficient training and better generalization. Bansal, Krizhevsky, and Ogaler [BKO18] propose a perception module that translates raw sensor observations to a mid-level representation. Their representation includes a top-down rendering of the environment where 2D boxes of vehicles are drawn along with a rendering of the road information and traffic light states. They use this mid-level representation as input to a recurrent neural network (RNN) which outputs the control command. These methods are relevant to our study in chapter 5, in that they simplify perception through compact representations. However, these affordances are hand-engineered and very low-dimensional. Thus, failures in design will lead to errors that cannot be recovered from.

3 Exploiting Recognition for Robust 3D Motion Estimation

Existing methods for 3D scene flow estimation often fail in the presence of large displacement or local ambiguities, e.g., at texture-less or reflective surfaces. However, these challenges are omnipresent in dynamic road scenes, which is the focus of this work. Our main motivation in this chapter is to overcome these challenges in 3D motion estimation by exploiting recognition cues such as bounding box detections or instance segmentations. Furthermore, we investigate the importance of recognition granularity, from coarse 2D bounding box estimates over 2D instance segmentations to fine-grained 3D object part predictions. We compute these cues using CNNs trained on a newly annotated dataset of stereo images and integrate them into a CRF-based model for robust 3D scene flow estimation - an approach we term Instance Scene Flow.

To obtain the bounding box or segmentation masks, we utilize an existing state-of-the-art approach - the multi-task network cascade (MNC) from Dai et al. [DHS16]. In contrast to [DHS16], we achieve improved outputs, by providing a dense depth map as additional input to the network. To train the CNN, we annotated 400 stereo images from the KITTI 2015 benchmark of Menze et al. [MG15] using in-house annotators. For fine-grained geometric recognition we train a new convolutional neural network (CNN) on 2D instance segmentations of MNC which predicts the continuous 3D object parts, also known as 3D object coordinates [Bra+14; Bra+16; Val+15; Mic+15; Tay+12], for each pixel inside the instance mask. The 3D object coordinates specify the relative 3D location of an object's surface point with respect to the object's local coordinate frame. Thus, they provide a detailed geometric cue for matching pixels in neighboring frames, even in the presence of large displacements. Our CRF framework is based on [MG15] and termed Instance Scene Flow. We validate the benefits of recognition cues for scene flow on the challenging KITTI 2015 benchmark [MG15]. Firstly, we conduct a detailed ablation study for the three levels of recognition granularity, i.e. 2D bounding box, 2D segmentation mask, and continuous 3D object parts. From this study, we conclude that the instance segmentation cue is by far strongest, in our setting. Secondly, we show that our Instance Scene Flow method significantly boosts the scene flow accuracy, in particular in challenging foreground regions. Alongside, we demonstrate the effectiveness of our method on the challenging KITTI 2015 scene flow benchmark where we achieve state-of-the-art performance at the time of submission. We analyze the importance of each recognition cue in an ablation study and observe that the instance segmentation cue is by far strongest, in our setting. We demonstrate the effectiveness of our method on the challenging KITTI 2015 scene flow benchmark where we achieve state-of-the-art performance at the time of submission.

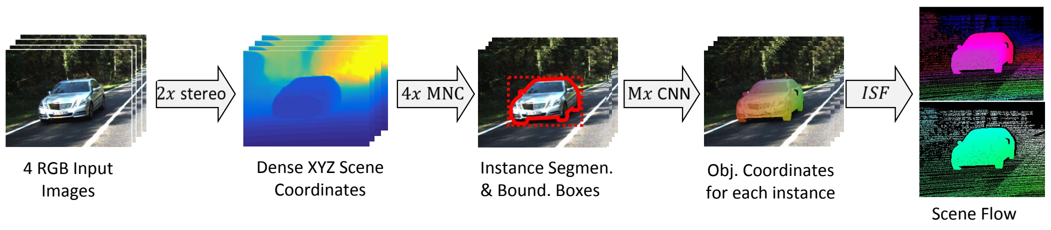

4 RGB Input Images
2x stereo
Dense XYZ Scene Coordinates
4x MNC
Instance Segmen. & Bound. Boxes
Mx CNN
Obj. Coordinates for each instance
ISF
Scene Flow

Figure 3.1: **Work flow for our approach**. *Note that each intermediate step uses as input all of the previous results. Given the four RGB input images (t/t+1, left/right) we compute 3D points (XYZ) for each pixel. For each of the four RGB,XYZ image-blocks we obtain instance segmentations, alongside bounding boxes. The M instances are processed individually to obtain object coordinates for each instance, using our object coordinates CNN. Finally, all this information is integrated into our Instance Scene Flow method (ISF) to produce the final output.*

3.1 Method

This section describes our approach to 3D scene flow estimation. The overall work flow of our approach is illustrated in Fig. 3.1. Given the 4 RGB images we extract 3D points (XYZ) for each pixel in camera coordinate system using a stereo method (see Section 3.2 for details). Based on the RGB and XYZ values, we train a multi network cascade (MNC) [DHS16] to predict 2D bounding boxes and 2D instance segmentations. We train a CNN to predict object coordinates for car instances. Finally, we integrate the bounding box, instance and object coordinates cues into a slanted-plane formulation and analyze the importance of each cue for the scene flow estimation task. The remainder of this section is structured as follows.

As our goal is to analyze the impact of different levels of recognition granularity, we first describe the inputs to our method in Section 3.1.1 and Section 3.1.2. In particular, we are interested in improving scene flow estimation of vehicles which are challenging due to their large motion and non-lambertian appearance. In Section 3.1.3, we finally show how these predictions can be integrated into a CRF model for 3D scene flow estimation.

3.1.1 2D Bounding Boxes and Instances

As discussed before, recognizing and segmenting objects is imperative for our approach. For this task, we use the state-of-the-art Multi-task Network Cascades (MNC) proposed by Dai et al. [DHS16] to obtain bounding boxes and segmentation masks of all cars present in the scene. Unlike the standard MNC framework which operates on RGB images, we provide the network with an RGB-XYZ image, where XYZ denotes the 3D scene coordinates, i.e., the 3D location of every pixel in the scene in camera coordinates. The 3D location for each pixel is computed from disparity maps, see Section 3.2 for details.

We pre-train the network using Pascal VOC and fine-tune it using 3200 coarse annotations of KITTI [GLU12] provided by [Che+14]. We obtained 200 images with 1902 pixel-accurate instance annotations from the KITTI 2015 Stereo benchmark [MG15] using in-house annotators. We further refined the model with these fine annotations. The final accuracy (IoU-50%) on the validation set of [Che+14] is 83% as compared to 78% when using only RGB images.

3.1.2 3D Object Coordinates

3D object coordinates specify the 3D location of the surface of an object with respect to a local coordinate frame. They can be viewed as a fine grained unique geometric labeling of the object's surface which is independent of the viewpoint and can be used to establish correspondences between frames, which we expect to be more robust to appearance changes than correspondences based on sparse feature matching.

While random forests have been used for estimating object coordinates when the target instance is known [Bra+14; Bra+16; Val+15; Kru+14], we found that a CNN-based approach leads to significantly better object coordinates predictions as also evidenced by our experiments in Section 3.2. We use a modified version of the encoder-decoder style CNN proposed in [TDB16] for estimating the object coordinates at each pixel. As above, the input to the CNN is an RGB-XYZ image as well as the instance prediction from the MNC. The output of the CNN is a 3 layer regression which stores the X, Y and Z coordinates for each point of the input.

An illustration of our architecture can be found in Fig. 3.2. We use an Encoder-Decoder style network similar to that of [TDB16]. The input to our network is the RGB image and the XYZ point cloud of a car instance that is predicted from the instance prediction network. The XYZ point cloud can be obtained using the estimated disparity and the camera calibration which is known. The network regresses the X, Y and Z values in object coordinate system for each pixel of the input. We consider this as fine grained unique geometric labeling of the object's surface points.

The encoder part of the network is a set of 5 convolutional layers with a stride of 2 at each of them (Conv0-Conv4), followed by 3 fully connected layers (FC1-FC3). Further, the FC3 layer is reshaped to a matrix again and follows through a set of deconvolutional layers (Dec4-Dec0). Each convolutional/deconvolutional layer is follwed by a nonlinear ReLU layer with negative slope 0.2 except for Dec0 after which a tanh layer is used for the output. While training the network, we present the RGB-XYZ features of an image as input and use the ground truth object coordinates for computing the loss. The network is trained by minimizing a robust and smooth Huber loss function [Gir15] using the Adam solver with a momentum of 0.9 and a learning rate of 1e-5.

We generate the ground truth object coordinates for the foreground cars using the following process. We obtain the car CAD model and the corresponding 6D pose annotations from Menze et al. [MG15]. Using the CAD model and the 6D pose, we render dense 3D object coordinates for each visible point on the surface of the car.

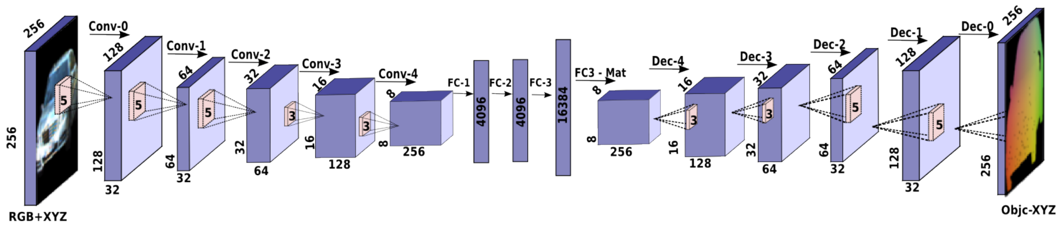


Figure 3.2: **Architecture of our Encoder Decoder Network for Object Coordinate Prediction.**

3.1.3 Scene Flow Model

We now describe our scene flow model which is based on [MG15] but adds two new components to it, in particular an instance sensitive appearance term and a term which encourages object coordinates to align across frames. For making the chapter self-contained, we specify the full model.

Notation: We first describe the notation of the inputs to the scene flow model which are pre-computed as described in the previous section and fixed during inference. Let $\mathcal{V} = \{0,1,2,3\}$ denote the set of views as illustrated in Fig. 3.3 and let $\mathbf{I}_v \in \mathbb{R}^{w \times h \times 3}$ denote the input image corresponding to each view. For each view $v \in \mathcal{V}$, we compute the following information: first, our MNC predicts instance label maps $\mathbf{M}_v \in \{0,\ldots,|M_v|\}^{w \times h}$ which determine the predicted semantic instance label for each pixel in each view. In our experiments, these maps are either coarse 2D bounding box segmentations or more accurate 2D instance segmentations which are both computed via MNC. Background pixels are assigned the label $\mathbf{M}_v(\mathbf{p}) = 0$ and foreground pixels are assigned positive labels. Note that instance labels do not correspond across frames as the correspondence is unknown a-priori and needs to be inferred jointly with the scene flow. Furthermore, we denote the 3D object coordinates predicted by the network with $\mathbf{C}_v \in [-1,1]^{w \times h \times 3}$.

We now describe the parameters of our model which are optimized during inference. Let $\mathcal{S}$ denote the set of superpixels in the reference view and $\mathcal{O}$ denote the set of objects in the scene. Each superpixel $i \in \mathcal{S}$ is associated with a region $\mathcal{R}_i$ in the reference image and a variable $\mathbf{s}_i = (\mathbf{n}_i, k_i)^\mathsf{T}$, where $\mathbf{n}_i \in \mathbb{R}^3$ describes a plane in 3D via $\mathbf{n}_i^\mathsf{T}\mathbf{x} = 1$. Further, let $k_i \in \{0,\ldots,|\mathcal{O}|\}$ index the object which the superpixel is associated with. Here $|\mathcal{O}|$ denotes an upper bound on the number of objects we expect to see, and $k = 0$ refers to the background object with $k > 0$ to other traffic participants. Each object $j \in \mathcal{O}$ is associated with a variable $\mathbf{o}_j \in SE(3)$ which describes its rigid body motion in 3D. Each superpixel associated with object j inherits its rigid motion parameters $\mathbf{o}_j \in SE(3)$. In combination with the plane parameters $\mathbf{n}_i$, this fully determines the 3D scene flow at each pixel inside the superpixel.

Energy Model: Given the left and right input images of two consecutive frames (Fig. 3.3), our goal is to infer the 3D geometry of each superpixel $\mathbf{n}_i$ in the reference view, the association to objects k_i and the rigid body motion of each object $\mathbf{o}_j$. We formulate the scene

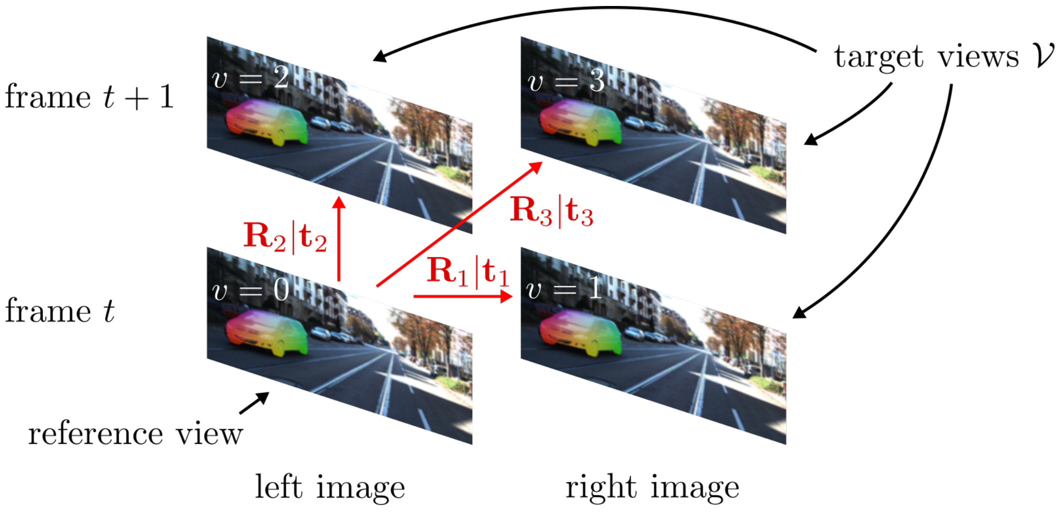

target views $\mathcal{V}$
frame $t+1$
$v = 2$
$v = 3$
$\mathbf{R}_2|\mathbf{t}_2$
$\mathbf{R}_3|\mathbf{t}_3$
$\mathbf{R}_1|\mathbf{t}_1$
frame t
$v = 0$
$v = 1$
reference view
left image
right image

Figure 3.3: **Geometric Relationship** *between reference and target views. Pixels in the reference view are mapped to a pixels in a target view via their depth and rigid body motion.*

flow estimation task as an energy minimization problem comprising data, smoothness and instance terms:

$$\hat{\mathbf{s}}, \hat{\mathbf{o}} = \underset{\mathbf{s},\mathbf{o}}{\operatorname{argmin}} \; \underbrace{\varphi(\mathbf{s},\mathbf{o})}_{\text{data}} + \underbrace{\psi(\mathbf{s})}_{\text{smooth.}} + \underbrace{\chi(\mathbf{s},\mathbf{o})}_{\text{instance}} \tag{3.1}$$

We summarize all variables involved in the optimization with $\mathbf{s} = \{\mathbf{s}_i | i \in \mathcal{S}\}$ and $\mathbf{o} = \{\mathbf{o}_i | i \in \mathcal{O}\}$. For clarity of exposition we omit all weight parameters of the model.

Data Term: Our data term encodes the assumption that corresponding points across all images should be similar in appearance:

$$\varphi(\mathbf{s},\mathbf{o}) = \sum_{i \in \mathcal{S}} \sum_{\mathbf{p} \in \mathcal{R}_i} \sum_{v \in \mathcal{V}} \varphi_v^{\mathrm{D}}(\mathbf{p},\mathbf{q}) \tag{3.2}$$

$$\mathbf{q} = \underbrace{\mathbf{K}\left(\mathbf{R}_v(\mathbf{o}_{k_i}) - \mathbf{t}_v(\mathbf{o}_{k_i})\,\mathbf{n}_i^{\mathsf{T}}\right)\mathbf{K}^{-1}}_{\text{homography (view 0} \rightarrow \text{view } v)} \mathbf{p} \tag{3.3}$$

Here, $\mathcal{V} = \{1,2,3\}$ denotes the set of target views and $\mathbf{q}$ is the location of pixel $\mathbf{p}$ in reference view 0 mapped into the target view $v \in \mathcal{V}$ according to the calibration matrix $\mathbf{K}$, the rigid body motion $\mathbf{R}_v|\mathbf{t}_v$ of the corresponding object $\mathbf{o}_{k_i}$ and the plane parameters of the associated superpixel $\mathbf{n}_i$. See Fig. 3.3 for an illustration.

The data cost $\varphi_v^{\mathrm{D}}(\mathbf{p},\mathbf{q})$ compares the appearance at pixel $\mathbf{p}$ in reference image 0 with the appearance at pixel $\mathbf{q}$ in the target view $v \in \mathcal{V}$. In our experiments, we use Census descriptors [ZW94] which are robust to simple photometric variations [MG15; VSR15; YMU14]. To guide the optimization process and overcome local minima we additionally add a robust ℓ_1 loss to φ_v^{D}. This loss measures the difference wrt. sparse DiscreteFlow correspondences [MHG15a] for the flow terms ($v = 2,3$) and depth estimates from SPS-stereo [YMU14] for the stereo term ($v = 1$).

Smoothness Term: The smoothness term encourages coherence of adjacent superpixels in terms of depth, orientation and motion. It decomposes as

$$\psi(\mathbf{s}) = \gamma_{ij}^{\mathrm{S}} \sum_{i\sim j} \psi_{ij}^{\mathrm{G}}(\mathbf{n}_i, \mathbf{n}_j) + \psi_{ij}^{\mathrm{M}}(\mathbf{s}_i, \mathbf{s}_j) \tag{3.4}$$

with the following geometry (G) and motion (M) terms:

$$\begin{aligned}
\psi_{ij}^{\mathrm{G}}(\mathbf{n}_i, \mathbf{n}_j) &= \sum_{\mathbf{p}\in\mathcal{B}_{ij}} \rho\left(d(\mathbf{n}_i,\mathbf{p}) - d(\mathbf{n}_j,\mathbf{p})\right) \\
&+ \rho\left(1 - |\mathbf{n}_i^{\mathrm{T}}\mathbf{n}_j|/(\|\mathbf{n}_i\|\|\mathbf{n}_j\|)\right), \\
\psi_{ij}^{\mathrm{M}}(\mathbf{s}_i, \mathbf{s}_j) &= \gamma_{ij}^{\mathrm{M}}(\mathbf{n}_i, \mathbf{n}_j)\,[k_i \neq k_j].
\end{aligned}$$

Here, $d(\mathbf{n},\mathbf{p})$ denotes the disparity of plane $\mathbf{n}$ at pixel $\mathbf{p}$ in the reference image, $\mathcal{B}_{ij}$ is the set of shared boundary pixels between superpixel i and superpixel j, and ρ is the robust ℓ_1 penalty. The instance-sensitive weight γ_{ij}^{S} is defined as

$$\gamma_{ij}^{\mathrm{S}} = 1 - \beta_{\mathrm{S}} \cdot [(i,j) \in \mathcal{M}] \tag{3.5}$$

where $\mathcal{M}$ denotes the set of adjacent superpixel pairs where exactly one of the superpixels lies on an object instance. $\beta \in [0,1]$ is a hyper-parameter which weighs down the costs of discontinuities for adjacent superpixels in $\mathcal{M}$. Note that this weighting is only possible in the presence of instance predictions.

The geometry-sensitive motion weight is defined as

$$\exp\left(-\frac{\lambda}{|\mathcal{B}_{ij}|}\sum_{\mathbf{p}\in\mathcal{B}_{ij}} \left(d(\mathbf{n}_i,\mathbf{p}) - d(\mathbf{n}_j,\mathbf{p})\right)^2\right) \\ \times |\mathbf{n}_i^{\mathrm{T}}\mathbf{n}_j|/(\|\mathbf{n}_i\|\|\mathbf{n}_j\|)$$

encouraging motion boundaries to align with 3D folds and discontinuities rather than within smooth surfaces.

Instance Term: The instance term $\chi(\mathbf{s},\mathbf{o})$ measures the compatibility of appearance and part-labeling induced by the 3D object coordinates when warping the detected instances into the next frame. It takes the following form

$$\chi(\mathbf{s},\mathbf{o}) = \sum_{i\in\mathcal{S}} \sum_{\mathbf{p}\in\mathcal{R}_i} \sum_{v\in\mathcal{V}} \chi_v^{\mathrm{I}}(\mathbf{p},\mathbf{q}) \tag{3.6}$$

with

$$\begin{aligned}
\chi_v^{\mathrm{I}}(\mathbf{p},\mathbf{q}) = \quad & [\mathbf{M}_0(\mathbf{p}) = 0 \vee \mathbf{M}_v(\mathbf{q}) = 0] \cdot \lambda \; + \\
& [\mathbf{M}_0(\mathbf{p}) > 0 \wedge \mathbf{M}_v(\mathbf{q}) > 0] \cdot \left(\chi_v^{\mathrm{A}}(\mathbf{p},\mathbf{q}) + \chi_v^{\mathrm{L}}(\mathbf{p},\mathbf{q})\right)
\end{aligned} \tag{3.7}$$

Here, $\mathbf{q}$ is calculated as in (3.3) and the appearance (A) potential and the part labeling (L)

Algorithm 1 Optimization

```
1:  Input: $\mathbf{I}_v$, $\mathbf{M}_v$, $\mathbf{C}_v$ for $v \in \mathcal{V}$
2:  Initialize $\mathbf{s}$ and $\mathbf{o}$ as described in “Initialization”
3:  for all iterations $n = 1, \dots, N$ do
4:      for all $i \in \mathcal{S}$ do
5:          Draw samples for $\mathbf{s}_i$ (Gaussian)
6:      end for
7:      for all $j \in \mathcal{O}$ do
8:          Draw samples for $\mathbf{o}_j$ (MCMC)
9:      end for
10:     Run TRW-S [Kol06] on discretized problem
11: end for
12: Output: $\hat{\mathbf{s}}$, $\hat{\mathbf{o}}$
```

potential are defined as

$$\begin{aligned} \chi_v^{\mathrm{A}}(\mathbf{p},\mathbf{q}) &= \|\mathbf{I}_0(\mathbf{p}) - \mathbf{I}_v(\mathbf{q})\|_1 && (3.8) \\ \chi_v^{\mathrm{L}}(\mathbf{p},\mathbf{q}) &= \|\mathbf{C}_0(\mathbf{p}) - \mathbf{C}_v(\mathbf{q})\|_1 && (3.9) \end{aligned}$$

and measures the difference in appearance $\mathbf{I}$ and 3D object coordinates $\mathbf{C}$ between image location $\mathbf{p}$ in the reference view and $\mathbf{q}$ in the target view, respectively. While the data term in (3.2) also evaluates appearance, we found that the Census descriptors work well mostly for textured background regions. In contrast, it returns noisy and unreliable results in the presence of textureless, specular surfaces such as on cars. However, as evidenced by our experiments, including an additional ℓ_1 constraint on appearance for instances leads to significantly better estimates in those cases. This observation is in accordance with recent works on direct visual odometry and SLAM [EKC16; NLD11; ESC14] which use similar measures to reliably estimate the camera pose in weakly textured environments. In contrast to them, here we exploit this constraint to estimate the relative pose of each individual weakly textured object in the scene.

The intuition behind this term is as follows: when warping the recognized instances from the reference frame into the target frame according to the estimated geometry and motion, the appearance as well as the part labeling induced by the object coordinates should agree. The term $[\mathbf{M}_0(\mathbf{p}) > 0 \wedge \mathbf{M}_v(\mathbf{q}) > 0]$ ensures that these constraints are only evaluated if both the reference and the target pixel belong to a detected instance (i.e., when $\mathbf{M} > 0$). However, as both χ_v^{A} and χ_v^{L} are positive terms the model prefers to associate instances with background regions which incurs no additional cost compared to associating instances with instances. We therefore incorporate an additional term which yields an appropriate bias λ and favors the association of instances.

Optimization: For optimizing (3.1), we leverage max-product particle belief propagation with TRW-S [Kol06] as proposed in [MG15]. At each iteration this optimization algorithm discretizes the continous variables by drawing samples around the current maximum-a-posteriori (MAP) solution and runs TRW-S until convergence before resampling. However, we found that this approach gets easily trapped in local minima, in particular due to the highly non-convex energy function associated with the appearance of the foreground objects.

We thus modify their sampling strategy as shown in Algorithm 1, which leads to better results. While the original algorithm [MG15] discretizes the continuous variables (in particular the rigid body motions $\mathbf{o}$) by sampling from a Gaussian centered at the current MAP estimate, we create samples by running a Markov chain based on the appearance term in (3.6) for each object individually. More specifically, our sampling energy warps all pixels inside an instance based on the predicted depth and the rigid body motion of the sample, and measures the photoconsistency between reference and target views using the ℓ_1 norm. This "informed" way of sampling ensures that TRW-S has access to high-quality samples compared to noisy estimates drawn around the current MAP solution. For sampling the geometry variables (i.e., normals), we follow [MG15].

