

Learning Navigation Priors based on Adaptive Data Aggregation

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieurin

im Rahmen des Studiums

Visual Computing

eingereicht von

Verena Widhalm, BSc

Matrikelnummer 01225862

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Univ. Prof. Dipl.-Ing. Dr. techn. Robert Sablatnig

Mitwirkung: Dipl.-Ing. Dr.techn. Manuel Keglevic

Daniel Steininger, MSc

Wien, 30. April 2023

Verena Widhalm

Robert Sablatnig



Learning Navigation Priors based on Adaptive Data Aggregation

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieurin

in

Visual Computing

by

Verena Widhalm, BSc

Registration Number 01225862

to the Faculty of Informatics

at the TU Wien

Advisor: Univ. Prof. Dipl.-Ing. Dr. techn. Robert Sablatnig

Assistance: Dipl.-Ing. Dr.techn. Manuel Keglevic
Daniel Steininger, MSc

Vienna, 30th April, 2023

Verena Widhalm

Robert Sablatnig

Erklärung zur Verfassung der Arbeit

Verena Widhalm, BSc

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 30. April 2023

Verena Widhalm

Danksagung

Nach erfolgreicher Beendigung meiner Masterarbeit, möchte ich mich bei einigen Personen die essenziell an der Fertigstellung dieser Arbeit mitgewirkt und unterstützt haben bedanken. Ich danke meinen Betreuern Manuel Keglevic, Robert Sablatnig und Daniel Steininger welche ihr umfangreiches Wissen in dem behandelten Forschungsbereich mit mir geteilt haben und mir konstruktive Ratschläge gaben aber auch auf begründete Probleme aufmerksam gemacht haben.

Ein Studienkollege meinte zu mir man muss die Masterarbeit wie einen Marathon sehen. Es ist ein langwieriger und anstrengender Prozess aber es lohnt sich am Ball zu bleiben. Nach Abschluss dieser Arbeit kann ich nur sagen ich gebe ihm vollkommen recht obwohl es sich in meinem Fall zum Ende hin wie ein Marathon mit abschließendem Zielsprint angefühlt hat. Danke Patrick für deine motivierenden Worte und andere inspirierende Diskussionen. Im Zuge dessen bedanke ich mich bei Birgit, Kati, Vanessa, Johannes und Eva welche mich beim Korrekturprozess begleitet haben.

Ein besonderer Dank gilt meiner Familie, die mich stets unterstützt hat und Rückhalt gab wodurch ich meine Fähigkeiten ungehindert verbessern und meine Ziele erreichen konnte. Insbesondere meiner Mutter Anita und meiner Nichte Helene die mich stets motivierten die Arbeit abzuschließen. Zuletzt möchte ich Alois für seine Geduld, Ausdauer und ständige mentale Unterstützung während der Erstellungsphase meiner Masterarbeit danken. Ich weiß die Hilfe von jedem einzelnen sehr zu schätzen.

Acknowledgements

It is an honor to acknowledge the persons who supported me throughout my studies and the final thesis. I am especially thankful to my advisors Manuel Keglevic, Robert Sablatnig and Daniel Steininger for sharing their knowledge with me and who always offered me advice and guidance as well as calling attention to possible problems.

One of my former fellow students told me to compare creating a master thesis with a marathon race. Like a race it is a hard and long process but at the end it is worth the effort. After finishing my thesis, I absolute agree with him but in my case, it felt like an exhausting marathon race ending with a finish sprint. Thank you, Patrick, for the motivating words and our inspiring discussions. I also want to thank Birgit, Kati, Vanessa, Johannes and Eva for supporting me during the correction phase.

Special thanks go to my family, who always supported me and gave me backing, which allowed me to improve my skills and achieve my goals unhindered. Especially my mother Anita and my niece Helene, who always motivated me to finish this thesis. Finally, I would like to thank Alois for his patience, perseverance and constant mental support during the writing phase of my master's thesis. I really appreciate everyone's help.