The hyper parameters in our model are estimated using block coordinate descent on a train/validation split.

Initialization: As we are presented with a complex NP hard optimization problem in (3.1), initialization of parameters is an important step. We describe the details of our initialization procedure in the following.

We initialize the geometry parameters using dense disparity maps and the object hypotheses/motion variables using sparse flow predictions and the predicted bounding boxes/instances. More specifically, we robustly fit all superpixel parameters to the disparity estimates and aggregate all sparse flow estimates for each instance to robustly fit a rigid body motion to it via RANSAC.

For the instance-based algorithms, we aggregate the sparse flow estimates directly based on the instance masks and robustly fit a rigid body motion to it via RANSAC. Given the high quality of the instance predictions, this leads to very robust initializations. For baselines which do not leverage recognition we follow [MG15] and initialize objects based on clustering sparse 3D scene flow estimates which disagree with the background motion. Based on this clustering, we initialize the associations and poses using robust fitting using RANSAC. We refer the reader to [MG15] for further details.

While we have also experimented with color histograms and pose prediction to support the assignment of instances across frames, we found that aggregating sparse flow vectors [MHG15a] and dense geometry [May+16] allows for correctly associating almost all objects. We thus don't use such an additional term in the model, which, however, could be easily integrated to solve more challenging association problems as present in the KITTI 2015 scene flow benchmark.

Runtime: Our MATLAB implementation with C++ wrappers requires on average 40 seconds for each of the 10 iterations of the optimization described in Algorithm 1. This leads to a runtime of 7 minutes for processing one scene (4 images) on a single i7 core running at 3.0 Ghz. In addition, the inputs to our methods namely, dense disparity maps, sparse flow predictions and predicted bounding boxes/instances require on average 3 minutes for processing one scene, leading to a total runtime of 10 minutes.

3.2 Experimental Evaluation

	D1			D2		
	bg	*fg*	*bg+fg*	*bg*	*fg*	*bg+fg*
OSF	4.00	8.86	4.74	5.16	17.11	6.99
ISF-BBox	3.94	8.81	4.69	5.10	10.77	5.97
ISF-SegMask	4.06	7.97	4.66	5.26	9.20	5.86
ISF-SegMask-ObjCoord	4.08	7.98	4.68	5.27	9.20	5.87
ISF-SegMask-CNNDisp	3.55	3.94	3.61	4.86	4.72	4.84

	Fl			SF		
	bg	*fg*	*bg+fg*	*bg*	*fg*	*bg+fg*
OSF	6.38	20.56	8.55	7.38	24.12	9.94
ISF-BBox	6.46	12.90	7.44	7.42	17.11	8.90
ISF-SegMask	6.72	10.78	7.34	7.74	14.60	8.79
ISF-SegMask-ObjCoord	6.72	10.84	7.35	7.75	14.66	8.80
ISF-SegMask-CNNDisp	6.36	7.31	6.50	7.23	8.72	7.46

Table 3.1: Quantitative results from ablation study on KITTI 2015 Validation Set. *We report the disparity (D1,D2), flow (Fl) and scene flow (SF) error averaged over our validation set for OSF[MG15] (no recognition input), ISF-BBox (bounding box input), ISF-SegMask (segmentation input) and ISF-SegMask-ObjCoord (segmentation + object coordinates input). Additionally, we also report results of ISF-SegMask-CNNDisp (ISF-SegMask with higher quality CNN based disparity input).*

3.2.1 Effect of recognition granularity

In this section, we study the impact of different levels of recognition granularity for estimating the 3D scene flow of dynamic (i.e., foreground) objects. In addition to the recognition cues, we use sparse optical flow from sparse DiscreteFlow correspondences [MHG15a] and dense disparity maps from SPS-stereo [YMU14] for both rectified frames. We obtain the superpixel boundaries using StereoSLIC[YMU13]. Table 3.1 provides a quantitative comparison of the performance of *OSF* [MG15] (no recognition input), *ISF-BBox* (2D bounding boxes as recognition input), *ISF-SegMask* (2D instance segmentations as recognition input) and *ISF-SegMask-ObjCoord* (2D instance segmentations in conjunction with 3D object coordinates as recognition input) on our validation set (which is a subset of the KITTI 2015[MG15] scene flow training set). We report the error with respect to four different outputs: disparities in the first and second frame (D1,D2), optical flow (FL) and scene flow (SF).

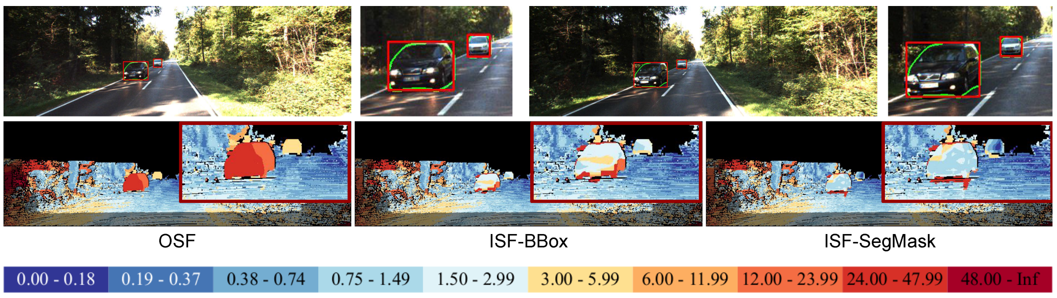


Figure 3.4: Qualitative Results from Ablation Study on KITTI 2015 Validation Set. *The top row shows predicted masks and bounding boxes which form the input to our method overlaid onto the left camera image at time t and $t+1$. Each figure in the bottom row (from left to right) shows scene flow error of OSF[MG15] (no recognition input), ISF-BBox (bounding box input) and ISF-SegMask (segmentation input) using the color scheme depicted in the legend. The red box shows zoom-ins.*

Our results indicate that recognition (both 2D bounding box and 2D instance segmentation) provides large improvements for optical flow estimation (and in turn scene flow estimation) on foreground parts of the scene. Note that we do not tackle the recognition of static background objects in this method which are estimated relatively well without such priors. Furthermore, as illustrated in Fig. 3.4, we note that instance segmentations as input improve performance in particular at the boundary of objects. We attribute this effect to the more fine grained nature of the 2D segmentation input compared to using rough 2D bounding boxes input. We remark that for evaluation, the KITTI 2015 benchmark considers only a subset of the cars in the scene as foreground, specifically dynamic cars within a threshold distance to the camera. In contrast, as defined in section 3.1.3, our scene flow estimation model considers all car instances detected by our CNN as foreground.

Moreover, we observe that 3D object coordinates do not increase performance beyond 2D instance segmentations (and hence yield the same estimates, i.e., the weight of the object coordinate terms are zero after optimization).

Next, we provide more qualitative analysis of the impact of different levels of recognition granularity for estimating the 3D scene flow of dynamic (i.e., foreground) objects. In particular, Fig. 3.5 to 3.9 compare the results of OSF[MG15] (no recognition input), *ISF-BBox* (2D bounding boxes as recognition input), *ISF-SegMask* (2D instance segmentations as recognition input) on our validation set (which is a subset of the KITTI 2015 training set). As 3D object coordinates do not increase performance beyond 2D instance segmentations, we do not visualize the result of our model using 3D object coordinates. Instead, we refer the reader to the next section for a detailed analysis why 3D object coordinates do not provide any further gains. For each scene and recognition granularity, Fig. 3.5 to 3.9 show the left input image at frame $t = 0$ and frame $t = 1$ along with the detected 2D bounding boxes, the 2D instance segmentation mask estimated by the network, the error images corresponding to the estimated disparity in the first frame (D1), the optical flow errors and the scene flow errors. Following [MG15], we use a logarithmic color coding where red shades represent errors above 3 pixels / 5% relative error and blue shades denote errors below 3 pixels / 5%

relative error. Our results indicate that recognition (both 2D bounding box and 2D instance segmentation) provide large improvements for optical flow estimation (and in turn scene flow estimation) on foreground parts of the scene (note that we do not tackle the recognition of static background objects in this method which are estimated relatively well without such priors).

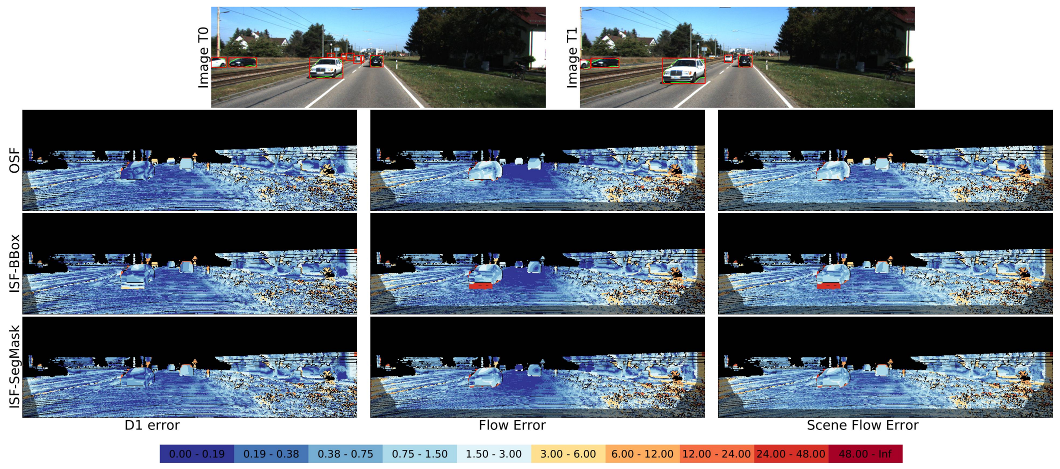


Figure 3.5: **Impact of Recognition Granularity on 3D Scene Flow Estimation.** *The top row shows two consecutive input images of the left camera along with the predicted 2D bounding boxes (red) and 2D instance segmentation masks (green) which form the input to our method. The figures below show (from top-to-bottom) the results of OSF[MG15] (no recognition input), ISF-BBox (bounding box input), ISF-SegMask (segmentation input) in terms of disparity (left), optical flow (middle) and scene flow (right) errors.*

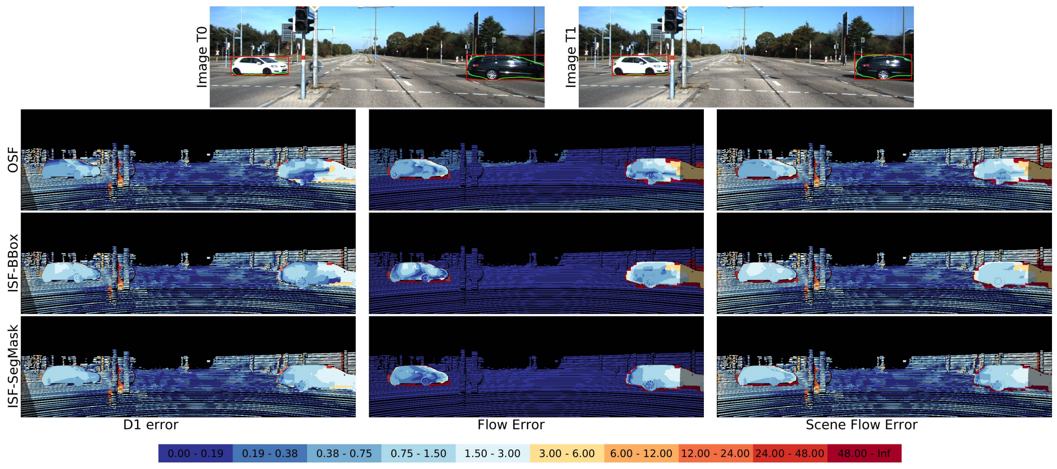


Figure 3.6: **Impact of Recognition Granularity on 3D Scene Flow Estimation.** *The top row shows two consecutive input images of the left camera along with the predicted 2D bounding boxes (red) and 2D instance segmentation masks (green) which form the input to our method. The figures below show (from top-to-bottom) the results of OSF[MG15] (no recognition input), ISF-BBox (bounding box input), ISF-SegMask (segmentation input) in terms of disparity (left), optical flow (middle) and scene flow (right) errors.*

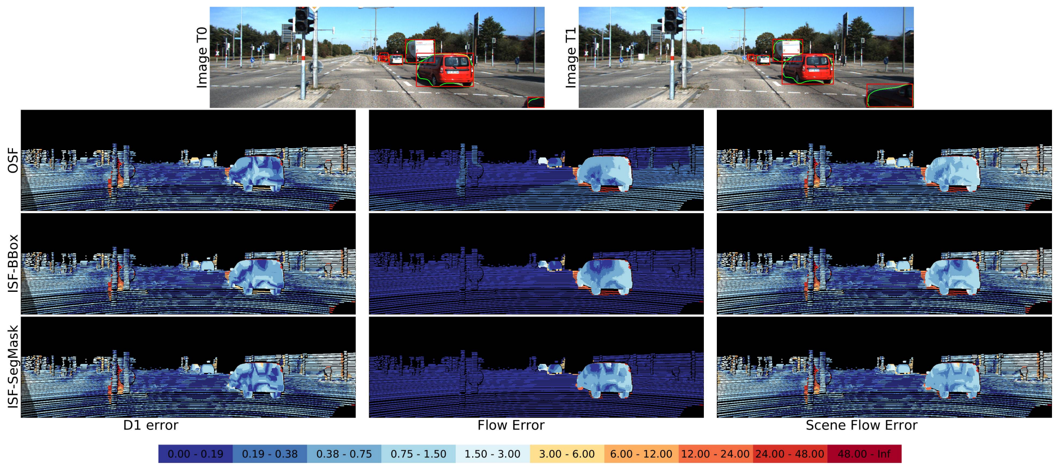


Figure 3.7: **Impact of Recognition Granularity on 3D Scene Flow Estimation.** *The top row shows two consecutive input images of the left camera along with the predicted 2D bounding boxes (red) and 2D instance segmentation masks (green) which form the input to our method. The figures below show (from top-to-bottom) the results of OSF[MG15] (no recognition input), ISF-BBox (bounding box input), ISF-SegMask (segmentation input) in terms of disparity (left), optical flow (middle) and scene flow (right) errors.*

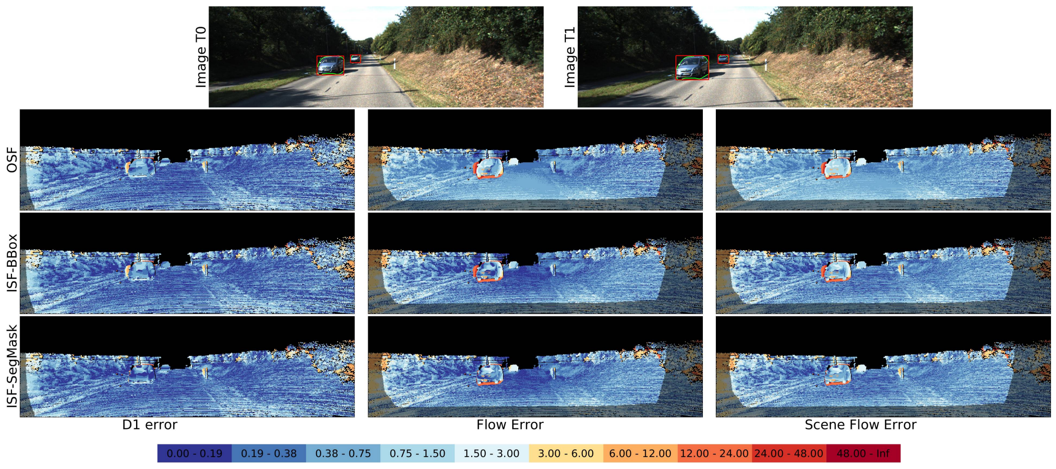


Figure 3.8: **Impact of Recognition Granularity on 3D Scene Flow Estimation.** *The top row shows two consecutive input images of the left camera along with the predicted 2D bounding boxes (red) and 2D instance segmentation masks (green) which form the input to our method. The figures below show (from top-to-bottom) the results of OSF[MG15] (no recognition input), ISF-BBox (bounding box input), ISF-SegMask (segmentation input) in terms of disparity (left), optical flow (middle) and scene flow (right) errors.*

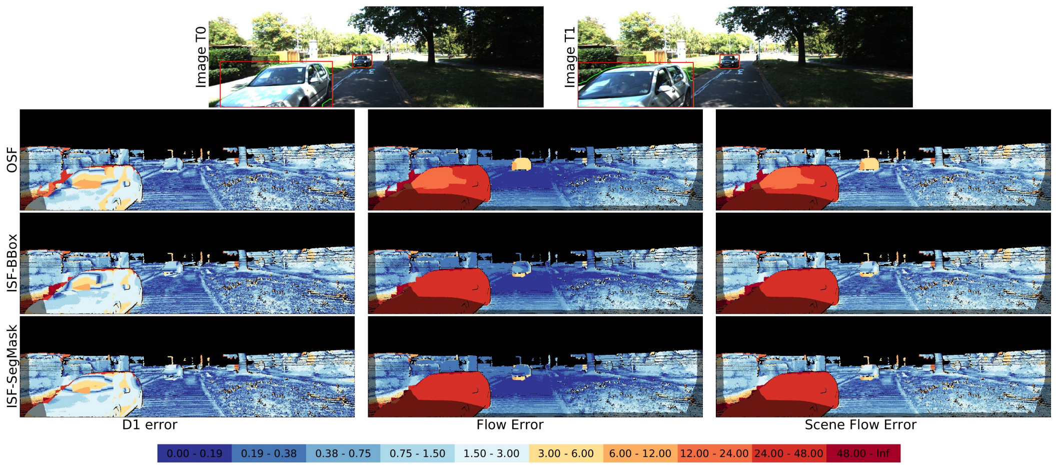


Figure 3.9: **Impact of Recognition Granularity on 3D Scene Flow Estimation.** *The top row shows two consecutive input images of the left camera along with the predicted 2D bounding boxes (red) and 2D instance segmentation masks (green) which form the input to our method. The figures below show (from top-to-bottom) the results of OSF[MG15] (no recognition input), ISF-BBox (bounding box input), ISF-SegMask (segmentation input) in terms of disparity (left), optical flow (middle) and scene flow (right) errors.*

3.2.2 Why are 3D object coordinates not important?

In this section, we study the effect of using 3D object coordinate cues to improve 3D scene flow estimation. Towards this goal, we evaluated our *ISF-SegMask-ObjCoord* method which includes a data term in the CRF penalizing the difference in 3D object coordinates between corresponding image locations. Figure 3.10 shows the influence of the 3D object coordinate term weight on the scene flow error. Surprisingly, we observe an increase in 3D scene flow error, in particularly for the foreground parts of the image, when increasing the importance of the 3D object coordinate cue.

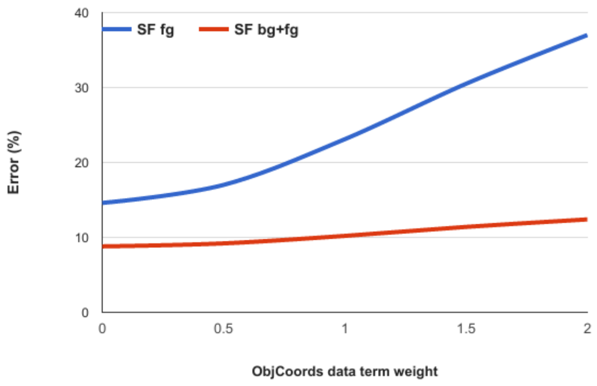


Figure 3.10: **Impact of 3D Object Coordinates.** *This figure shows the 3D scene flow error evaluated at all pixels in the image (red) and only the foreground pixels (blue) with respect to the weight of the object coordinates data term.*

We now analyze the reason why 3D object coordinates fail to improve 3D scene flow estimation beyond simpler 2D instance segmentation cues. We hypothesize that the accuracy of 3D object coordinate predictions from modern CNN-based methods is below what is required to further improve 3D scene flow estimation due to the high level of accuracy requested by current benchmarks (i.e., 3 pixels error in KITTI). In order to validate this hypothesis, we computed optical flow for the foreground part of the scene by finding correspondences between the reference and target frame using the 3D object coordinates. We then compare this flow with the optical flow results of our CRF-based *ISF-SegMask* method after initialization and with the final results after optimization. Quantitatively, we find that the optical flow computed via 3D object coordinate matching improves upon *ISF-SegMask* initialization and final flow output for only 1.4% and 1% of the foreground pixels, respectively. For the remaining pixels, however, the quality of the 3D object coordinates does not reach the level required for accurate 3D scene flow estimation. Fig. 3.11 to 3.16 illustrate this observation on a few examples from our validation set. The figures show that flow accuracy from object coordinate matching is either worse or similar at places where *ISF-SegMask* fails. Therefore, 3D object coordinates fail in improving 3D scene flow estimation using our

CRF framework.

Remark: We compare the quality of our 3D object coordinate predictions to state-of-the-art approaches using random forests in the last section of this chapter and demonstrate significantly better performance when using our convolutional neural network architecture. This demonstrates that even state-of-the-art object coordinate predictions are not helpful for improving 3D scene flow due to the high level of accuracy required by current benchmarks (i.e., 3 pixels error in KITTI).

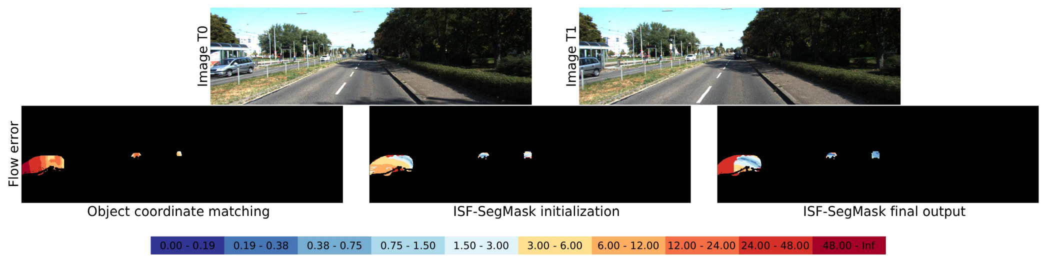


Figure 3.11: **Qualitative Analysis of Flow from 3D Object Coordinate on KITTI 2015 Validation Set.** *The top row shows two consecutive frames of the left camera. Each figure in the bottom row shows optical flow errors on foreground pixels in the reference view using the color scheme depicted in the legend. In particular, we compare results when flow is obtained directly from the 3D object coordinate predictions (left) to our ISF-SegMask method before (middle) and after (right) optimization.*

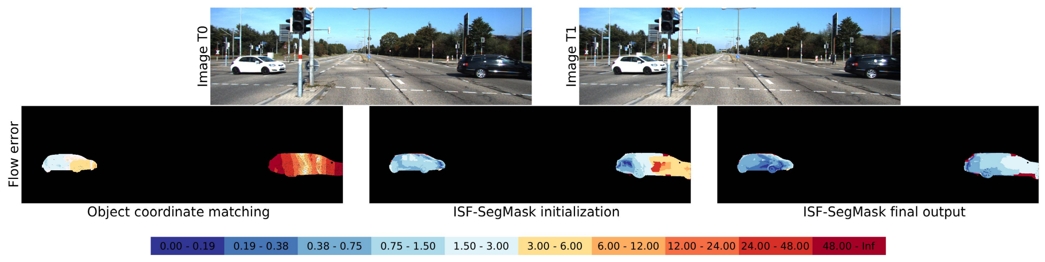


Figure 3.12: **Qualitative Analysis of Flow from 3D Object Coordinate on KITTI 2015 Validation Set.** *The top row shows two consecutive frames of the left camera. Each figure in the bottom row shows optical flow errors on foreground pixels in the reference view using the color scheme depicted in the legend. In particular, we compare results when flow is obtained directly from the 3D object coordinate predictions (left) to our ISF-SegMask method before (middle) and after (right) optimization.*

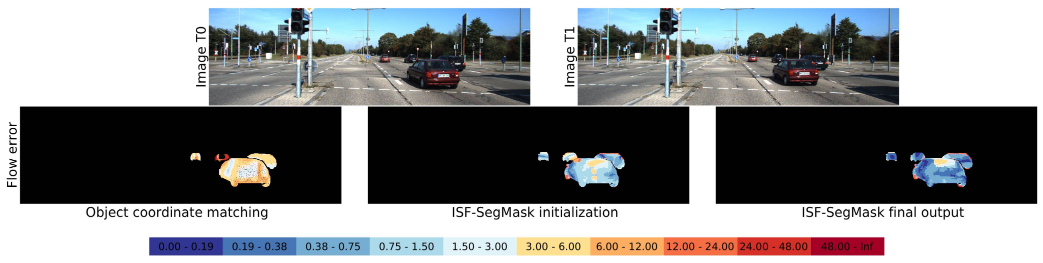


Figure 3.13: **Qualitative Analysis of Flow from 3D Object Coordinate on KITTI 2015 Validation Set.** *The top row shows two consecutive frames of the left camera. Each figure in the bottom row shows optical flow errors on foreground pixels in the reference view using the color scheme depicted in the legend. In particular, we compare results when flow is obtained directly from the 3D object coordinate predictions (left) to our ISF-SegMask method before (middle) and after (right) optimization.*

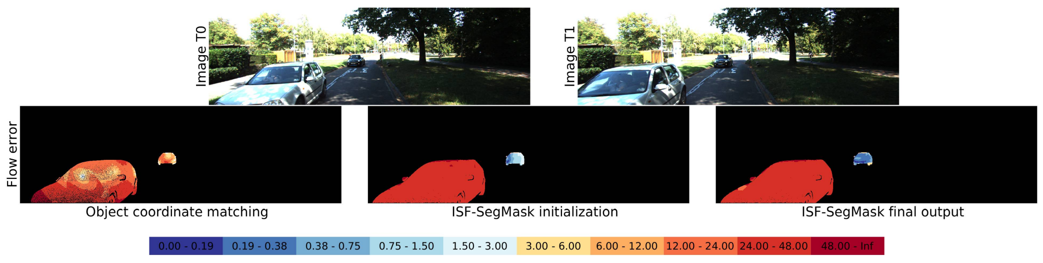


Figure 3.14: **Qualitative Analysis of Flow from 3D Object Coordinate on KITTI 2015 Validation Set.** *The top row shows two consecutive frames of the left camera. Each figure in the bottom row shows optical flow errors on foreground pixels in the reference view using the color scheme depicted in the legend. In particular, we compare results when flow is obtained directly from the 3D object coordinate predictions (left) to our ISF-SegMask method before (middle) and after (right) optimization.*

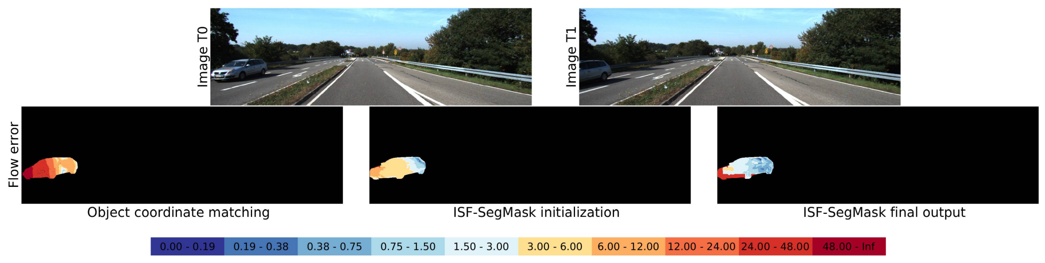


Figure 3.15: **Qualitative Analysis of Flow from 3D Object Coordinate on KITTI 2015 Validation Set.** *The top row shows two consecutive frames of the left camera. Each figure in the bottom row shows optical flow errors on foreground pixels in the reference view using the color scheme depicted in the legend. In particular, we compare results when flow is obtained directly from the 3D object coordinate predictions (left) to our ISF-SegMask method before (middle) and after (right) optimization.*

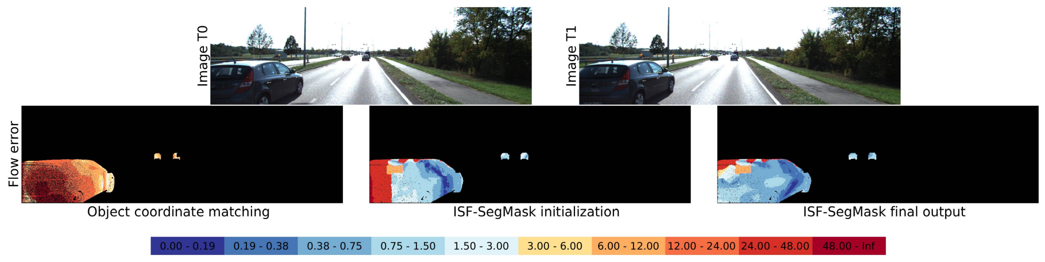


Figure 3.16: **Qualitative Analysis of Flow from 3D Object Coordinate on KITTI 2015 Validation Set.** *The top row shows two consecutive frames of the left camera. Each figure in the bottom row shows optical flow errors on foreground pixels in the reference view using the color scheme depicted in the legend. In particular, we compare results when flow is obtained directly from the 3D object coordinate predictions (left) to our ISF-SegMask method before (middle) and after (right) optimization.*

3.2.3 Experiments with CNN based disparity as input

Furthermore, we evaluated our method with the optimal level of recognition granularity selected based on the ablation study (*ISF-SegMask*) on higher quality disparity inputs computed from DispNetC [May+16] and MC-CNN-acrt [ŽL16]. In particular, combining DispNetC results for instance pixels and MC-CNN-acrt results for the rest performed best on our validation set. We report the results of our top performing method *ISF-SegMask-CNNDisp* on the validation set in Table 3.1.

	D1			D2		
	bg	*fg*	*bg+fg*	*bg*	*fg*	*bg+fg*
PRSM* [VSR13]	3.02	10.52	**4.27**	5.13	15.11	6.79
OSF+TC* [NŠ17]	4.11	9.64	5.03	5.18	15.12	6.84
OSF [MG15]	4.54	12.03	5.79	5.45	19.41	7.77
ISF-SegMask-CNNDisp	**4.12**	**6.17**	4.46	**4.88**	**11.34**	**5.95**

	Fl			SF		
	bg	*fg*	*bg+fg*	*bg*	*fg*	*bg+fg*
PRSM* [VSR13]	**5.33**	13.40	6.68	6.61	20.79	8.97
OSF+TC* [NŠ17]	5.76	13.31	7.02	7.08	20.03	9.23
OSF [MG15]	5.62	18.92	7.83	7.01	26.34	10.23
ISF-SegMask-CNNDisp	5.40	**10.29**	**6.22**	**6.58**	**15.63**	**8.08**

Table 3.2: Quantitative Results on the KITTI 2015 Scene Flow Evaluation Server. *These tables shows the disparity (D1/D2), flow (Fl) and scene flow (SF) errors averaged over all 200 test images for our method in comparison to other leading methods (OSF[MG15], OSF-TC[NŠ17] and PRSM[VSR13]) on the KITTI 15 benchmark. Methods with * use more than two temporal frames.*

3.2.4 Results on the KITTI Benchmark

In this section, we present results of our top performing method *ISF-SegMask-CNNDisp* evaluated on the KITTI 2015 scene flow evaluation server. Table 3.2 compares the performance of our method with other leading methods on the benchmark: PRSM [VSR15], OSF [MG15] and OSF-TC [NŠ17]. We obtain state-of-the-art performance at the time of submission, even including anonymous submissions. Notably, our method also outperforms methods which use more than two temporal frames (OSF-TC [NŠ17] and PRSM[VSR15]). Figure 3.17 shows scene flow errors for examples where other leading methods fail to effectively estimate scene flow in foreground regions. Scene flow estimation is challenging for this region as the texture-less car undergoes large displacement and is occluded in the second frame. Our method employing recognition cues performs comparatively better than other methods. Without recognition cues, only very few matches could be established in those regions and most of them would be wrong.

3.2.5 Random Forests vs. CNNs for Predicting Object Coordinates

In this section, we compare our object coordinate predictions to those of an existing state-of-the-art approach using random forests as presented by Brachmann et al. [Bra+14]. For this comparison, we use the same random forest (RF) implementation provided by [Bra+14]. In particular, their implementation estimates the pixel-wise object coordinates using an RGB-Depth patch of size 20x20 at each pixel of the instance. We train a random forest with 3 trees and maximum depth of 64 by sampling 3 million random points during training.

We remark that in [Bra+14], object coordinates are predicted only for a fixed set of known

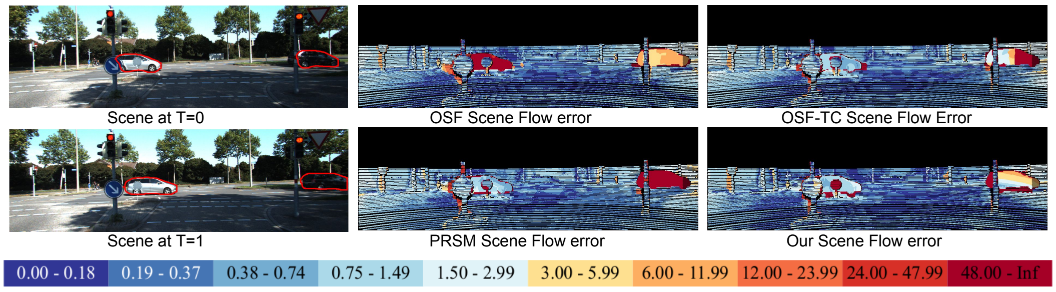


Figure 3.17: **Qualitative Comparison on KITTI-15 Test Set.** *The first column shows the input images, followed by scene flow error maps of OSF[MG15], OSF-TC[NŠ17], PRSM[VSR13] and our method using the color scheme depicted in the legend.*

3D objects. On the contrary, our case is a lot more complex due to the large intra-class variation among cars (hatchback, station wagon, SUV etc.). This presents a challenge to the prediction model as it has to generalize well for all of the possible car types. We found that our CNN based model generalized better to various types of cars and is more accurate in predicting object coordinates compared the state-of-the-art RF based approach. The average euclidean error of the predicted object coordinates using our CNN is 0.6 meters while random forests acchieve an average error of 2.89 meters. We believe that with more training data the performance of our object coordinate predictions could be further improved.

In Fig. 3.18, we compare the object coordinate predictions from our CNN to those of the RF. For illustration purposes, we have colorized the object coordinate ground truth. As clearly visible, the continuous smooth change of colours indicate a smooth fine grained part labelling of the car. In addition, we show the corresponding error map. In contrast to our predictions, the object coordinate predictions from the RF are extremely noisy and thus lead to very large errors. We conclude that random forests can't cope well with the appearance changes induced by intra-class variations present in our dataset.

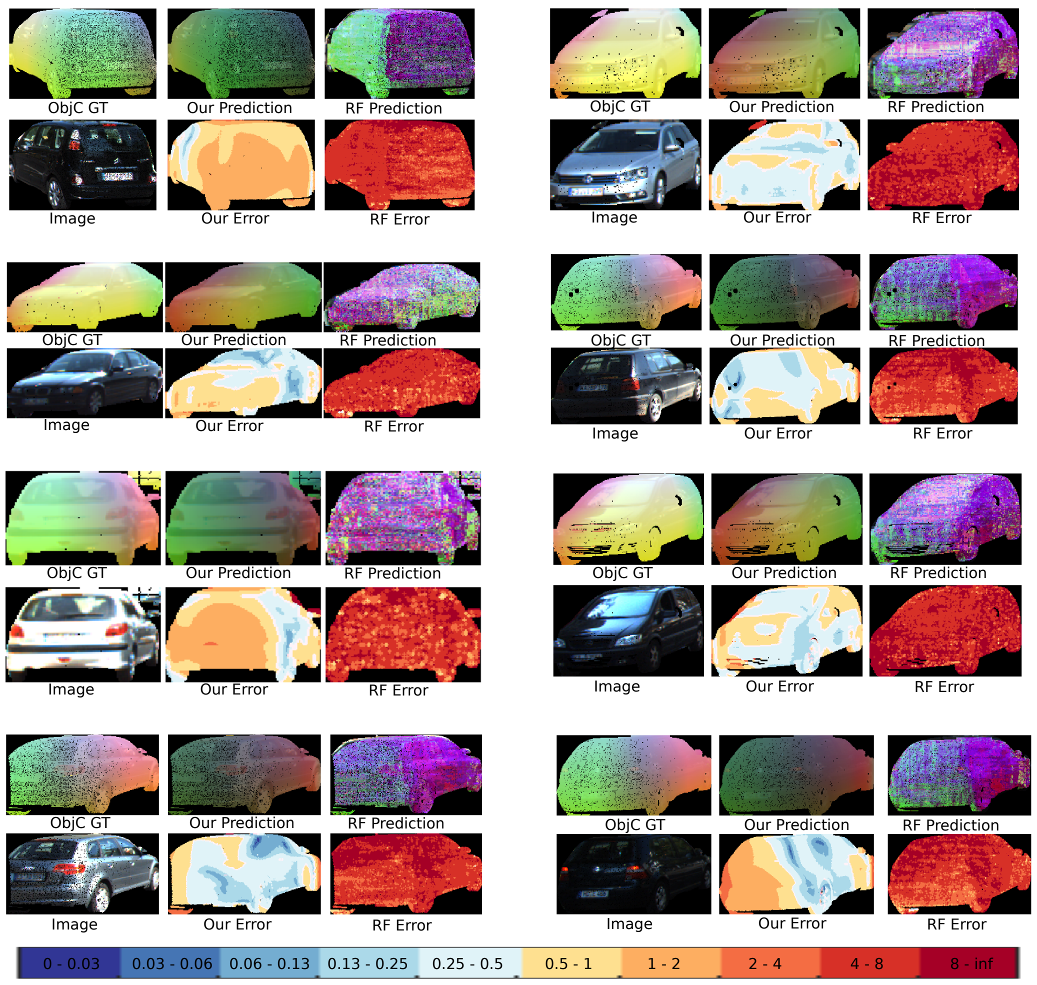


Figure 3.18: *This figure shows a qualitative comparison of object coordinate predictions from our CNN and a baseline Random Forest (RF) approach. In each of the illustrations, we show the ground truth object coordinates (ObjC GT), predictions from our CNN (Our Prediction), predictions from RF (RF Prediction), corresponding pixel wise error maps (Our Error, RF Error) and also the car instance for which the object coordinates are predicted. While the error of our CNN based object coordinate predictions is mostly below 1 meter, the RF based approach leads to much larger errors.*

4 Learning Representations for Rigid Motion Estimation from Point Clouds

In the previous chapter we proposed an approach to address the problem with large displacements or local ambiguities by regularization of scene flow estimation with recognition cues such as semantic grouping and fine-grained geometric recognition. However, stereo-based scene flow methods suffer from a fundamental flaw, the "curse of two-view geometry": it can be shown that the depth error grows quadratically with the distance to the observer [Len+11]. This causes problems for the large baselines and object depths often found in self-driving cars, as illustrated in Fig. 4.1 (top).

In this chapter, we extend the idea of semantic grouping as a regularizer for 3D motion to 3D point cloud measurements from laser scanners. Towards this goal, we jointly predict pointwise 3D scene flow, pointwise rigid motion as well as the 3D bounding box. We then semantically group the pointwise rigid motion estimates over the predicted detections to obtain rigid body motion of objects in the scene. While the prospect of estimating 3D scene flow from unstructured point clouds is promising, it is also a challenging task. Because of the sparse and non-uniform nature of the point clouds, as well as the missing appearance information, the data association problem is complicated. Moreover, characteristic patterns produced by the scanner, such as the circular rings in Fig. 4.1 (bottom), move with the observer and can easily mislead local correspondence estimation algorithms. Furthermore, we show that the traditional global representation of rigid body motion cannot be directly inferred by CNNs. Finally, for training our deep network, a large labeled dataset is required. To address these challenges, we propose *PointFlowNet*, a generic model for learning 3D scene flow from pairs of unstructured 3D point clouds. We propose a novel translation equivariant representation for pointwise rigid motion which can be inferred by CNNs. We create large, diverse training datasets by augmenting real scans from KITTI with virtual CAD models of cars. We then use our LiDAR simulator that realistically modeling occlusions and simulating sensor noise to generate the augmented scans.

4.1 Method

We start by formally defining our problem. Let $\mathbf{P}_t \in \mathbb{R}^{N\times 3}$ and $\mathbf{P}_{t+1} \in \mathbb{R}^{M\times 3}$ denote the input 3D point clouds at frames t and $t+1$, respectively. Our goal is to estimate

- the 3D scene flow $\mathbf{v}_i \in \mathbb{R}^3$ and the 3D rigid motion $\mathbf{R}_i \in \mathbb{R}^{3\times 3}$, $\mathbf{t}_i \in \mathbb{R}^3$ at each of the N points in the reference point cloud at frame t, and
- the location, orientation, size and rigid motion of every moving object in the scene (in our experiments, we focus solely on cars).

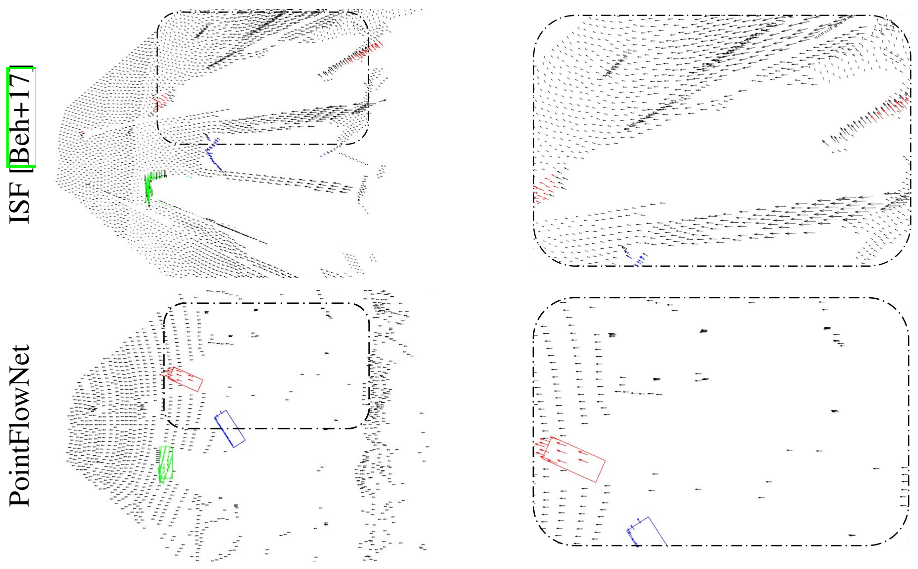


Figure 4.1: **Motivation for LIDAR based Scene Flow.** *To motivate the use of LIDAR sensors in the context of autonomous driving, we provide a qualitative comparison of the state-of-the-art image-based scene flow method ISF [Beh+17] (top) to our LIDAR-based PointFlowNet (bottom) using a scene from the KITTI 2015 dataset [MG15]. The left column shows the output of the two methods. The right column shows a zoomed-in version of the inlet. While the image-based result suffers from the "curse of two-view geometry" (with noisy geometry, and non-uniform background movement), our LIDAR-based approach is also accurate in distant regions. Moreover, ISF relies on instance segmentation in the image space for detecting objects: depth estimation errors at the boundaries lead to objects being split into two 3D clusters (e.g., the red car). For clarity, we visualize only a subset of the points.*

The overall network architecture of our approach is illustrated in Figure 4.2. The network comprises five main components: (1) feature encoding layers, (2) context encoding layers (3) scene flow estimation, ego-motion estimation and 3D object detection layers, (4) rigid motion estimation layers and (5) object motion decoder. In the following, we provide a detailed description for each of these components as well as the loss functions.

4.1.1 Feature Encoder

The feature encoding layers take a raw point cloud as input, partition the space into voxels, and describe each voxel with a feature vector. The simplest form of aggregation is binarization, where any voxel containing at least one point is set to 1 and all others are zero. However, better results can be achieved by aggregating high-order statistics over the voxel [Qi+17a; Qi+17b; ZT18; Su+18; PZZ17; Che+17b]. In this chapter, we leverage the feature encoding recently proposed by Zhou et al. [ZT18], which has demonstrated state-of-the-art results for 3D object detection from point clouds.

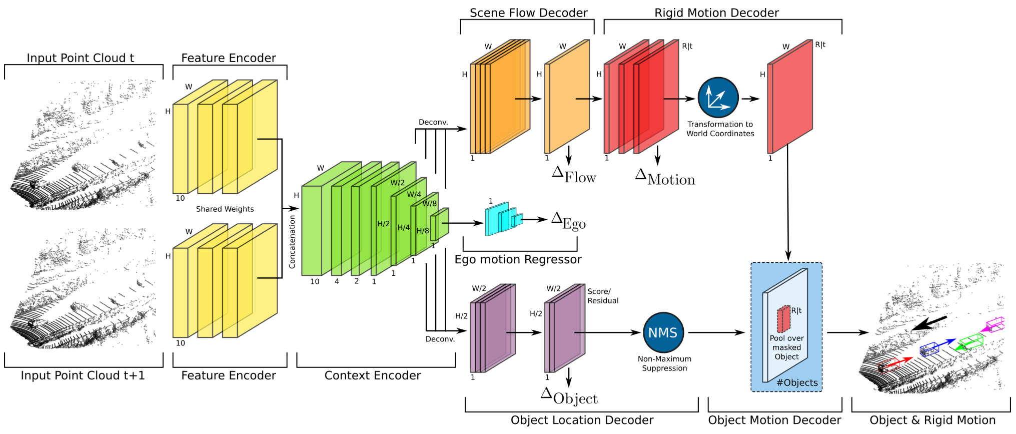


Figure 4.2: **Network Architecture.** *The* **feature encoder** *takes a raw LIDAR point cloud as input, groups the points into $W \times H \times 10$ voxels, and outputs 128D feature maps (for clarity, the size of the feature maps is not shown in the figure) which are concatenated and passed to the* **context encoder**. *The context encoder learns a global representation by interleaving convolution with strided convolution layers and "flattening" the third dimension (height above ground), i.e., we assume that 3D objects cannot be located on top of each other and that 3D scene points that project to the same location in the ground plane undergo the same 3D motion. Feature maps at different resolutions are upsampled, stacked and fed into the decoding branches. 3D scene flow is computed for every input voxel in the* **scene flow decoder** *and the result is passed to the* **rigid motion decoder**, *which infers a rigid body transformation for every point. In parallel, the* **ego-motion regressor**, *further downsamples the feature map by interleaving convolutional layers with strided convolutional layers and a fully connected layer at the end to regress rigid motion for the ego vehicle. In addition, the* **object decoder** *predicts the location and size (i.e., 3D bounding box) of objects in the scene. Finally, the* **object motion decoder** *takes the point-wise rigid body motions as input and predicts the object rigid motions by pooling the rigid motion field over the detected 3D objects.*

We briefly summarize this encoding, but refer the reader to [ZT18] for more details. We subdivide the 3D space of each input point cloud into equally spaced voxels and group points according to the voxel they reside in. To reduce bias with respect to LIDAR point density, a fixed number of T points is randomly sampled for all voxels containing more than T points. Each voxel is processed with a stack of Voxel Feature Encoding (VFE) layers to capture local and global geometric properties of its contained points. As more than 90% of the voxels in LIDAR scans tend to be empty, we only process non-empty voxels and store the results in a sparse 4D tensor.

We remark that alternative representations, e.g., those that directly encode the raw point cloud [Wan+18; GWL18], could be a viable alternative to voxel representations. However, as the representation is not the main focus of this approach, we will leave such an investigation to future work.

4.1.2 Context Encoder

As objects in a street scene are restricted to the ground plane, we only estimate objects and motions on this plane: we assume that 3D objects cannot be located on top of each other and that 3D scene points directly above each other undergo the same 3D motion. This is a valid assumption for our autonomous driving scenario, and greatly improves memory efficiency. Following [ZT18], the first part of the context encoder vertically downsamples the voxel feature map by using three 3D convolutions with vertical stride 2. The resulting 3D feature map is reshaped by stacking the remaining height slices as feature maps to yield a 2D feature map. The resulting 2D feature map is provided to three blocks of 2D convolutional layers. The first layer of each block downsamples the feature map via a convolution with stride 2, followed by a series of convolution layers with stride 1.

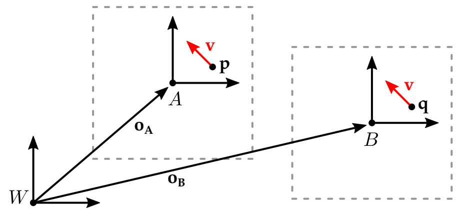
4.1.3 3D Detection, Ego-motion and 3D Scene Flow

Next, the network splits up in three branches for respectively ego-motion estimation, 3D object detection and 3D scene flow estimation. As there is only one observer, the ego-motion branch further downsamples the feature map by interleaving convolutional layers with strided convolutional layers and finally using a fully connected layer to regress a 3D ego-motion (movement in the ground-plane and rotation around the vertical). For the other two tasks, we upsample the output of the various blocks using up-convolutions: to half the original resolution for 3D object detection, and to the full resolution for 3D scene flow estimation. The resulting features are stacked and mapped to the training targets with one 2D convolutional layer each. We regress a 3D vector per voxel for the scene flow, and follow [ZT18] for the object detections: regressing likelihoods for a set of proposal bounding boxes and regressing the residuals (translation, rotation and size) between the positive proposal boxes and corresponding ground truth boxes. A proposal bounding box is called positive if it has the highest Intersection over Union (IoU, in the ground plane) with a ground truth detection, or if its IoU with any ground truth box is larger than 0.6, as in [ZT18].