Kurzfassung

Autonomes Fahren lässt sich in 6 Level der Autonomie unterteilen. Während bei Level 0 keine Automatisierung am Fahrzeug vorgesehen ist, beschreibt Level 5 ein voll automatisiertes Fahrzeug das ganz ohne menschlichen Fahrer auskommt [Int18]. Die Fahrzeuge der Autobauer haben derzeit Level 2 erreicht und sind kurz davor Level 3 abzuschließen. Bei Level 3 muss noch eine Person im Fahrzeug anwesend und aufmerksam sein aber kann die Hände vom Lenkrad nehmen und das Fahrzeug fährt selbstständig [QLL22]. Um die nächste Autonomiephase zu erreichen sind weitere Entwicklungen im Bereich des Maschinellen Lernens erforderlich. Damit Autonomes Fahren generell möglich ist, muss das Fahrzeug während des Fahrprozesses wissen wo es sich befindet und weiters seine unmittelbare Umgebung wahrnehmen. Um das zu gewährleisten, benötigen Neuronale Netze ein menschenähnliches Szenenverständnis. Dies geschieht, indem sie spezifische Fahrdaten von verschiedenen bildgebenden Sensoren wie etwa Bilder von RGB Kameras über einen längeren Zeitraum analysieren. Die daraus gewonnenen Daten und abgeleiteten Erkenntnisse unterstützen autonome Fahrzeuge dabei ihre Umgebung zu erkennen und zu verstehen. Diese gesammelten Informationen können auch an andere Fahrzeuge weitergegeben werden und können genutzt werden, um das Fahrverhalten auf kommende Ereignisse anzupassen. Dadurch sind zum Beispiel Hindernisse und deren geografische Positionen für autonome Fahrzeuge bereits bekannt, obwohl sie diese selbst mit ihren Sensoren noch nicht erfasst haben. Es muss jedoch auch eine Phase der Koexistenz berücksichtigt werden in der sich sowohl autonome als auch von Menschen gesteuerte Fahrzeuge im Straßenverkehr über einen langen Zeitraum gemeinsam bewegen können. Daher sind statische Infrastrukturen wie zum Beispiel Verkehrszeichen und Ampelanlagen für die Sicherheit und das richtige Verhalten im Straßenverkehr unerlässlich. Für den Fahrprozess ist die Eigenlokalisierung des autonomen Fahrzeugs wichtig, um sich zu orientieren und eine passende Wegstrecke zu finden. Jedoch ist die Eigenlokalisierung teilweise zu ungenau. Grund dafür ist ein gestörtes Signal, von welchem die ungenauen GPS Daten produziert werden. Um die Lokalisierung des Fahrzeugs als auch von anderen Objekten zu gewährleisten, muss die Umgebung durch einen Sensor wie zum Beispiel eine Kamera wahrgenommen werden. Jedoch wird die Wahrnehmung der Umgebung auf Basis von Kamerabildern erschwert durch gewisse Störfaktoren wie etwa durch Aufnahmeartefakte, Verkehrshindernisse während der Fahrt oder erschwertes Detektieren und Tracken von Objekten während dem Fahrprozess auf Grund von bewegten Kameraaufnahmen während der Fahrt.

Um diese Herausforderungen zu meistern und eine robuste Wahrnehmung mit anschließender Lokalisierung zu unterstützen, wird im Zuge dieser Arbeit eine Kombination aus Convolutional Neural Networks untersucht. Die Objektlokalisierung im Rahmen dieser Arbeit ist das Ergebnis von drei Hauptkomponenten: ein Modul für Datenvorverarbeitung und -aggregation, ein Modul zur Objekterkennung und -klassifikation und ein Lokalisierungsmodul. Während das erste vorverarbeitete Daten aufbereitet um die Modelle zu trainieren und optimieren, extrahieren die resultierenden Modelle in den Erkennungs- und Klassifizierungsmodulen die straßenbezogenen Informationen aus den Bilddaten. Es werden dabei Autos, Radfahrer, Lastwagen, Personen, Verkehrsampeln und Verkehrsschilder entlang der Fahrbahn identifiziert. Durch Optimierungen unter Verwendung der zeitlichen Komponente werden die Ergebnisse mit Hilfe von Optical Flow und Object Tracking verbessert. Abschließend erfolgt im Lokalisierungsmodul die finale Berechnung und Positionierung der Objekte in einer Straßenkarte.

Das Projekt auto.Bus – Seestadt wird gefördert bzw. finanziert im Rahmen des FTI-Programms Mobilität der Zukunft durch das Bundesministerium für Klimaschutz und von der Österreichischen Forschungsförderungsgesellschaft (FFG) abgewickelt.