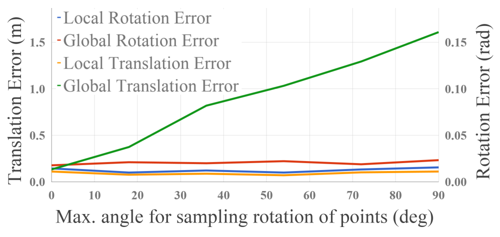
4.1.4 Rigid Motion Decoder

We now wish to infer per-pixel and per-object rigid body motions from the previously estimated 3D scene flow. For a single point in isolation, there are infinitely many rigid body motions that explain a given 3D scene flow: this ambiguity can be resolved by considering the local neighborhood.

It is unfortunately impossible to use a convolutional neural network to regress rigid body motions that are represented in global world coordinates, as the conversion between scene flow and global rigid body motion depends on the location in the scene: while convolutional layers are translation equivariant, the mapping to be learned is not. Identical regions of flow lead to different global rigid body motions, depending on the location in the volume, and a fully convolutional network cannot model this. In the following, we first prove that the rigid motion in the world coordinate system is not translation equivariant. Subsequently, we introduce our proposed rigid motion representation in local coordinates and show it to be translation equivariant and therefore amenable to fully convolutional inference.



(a) Local (A,B) and World Coordinate System (W)



(b) Quantitative Comparison

Figure 4.3: **Rigid Motion Estimation.** *In (a), indices A and B denote the coordinate system of points* $\mathbf{p}$ *and* $\mathbf{q}$ *at origin* $\mathbf{o}_A$ *and* $\mathbf{o}_B$*, respectively. The same scene flow* $\mathbf{v}$ *can locally be explained with the same rigid body motion* $(\mathbf{R}_L, \mathbf{t}_L)$*, but requires different translations* $\mathbf{t}_W^{\mathbf{p}} \neq \mathbf{t}_W^{\mathbf{q}}$ *in the global coordinate system. A simple example (b) provides empirical evidence that translation cannot be learned in global coordinates with a CNN. Using global coordinates, the translation error increases significantly with the magnitude of rotation (green). There is no such increase in error when using local coordinates (orange).*

Let us assume a point $\mathbf{p}$ in world coordinate system W and let A denote a local coordinate system with origin $\mathbf{o}_A$ as illustrated in Fig. 4.3a. A scene flow vector $\mathbf{v}$ is explained by rigid body motion $(\mathbf{R}_A, \mathbf{t}_A)$, represented in local coordinate system A with origin $\mathbf{o}_A$, if and only if:

$$\mathbf{v} = [\mathbf{R}_A(\mathbf{p} - \mathbf{o}_A) + \mathbf{t}_A)] - (\mathbf{p} - \mathbf{o}_A) \tag{4.1}$$

Now assume a second world location $\mathbf{q}$, also with scene flow $\mathbf{v}$ as in Fig. 4.3a. Let B denote a second local coordinate system with origin $\mathbf{o}_B$ such that $\mathbf{p}$ and $\mathbf{q}$ have the same local coordinates in their respective coordinate system, i.e., $\mathbf{p} - \mathbf{o}_A = \mathbf{q} - \mathbf{o}_B$. We now prove the following two claims:

1. There exists no rigid body motion $\mathbf{R}_W, \mathbf{t}_W$ represented in world coordinate system W that explains the scene flow $\mathbf{v}$ for both $\mathbf{p}$ and $\mathbf{q}$, unless $\mathbf{R}_W = \mathbf{I}$.

2. Any rigid body motion $(\mathbf{R}_A, \mathbf{t}_A)$ explaining scene flow $\mathbf{v}$ for $\mathbf{p}$ in system A also does so for $\mathbf{q}$ in system B.

Towards this goal, we introduce the notation $(\mathbf{R}_W^{\mathbf{p}}, \mathbf{t}_W^{\mathbf{p}})$ to indicate rigid motion in world coordinates W induced by $\mathbf{v}_{\mathbf{p}}$.

Claim 4.1.1.

$$\begin{aligned} \forall \mathbf{p}, \mathbf{q} \in \mathbb{R}^3, \mathbf{p} - \mathbf{o}_A = \mathbf{q} - \mathbf{o}_B, \mathbf{o}_A \neq \mathbf{o}_B : \\ \mathbf{v}_{\mathbf{p}} = \mathbf{v}_{\mathbf{q}} \implies \mathbf{R}_W^{\mathbf{p}} \neq \mathbf{R}_W^{\mathbf{q}} \quad & or \\ \mathbf{t}_W^{\mathbf{p}} \neq \mathbf{t}_W^{\mathbf{q}} \quad & or \\ \mathbf{R}_W^{\mathbf{p}} = \mathbf{R}_W^{\mathbf{q}} = \mathbf{I} \end{aligned} \tag{4.2}$$

Proof of Claim 1. *From* $\mathbf{v_p} = \mathbf{v_q}$ *we get*

$$\begin{aligned}
\mathbf{R}^{\mathbf{p}}_W \mathbf{p} + \mathbf{t}^{\mathbf{p}}_W - \mathbf{p} &= \mathbf{R}^{\mathbf{q}}_W \mathbf{q} + \mathbf{t}^{\mathbf{q}}_W - \mathbf{q} \\
\mathbf{R}^{\mathbf{p}}_W \mathbf{p} + \mathbf{t}^{\mathbf{p}}_W &= \mathbf{R}^{\mathbf{q}}_W (\mathbf{p} - \Delta\mathbf{o}) + \mathbf{t}^{\mathbf{q}}_W + \Delta\mathbf{o} \\
\left(\mathbf{R}^{\mathbf{p}}_W - \mathbf{R}^{\mathbf{q}}_W\right) \mathbf{p} &= \left(\mathbf{I} - \mathbf{R}^{\mathbf{q}}_W\right) \Delta\mathbf{o} + \left(\mathbf{t}^{\mathbf{q}}_W - \mathbf{t}^{\mathbf{p}}_W\right)
\end{aligned}$$

where $\Delta\mathbf{o} = \mathbf{o}_A - \mathbf{o}_B$*. Now, we assume that* $\mathbf{R}^{\mathbf{p}}_W = \mathbf{R}^{\mathbf{q}}_W$ *and that* $\mathbf{t}^{\mathbf{p}}_W = \mathbf{t}^{\mathbf{q}}_W$ *(in all other cases the claim is already fulfilled). In this case, we have* $\Delta\mathbf{o} = \mathbf{R}^{\mathbf{p}}_W \Delta\mathbf{o}$*. However, any rotation matrix representing a non-zero rotation has no real eigenvectors. Hence, as* $\mathbf{o}_A \neq \mathbf{o}_B$*, this equality can only be fulfilled if* $\mathbf{R}^{\mathbf{p}}_W$ *is the identity matrix.* ■

Claim 4.1.2.

$$\begin{aligned}
\forall \mathbf{p}, \mathbf{q} \in \mathbb{R}^3, \mathbf{p} - \mathbf{o}_A = \mathbf{q} - \mathbf{o}_B, \mathbf{o}_A \neq \mathbf{o}_B : \\
\mathbf{v} = \mathbf{R}(\mathbf{p} - \mathbf{o}_A) + \mathbf{t} + (\mathbf{p} - \mathbf{o}_A) \\
\implies \mathbf{v} = \mathbf{R}(\mathbf{q} - \mathbf{o}_B) + \mathbf{t} + (\mathbf{q} - \mathbf{o}_B)
\end{aligned} \tag{4.3}$$

Proof of Claim 2. *Trivially from* $\mathbf{p} - \mathbf{o}_A = \mathbf{q} - \mathbf{o}_B$. ■

The first proof shows the non-stationarity of rigid body motions represented in global coordinates, while the second proof shows that the rigid motion represented in local coordinates is stationary and can therefore be learned by a translation equivariant convolutional neural network.

We provide a simple synthetic experiment in Figure 4.3 to empirically confirm this analysis. Towards this goal, we warp a grid of 10×10 points by random rigid motions, and then try to infer these rigid motions from the resulting scene flow: as expected, the estimation is only successful using local coordinates. Note that a change of reference system only affects the translation component while the rotation component remains unaffected. Motivated by the preceding analysis, we task our CNN to predict rigid motion in local coordinates, followed by a deterministic layer which transforms local coordinates into global coordinates as follows:

$$\begin{aligned}
\mathbf{R}_L &= \mathbf{R}_W \qquad & \mathbf{t}_L &= (\mathbf{R}_W - \mathbf{I})\mathbf{o}_L + \mathbf{t}_W \\
\mathbf{R}_W &= \mathbf{R}_L \qquad & \mathbf{t}_W &= (\mathbf{I} - \mathbf{R}_W)\mathbf{o}_L + \mathbf{t}_L
\end{aligned} \tag{4.4}$$

In our case, the origin of the world coordinate system W coincides with the LIDAR scanner and the origin of the local coordinate systems is located at the center of each voxel.

4.1.5 Object Motion Decoder

Finally, we combine the results of 3D object detection and rigid motion estimation into a single rigid motion for each detected object. We first apply non-maximum-suppression (NMS) using detection threshold τ, yielding a set of 3D bounding boxes. To estimate the rigid body motion of each detection, we pool the predicted rigid body motions over the corresponding voxels (i.e., the voxels in the bounding box of the detection) by computing the median translation and rotation. Note that this is only possible as the rigid body motions have been converted back into world coordinates.

4.1.6 Loss Functions

This section describes the loss functions used by our approach. While it seems desirable to define a rigid motion loss directly at object level, this is complicated by the need for differentiation through the non-maximum-suppression step and the difficulty associating to ground truth objects. Furthermore, balancing the influence of an object loss across voxels is much more complex than applying all loss functions directly at the voxel level. We therefore use auxiliary voxel-level loss functions. Our loss comprises four parts:

$$\mathcal{L} = \alpha \mathcal{L}_{flow} + \beta \mathcal{L}_{rigmo} + \gamma \mathcal{L}_{ego} + \mathcal{L}_{det} \tag{4.5}$$

Here, α, β, γ are positive constants for balancing the relative importance of the task specific loss functions. In order to fix the balancing weights in our multi-task loss function, we performed a hyperparameter search over the loss function weights, maximizing the performance over the scene flow and object rigid motion. We set $\alpha = 4.0$, $\beta = 1.0$, $\gamma = 1.0$ in Eqn. 4.5.

We now describe the task-specific loss functions in more detail.

Scene Flow Loss: The scene flow loss is defined as the average ℓ_1 distance between the predicted scene flow and the true scene flow at every voxel

$$\mathcal{L}_{flow} = \frac{1}{K} \sum_j \left\| \mathbf{v}_j - \mathbf{v}_j^* \right\|_1 \tag{4.6}$$

where $\mathbf{v}_j \in \mathbb{R}^3$ and $\mathbf{v}_j^* \in \mathbb{R}^3$ denote the regression estimate and ground truth scene flow at voxel j, and K is the number of non-empty voxels.

Rigid Motion Loss: The rigid motion loss is defined as the average ℓ_1 error between the predicted translation $\mathbf{t}_j \in \mathbb{R}^2$ and its ground truth $\mathbf{t}_j^* \in \mathbb{R}^2$ in the local coordinate system and the average ℓ_1 error between the predicted rotation θ_j around the Z-axis and its ground truth θ_j^* at every voxel j.

$$\mathcal{L}_{rigmo} = \frac{1}{K} \sum_j \left\| \mathbf{t}_j - \mathbf{t}_j^* \right\|_1 + \lambda \left\| \theta_j - \theta_j^* \right\|_1 \tag{4.7}$$

where λ is a positive constant to balance the relative importance of the two terms. The conversion from world coordinates to local coordinates is given by (see also Eq. 4.4)

$$\mathbf{R}_L = \mathbf{R}_W(\theta_j) \qquad \mathbf{t}_L = (\mathbf{R}_W(\theta_j) - \mathbf{I})\, \mathbf{p}_j + \mathbf{t}_W \tag{4.8}$$

where $\mathbf{p}_j \in \mathbb{R}^2$ specifies the position of voxel j in the XY-plane in world coordinates and $\mathbf{R}_W(\theta_j)$ is the rotation matrix corresponding to rotation θ_j around the Z-axis.

Ego-motion Loss: Similarly, the ego-motion loss is defined as the ℓ_1 distance between the predicted background translation $\mathbf{t}_{BG} \in \mathbb{R}^2$ and its ground truth $\mathbf{t}_{BG}^* \in \mathbb{R}^2$ and the predicted rotation θ_{BG} and its ground truth θ_{BG}^*:

$$\mathcal{L}_{ego} = \|\mathbf{t}_{BG} - \mathbf{t}_{BG}^*\|_1 + \lambda \|\theta_{BG} - \theta_{BG}^*\|_1 \tag{4.9}$$

Detection Loss: Following [ZT18], we define the detection loss as follows:

$$\begin{aligned}\mathcal{L}_{det} =& \frac{1}{M_{pos}} \sum_k \mathcal{L}_{cls}(p_k^{pos}, 1) + \mathcal{L}_{reg}(\mathbf{r}_k, \mathbf{r}_k^*) \\ &+ \frac{1}{M_{neg}} \sum_l \mathcal{L}_{cls}(p_l^{neg}, 0)\end{aligned} \tag{4.10}$$

where p_k^{pos} and p_l^{neg} represent the softmax output for positive proposal boxes a_k^{pos} and negative proposal boxes a_l^{neg}, respectively. $\mathbf{r}_k \in \mathbb{R}^7$ and $\mathbf{r}_k^* \in \mathbb{R}^7$ denote the regression estimates and ground truth residual vectors (translation, rotation and size) for the positive proposal box k, respectively. M_{pos} and M_{neg} represent the number of positive and negative proposal boxes. $\mathcal{L}_{cls}$ denotes the binary cross entropy loss, while $\mathcal{L}_{reg}$ represents the smooth ℓ_1 distance function.

4.2 Experimental Evaluation

We now evaluate the performance of our method on the KITTI object detection dataset [GLU12] as well as an extended version, which we have augmented by simulating virtual objects in each scene.

Eval. Dataset	Training Dataset	Scene Flow (m)			Object Motion		Ego-motion	
		FG	BG	All	Rot.(rad)	Tr.(m)	Rot.(rad)	Tr.(m)
K	K	0.23	**0.14**	**0.14**	**0.004**	0.30	**0.004**	**0.09**
K	K+AK	**0.18**	**0.14**	**0.14**	**0.004**	**0.29**	**0.004**	**0.09**
K+AK	K	0.58	**0.14**	0.18	**0.010**	0.57	**0.004**	0.14
K+AK	K+AK	**0.28**	**0.14**	**0.16**	**0.011**	**0.48**	**0.004**	**0.12**

Table 4.1: **Ablation Study** *on our KITTI and Augmented KITTI validation datasets, abbreviated with K and AK, respectively.*

4.2.1 Datasets

KITTI: For evaluating our approach, we use 61 sequences of the training set in the KITTI object detection dataset [GLU12], containing a total of 20k frames. As there is no pointcloud-based scene flow benchmark in KITTI, we perform our experiments on the original training set. Towards this goal, we split the original training set into 70% train, 10% validation, 20% test sequences, making sure that frames from the same sequence are not used in different splits.

Augmented KITTI: However, the official KITTI object detection datasets lacks cars with a diverse range of motions. To generate more salient training example, we generate a

Eval. Dataset	Method	Scene Flow (m)			Object Motion		Ego-motion	
		FG	BG	All	Rot.(rad)	Tr.(m)	Rot.(rad)	Tr.(m)
K	ICP+Det.	0.56	0.43	0.44	0.22	6.27	**0.004**	0.44
K	3DMatch+Det.	0.89	0.70	0.71	0.021	1.80	**0.004**	0.68
K	FPFH+Det.	3.83	4.24	4.21	0.299	14.23	0.135	4.27
K	Dewan et al.+Det.	0.55	0.41	0.41	0.008	0.55	**0.006**	0.39
K	Ours	**0.29**	**0.15**	**0.16**	**0.004**	**0.19**	**0.005**	**0.12**
K+AK	ICP+Det.	0.74	0.48	0.50	0.226	6.30	**0.005**	0.49
K+AK	3DMatch+Det.	1.14	0.77	0.80	0.027	1.76	**0.004**	0.76
K+AK	FPFH+Det.	4.00	4.39	4.36	0.311	13.54	0.122	4.30
K+AK	Dewan et al.+Det.	0.60	0.52	0.52	0.014	0.75	**0.006**	0.46
K+AK	Ours	**0.34**	**0.18**	**0.20**	**0.011**	**0.50**	**0.005**	**0.15**

Table 4.2: **Comparison to Baselines** *on test sets of KITTI and Augmented KITTI, abbreviated with K and AK, respectively.*

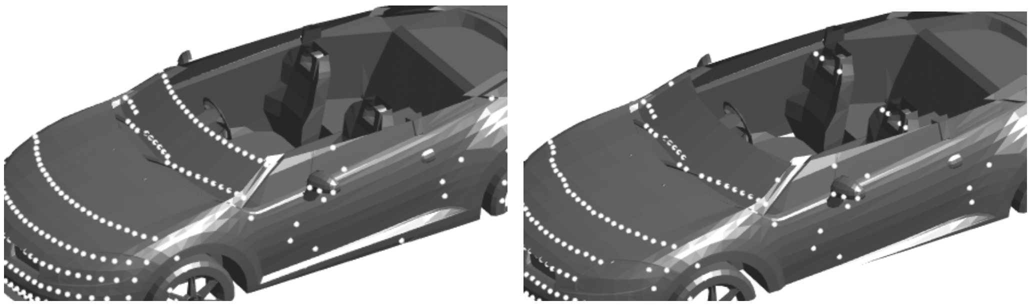


Figure 4.4: **Data Augmentation.** *Simulating LIDAR measurements based on 3D meshes would result in measurements at transparant surfaces such as windows (**left**), wheres a real LIDAR scanner measures interior points instead. Our simulation replicates the behavior of LIDAR scanners by taking into account model transparency and learning the noise model from real KITTI scans (**right**).*

realistic mixed reality LiDAR dataset exploiting a set of high quality 3D CAD models of cars [FDU12] by taking the characteristics of real LIDAR scans into account.

Figure 4.5 describes our workflow for generating the Augmented KITTI. We start by fitting the ground plane using RANSAC 3D plane fitting; this allows us to detect obstacles and hence the drivable region. In a second step, we randomly place virtual cars in the drivable region, and simulate a new LIDAR scan that includes these virtual cars. Our simulator uses a noise model learned from the real KITTI scanner by fitting a Gaussian distribution conditioned on the horizontal and vertical angle of the rays, based on KITTI LIDAR scans. Our simulator also produces missing estimates at transparent surfaces by ignoring them with a probability equal to their transparency value provided by the CAD models, as illustrated in Figure 4.4. Additionally, we remove points in the original scan which become occluded by the augmented car by tracing a ray between each point and the

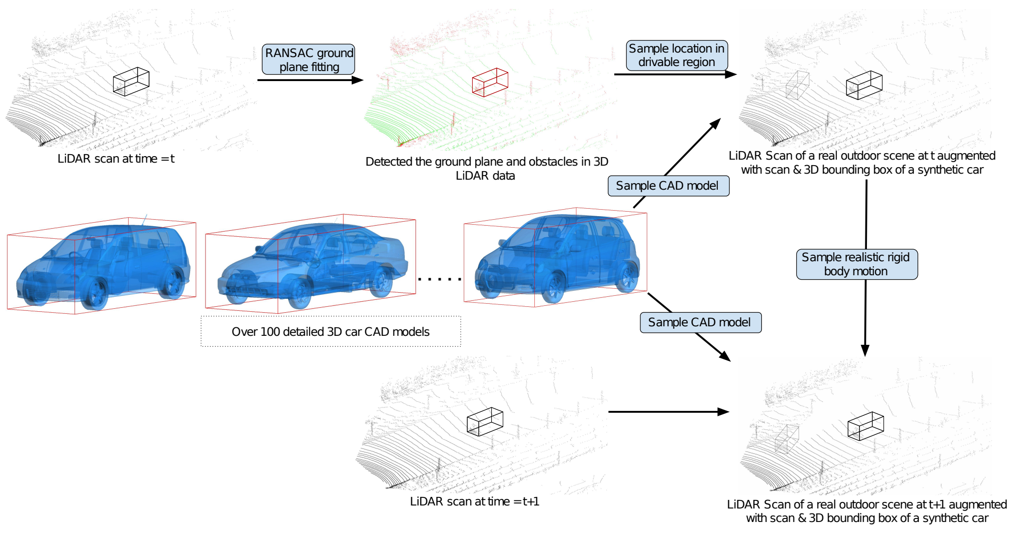

LiDAR scan at time =t
RANSAC ground plane fitting
Detected the ground plane and obstacles in 3D LiDAR data
Sample location in drivable region
LiDAR Scan of a real outdoor scene at t augmented with scan & 3D bounding box of a synthetic car
Sample CAD model
Over 100 detailed 3D car CAD models
Sample realistic rigid body motion
Sample CAD model
LiDAR scan at time =t+1
LiDAR Scan of a real outdoor scene at t+1 augmented with scan & 3D bounding box of a synthetic car

Figure 4.5: **Augmented KITTI.** *Workflow for generating the Augmented KITTI dataset.*

LIDAR, and removing those points whose ray intersects with the car mesh. Finally, we sample the augmented car's rigid motion using a simple approximation of the Ackermann steering geometry, place the car at the corresponding location in the next frame, and repeat the LIDAR simulation. We generate 20k such frames with 1 to 3 augmented moving cars per scene. We split the sequences into 70% train, 10% validation, 20% test similar to our split of the original KITTI dataset.

4.2.2 Baseline Methods

We compare our method to four baselines: a point cloud-based method using a CRF [Dew+16b], two point-matching methods, and an Iterative Closest Point [BM92] (ICP) baseline.

Dewan et al. [Dew+16b] estimate per-point rigid motion. To arrive at object-level motion and ego-motion, we pool the estimates over our object detections and over the background. As they only estimate valid scene flow for a subset of the points, we evaluate [Dew+16b] only on those estimates and the comparison is therefore inherently biased in their favor.

Method Matching 3D Descriptors yield a scene flow estimate for each point in the reference point cloud by finding correspondences of 3D features in two timesteps. We evaluate two different descriptors: 3D Match [Zen+17], a learnable 3D descriptor trained on KITTI and Fast Point Feature Histogram features (FPFH) [RBB09]. Based on the per-point scene flow, we fit rigid body motions to each of the objects and to the background, again using the object detections from our pipeline for a fair comparison.

Iterative Closest Point (ICP) [BM92] outputs a transformation relating two point clouds to each other using an SVD-based point-to-point algorithm. We estimate object rigid motions by fitting the points of each detected 3D object in the first point cloud to the entire second point cloud.

Evaluation Metrics: We quantify performance using several metrics applied to both the detected objects and the background. To quantify the accuracy of the estimates independently from the detection accuracy, we only evaluate object motion on true positive detections.

- For 3D scene flow, we use the average endpoint error between the prediction and the ground truth.
- Similarly, we list the average rotation and translation error averaged over all of the detected objects, and averaged over all scenes for the observer's ego-motion.

4.2.3 Experimental Results

The importance of simulated augmentation: To quantify the value of our proposed LIDAR simulator for realistic augmentation with extra cars, we compare the performance of our method trained on the original KITTI object detection dataset with our method trained on both KITTI and Augmented KITTI. Table 4.1 shows the results of this study. Our analysis shows that training using a combination of KITTI and augmented KITTI leads to significant performance gains, especially when evaluating on the more diverse vehicle motions in the validation set of Augmented KITTI.

Comparison with the baselines: Table 4.2 summarizes the complete performance comparison on the KITTI test set. Note that the comparison with Dewan et al. [Dew+16b] is biased in their favor, as mentioned earlier, as we only evaluate their accuracy on the points they consider accurate. Regardless, our method outperforms all baselines. Additionally, we observe that the ICP-based method exhibits large errors for object motions. This is because of objects with few points: ICP often performs very poorly on these, but while their impact on the dense evaluation is small they constitute a relatively larger fraction of the object-based evaluation. Figures 4.6 - 4.13 show qualitative comparison of our method with the best performing baseline methods on examples from the test set of the Augmented KITTI dataset. For clarity, we visualize only a subset of the points. The qualitative results show that our method predicts motion for both background and foreground parts of the scene with higher accuracy than all the baselines on a diverse range of scenes and motions.

Regarding execution time, our method requires 0.5 seconds to process one point cloud pair. In comparison, Dewan et al. (4 seconds) and the 3D Match- and FPFH-based approaches (100 and 300 seconds, respectively) require significantly longer, while the ICP solution also takes 0.5 seconds but performs considerably worse.