Abstract

There are 6 levels of autonomous driving, whereby level 0 means the driver is steering the car with no automation and level 5 is fully automation and can drive without any human driver [Int18]. Today car manufacturers are above level 2 and close to level 3 where the autonomous vehicle is driving by its own but it is required that the driver take over the control of the vehicle if it is requested by the vehicle [QLL22]. To reach the next level, further developments of machine learning are essential. To enable autonomous driving, the vehicle must know where it is during the driving process and perceive its immediate surroundings. Therefore, algorithms for autonomous driving need human like scene understanding by analyzing specific driving data from various sensors like images from a RGB camera over time. The collected data and the derived information from the data supports the autonomous vehicles to recognize and understand their surroundings. To further improve autonomous driving, it is necessary to get access to accurate information of obstacles and their geolocations. Autonomous vehicles are able to share and exchange the collected information with other autonomous vehicles. This means, for example, that obstacles and its geographical positions are already known to autonomous vehicles, even though they have not yet detected the obstacles itself with their sensors. However, stationary infrastructure is still essential like traffic signs and lights are essential for safety and proper behavior in road traffic due to the fact that autonomous vehicles and human-driven cars have to share the roads as long as humans are still driving cars by their own. Self-localization is needed for autonomous driving even though the localization can be inaccurate due to a disturbed GPS signal. To ensure the localization of the vehicle as well as other objects, the driving environment must be perceived by a sensor such as a camera. However, the environmental perception by image sequences is hampered by certain disturbing factors such as recording artifacts, traffic obstacles while driving or moving background. To address the challenges and to support a robust visualization, a combined approach based on convolutional neural networks is introduced. Within this thesis the performance of traffic sign localization and self-localization are the result of three major factors, namely: the data preprocessing and aggregation approach, the object detection and classification part and the localization module. While the first one prepares preprocessed data to train the models, the next two modules, the detection and classification modules extract the road-related information. The models are trained to identify cars, bicycles, trucks, persons, traffic lights and traffic signs along the traffic area. By using the temporal data, the results of the previous modules are improved. For that,

optical flow and object tracking are used. The localization module performs the final localization of the objects within the street map.

The project auto.Bus – Seestadt has received funding from the Mobility of the Future programme. Mobility of the Future is a research, technology and innovation funding programme of the Republic of Austria, Ministry of Climate Action. The Austrian Research Promotion Agency (FFG) has been authorised for the programme management.

Contents

Kurzfassung	xi
Abstract	xiii
Contents	xv
1 Introduction	1
1.1 Motivation	1
1.2 Problem Description	3
1.3 Contributions	4
1.4 Structure Of The Work	5
2 Related Work	7
2.1 Dataset Bias	7
2.2 Visual Recognition	8
2.3 Spatial and Temporal Propagation	12
2.4 Localization	13
2.5 Summary	14
3 Methodology	15
3.1 Object Space	16
3.2 Object Detection and Classification	18
3.3 Refinement by Optical Flow and Object Categorization	21
3.4 Calculate Object Localization	28
3.5 Visualization	30
3.6 Implementation Details	31
3.7 Summary	32
4 Datasets	33
4.1 Datasets for Urban Object Detection	34
4.2 Traffic Sign and Traffic Light Classification	37
4.3 Data Aggregation	39
4.4 Test Data	43
4.5 Summary	51
	xv

5	Evaluation	53
5.1	Evaluation Metrics	53
5.2	Evaluation of Object Detection	56
5.3	Evaluation of Refinement	64
5.4	Evaluation of Localization	70
5.5	Limitations	73
5.6	Summary	74
6	Conclusion	79
	List of Figures	81
	List of Tables	87
	Acronyms	89
	Bibliography	91

Introduction

Human drivers make mistakes during the driving process shown by the NHTSA study [Sin15], which revealed that 94% of road accidents occurred due to human errors. The number of people who die each year as a result of traffic accidents is increasing steadily which results in 1.35 million traffic fatalities worldwide in 2016 [O⁺18]. The human error-prone driving style results from emotional characteristics like overestimating the own driving skills and becoming panicked or distracted by the environment and physical exceptional situations like sleepiness or drunkenness [SBR21]. From this perspective, there is an opportunity to increase car and road safety through reducing the amount of workload for human vehicle operators. This results in the growing interest regarding assistance systems and self-driving cars.

1.1 Motivation

Advanced Driver-Assistance Systems (ADAS) are promising to assist drivers with self-driving functions as well as connecting the vehicle with other cars [APS20]. Currently available ADAS support the driver with basic functionalities like lane-keeping or parking assistance. In order to handle autonomous driving, the ADAS domain needs to reach higher levels of autonomy described by the taxonomy of vehicle driving autonomous systems [Int18].