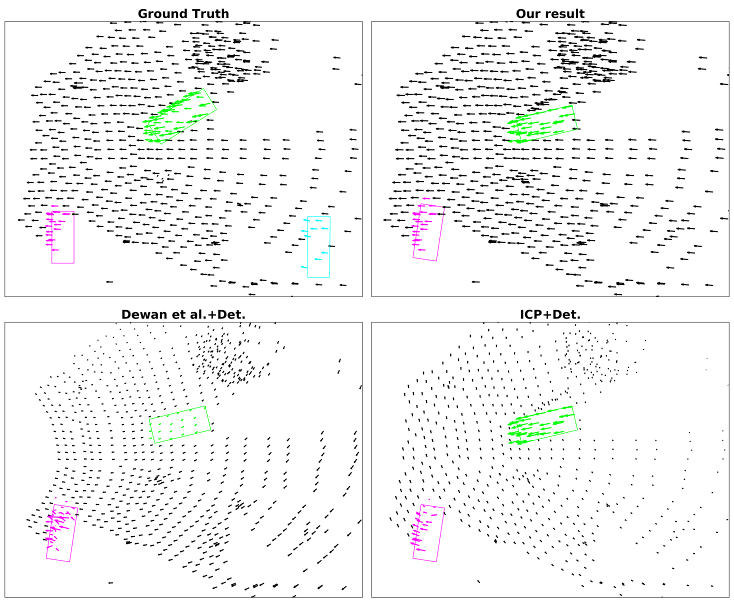


Figure 4.6: **Qualitative Comparison** *of our method with the best performing baseline methods on an example from the test set of the Augmented KITTI dataset.*

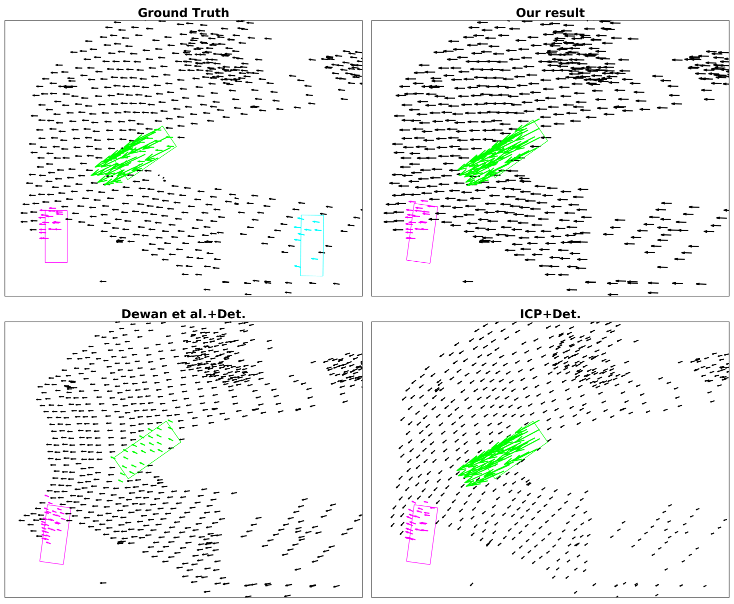


Figure 4.7: **Qualitative Comparison** *of our method with the best performing baseline methods on an example from the test set of the Augmented KITTI dataset.*

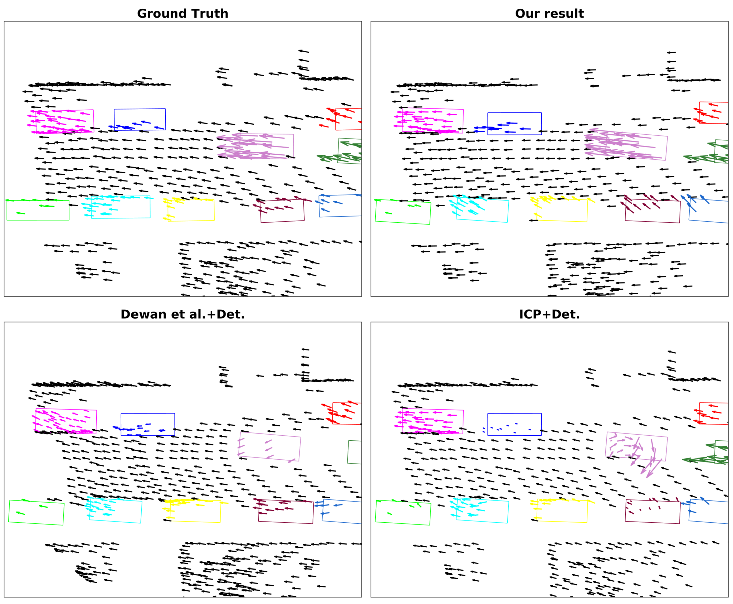


Figure 4.8: **Qualitative Comparison** *of our method with the best performing baseline methods on an example from the test set of the Augmented KITTI dataset.*

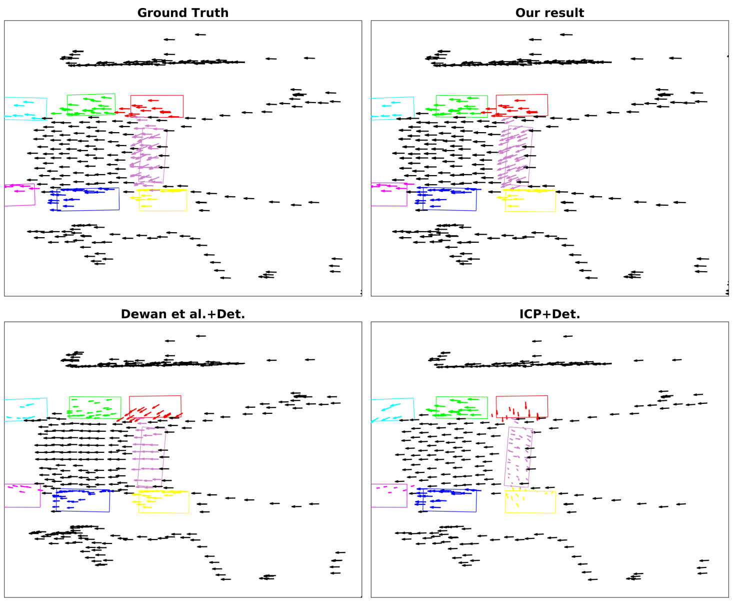


Figure 4.9: **Qualitative Comparison** *of our method with the best performing baseline methods on an example from the test set of the Augmented KITTI dataset.*

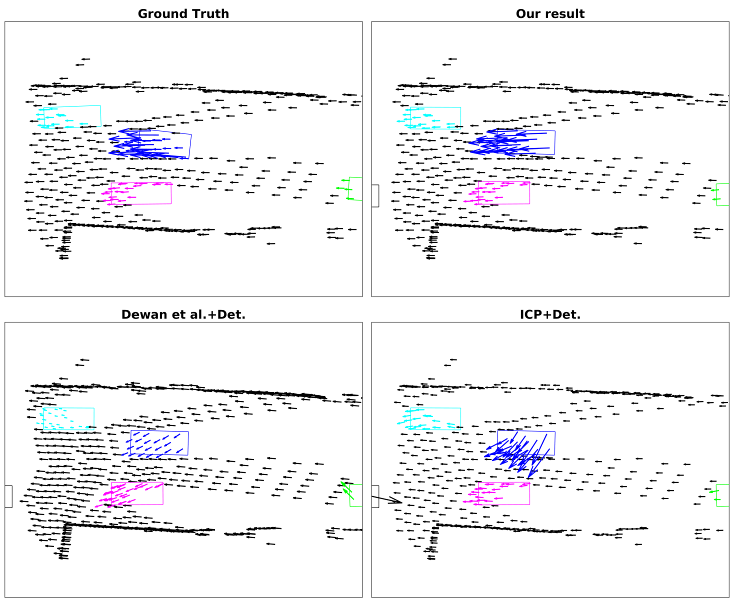


Figure 4.10: **Qualitative Comparison** *of our method with the best performing baseline methods on an example from the test set of the Augmented KITTI dataset.*

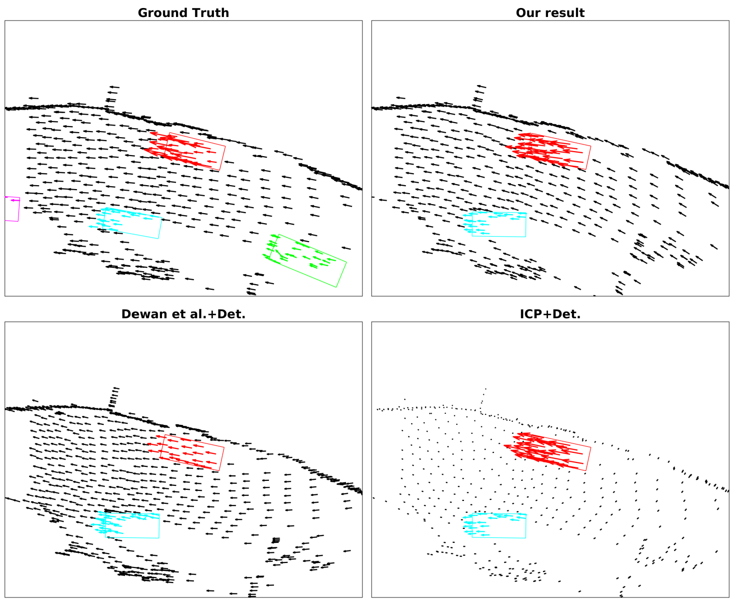


Figure 4.11: **Qualitative Comparison** *of our method with the best performing baseline methods on an example from the test set of the Augmented KITTI dataset.*

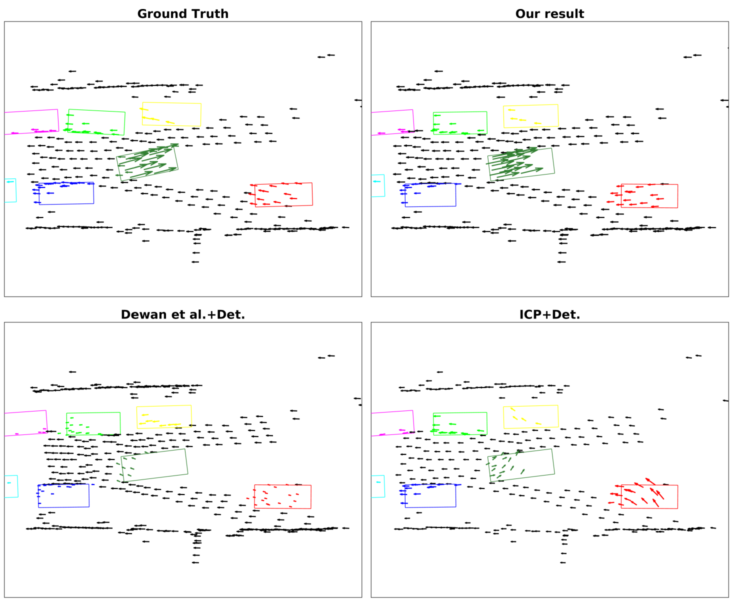


Figure 4.12: **Qualitative Comparison** *of our method with the best performing baseline methods on an example from the test set of the Augmented KITTI dataset.*

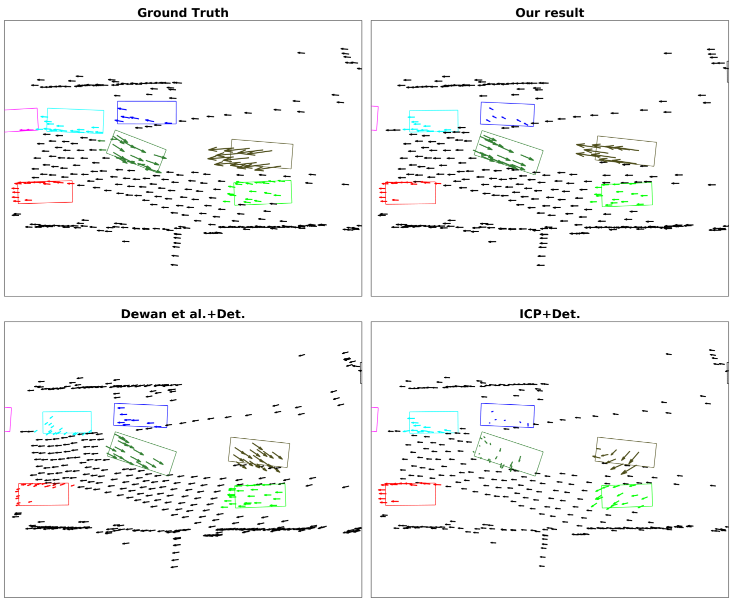


Figure 4.13: **Qualitative Comparison** *of our method with the best performing baseline methods on an example from the test set of the Augmented KITTI dataset.*

5 Label Efficient Visual Abstractions for Autonomous Driving

In chapter 3 and chapter 4, we proposed to exploit recognition cues towards improving performance of 3D scene flow methods. Typically, the scene flow estimation is further employed in the modular autonomous driving pipeline to estimate the future positions of other traffic actors and thus allows for safe planning of self driving vehicle motion. In this chapter, we consider an alternative to modular pipelines, that is, end-to-end learning-based models which try to learn a policy, i.e. a function from observations to actions using a generic model such as a deep neural network. The network parameters of end-to-end driving methods are typically learned via behavior cloning by replicating the behavior of a teacher using expert demonstrations. However, these methods for end-to-end learned driving models working directly with raw image sensor observations fail to generalize to different visual conditions like change in weather or city neighborhood.

Therefore, there has been a surge in interest on using visual priors for learning end-to-end control policies with improved performance, generalization and sample efficiency [ZKK19; Mou+19; Sax+19]. Instead of learning to act directly from image observations, which is challenging due to the high-dimensional input, a visual prior is enforced by decomposing the task into intermediate sub-tasks. These intermediate visual sub-tasks, e.g., object detection, segmentation, depth and motion estimation, optimized independently, are then fed as an input to a policy learning algorithm, e.g., [Sax+19]. A useful visual prior needs to encode the right assumptions about the environment in order to simplify policy learning. In the case of autonomous driving, semantic segmentation encodes the fact that certain pixels in an image can be treated similarly: e.g. the agent can drive on roads but not sidewalks; the agent must not collide with other vehicles or pedestrians. Semantic segmentation has also been shown to act as a powerful visual prior for driving agents [SS16; Mül+18]. While beneficial for learning robust policies, such intermediate sub-tasks require explicit supervision in the form of additional manual annotations. For several visual priors, obtaining these annotations can be time consuming, tedious, and prohibitive.

In parallel, significant research effort has been devoted into semantic segmentation of street scenes in recent years, where images from a camera sensor mounted on a vehicle are segmented into classes such as road, sidewalk, and pedestrian [BFC09; Cor+16; Yu+18; Hua+18; GLU12]. It is widely believed that this kind of accurate scene understanding is key for robust self-driving vehicles. Existing state-of-the-art methods [Zha+17] optimize for image-level metrics such as mIoU, which is challenging as it requires a combination of coarse contextual reasoning and fine pixel-level accuracy [Far+13]. The emphasis on such image-level requirements has resulted in large segmentation benchmarks, i.e., thousands of images, with high labeling costs. However, the development of such benchmarks, in terms of annotation type and cost, is often done independently of the actual driving task which encompasses optimizing metrics such as distance traveled per human intervention.

This motivates our study of *visual abstractions*, which are compact semantic segmentation-based representations of the scene with fewer classes, coarser annotation, and learned with little supervision (only several hundred images). We consider the following question: when used as visual priors for policy learning, are representations obtained from datasets with lower annotation costs competitive in terms of driving ability? In particular, we seek to quantify the impact of reducing segmentation annotation costs on learned behavior cloning agents. We analyze several segmentation-based intermediate representations. We use these *visual abstractions* to systematically study the trade-off between annotation efficiency and driving performance, i.e., the types of classes labeled, the number of image samples used to learn the visual abstraction model, and their granularity (e.g., object masks vs. 2D bounding boxes). Our analysis uncovers several practical insights into how segmentation-based visual abstractions can be exploited in a more label efficient manner. Surprisingly, we find that state-of-the-art driving performance can be achieved with orders of magnitude reduction in annotation cost. Beyond label efficiency, we find several additional training benefits when leveraging visual abstractions, such as a significant reduction in the variance of the learned policy when compared to state-of-the-art end-to-end driving models. Towards addressing this research question, we systematically analyze the performance of varying intermediate representations on the recent NoCrash benchmark of the CARLA urban driving simulator [Dos+17; Cod+19]. Our analysis uncovers several new results regarding label efficient representations.

5.1 Method

As illustrated in Fig. 5.1, we consider a modular approach that comprises two learned mappings, one from the RGB image to a semantic label map and one from the semantic label map to control. To learn these mappings, we use two image-based datasets, (i) $S = \{\mathbf{x}^i, \mathbf{s}^i\}_{i=1}^{n_s}$ which consists of n_s images annotated with semantic labels, and (ii) $C = \{\mathbf{x}^i, \mathbf{c}^i\}_{i=1}^{n_c}$ which consists of n_c images annotated with expert driving controls. First, we train the parameters of a visual abstraction model a_ϕ parameterized by ϕ using the segmentation dataset S. The trained visual abstraction stack is then applied to transform C resulting in a control dataset $C_\phi = \{a_\phi(\mathbf{x}^i), \mathbf{c}^i\}_{i=1}^{n_c}$ on which we train a driving policy π_θ with parameters θ. At test time, control values are obtained for an image $\mathbf{x}^*$ by composing the two learned mappings, $\mathbf{c}^* = \pi_\theta(a_\phi(\mathbf{x}^*))$.

In this section, we discuss the core questions we aim to answer, followed by a description of the visual abstractions and driving agent considered in our study.

5.1.1 Research Questions

We aim to build a segmentation dataset S that is cost-effective, yet encodes all relevant information for policy learning. We are interested in the following questions:

Can selecting specific classes ease policy learning? A semantic segmentation $\mathbf{s}$ assigns

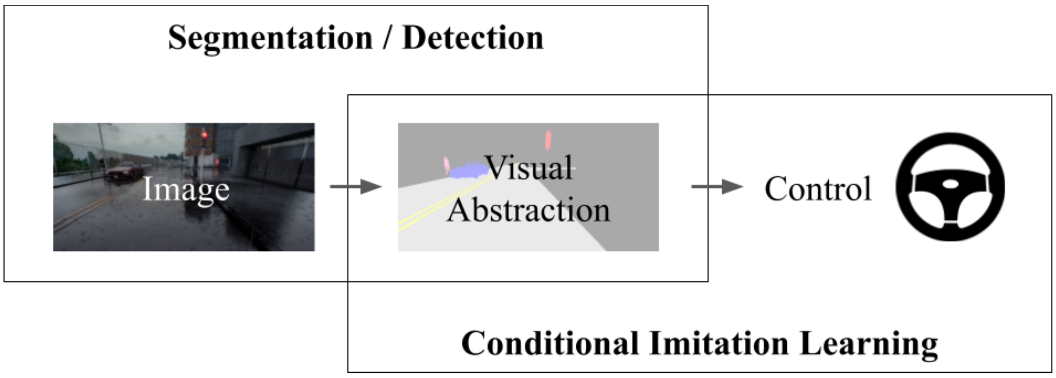


Figure 5.1: High-level overview of the proposed study. *We investigate different segmentation-based visual abstractions by pairing them with a conditional imitation learning framework for autonomous driving.*

each pixel to a discrete category $k \in \{1,\ldots,K\}$. Knowing whether a pixel belongs to the building class or tree class may provide no additional information to a driving agent, if it knows that the pixel does not belong to the road, vehicle or pedestrian class. We are interested in understanding the impact of the set of categories on the driving task.

Are semantic representations trained with few images competitive? In a policy learning setting, the training of the driving agent may be able to automatically compensate for some drop in performance of the segmentation model. We aim to determine if a parsimonious training dataset obtained by reducing the number of training images n_s for the segmentation model can achieve satisfactory performance.

Is fine-grained annotation important? Exploiting coarse annotations such as 2D bounding boxes instead of pixel-accurate segmentation masks can alleviate the key challenge in building segmentation models: annotation costs [Zla+18]. If fine-grained annotation can be avoided, we are interested in how to select a_ϕ to exploit coarse annotation during training.

Are visual abstractions able to reduce the variance which is typically observed when training agents using behavior cloning? Significant difference in performance of behavior cloning policies is caused as a result of changing the training seed or the sampling of the training data [Cod+19]. This is problematic in the context of autonomous driving where evaluating an agent is expensive and time-consuming, making it difficult to assess if changes in performance are a result of algorithmic improvements or random training seeds. Since visual priors abstract out certain aspects of the input such as illumination and weather, we are interested in investigating their effect on reducing the variance in policies with different random training seeds.

5.1.2 Visual Abstractions

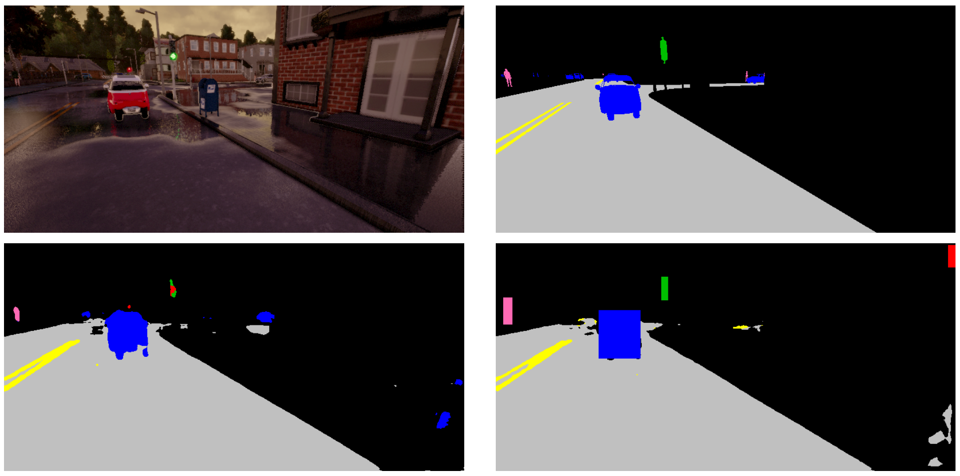


Figure 5.2: Visual Abstractions. *For our analysis, we consider three visual abstractions based on semantic segmentation. From top left in clockwise order, we illustrate the camera image, Privileged Segmentation, Standard Segmentation, and Hybrid Detection and Segmentation based visual abstractions. We show the comparison for 6 semantic class labels.*

For our analysis, we consider three visual abstractions based on semantic segmentation as illustrated in Fig. 5.2.

Privileged Segmentation: As an upper bound, the ground-truth semantic labels (available from the simulator) can be used directly as an input to the driving agent. This form of privileged information is useful for ablative analysis.

Standard Segmentation: For standard pixel-wise segmentation over all classes, our perception stack is based on a ResNet and Feature Pyramid Network (FPN) backbone [He+16; Lin+17a], with a fully-convolutional segmentation head [LSD15].

Hybrid Detection and Segmentation: To exploit coarser annotation, we use a hybrid architecture that distinguishes between stuff and thing classes [HK08]. The architecture consists of a segmentation model trained on stuff classes annotated with semantic labels (e.g. road, lane marking) and a detection model based on Faster-RCNN [Ren+15] trained on thing classes annotated with bounding boxes (e.g. pedestrian, vehicle). The final visual abstraction per pixel is obtained by overlaying the pixels of detected bounding boxes on top of the predicted segmentation, based on a pre-defined class priority. Similar hybrid architectures have been found useful previously in the urban street scene semantic segmentation setting, since detectors typically have different inductive biases than segmentation networks [Sal+18].

5.1.3 Driving Agent

Conditional Imitation Learning: In general, behavior cloning involves a supervised learning method which is used to learn a mapping from observations to expert actions. However, sensor input alone is not always sufficient to infer optimal control. For example, at intersections, whether the car should turn or keep straight cannot be inferred from camera images alone without conditioning on the goal. We therefore follow [Cod+18; Cod+19] and condition on a navigational command. The navigational command represents driver intentions such as the direction to follow at the next intersection. Our agent is a neural network π_θ with parameters θ, which maps a semantic representation $\mathbf{s} \in \mathcal{S}$, navigational command $\mathbf{n} \in \mathcal{N}$, and measured velocity $v \in \mathbb{R}^+$ to a control value $\mathbf{c} \in \mathcal{C}$:

$$\pi_\theta : \mathcal{S} \times \mathcal{N} \times \mathbb{R}^+ \to \mathcal{C} \tag{5.1}$$

Imitation Loss: In order to learn the policy parameters θ, we minimize an imitation loss $\mathcal{L}_{\text{imitation}}$ defined as follows:

$$\mathcal{L}_{\text{imitation}} = ||\mathbf{c} - \hat{\mathbf{c}}||_1 \tag{5.2}$$

Here, $\hat{\mathbf{c}} = \pi_\theta(\mathbf{s}, \mathbf{n}, v)$ is the control value predicted by our agent and $|| \cdot ||_1$ denotes the L_1 norm.

Velocity Loss: Recordings of expert drivers have an inherent inertia bias, where most of the samples with low velocity also have low acceleration. It is critical to not overly correlate these since the vehicle would prefer to never start after slowing down. As demonstrated in [Cod+19], predicting the current vehicle velocity as auxiliary task can alleviate this issue. We thus also use a velocity prediction loss:

$$\mathcal{L}_{\text{velocity}} = ||v - \hat{v}||_1 \tag{5.3}$$

Architecture: The architecture used for our driving agent is based on the CILRS model [Cod+19], summarized in Fig. 5.3. The visual abstraction is initially processed by an embedding branch, which typically consists of several convolutional layers. We flatten the output of the embedding branch and combine it with the measured vehicle velocity v using fully-connected layers. Since the space of navigational commands is typically discrete for driving, we use a conditional module to select one of several command branches based on the input command. The command branch outputs control values. Additionally, the output of the embedding branch is used for predicting the current vehicle speed, which is compared to the actual vehicle speed in the velocity loss $\mathcal{L}_{\text{velocity}}$ defined in Eq. 5.3. The final loss function for training is a weighted sum of the two components, with a scalar weight λ:

$$\mathcal{L} = \mathcal{L}_{\text{imitiation}} + \lambda \mathcal{L}_{\text{velocity}} \tag{5.4}$$

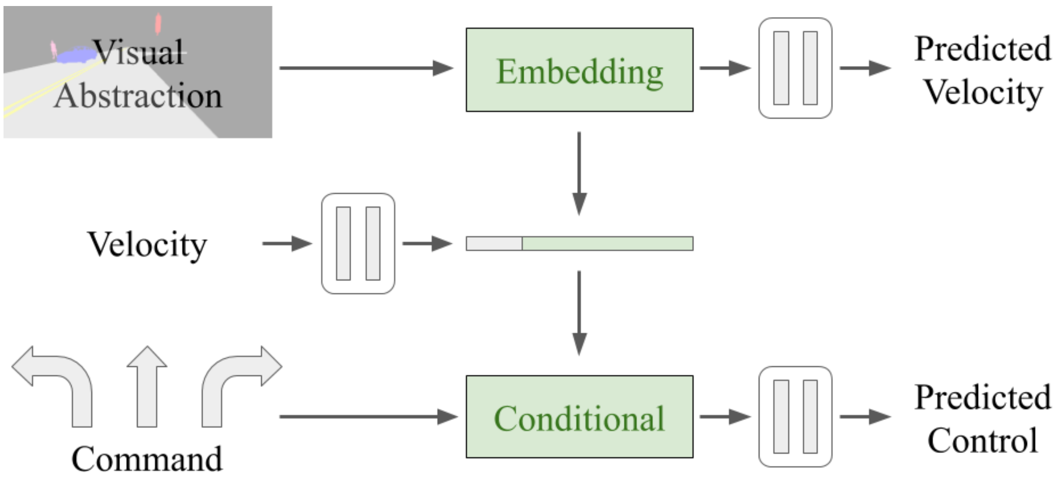


Figure 5.3: Driving agent architecture. *Given a segmentation-based visual abstraction, current vehicle velocity, and discrete navigational command, the CILRS model predicts a control value [Cod+19].*

5.2 Experiments

In this section, we present a series of experiments on the open-source driving simulator CARLA [Dos+17] to answer the questions raised in Section 5.1.1. We perform our analysis by training and evaluating driving agents on the NoCrash benchmark of CARLA (version 0.8.4) [Cod+19].

5.2.1 Task

The CARLA simulation environment consists of two map layouts, called Town 01 & Town 02, which can be augmented by 14 weather conditions. We use the provided ‘autopilot’ expert mode for data collection. All training and validation data is collected in Town 01 with the four weather conditions specified as the ‘train’ conditions in the NoCrash benchmark. The number of external agents is uniformly sampled from the range $[80, 160]$. Town 02 is reserved for testing.

Evaluation: The primary evaluation criterion in CARLA is the percentage of successfully completed episodes during an evaluation phase, referred to as Success Rate (SR). Successful navigation requires driving the vehicle from a starting position to a specified destination within a fixed time limit. Failure can be a result of collision with pedestrians, vehicles or static objects; or inability to reach the destination within the time limit (timeout). The benchmark consists of three levels of traffic density: Empty, Regular and Dense involving 0, 65 and 220 external agents respectively. We perform evaluation in Town 02 for two ‘test’ weather conditions of the NoCrash benchmark that are unseen during training.

5.2.2 Datasets

Table 5.1: Summary of control datasets. *The third column indicates the number of labeled images used for training our detection and segmentation models. The fourth column indicates the approximate cost of annotating these images.*

Name	Classes	Labeled Images	Cost (Hours)
Standard-Large-14	14	6400	7500
Standard-Large	6	6400	3200
Standard	6	1600	800
Standard-Small	6	400	200
Hybrid	6	1600	50

Perception: We collect a training set of 6400 images from Town 01 for training our detection and segmentation models. The images are annotated with 2D semantic labels for 14 classes, and 2D object boxes for 4 of these classes. These annotations are provided by the CARLA simulator.

Control: Following [Cod+19], we collect approximately 10 hours of driving frames and corresponding expert controls for imitation learning using the autopilot of the CARLA simulator. The images are sampled from three cameras facing different directions (left, center and right) on the car at 10 frames per second. We create five variants of this control dataset (summarized in Table 5.1) by transforming the input RGB images to visual abstractions. The Standard-Large-14 dataset is generated using a segmentation network trained to segment the input into fourteen semantic classes: 'road', 'lane marking', 'vehicle', 'pedestrian', 'green light', 'red light', 'sidewalk', 'building', 'fence', 'pole','vegetation', 'wall', 'traffic sign' and 'other'. The remaining datasets, namely Standard-Large, Standard, Standard-Small, and Hybrid, use a reduced set of six classes: 'road', 'lane marking', 'vehicle', 'pedestrian', 'green light' and 'red light'. While the Standard datasets are generated using a segmentation network, the Hybrid dataset is generated using the combination of a 2-class segmentation model and 4-class detection model.

The study of [Zla+18] provides approximate labeling costs based on a labeling error threshold, for a dataset of similar visual detail as ours. The annotation time reported for the Standard abstraction (fine labeling) is around 300 seconds per image and per class. Our Hybrid visual abstraction roughly corresponds to a 32-pixel labeling error, which requires approximately 20 seconds per image and per class. Based on these statistics, we include the estimated annotation time for each visual abstraction in Table 5.1.

In addition, we also collect a Privileged dataset comprising the ground truth semantic segmentation for the input, which we use for ablation studies involving privileged agents.

5.2.3 Implementation Details

Our perception models are based on a ResNet-50 FPN backbone pre-trained on the MS-COCO dataset [Lin+14]. We finetune this model for 3k iterations on the perception dataset with the hyper-parameters set to the default of the Detectron2[1] detection and segmentation algorithms.

The driving agents use a ResNet-18 model in the 'embedding' branch (see Fig. 5.1). We process the velocity input with two fully-connected layers of 128 units each, which is combined with the ResNet output in another fully-connected layer of 512 units. The velocity prediction branch and each command branch encompass two fully-connected layers of 256 units. We train each model from scratch for 200k iterations using the default training hyper-parameters of the COiLTRAiNE[2] framework. This is the standardized repository used to train imitation learning agents on CARLA [Cod+19; Zha+19], which currently supports CARLA version 0.8.4.

5.2.4 Results

Identifying Most Relevant Classes: In our first experiment, our goal is to analyze the impact of training policies with a reduced set of annotated classes. For this, we train and evaluate agents using the Privileged dataset. As a baseline, we use a semantic representation consisting of all fourteen classes in the dataset. We then evaluate a reduced subset of seven classes that we hypothesize to be the most relevant: 'road', 'sidewalk', 'lane marking', 'vehicle', 'pedestrian', 'green light' and 'red light'. Next, we evaluate representations that consist of six and five classes, by excluding 'sidewalk' and then 'lane marking' from the seven-class representation. For the five-class representation, we re-label 'lane marking' as 'road'. The driving performance for these representations is summarized in Fig. 5.4. We show the percentage of evaluation episodes in the test environment where the agent succeeded, collided with an obstacle or timed out.

We note from our results that perfect segmentation accuracy does not mean perfect overall perception. The fourteen-class model does not achieve perfect driving in any of the three traffic conditions. Even with perfect perception, limitations of using behavior cloning methods such as covariate shift, where the states encountered at train time differ from those encountered at test time, can lead to non-optimal driving behavior. Further, the higher relative dimensionality when using fourteen classes, which includes fine details of classes such as fences, buildings, and vegetation, makes it harder for the agent to identify the right features important for generalization. This is reflected by the fact that the seven-class representation outperforms the agent based on fourteen classes in all three traffic conditions. We empirically observe that the fourteen-class agent is more conservative in its driving style, and more susceptible to timeouts.

[1] `https://github.com/facebookresearch/detectron2`
[2] `https://github.com/felipecode/coiltraine`

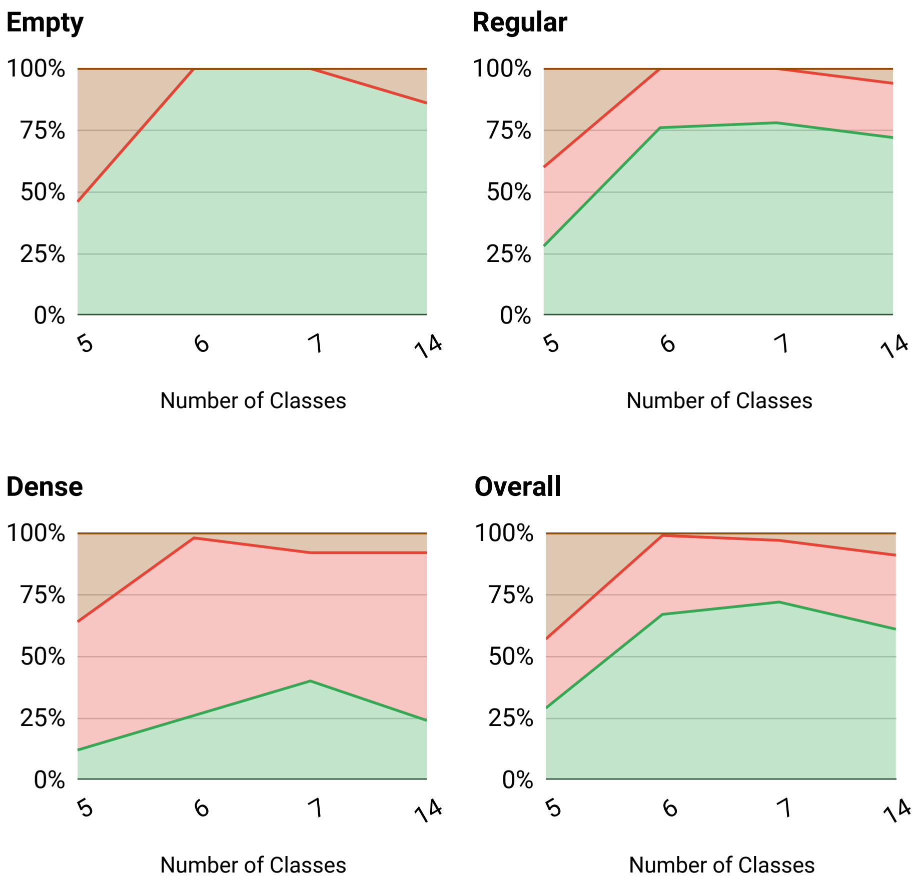


Figure 5.4: ***Identifying most relevant classes.*** *Success/collision/timeout percentages on the test environment (Town 02 Test Weather) of the CARLA NoCrash benchmark. For this ablation study, we use ground truth segmentation as inputs to the behavior cloning agent. Reduction from fourteen to seven or six classes leads to a slight increase in success rate, but further reduction to five classes leads to a large number of failures.*

The six-class representation that excludes sidewalk segmentation achieves similar performance to seven classes in empty and regular traffic. We therefore additionally compare the six-class and fourteen-class representations using inferred visual abstractions without privileged information, in order to analyze if the same trends observed in Fig. 5.4 hold. Specifically, we compare the Standard-Large-14 and Standard-Large datasets as described in Table 5.1. These datasets are generated using fourteen-class and six-class segmentation networks respectively. Example visual abstractions for fourteen-class and six-class segmentation privileged and standard agents are illustrated in Fig. 5.5. The success rates of these trained models are shown in Fig. 5.6. Additionally, we show the performance of the corresponding six-class and fourteen-class Privileged agents for reference. We observe that the six-class representation consistently maintains or improves upon the performance of agents trained using all fourteen classes. The six-class approach helps to reduce annotation costs by removing the requirement of assigning labels to several classes such as poles and vegetation, which can be time-consuming due to thin structures with a lot of fine detail.

Interestingly, we observe from Fig. 5.4 that using only five classes leads to a significant reduction in performance, with the overall success rate dropping from 67% to 29%. This drastic change indicates that the lane marking class is of very high importance for learning driving policies, and the task becomes hard to solve without this class even with perfect segmentation accuracy on all other classes. Based on the consistent performance of the six-class visual abstraction in both Fig. 5.4 and Fig. 5.6, we choose this representation to perform a more detailed analysis of trade-offs related to labeling quality and quantity.

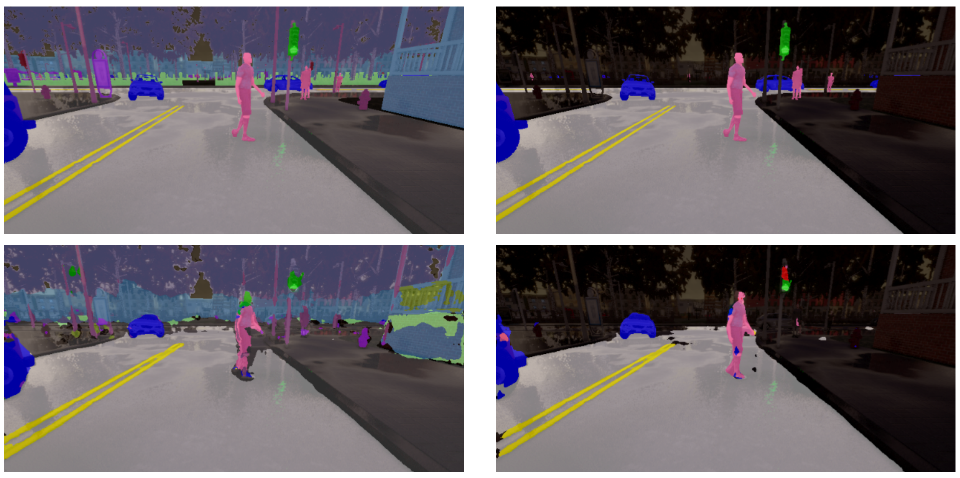


Figure 5.5: Identifying Most Relevant Classes. *In the top and bottom rows, we illustrate the Privileged Segmentation and Standard Segmentation respectively for fourteen (left) and six (right) semantic classes.*

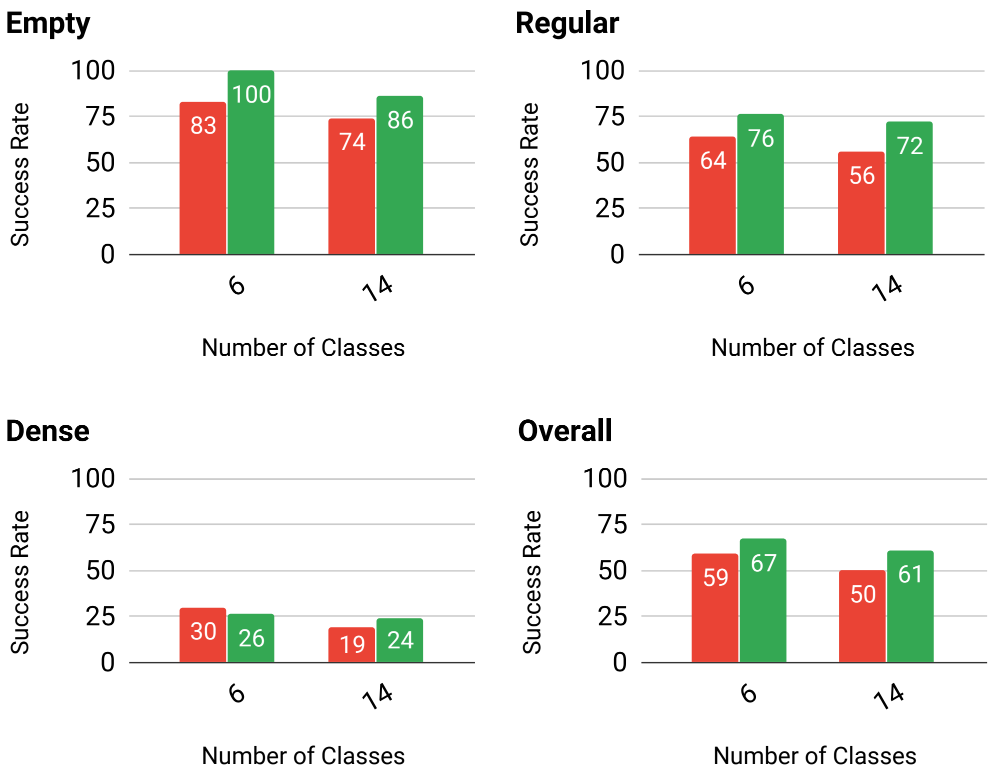


Figure 5.6: Evaluating the six-class representation. *Success Rate (Privileged/Standard) on the test environment (Town 02 Test Weather) of the CARLA NoCrash benchmark. The six-class representation consistently leads to better performance than the fourteen-class representation while simultaneously having lower annotation costs.*

Number of Annotated Images: In our second experiment, we study the impact of reducing annotation quantity by training agents using the Standard-Large, Standard, and Standard-Small datasets from Table 5.1. Reducing from Standard-Large to Standard-Small, each dataset has 4 times less samples (and therefore 4 times less labeling cost) than the preceding one. Our results, presented as the mean success, collision and timeout percentages over 5 different training seeds for the behavior cloning agent, are summarized in Fig. 5.7.

We observe no significant differences in overall downstream driving task performance between the agents trained on 6400 or 1600 samples, and a slight drop when using 400 images. Taking a closer look at the driving performance, we observe that the number of collisions in dense traffic is slightly lower for 6400 samples, but success rate is also slightly decreased on empty conditions. This shows that for our task, when focusing on only the most salient classes, a few hundred images are sufficient to obtain robust driving policies. In contrast, prior work that exploits semantic information on CARLA uses fine-grained annotation for several hours of driving data (up to millions of images) [Zha+19].

From Fig. 5.7, we clearly observe a saturation in overall performance beyond 1600 training samples. We therefore study the impact of granularity of annotation in more detail while fixing the dataset size to 1600 samples.

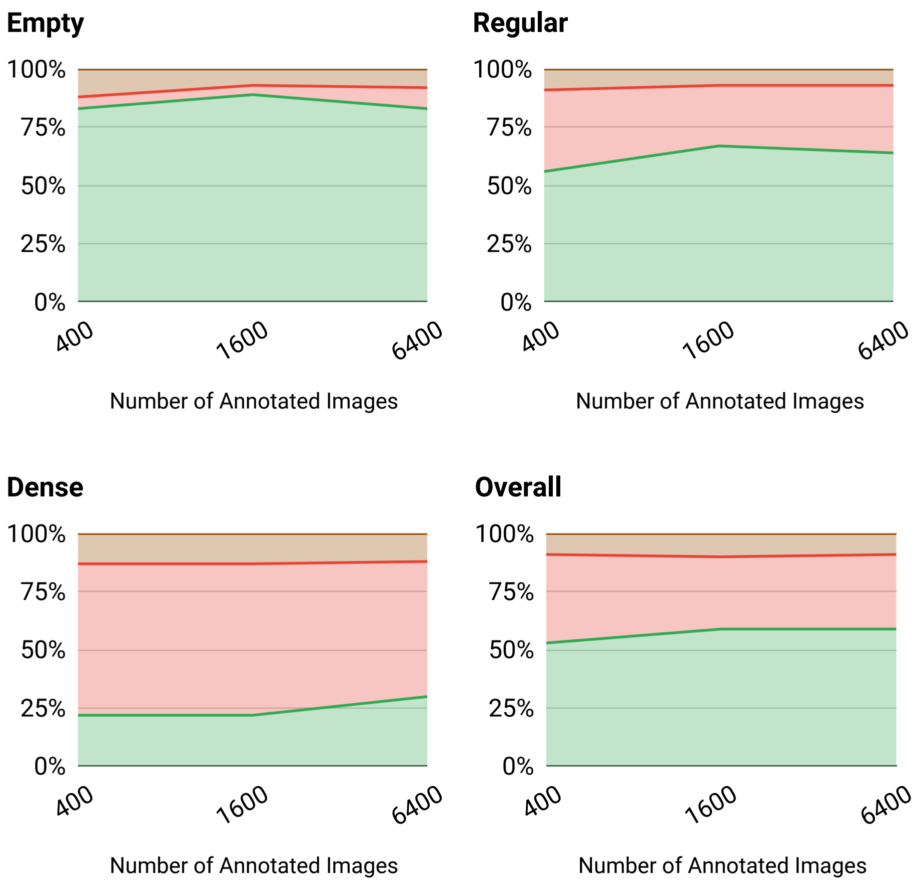


Figure 5.7: ***Comparing visual abstractions as annotation quantity is reduced.*** *Success/collision/timeout percentages on the test environment (Town 02 Test Weather). Mean over 5 random training seeds. Performance remains consistent with 6400 or 1600 annotated images, with a slight drop as the training dataset for the visual abstraction is reduced to 400 images.*

Coarse Annotation: In our third experiment, we analyze the impact of using the Hybrid visual abstraction that utilizes coarse supervision during training, requiring only an estimated 50 hours to annotate. We present a comparison of the Hybrid and Standard visual abstractions in Fig. 5.8. The results are presented as the mean driving success rate of five training seeds. We find the Hybrid abstraction to improve performance on tasks involving external dynamic agents, i.e., regular and dense settings. Since objects are input as rectangles, without additional noise introduced by a standard segmentation network, we hypothesize that Hybrid abstractions are able to simplify policy learning (see Fig. 1.6). Overall, the performance of the Hybrid agent is on par with that of the Standard agent despite having approximately 15 times lower annotation costs (see Table 5.1).

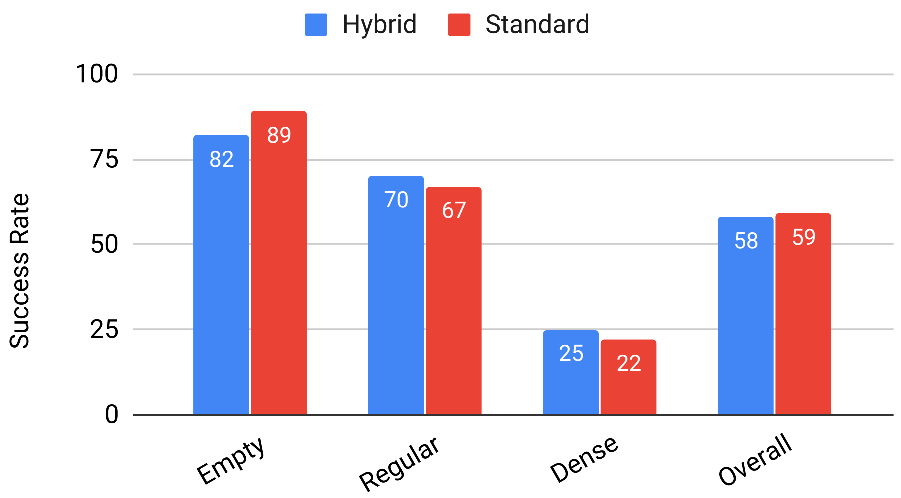


Figure 5.8: Evaluating the Hybrid visual abstraction. *Success rate on the test environment (Town 02 Test Weather) as the quality of annotation is reduced. Mean over 5 random training seeds. Overall, the performance of the Hybrid abstraction matches Standard segmentation despite having a reduction in annotation costs of several orders of magnitude.*

Variance Between Training Runs: In our fourth experiment, we investigate the impact of semantic representations on the variance between results for different training runs. In order to conduct a fair comparison, we use the same raw dataset for training all the models in this study. This ensures that there is no variance caused by the training data distribution. Similar to [Cod+19], the raw training data was collected by the standard CARLA data-collector framework[3].

We compare our approach to CILRS [Cod+19], which uses the weights of a network pre-trained on ImageNet [Rus+15] to reduce variance due to random initialization of the policy parameters. The only remaining source of variance in training is the random sampling of data mini-batches that occurs during stochastic gradient descent. However, existing studies have still reported high variance in CILRS models between training runs [Cod+19]. For our approach, we choose the agent trained with the Hybrid abstraction. The results of five different training seeds along with the mean, standard deviation and coefficient of variation (standard deviation normalized by the mean) for each method are shown in Table 5.2.