Interacting with others offers the opportunity to collect additional data which can be transformed into knowledge about the environment of the vehicle [Kir15]. However, while the information exchange between connected vehicles is integrated in autonomous and humane-driven vehicles, the autonomous ones have further the opportunity for traffic- and infrastructure-related data exchange to adapt the autonomous driving style and improve their traffic steering [SM16]. This can enhance traffic operations, which leads to decreased traffic delays and to increased road capacity. For example, to avoid stopping at signal lights through automatically adapting the driving speed [DYC21] or by optimizing

the gap acceptance at roundabouts [MGE21]. As long as human-driven vehicles exist the road environment must be shared between human-driven and autonomous vehicles. Therefore, conditions must be set for mixed driving.

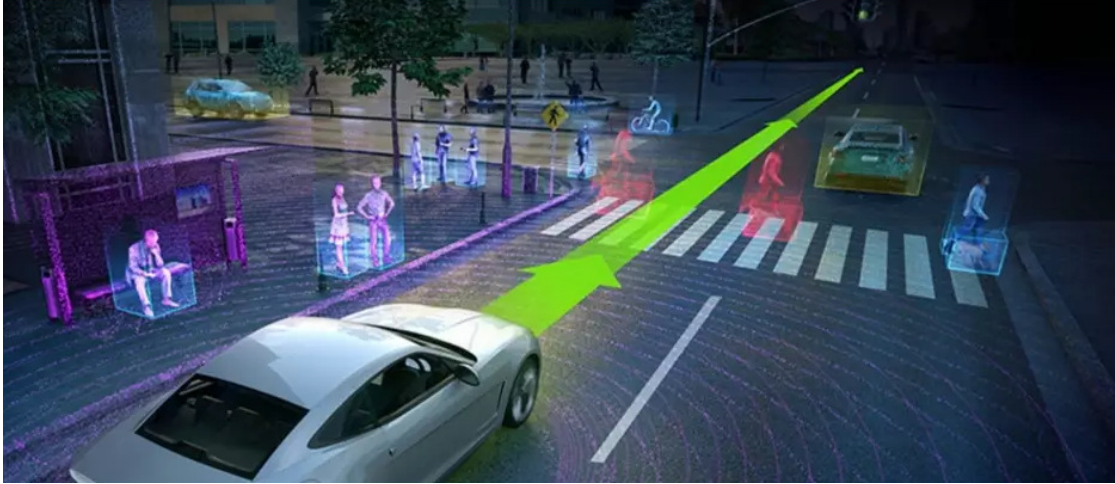


Figure 1.1: Scenery of a traffic square containing different dynamic objects [Mad19] which cope different essential research topics like object detection, multi object tracking, scene understanding up to behavior prediction.

Figure 1.1 shows the complexity of a common traffic scene within a city. The road can have multiple crossings and driving lanes. Persons are waiting on the bus station or moving through the cross walk, other cars are parking sideways next to the street or driving through the crossing and bicycles driving on the road or on specific cycle paths on the street. Stationary navigation priors, like traffic lights and traffic signs, are utilized from motorized vehicles, bicycles or pedestrians to interact with others. This results in a large number of road users with various moving behaviors. Due to prior knowledge of humans about the visual world in terms of moving and not moving objects and the semantic context within a scene, humans can make forecast dependent on the current traffic scene [BCA⁺16]. Therefore, autonomous vehicles need human skills like scene understanding to interpret the scene. To facilitate this the creation of an environmental representation is needed. The road users as well as common traffic regulations are prerequisites within this environment. Therefore, the recognition of the surrounding is a key feature to interpret the world and its influence on the style of driving. This is achieved through sensors like RGB cameras and further analyses based on the captured images to monitor the current road situation during the driving time.

To recognize objects within the driving surrounding, they have to be identified within the captured images during driving. Thereby object variability increases the self-driving complexity as well as various interfering factors hamper autonomous systems from making right decisions like over- or underexposure, unexpected objects or crowded scenes including various road users. To cope with the large variety, Convolutional Neural Network (CNN)s

[LBD⁺89] can be used, which learn object representations from image training data. The networks are systems that mimic the human nervous system which makes the training comparable to humans learning from experience [Agg18]. By increasing the experience they can collect of a specific topic, the higher the probability to create specialist knowledge.

1.2 Problem Description

Navigation is essential for driving in general. Zhou et al. [ZLK⁺17] state that an autonomous vehicle must know the place and context, in which an object appears to make assumptions and predictions concerning to future events. Thus, if the vehicle wants to get from one place to another it needs to know its position during the whole driving process. The present and future location of the self-driving vehicle must be provided accurately to be able to navigate through the traffic scene. But Global Positioning System (GPS) has a lack of accuracy due to interference by buildings, bridges or trees and for indoor or underground applications [gov22]. Even in the best-case scenario without interference the GPS signals of smartphones are typically accurate to within a radius of 4.9 meters when measuring outdoor under the open sky [VDE15].