We observe that for CILRS, the best training seed has double the average success rate of the worst training seed, leading to extremely large variance. In particular, on dense traffic conditions, the success rate ranges from 0 to 18. This amount of variance is problematic when trying to analyze or compare different approaches. In contrast, there is less variance

[3] `https://github.com/carla-simulator/data-collector`

Table 5.2: Variance between random training seeds. *Percentage Success Rate on Town 02 Test Weather for five training seeds on Empty (E), Regular (R), and Dense (D) conditions, as well as the average Overall (O) success rate. Max and min values indicated in* ***bold****. Our approach significantly reduces the standard deviation and coefficient of variation (CV).*

Task	Seed 1	Seed 2	Seed 3	Seed 4	Seed 5	Mean ↑	Std ↓	CV ↓
				CILRS [Cod+19]				
E	**26**	44	42	**48**	46	41.20	8.79	0.21
R	**24**	26	30	32	**40**	30.40	6.23	0.20
D	**0**	2	4	4	**18**	5.60	7.13	1.27
O	**17**	24	25	28	**34**	25.60	6.18	0.24
				Hybrid				
E	**76**	80	82	78	**90**	81.20	5.40	0.06
R	**64**	68	72	72	**72**	69.60	3.57	0.05
D	28	22	**18**	**34**	22	24.80	6.26	0.25
O	**55**	56	57	61	**61**	58.00	2.82	0.04

observed when training with the Hybrid visual abstraction. Specifically, there is a significant reduction in the standard deviation across all traffic conditions, and the coefficient of variation is reduced by an order of magnitude.

We would like to emphasize that the variance reported in Table 5.2 comes from *training* with different random seeds. The training seed is the primary cause of variance, in addition to secondary evaluation variance which is caused by the random dynamics in the simulator. The existing practice for state-of-the-art methods on CARLA is to report only the evaluation variance by running multiple evaluations of a single training seed. We argue (given our findings) that for fair comparison, future studies should additionally report results by varying the training seed and providing the mean and standard deviation (as in Table 5.2).

Comparison to State-of-the-Art: In our final experiment, we compare our approach to CAL [SSG18], CILRS [Cod+19], LaTeS [Zha+19] and LSD [Ohn+20], which is the state-of-the-art driving agent on the NoCrash benchmark with CARLA version 0.8.4. For fair comparison, we report percentage success rate with the mean and standard deviation over three different evaluations of the best model for each approach. We further report the results of the expert autopilot used for training on the CARLA simulator as an upper bound. Our results are summarized in Table 5.3. We would additionally like to mention that LBC [Che+19], which is the state-of-the-art for a different CARLA version (0.9.6) cannot be directly compared to these methods due to the reliance on several different forms of privileged information (such as the 3D position and orientation of all external dynamic agents).

Conditional Affordance Learning (CAL) [SSG18], which maps an input image to six scalar 'affordances' that are used by a hand-designed controller for driving, is unable to achieve satisfactory performance. We rerun CILRS [Cod+19] using the author-provided best model, and notice that the rerun numbers (reported in Table 5.3) differ significantly from the CILRS models we trained in our experiments (reported in Table 5.2) despite using

Table 5.3: Comparison to state-of-the-art. *Percentage Success Rate on Town 02 of the CARLA NoCrash benchmark, presented as mean and standard deviation over three evaluations of the same model for Empty (E), Regular (R), and Dense (D) conditions. * Indicates our rerun of the best model provided by the authors of [Cod+19]. Models trained with visual abstractions obtain state-of-the-art results.*

Task	CAL	CILRS*	LaTeS	LSD	Standard	Hybrid	Expert
			Train Weather				
E	36 ± 3	65 ± 2	92 ± 1	**94 ± 1**	91 ± 2	87 ± 1	96 ± 0
R	26 ± 2	46 ± 2	74 ± 2	68 ± 2	77 ± 1	**82 ± 1**	91 ± 0
D	9 ± 1	20 ± 1	29 ± 3	30 ± 4	27 ± 7	**41 ± 1**	41 ± 2
			Test Weather				
E	25 ± 3	71 ± 2	83 ± 1	**95 ± 1**	**95 ± 1**	79 ± 1	96 ± 0
R	14 ± 2	59 ± 4	68 ± 7	65 ± 4	**75 ± 6**	71 ± 1	92 ± 0
D	10 ± 0	31 ± 3	29 ± 2	**32 ± 3**	29 ± 5	**32 ± 5**	45 ± 2

the same author-provided codebase. The authors do not release the specific dataset used for training their best model, which could explain the difference in performance. However, our models significantly outperform both the CILRS models we trained in our experiments (reported in Table 5.2) and author-provided best model (reported in Table 5.3) on every evaluation setting of the benchmark.

The authors of LaTeS [Zha+19] train a teacher network that takes ground truth segmentation masks as inputs and outputs low-level driving controls. A second student network which outputs driving controls by taking only RGB images as inputs is trained with an additional loss enforcing its latent embeddings to match the teacher network. The training of their teacher network requires fine-grained semantic segmentation labels for each sample used to train the driving policy (hundreds of thousands of images). In contrast, the models trained on our Standard and Hybrid datasets require only a few hundred fine or coarsely labeled images respectively, and outperform LaTeS in the majority of the evaluation settings.

LSD [Ohn+20] uses a mixture model trained using demonstrations and further refined by optimizing directly for the driving task in terms of a reward function. In contrast to our approach, this method uses no image-level annotations, but directly optimizing the policy with a reward is challenging outside of simulated environments. While this approach is slightly better at navigating empty conditions, our models outperform it in regular and dense traffic.

6 Conclusion

In the last few years deep neural networks have shown great success in solving recognition tasks like object detection and semantic segmentation. While recognition has important direct applications in autonomous driving, it can be exploited further to improve performance of other tasks required to solve autonomous driving. In this thesis, we investigated the effectiveness of recognition as a regularizer in the following two prominent problems in autonomous driving.

Exploiting Recognition for Robust 3D Motion Estimation. In chapter 3, our main contribution is to improve performance of 3D scene flow estimation methods in presence of large displacement or local ambiguities by exploiting recognition cues as a prior. We studied the impact of different levels of recognition granularity on the problem of estimating scene flow for dynamic foreground objects. Our results indicate that recognition cues such as 2D bounding box and 2D instance segmentation provide large improvements on foreground parts of the scene in the presence of challenges such as large displacement or local ambiguities. Furthermore, we showed that our method achieves state-of-the-art performance on the challenging KITTI scene flow benchmark.

In chapter 4, we have proposed a learning-based solution for estimating scene flow and rigid body motion from unstructured point clouds. Towards this goal, we proposed to exploit recognition cues like 3D object bounding boxes to semantically group the pointwise scene flow vectors for every semantic object in the scene. Our model simultaneously detects objects in the point clouds, estimates dense scene flow and rigid motion for all points in the cloud, and estimates object rigid motion for all detected objects as well as the observer. We have shown that a global rigid motion representation is not amenable to fully convolutional estimation, and propose to use a local representation. Our approach outperforms all evaluated baselines, yielding more accurate object motions in less time.

While our method proposed in chapter 3 achieves state-of-the-art performance on scene flow estimation from stereo images, it has limited application in autonomous driving. This is because our method takes several minutes to process each frame because of the inefficient optimization step. We address this problem in chapter 4 by introducing a CNN for fast inference of scene flow from LIDAR data. Therefore, we expect to develop a real-time inference algorithm for scene flow estimation from stereo images, potentially by replacing the inefficient CRF optimization with a fast GPU based optimizer.

Furthermore, most self driving vehicles have both cameras and LIDAR sensor collecting observations. While camera sensors can collect high resolution data with color information, LIDAR sensors work better with some materials and in some lighting and weather conditions. LIDAR sensors can also provide 360 degree 3D scan of the scene. Therefore, in future, we plan to extend our scene flow methods to fuse observations from both camera and laser scanners for higher redundancy in the input and in turn a more robust estimation.

In chapter 4, in order to process the LIDAR scans, we apply the standard 3D convolution operations. Specifically, we discretize the point cloud to a grid of voxels. However, because of the sparse nature of the LIDAR data, most of the space is empty, which makes the approach computationally and memory inefficient. To alleviate this problem, in future we expect to employ the continuous convolution that operates over non-grid structured data as proposed in Wang et al. [Wan+18].

Exploiting Recognition for End-to-End Learned Autonomous Driving. In chapter 5, we take a step towards understanding how to efficiently leverage recognition-based representations in order to learn robust driving policies in a cost-effective manner. As fine-grained semantic segmentation annotation is costly to obtain, and methods are often developed independently of the final driving task, we systematically quantify the impact of reducing annotation costs on a learned driving policy. Based on our experiments, we find that more detailed annotation does not necessarily improve actual driving performance. We show that with only a few hundred annotated images, that can be labeled in approximately 50 hours, segmentation-based visual abstractions can lead to significant improvements over end-to-end methods, in terms of both performance and variance with respect to different training seeds. Due to the modularity of this approach, its benefits can be extended to alternate policy learning techniques such as reinforcement learning. We believe that our findings will be useful to guide the development of better segmentation datasets and autonomous driving policies in the future.

Our analysis in chapter 5 is performed on images from a autonomous driving simulator. While modern simulators are getting closer to photo-realism, they still cannot offer the complexity of a real world environment. Therefore, in future, we expect to validate the claims made in chapter 5 in a real world setting.

Furthermore, existing state-of-the-art semantic segmentation methods [Zha+17] optimize for image-level metrics such as mIoU, which is challenging as it requires a combination of coarse contextual reasoning and fine pixel-level accuracy [Far+13]. However, offline metrics such as mIoU may not be optimal for predicting the performance of downstream autonomous driving task. Therefore, in the future, we expect to design metrics for semantic segmentation which are comparatively less challenging to optimize at the same time are a better indicator of downstream task performance.

A Publications

Publications [Beh+17; Beh+19; Beh+20] are covered in this thesis:

- A. Behl, O. H. Jafari, S. K. Mustikovela, H. A. Alhaija, C. Rother, and A. Geiger. "Bounding Boxes, Segmentations and Object Coordinates: How Important is Recognition for 3D Scene Flow Estimation in Autonomous Driving Scenarios?" In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*. 2017

- A. Behl, D. Paschalidou, S. Donne, and A. Geiger. "PointFlowNet: Learning Representations for Rigid Motion Estimation from Point Clouds". In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2019

- A. Behl, K. Chitta, A. Prakash, E. Ohn-Bar, and A. Geiger. "Label Efficient Visual Abstractions for Autonomous Driving". In: *International Conference on Intelligent Robots and Systems (IROS)*. 2020

Other publications during the Ph.D. [Jan+20; Pra+20; Ohn+20] that are not completely covered in this thesis:

- J. Janai, F. Güney, A. Behl, and A. Geiger. *Computer Vision for Autonomous Vehicles: Problems, Datasets and State of the Art*. Foundations, Trends in Computer Graphics, and Vision, 2020

- A. Prakash, A. Behl, E. Ohn-Bar, K. Chitta, and A. Geiger. "Exploring Data Aggregation in Policy Learning for Vision-based Urban Autonomous Driving". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020

- E. Ohn-Bar, A. Prakash, A. Behl, K. Chitta, and A. Geiger. "Learning Situational Driving". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020

I declare that this thesis has been created by me based on my own original research mentioned above. My advisor Prof. Dr. Andreas Geiger was involved in all projects. Prof. Dr. Andreas Geiger contributed ideas and text to my publications. Prof. Dr. Andreas Geiger also contributed illustrative figures. I also use content from [Jan+20] in this thesis. The related work (chapter 2) is paritally based on Janai et al.[Jan+20], which was a collaboration with Dr. Joel Janai, Dr. Fatma Guney and Prof. Dr. Andreas Geiger. The other sources, including data and code, are referenced in the text. All the experiments in this work are the result of my own work unless otherwise stated.

Bibliography

[Arb+14] P. A. Arbeláez, J. Pont-Tuset, J. T. Barron, F. Marqués, and J. Malik. "Multiscale Combinatorial Grouping". In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2014.

[Asv+16] A. Asvadi, P. Girao, P. Peixoto, and U. Nunes. "3D object tracking using RGB and LIDAR data". In: *Proc. IEEE Conf. on Intelligent Transportation Systems (ITSC)*. 2016.

[AT17] A. Arnab and P. H. S. Torr. "Pixelwise Instance Segmentation with a Dynamically Instantiated Network". In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2017.

[Bai+16] M. Bai, W. Luo, K. Kundu, and R. Urtasun. "Exploiting Semantic Information and Deep Matching for Optical Flow". In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2016.

[Beh+17] A. Behl, O. H. Jafari, S. K. Mustikovela, H. A. Alhaija, C. Rother, and A. Geiger. "Bounding Boxes, Segmentations and Object Coordinates: How Important is Recognition for 3D Scene Flow Estimation in Autonomous Driving Scenarios?" In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*. 2017.

[Beh+19] A. Behl, D. Paschalidou, S. Donne, and A. Geiger. "PointFlowNet: Learning Representations for Rigid Motion Estimation from Point Clouds". In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2019.

[Beh+20] A. Behl, K. Chitta, A. Prakash, E. Ohn-Bar, and A. Geiger. "Label Efficient Visual Abstractions for Autonomous Driving". In: *International Conference on Intelligent Robots and Systems (IROS)*. 2020.

[Ben+14] R. Benenson, M. Omran, J. H. Hosang, and B. Schiele. "Ten Years of Pedestrian Detection, What Have We Learned?" In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2014.

[BFC09] G. J. Brostow, J. Fauqueur, and R. Cipolla. "Semantic Object Classes in Video: A High-definition Ground Truth Database". In: *Pattern Recognition Letters* 30.2 (Jan. 2009), pp. 88–97.

[BKC17] V. Badrinarayanan, A. Kendall, and R. Cipolla. "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation". In: *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)* 39.12 (2017), pp. 2481–2495.

[BKO18] M. Bansal, A. Krizhevsky, and A. Ogaler. “ChauffeurNet: Learning to Drive by Imitating the Best and Synthesizing the Worst”. In: *arXiv.org* abs/1812.03079 (2018).

[BM92] P. Besl and H. McKay. “A method for registration of 3D shapes”. In: *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)* 14 (1992), pp. 239–256.

[BMK13] T. Basha, Y. Moses, and N. Kiryati. “Multi-view Scene Flow Estimation: A View Centered Variational Approach”. In: *International Journal of Computer Vision (IJCV)* 101.1 (2013), pp. 6–21.

[Boj+16] M. Bojarski, D. D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba. “End to End Learning for Self-Driving Cars”. In: *arXiv.org* 1604.07316 (2016).

[Bra+14] E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton, and C. Rother. “Learning 6D Object Pose Estimation Using 3D Object Coordinates”. In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2014.

[Bra+16] E. Brachmann, F. Michel, A. Krull, M. Y. Yang, S. Gumhold, and C. Rother. “Uncertainty-Driven 6D Pose Estimation of Objects and Scenes From a Single RGB Image”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[Bro+00] A. Broggi, M. Bertozzi, A. Fascioli, and M. Sechi. “Shape-based Pedestrian Detection”. In: *Proc. IEEE Intelligent Vehicles Symposium (IV)*. 2000.

[BU17] M. Bai and R. Urtasun. “Deep Watershed Transform for Instance Segmentation”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 2858–2866.

[Cai+16] Z. Cai, Q. Fan, R. S. Feris, and N. Vasconcelos. “A Unified Multi-scale Deep Convolutional Neural Network for Fast Object Detection”. In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2016.

[Car+12] J. Carreira, R. Caseiro, J. P. Batista, and C. Sminchisescu. “Semantic Segmentation with Second-Order Pooling”. In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2012, pp. 430–443.

[Che+14] L.-C. Chen, S. Fidler, A. L. Yuille, and R. Urtasun. “Beat the MTurkers: Automatic Image Labeling from Weak 3D Supervision”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2014.

[Che+15a] C. Chen, A. Seff, A. L. Kornhauser, and J. Xiao. “DeepDriving: Learning Affordance for Direct Perception in Autonomous Driving”. In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*. 2015.

[Che+15b] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. “Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs”. In: *Proc. of the International Conf. on Learning Representations (ICLR)*. 2015.

[Che+15c] X. Chen, K. Kundu, Y. Zhu, A. G. Berneshawi, H. Ma, S. Fidler, and R. Urtasun. “3D Object Proposals for Accurate Object Class Detection”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2015.

[Che+17a] L. Chen, G. Papandreou, F. Schroff, and H. Adam. “Rethinking Atrous Convolution for Semantic Image Segmentation”. In: *arXiv.org* 1706.05587 (2017).

[Che+17b] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia. “Multi-View 3D Object Detection Network for Autonomous Driving”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2017.

[Che+18a] L. Chen, A. Hermans, G. Papandreou, F. Schroff, P. Wang, and H. Adam. “MaskLab: Instance Segmentation by Refining Object Detection With Semantic and Direction Features”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 4013–4022.

[Che+18b] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. “DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs”. In: *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)* 40.4 (2018), pp. 834–848.

[Che+19] D. Chen, B. Zhou, V. Koltun, and P. Krähenbühl. “Learning by Cheating”. In: *Proc. Conf. on Robot Learning (CoRL)*. 2019.

[Cod+18] F. Codevilla, M. Miiller, A. López, V. Koltun, and A. Dosovitskiy. “End-to-End Driving Via Conditional Imitation Learning”. In: *Proc. IEEE International Conf. on Robotics and Automation (ICRA)*. 2018.

[Cod+19] F. Codevilla, E. Santana, A. M. López, and A. Gaidon. “Exploring the Limitations of Behavior Cloning for Autonomous Driving”. In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*. 2019.

[Cor+16] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. “The Cityscapes Dataset for Semantic Urban Scene Understanding”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[CPN18] A. D. Costea, A. Petrovai, and S. Nedevschi. “Fusion Scheme for Semantic and Instance-level Segmentation”. In: *Proc. IEEE Conf. on Intelligent Transportation Systems (ITSC)*. 2018, pp. 3469–3475.

[CS12] J. Carreira and C. Sminchisescu. “CPMC: Automatic Object Segmentation Using Constrained Parametric Min-Cuts”. In: *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)* 34.7 (2012), pp. 1312–1328.

[CVN17] A. D. Costea, R. Varga, and S. Nedevschi. “Fast Boosting Based Detection Using Scale Invariant Multimodal Multiresolution Filtered Features”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 993–1002.

[Dai+16] J. Dai, K. He, Y. Li, S. Ren, and J. Sun. “Instance-Sensitive Fully Convolutional Networks”. In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2016, pp. 534–549.

[Dew+16a] A. Dewan, T. Caselitz, G. D. Tipaldi, and W. Burgard. “Motion-based detection and tracking in 3D LiDAR scans”. In: *Proc. IEEE International Conf. on Robotics and Automation (ICRA)*. 2016.

[Dew+16b] A. Dewan, T. Caselitz, G. D. Tipaldi, and W. Burgard. “Rigid scene flow for 3D LiDAR scans”. In: *Proc. IEEE International Conf. on Intelligent Robots and Systems (IROS)*. 2016.

[DG88] E. D. Dickmanns and V. Graefe. “Dynamic monocular machine vision”. In: *Machine Vision and Applications (MVA)* 1.4 (1988), pp. 223–240.

[DHS15] J. Dai, K. He, and J. Sun. “Convolutional feature masking for joint object and stuff segmentation”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 3992–4000.

[DHS16] J. Dai, K. He, and J. Sun. “Instance-Aware Semantic Segmentation via Multi-Task Network Cascades”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[DM92] E. D. Dickmanns and B. D. Mysliwetz. “Recursive 3-D road and relative ego-state recognition”. In: *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)* 14.2 (Feb. 1992), pp. 199–213.

[DON11] R. Danescu, F. Oniga, and S. Nedevschi. “Modeling and Tracking the Driving Environment With a Particle-Based Occupancy Grid”. In: *IEEE Trans. on Intelligent Transportation Systems (TITS)* 12.4 (2011), pp. 1331–1342.

[Dos+15] A. Dosovitskiy, P. Fischer, E. Ilg, P. Haeusser, C. Hazirbas, V. Golkov, P. v.d. Smagt, D. Cremers, and T. Brox. “FlowNet: Learning Optical Flow with Convolutional Networks”. In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*. 2015.

[Dos+17] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun. “CARLA: An Open Urban Driving Simulator”. In: *Proc. Conf. on Robot Learning (CoRL)*. 2017.

[DT05] N. Dalal and B. Triggs. “Histograms of oriented gradients for human detection”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2005.

[Du+18] X. Du, M. H. Ang, S. Karaman, and D. Rus. “A General Pipeline for 3D Detection of Vehicles”. In: *Proc. IEEE International Conf. on Robotics and Automation (ICRA)*. 2018, pp. 3194–3200.

[EKC16] J. Engel, V. Koltun, and D. Cremers. “Direct Sparse Odometry”. In: *arXiv.org* 1607.02565 (2016).

[Eng+17] M. Engelcke, D. Rao, D. Z. Wang, C. H. Tong, and I. Posner. “Vote3Deep: Fast object detection in 3D point clouds using efficient convolutional neural networks”. In: *Proc. IEEE International Conf. on Robotics and Automation (ICRA)*. 2017, pp. 1355–1361.

[ESC14] J. Engel, T. Schöps, and D. Cremers. “LSD-SLAM: Large-Scale Direct Monocular SLAM”. In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2014.

[Far+13] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. “Learning Hierarchical Features for Scene Labeling”. In: *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)* 35.8 (2013), pp. 1915–1929.

[FDU12] S. Fidler, S. Dickinson, and R. Urtasun. “3D Object Detection and Viewpoint Estimation with a Deformable 3D Cuboid Model”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Dec. 2012.

[Fra+05] U. Franke, C. Rabe, H. Badino, and S. Gehrig. “6D-Vision: Fusion of Stereo and Motion for Robust Environment Perception”. In: *Proc. of the DAGM Symposium on Pattern Recognition (DAGM)*. 2005.

[GF16] G. Ghiasi and C. C. Fowlkes. “Laplacian Pyramid Reconstruction and Refinement for Semantic Segmentation”. In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2016.

[GG15] F. Güney and A. Geiger. “Displets: Resolving Stereo Ambiguities using Object Knowledge”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2015.

[Gir+14] R. Girshick, J. Donahue, T. Darrell, and J. Malik. “Rich feature hierarchies for accurate object detection and semantic segmentation”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2014.

[Gir15] R. B. Girshick. “Fast R-CNN”. In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*. 2015.

[GLU12] A. Geiger, P. Lenz, and R. Urtasun. “Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2012.

[GRB08] C. Galleguillos, A. Rabinovich, and S. J. Belongie. “Object categorization using co-occurrence, location and appearance”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2008.

[GWL18] F. Groh, P. Wieschollek, and H. P. A. Lensch. “Flex-Convolution (Million-Scale Point-Cloud Learning Beyond Grid-Worlds)”. In: *Proc. of the Asian Conf. on Computer Vision (ACCV)*. Dezember 2018.

[Har+14] B. Hariharan, P. A. Arbeláez, R. B. Girshick, and J. Malik. "Simultaneous Detection and Segmentation". In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2014.

[Har+15] B. Hariharan, P. A. Arbeláez, R. B. Girshick, and J. Malik. "Hypercolumns for object segmentation and fine-grained localization". In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 447–456.

[HCL19] M. Henaff, A. Canziani, and Y. LeCun. "Model-Predictive Policy Learning with Uncertainty Regularization for Driving in Dense Traffic". In: *arXiv.org* abs/1901.02705 (2019).

[HD07] F. Huguet and F. Devernay. "A Variational Method for Scene Flow Estimation from Stereo Sequences". In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*. 2007.

[He+14] K. He, X. Zhang, S. Ren, and J. Sun. "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition". In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2014.

[He+16] K. He, X. Zhang, S. Ren, and J. Sun. "Deep Residual Learning for Image Recognition". In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[He+17] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick. "Mask R-CNN". In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*. 2017, pp. 2980–2988.

[Hel+16] D. Held, J. Levinson, S. Thrun, and S. Savarese. "Robust real-time tracking combining 3D shape, color, and motion". In: *International Journal of Robotics Research (IJRR)* 35.1-3 (2016), pp. 30–49.

[HFR14] M. Hornacek, A. Fitzgibbon, and C. Rother. "SphereFlow: 6 DoF Scene Flow from RGB-D Pairs". In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2014.

[HHS17] Z. Hayder, X. He, and M. Salzmann. "Boundary-Aware Instance Segmentation". In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 587–595.

[HK08] G. Heitz and D. Koller. "Learning spatial context: using stuff to find things". In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2008.

[HR16] J. Hur and S. Roth. "Joint Optical Flow and Temporally Consistent Semantic Segmentation". In: *Proc. of the European Conf. on Computer Vision (ECCV) Workshops*. 2016.