To improve navigation and localization, visual features derived from camera input can be used. Objects with known location within the scene nearby the self-driving vehicle are detected and identified. This is achieved by using CNNs. Trained models based on selected networks predict and classify objects within the scene. However, the models are not faultless. Objects within the image can be missed by the model or other objects can be misidentified. This means that the used data to create the needed models are essential. Data variety and granularity differs between different datasets. For example, the data captured in a single street results in a low data variety which results in a low performance of the trained model if it is used in other locations containing unseen objects or objects with unexpected appearance. The label granularity of a specific class like traffic lights varies too. For example, in case that the dataset distinguishes between the state of the active lights and its orientation, the label granularity is higher than in case where just the label “traffic light” is provided. The active light and orientation allow further assumptions like a red traffic light on the current street means that the car has to stop and must wait until the traffic light is switching to green. Of course, if datasets with different label granularity are combined, the labels have to be transformed into the lower label granularity.

The autonomous vehicle is moving on the road which means that the camera mounted on the vehicle is moving too. Resulting in a moving image background which is a challenging task. Furthermore, other physical objects can move independently within the scene. Temporal occlusions of the objects which have to be recognized during the capturing time is additionally challenging. To limit the issues caused by temporal occlusions, it is essential to have preliminary knowledge regarding which objects in an image tend to move in the real world and which parts are stationary on a fixed position.

The results must be accurate for mapping the results, placing the vehicle on a street map and to feed further information into route planning and navigation algorithms.

1.3 Contributions

The goal of this thesis is to get a deeper scene understanding by computer vision techniques to be able to support further mapping techniques based on monocular image sequences from a moving camera. In the course of this thesis a framework is developed, which detects, classifies and categorizes objects within a traffic scene to finally localize the recognized derived objects and the autonomous vehicle itself within a street map. This improves localization by using current available Deep Learning (DL) techniques.

Therefore, two methods are proposed: driver self-localization and localizing stationary infrastructure. Both topics are useful for autonomous driving to localize themselves and other objects within the scene. The selected infrastructures for localization have a standardized size and are defined as static if they are mounted on fixed positions. The first method assumes that the GPS positions of the stationary objects are known. By concentrating on stationary infrastructure like traffic signs and traffic lights, they can serve as navigation priors for readjustment of the GPS signal of the vehicle which leads to an improved GPS localization. The second method assumes that the GPS signal of the autonomous vehicle is accurate within a radius of 15 centimeters which can be achieved with GPS RTK provided by the vehicle manufacturer. By automatically marking the traffic signs and traffic lights during the drive and calculating their positions in world coordinates, they are added into a street map. This leads to increased robustness of street maps by automatically update them to integrate temporal traffic sign changes like from construction zones.

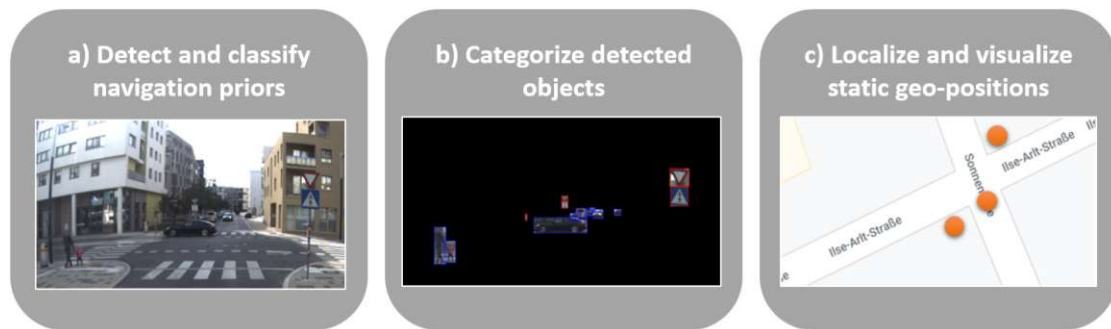


Figure 1.2: Three main aspects to cope within this thesis: (a) Objects must be detected and classified within the traffic scene. (b) The detections are improved by temporal propagation which is achieved by robust classification and occlusion handling over consecutive frames. (c) Finally, the relevant detected objects are geolocalized within the scene to estimate the main camera position in space.