[HRF13] E. Herbst, X. Ren, and D. Fox. "RGB-D flow: Dense 3D motion estimation using color and depth." In: *Proc. IEEE International Conf. on Robotics and Automation (ICRA)*. 2013.

[Hua+18] X. Huang, P. Wang, X. Cheng, D. Zhou, Q. Geng, and R. Yang. "The ApolloScape Open Dataset for Autonomous Driving and its Application". In: *arXiv.org* 1803.06184 (2018).

[HZC04] X. He, R. S. Zemel, and M. A. Carreira-Perpinan. "Multiscale Conditional Random Fields for Image Labeling". In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2004.

[HZR06] X. He, R. S. Zemel, and D. Ray. "Learning and Incorporating Top-Down Cues in Image Segmentation". In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2006.

[Ilg+17] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. "FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks". In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)* (2017).

[Jan+20] J. Janai, F. Güney, A. Behl, and A. Geiger. *Computer Vision for Autonomous Vehicles: Problems, Datasets and State of the Art*. Foundations, Trends in Computer Graphics, and Vision, 2020.

[Ken+17] A. Kendall, H. Martirosyan, S. Dasgupta, and P. Henry. "End-to-End Learning of Geometry and Context for Deep Stereo Regression". In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*. 2017.

[Ken+18] A. Kendall, J. Hawke, D. Janz, P. Mazur, D. Reda, J. M. Allen, V. D. Lam, A. Bewley, and A. Shah. "Learning to Drive in a Day". In: *arXiv.org* abs/1807.00412 (2018).

[KH05] S. Kumar and M. Hebert. "A Hierarchical Field Framework for Unified Context-Based Classification". In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*. 2005.

[Kir+17] A. Kirillov, E. Levinkov, B. Andres, B. Savchynskyy, and C. Rother. "InstanceCut: From Edges to Instances with MultiCut". In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 7322–7331.

[Kir+19a] A. Kirillov, R. B. Girshick, K. He, and P. Dollár. "Panoptic Feature Pyramid Networks". In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)* (2019).

[Kir+19b] A. Kirillov, K. He, R. B. Girshick, C. Rother, and P. Dollár. "Panoptic Segmentation". In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 9404–9413.

[KK11] P. Krähenbühl and V. Koltun. "Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2011.

[KLT09] P. Kohli, L. Ladicky, and P. H. S. Torr. "Robust Higher Order Potentials for Enforcing Label Consistency". In: *International Journal of Computer Vision (IJCV)* 82.3 (2009), pp. 302–324.

[Kol06] V. Kolmogorov. "Convergent Tree-Reweighted Message Passing for Energy Minimization". In: *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)* 28.10 (2006), pp. 1568–1583.

[Kru+14] A. Krull, F. Michel, E. Brachmann, S. Gumhold, S. Ihrke, and C. Rother. "6-DOF Model Based Tracking via Object Coordinate Regression". In: *Proc. of the Asian Conf. on Computer Vision (ACCV)*. 2014.

[KSH12] A. Krizhevsky, I. Sutskever, and G. E. Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2012.

[KT99] V. R. Konda and J. N. Tsitsiklis. "Actor-Critic Algorithms". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 1999, pp. 1008–1014.

[Ku+18] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander. "Joint 3D Proposal Generation and Object Detection from View Aggregation". In: *Proc. IEEE International Conf. on Intelligent Robots and Systems (IROS)*. 2018, pp. 1–8.

[Lad+09] L. Ladicky, C. Russell, P. Kohli, and P. Torr. "Associative hierarchical CRFs for object class image segmentation". In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*. 2009.

[Lad+14] L. Ladicky, C. Russell, P. Kohli, and P. H. S. Torr. "Associative Hierarchical Random Fields". In: *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)* 36.6 (2014), pp. 1056–1077.

[Las+17] M. Laskey, J. Lee, R. Fox, A. D. Dragan, and K. Goldberg. "DART: Noise Injection for Robust Imitation Learning". In: *Proc. Conf. on Robot Learning (CoRL)*. 2017, pp. 143–156.

[LAT18] Q. Li, A. Arnab, and P. H. S. Torr. "Weakly- and Semi-supervised Panoptic Segmentation". In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2018, pp. 106–124.

[Len+11] P. Lenz, J. Ziegler, A. Geiger, and M. Roser. "Sparse Scene Flow Segmentation for Moving Object Detection in Urban Environments". In: *Proc. IEEE Intelligent Vehicles Symposium (IV)*. 2011.

[Li+17] Y. Li, H. Qi, J. Dai, X. Ji, and Y. Wei. "Fully Convolutional Instance-Aware Semantic Segmentation". In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 4438–4446.

[Li+18] G. Li, M. Mueller, V. Casser, N. Smith, D. L. Michels, and B. Ghanem. "Teaching UAVs to Race With Observational Imitation Learning". In: *arXiv.org* abs/1803.01129 (2018).

[Lia+17] Z. Liang, Y. Feng, Y. Guo, H. Liu, L. Qiao, W. Chen, L. Zhou, and J. Zhang. "Learning Deep Correspondence through Prior and Posterior Feature Constancy". In: *arXiv.org* 1712.01039 (2017).

[Lia+18] X. Liang, T. Wang, L. Yang, and E. Xing. “CIRL: Controllable Imitative Reinforcement Learning for Vision-based Self-driving”. In: *arXiv.org* abs/1807.03776 (2018).

[Lin+14] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. “Microsoft COCO: Common Objects in Context”. In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2014.

[Lin+17a] T. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie. “Feature Pyramid Networks for Object Detection”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2017.

[Lin+17b] T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár. “Focal Loss for Dense Object Detection”. In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*. 2017, pp. 2999–3007.

[Liu+16] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C. Fu, and A. C. Berg. “SSD: Single Shot MultiBox Detector”. In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2016.

[LSD15] J. Long, E. Shelhamer, and T. Darrell. “Fully Convolutional Networks for Semantic Segmentation”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2015.

[Lv+16] Z. Lv, C. Beall, P. Alcantarilla, F. Li, Z. Kira, and F. Dellaert. “A Continuous Optimization Approach for Efficient and Accurate Scene Flow”. In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2016.

[LZX16] B. Li, T. Zhang, and T. Xia. “Vehicle Detection from 3D Lidar Using Fully Convolutional Network”. In: *Proc. Robotics: Science and Systems (RSS)*. 2016.

[May+16] N. Mayer, E. Ilg, P. Haeusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. “A Large Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[MBH10] D. Munoz, J. A. Bagnell, and M. Hebert. “Stacked Hierarchical Labeling”. In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2010.

[MG15] M. Menze and A. Geiger. “Object Scene Flow for Autonomous Vehicles”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2015.

[MHG15a] M. Menze, C. Heipke, and A. Geiger. “Discrete Optimization for Optical Flow”. In: *Proc. of the German Conference on Pattern Recognition (GCPR)*. 2015.

[MHG15b] M. Menze, C. Heipke, and A. Geiger. “Joint 3D Estimation of Vehicles and Scene Flow”. In: *Proc. of the ISPRS Workshop on Image Sequence Analysis (ISA)*. 2015.

[Mic+15] F. Michel, A. Krull, E. Brachmann, M. Y. Yang, S. Gumhold, and C. Rother. “Pose Estimation of Kinematic Chain Instances via Object Coordinate Regression”. In: *Proc. of the British Machine Vision Conf. (BMVC)*. 2015.

[Mou+19] A. Mousavian, A. Toshev, M. Fiser, J. Kosecká, A. Wahid, and J. Davidson. “Visual Representations for Semantic Target Driven Navigation”. In: *Proc. IEEE International Conf. on Robotics and Automation (ICRA)*. 2019.

[MS13] F. Moosmann and C. Stiller. “Joint Self-Localization and Tracking of Generic Objects in 3D Range Data”. In: *Proc. IEEE International Conf. on Robotics and Automation (ICRA)*. 2013.

[MSS18] A. Mehta, A. Subramanian, and A. Subramanian. “Learning End-to-end Autonomous Driving using Guided Auxiliary Supervision”. In: *arXiv.org* abs/1808.10393 (2018).

[Mül+18] M. Müller, A. Dosovitskiy, B. Ghanem, and V. Koltun. “Driving Policy Transfer via Modularity and Abstraction”. In: *Proc. Conf. on Robot Learning (CoRL)*. 2018.

[NLD11] R. A. Newcombe, S. Lovegrove, and A. J. Davison. “DTAM: Dense tracking and mapping in real-time”. In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*. 2011.

[NŠ17] M. Neoral and J. Šochman. “Object Scene Flow with Temporal Consistency”. In: *Proc. of the Computer Vision Winter Workshop (CVWW)*. 2017.

[Ohn+20] E. Ohn-Bar, A. Prakash, A. Behl, K. Chitta, and A. Geiger. “Learning Situational Driving”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020.

[PCD15] P. H. O. Pinheiro, R. Collobert, and P. Dollár. “Learning to Segment Object Candidates”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2015, pp. 1990–1998.

[Pin+16] P. O. Pinheiro, T. Lin, R. Collobert, and P. Dollár. “Learning to Refine Object Segments”. In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2016, pp. 75–91.

[PKF07] J.-P. Pons, R. Keriven, and O. Faugeras. “Multi-View Stereo Reconstruction and Scene Flow Estimation with a Global Image-Based Matching Score”. In: *International Journal of Computer Vision (IJCV)* 72.2 (2007), pp. 179–193.

[Pom88] D. Pomerleau. “ALVINN: An Autonomous Land Vehicle in a Neural Network”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 1988.

[Pon+17] J. Pont-Tuset, P. Arbelaez, J. T. Barron, F. Marqués, and J. Malik. “Multiscale Combinatorial Grouping for Image Segmentation and Object Proposal Generation”. In: *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)* 39.1 (2017), pp. 128–140.

[Pra+20] A. Prakash, A. Behl, E. Ohn-Bar, K. Chitta, and A. Geiger. “Exploring Data Aggregation in Policy Learning for Vision-based Urban Autonomous Driving”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020.

[PZZ17] P. Purkait, C. Zhao, and C. Zach. “SPP-Net: Deep Absolute Pose Regression with Synthetic Views”. In: *arXiv.org* 1712.03452 (2017).

[Qi+17a] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. “Pointnet: Deep learning on point sets for 3d classification and segmentation”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2017.

[Qi+17b] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. “PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2017.

[Qi+18] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas. “Frustum pointnets for 3d object detection from rgb-d data”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)* (2018).

[Qui+14] J. Quiroga, T. Brox, F. Devernay, and J. L. Crowley. “Dense Semi-Rigid Scene Flow Estimation from RGB-D images”. In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2014.

[Rab+07] A. Rabinovich, A. Vedaldi, C. Galleguillos, E. Wiewiora, and S. Belongie. “Objects in Context”. In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*. 2007.

[Rab+10] C. Rabe, T. Mueller, A. Wedel, and U. Franke. “Dense, Robust, and Accurate Motion Field Estimation from Stereo Image Sequences in Real-Time”. In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2010.

[RB] S. Ross and D. Bagnell. “Efficient Reductions for Imitation Learning”. In: *Conference on Artificial Intelligence and Statistics (AISTATS)*.

[RBB09] R. B. Rusu, N. Blodow, and M. Beetz. “Fast Point Feature Histograms (FPFH) for 3D registration”. In: *Proc. IEEE International Conf. on Robotics and Automation (ICRA)*. 2009.

[Red+16] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi. “You Only Look Once: Unified, Real-Time Object Detection”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 779–788.

[Ren+15] S. Ren, K. He, R. B. Girshick, and J. Sun. “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2015.

[Ren+17] S. Ren, K. He, R. B. Girshick, and J. Sun. “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”. In: *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)* 39.6 (2017), pp. 1137–1149.

[RF17] J. Redmon and A. Farhadi. “YOLO9000: Better, Faster, Stronger”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2017.

[RKB04] C. Rother, V. Kolmogorov, and A. Blake. “GrabCut: interactive foreground extraction using iterated graph cuts”. In: *ACM Trans. on Graphics*. Vol. 23. 3. 2004, pp. 309–314.

[Rus+15] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. “ImageNet Large Scale Visual Recognition Challenge”. In: *International Journal of Computer Vision (IJCV)* 115.3 (2015), pp. 211–252.

[Sal+18] F. S. Saleh, M. S. Aliakbarian, M. Salzmann, L. Petersson, and J. M. Alvarez. “Effective Use of Synthetic Data for Urban Scene Semantic Segmentation”. In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2018.

[Sax+19] A. Sax, B. Emi, A. R. Zamir, L. J. Guibas, S. Savarese, and J. Malik. “Learning to Navigate Using Mid-Level Visual Priors.” In: *Proc. Conf. on Robot Learning (CoRL)*. 2019.

[Sel+17] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. “Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization”. In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*. 2017, pp. 618–626.

[Ser+13] P. Sermanet, K. Kavukcuoglu, S. Chintala, and Y. LeCun. “Pedestrian Detection with Unsupervised Multi-stage Feature Learning”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2013.

[Ser+14] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. “OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks”. In: *Proc. of the International Conf. on Learning Representations (ICLR)*. 2014.

[Sev+16] L. Sevilla-Lara, D. Sun, V. Jampani, and M. J. Black. “Optical Flow with Semantic Segmentation and Localized Layers”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[Sha+16] S. Sharifzadeh, I. Chiotellis, R. Triebel, and D. Cremers. “Learning to Drive using Inverse Reinforcement Learning and Deep Q-Networks”. In: *Advances in Neural Information Processing Systems (NeurIPS) Workshops*. 2016.

[Sho+09] J. Shotton, J. Winn, C. Rother, and A. Criminisi. “TextonBoost for Image Understanding: Multi-Class Object Recognition and Segmentation by Jointly Modeling Texture, Layout, and Context”. In: *International Journal of Computer Vision (IJCV)* 81 (2009), pp. 2–23.

[SS16] S. Shalev-Shwartz and A. Shashua. “On the Sample Complexity of End-to-end Training vs. Semantic Abstraction Training”. In: *arXiv.org* 1604.06915 (2016).

[SSG18] A. Sauer, N. Savinov, and A. Geiger. “Conditional Affordance Learning for Driving in Urban Environments”. In: *Proc. Conf. on Robot Learning (CoRL)*. 2018.

[Su+18] H. Su, V. Jampani, D. Sun, S. Maji, E. Kalogerakis, M. Yang, and J. Kautz. “SPLATNet: Sparse Lattice Networks for Point Cloud Processing”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2018.

[Sun+13] D. Sun, J. Wulff, E. Sudderth, H. Pfister, and M. Black. “A fully-connected layered model of foreground and background flow”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2013.

[Sun+18] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz. “PWC-Net: CNNs for Optical Flow Using Pyramid, Warping, and Cost Volume”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2018.

[SZ15] K. Simonyan and A. Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition”. In: *Proc. of the International Conf. on Learning Representations (ICLR)*. 2015.

[Sze+15] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. “Going deeper with convolutions”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2015.

[Tan+14] G. Tanzmeister, J. Thomas, D. Wollherr, and M. Buss. “Grid-based mapping and tracking in dynamic environments using a uniform evidential environment representation”. In: *Proc. IEEE International Conf. on Robotics and Automation (ICRA)*. 2014.

[Tay+12] J. Taylor, J. Shotton, T. Sharp, and A. W. Fitzgibbon. “The Vitruvian manifold: Inferring dense correspondences for one-shot human pose estimation”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2012.

[TDB16] M. Tatarchenko, A. Dosovitskiy, and T. Brox. “Multi-view 3D Models from Single Images with a Convolutional Network”. In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2016.

[Tho+88] C. Thorpe, M. H. Hebert, T. Kanade, and S. A. Shafer. “Vision and Navigation for the Carnegie-Mellon Navlab”. In: *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)* 10.3 (May 1988), pp. 362–372.

[TLF10] E. Tola, V. Lepetit, and P. Fua. “Daisy: An Efficient Dense Descriptor Applied to Wide Baseline Stereo”. In: *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)* 32.5 (May 2010), pp. 815–830.

[TSS10] F. Tombari, S. Salti, and L. di Stefano. “Unique Signatures of Histograms for Local Surface Description”. In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2010.

[TWM20] M. Toromanoff, E. Wirbel, and F. Moutarde. “End-to-End Model-Free Reinforcement Learning for Urban Driving using Implicit Affordances”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2020.

[Uhr+16] J. Uhrig, M. Cordts, U. Franke, and T. Brox. “Pixel-Level Encoding and Depth Layering for Instance-Level Semantic Labeling”. In: *Proc. of the German Conference on Pattern Recognition (GCPR)*. 2016, pp. 14–25.

[Uij+13] J. R. Uijlings, K. E. van de Sande, T. Gevers, and A. W. Smeulders. “Selective search for object recognition”. In: *International Journal of Computer Vision (IJCV)* 104.2 (2013), pp. 154–171.

[Ush+17] A. K. Ushani, R. W. Wolcott, J. M. Walls, and R. M. Eustice. “A learning approach for real-time temporal scene flow estimation from LIDAR data”. In: *Proc. IEEE International Conf. on Robotics and Automation (ICRA)*. 2017.

[Val+10] L. Valgaerts, A. Bruhn, H. Zimmer, J. Weickert, C. Stoll, and C. Theobalt. “Joint Estimation of Motion, Structure and Geometry from Stereo Sequences”. In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2010.

[Val+15] J. P. C. Valentin, M. Nießner, J. Shotton, A. W. Fitzgibbon, S. Izadi, and P. H. S. Torr. “Exploiting uncertainty in regression forests for accurate camera relocalization”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2015.

[Ved+05] S. Vedula, P. Rander, R. Collins, and T. Kanade. “Three-dimensional scene flow”. In: *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)* 27.3 (2005), pp. 475–480.

[Ved+99] S. Vedula, S. Baker, P. Rander, R. Collins, and T. Kanade. “Three-dimensional scene flow”. In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*. 1999.

[VJS05] P. A. Viola, M. J. Jones, and D. Snow. “Detecting pedestrians using patterns of motion and appearance”. In: *International Journal of Computer Vision (IJCV)* 63(2) (2005), pp. 153–161.

[VRS14] C. Vogel, S. Roth, and K. Schindler. “View-Consistent 3D Scene Flow Estimation over Multiple Frames”. In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2014.

[VRT10] J. van de Ven, F. Ramos, and G. D. Tipaldi. “An integrated probabilistic model for scan-matching, moving object detection and motion estimation”. In: *Proc. IEEE International Conf. on Robotics and Automation (ICRA)*. 2010.

[VSR11] C. Vogel, K. Schindler, and S. Roth. “3D scene flow estimation with a rigid motion prior”. In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*. 2011.

[VSR13] C. Vogel, K. Schindler, and S. Roth. “Piecewise Rigid Scene Flow”. In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*. 2013.

[VSR15] C. Vogel, K. Schindler, and S. Roth. “3D scene flow estimation with a piecewise rigid scene model”. In: *International Journal of Computer Vision (IJCV)* 115.1 (2015), pp. 1–28.

[VT07] J. J. Verbeek and B. Triggs. "Scene Segmentation with CRFs Learned from Partially Labeled Images". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2007, pp. 1553–1560.

[Wan+18] S. Wang, S. Suo, W.-C. Ma, A. Pokrovsky, and R. Urtasun. "Deep Parametric Continuous Convolutional Neural Networks". In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. June 2018.

[Wan+19a] D. Wang, C. Devin, Q. Cai, P. Krähenbühl, and T. Darrell. "Monocular Plan View Networks for Autonomous Driving". In: *arXiv.org* abs/1905.06937 (2019).

[Wan+19b] D. Wang, C. Devin, Q. Cai, F. Yu, and T. Darrell. "Deep Object Centric Policies for Autonomous Driving". In: *Proc. IEEE International Conf. on Robotics and Automation (ICRA)*. 2019.

[Wed+08] A. Wedel, C. Rabe, T. Vaudrey, T. Brox, U. Franke, and D. Cremers. "Efficient Dense Scene Flow from Sparse or Dense Stereo Data". In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2008.

[Wed+11] A. Wedel, T. Brox, T. Vaudrey, C. Rabe, U. Franke, and D. Cremers. "Stereoscopic scene flow computation for 3D motion understanding". In: *International Journal of Computer Vision (IJCV)* 95.1 (2011), pp. 29–51.

[Wei+13] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid. "DeepFlow: Large Displacement Optical Flow with Deep Matching". In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*. 2013.

[WP15] D. Z. Wang and I. Posner. "Voting for Voting in Online Point Cloud Object Detection". In: *Proc. Robotics: Science and Systems (RSS)*. 2015.

[WSH19] Z. Wu, C. Shen, and A. van den Hengel. "Wider or Deeper: Revisiting the ResNet Model for Visual Recognition". In: *Pattern Recognition* 90 (2019), pp. 119–133.

[Wym+15] B. Wymann, C. Dimitrakakisy, A. Sumnery, E. Espié, and C. Guionneauz. *TORCS: The open racing car simulator*. 2015.

[Xia+17] Y. Xiang, W. Choi, Y. Lin, and S. Savarese. "Subcategory-Aware Convolutional Neural Networks for Object Proposals and Detection". In: *Proc. of the IEEE Winter Conference on Applications of Computer Vision (WACV)*. 2017, pp. 924–933.

[Xio+19] Y. Xiong, R. Liao, H. Zhao, R. Hu, M. Bai, E. Yumer, and R. Urtasun. "UPSNet: A Unified Panoptic Segmentation Network". In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2019.

[Xu+17] H. Xu, Y. Gao, F. Yu, and T. Darrell. "End-to-End Learning of Driving Models from Large-Scale Video Datasets". In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 3530–3538.

[YCL16] F. Yang, W. Choi, and Y. Lin. "Exploit All the Layers: Fast and Accurate CNN Object Detector with Scale Dependent Pooling and Cascaded Rejection Classifiers". In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[YK16] F. Yu and V. Koltun. "Multi-Scale Context Aggregation by Dilated Convolutions". In: *Proc. of the International Conf. on Learning Representations (ICLR)*. 2016.

[YMU13] K. Yamaguchi, D. McAllester, and R. Urtasun. "Robust Monocular Epipolar Flow Estimation". In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2013.

[YMU14] K. Yamaguchi, D. McAllester, and R. Urtasun. "Efficient joint segmentation, occlusion labeling, stereo and flow estimation". In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2014.

[Yu+18] F. Yu, W. Xian, Y. Chen, F. Liu, M. Liao, V. Madhavan, and T. Darrell. "BDD100K: A Diverse Driving Video Database with Scalable Annotation Tooling". In: *arXiv.org* 1805.04687 (2018).

[Yu+19] F. Yu, W. Xian, Y. Chen, F. Liu, M. Liao, V. Madhavan, and T. Darrell. *Berkeley DeepDrive*. `https://bdd-data.berkeley.edu/`. Online: accessed 05-June-2019. 2019.

[Zen+17] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser. "3DMatch: Learning Local Geometric Descriptors from RGB-D Reconstructions". In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2017.

[Zen+19] W. Zeng, W. Luo, S. Suo, A. Sadat, B. Yang, S. Casas, and R. Urtasun. "End-To-End Interpretable Neural Motion Planner". In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2019.

[ZFU16] Z. Zhang, S. Fidler, and R. Urtasun. "Instance-Level Segmentation for Autonomous Driving with Deep Densely Connected MRFs". In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[Zha+15] Z. Zhang, A. G. Schwing, S. Fidler, and R. Urtasun. "Monocular Object Instance Segmentation and Depth Ordering with CNNs". In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*. 2015.

[Zha+17] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. "Pyramid Scene Parsing Network". In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2017.

[Zha+19] A. Zhao, T. He, Y. Liang, H. Huang, G. Van den Broeck, and S. Soatto. "LaTeS: Latent Space Distillation for Teacher-Student Driving Policy Learning". In: *arXiv.org* 1912.02973 (2019).

[Zhe+15] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. S. Torr. “Conditional Random Fields as Recurrent Neural Networks”. In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*. 2015.

[Zhu+16] Y. Zhu, J. Wang, C. Zhao, H. Guo, and H. Lu. “Scale-adaptive Deconvolutional Regression Network for Pedestrian Detection”. In: *Proc. of the Asian Conf. on Computer Vision (ACCV)*. 2016.

[ZKK19] B. Zhou, P. Krähenbühl, and V. Koltun. “Does computer vision matter for action?” In: *Science Robotics* 4.30 (2019).

[ŽL16] J. Žbontar and Y. LeCun. “Stereo Matching by Training a Convolutional Neural Network to Compare Image Patches”. In: *Journal of Machine Learning Research (JMLR)* 17.65 (2016), pp. 1–32.

[Zla+18] A. Zlateski, R. Jaroensri, P. Sharma, and F. Durand. “On the Importance of Label Quality for Semantic Segmentation”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2018.

[ZT18] Y. Zhou and O. Tuzel. “VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection”. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2018.

[ZW94] R. Zabih and J. Woodfill. “Non-parametric local transforms for computing visual correspondence”. In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 1994.