In Figure 1.2 the steps of the developed pipeline are visualized. At first, to identify the defined objects within an image captured during the drive, robust scene understanding

techniques are employed which are based on consecutively detecting and classifying objects and obstacles within the scene. To increase the quality of the trained model and to decrease the training time, data aggregation techniques are applied. In the next step, wrong and missing predictions are considered to improve the final localization by using prior knowledge to make assumptions about objects interacting or appearing with others. Combining scene understanding techniques with other computer vision topics such as multi object tracking achieves a continuous spatial and temporal propagation of the estimated GPS locations. Subsequently, the objects in the scene are categorized into static, which means a fixed geographical position, and dynamic ones, like cars or pedestrians which tend to move on the urban areas. Consequently, static objects which are occluded by dynamic ones can be noticed so that object tracking and object localization do not get lost which increases the robustness. The approach is evaluated on monocular RGB image sequences captured from a moving camera mounted on an autonomous bus.

To summarize, this thesis introduces (1) a combination of CNN approaches based on (2) adaptive data aggregation to decrease training time and increase the accuracy of the final model to identify essential objects along the traffic area. Furthermore, the temporal factor (3) is essential to handle occlusions and to decrease missing or wrong predictions. Finally, (4) the positions of the detected objects are estimated and (5) visualized into a street map. Thereby, the following research questions are examined:

- Which functions are needed for autonomous vehicles to understand their surrounding with regard to current computer vision techniques?
- How can CNNs be combined to perform object geolocalization?
- How to merge various tasks to support a robust visualization process?

1.4 Structure Of The Work

This thesis is organized as follows. Section 2 gives a wide overview of commonly used state-of-the-art of datasets and their strengths and weaknesses, visual recognition like object detection and object classification, spatial and temporal propagation as well as different localization techniques. In Section 3, the selected methods and their adaptations are discussed. The used data for training and evaluation purposes are presented in Section 4. Section 5 focuses on evaluating the customized approach and summarizes the outcome. Finally, Section 6 concludes the thesis and shows potential future research topics.

Related Work

Traditional navigation tasks consist of three main components as defined by Kim et al. [KJY18]. First, visual features are extracted from the camera input of a scene. Following, the current positions of the objects are calculated based on the resulting visual features. Finally, rules are defined to move the autonomous vehicle by its own. Within this thesis the first two components are investigated to provide an improved self-localization and a street map including stationary navigation priors based on trained models during the driving process. To accomplish this task, the proposed approach combines concepts of multiple computer vision topics.

Within this section, the relevant steps data aggregation, visual recognition, temporal propagation and localization mentioned in Section 1 are discussed in detail. Since the proposed approach contains multiple challenging research topics like object detection and classification, object tracking and object localization, a concluding review is beyond the scope of this section. Therefore, for each topic representative literature based on monocular RGB images is provided and their differences are highlighted, starting with a short introduction of qualitative and quantitative aspects of datasets and their impact on object detection and classification performance. Subsequently, insights are given into relevant state-of-the-art learning techniques. Afterwards, techniques for spatial and temporal propagation are examined. The chapter finishes by presenting work of existing localization processes.

2.1 Dataset Bias

Since the aim of supervised learning is to find robust features which generalize to new unseen data, the performance of models strongly depends on the quality, quantity and variability of the annotated training data. Therefore, the dataset which is used to train the model is an essential factor for the performance of the resulting model. The detectable objects can have an increased variability and complexity, for example different shapes

and appearances or varying object size from at least 100 pixels up to the whole image area. The quality and quantity of the annotated dataset as well as the image recording conditions like viewpoint, camera to object distance and image resolution are essential. This results in the hypotheses that any finite dataset for a visual task can only describe specific aspects of specific regions from the whole visual world [TE⁺11].

As such, a dataset needs a balance between generalization and specialization. While the generalization degree has to meet the specific needs for the underlying task, the specialization degree must handle the whole range of fine-grained object categories. An unbalanced dataset results in a dataset bias. Different papers try to analyze, measure and categorize this dataset bias [TPCT17] [TE⁺11] [MS15] [GDL⁺17]. Gauen et al. [GDL⁺17] describe, analyze and compare different visual datasets for object detection in terms of label distribution and size within the image space. As a result, the authors state the increasing significance of meta information of a dataset like label distribution when selecting the data to train a model for a specific learning problem.

Instead of manipulating the datasets itself to decrease the dataset bias, there are techniques to adapt the data and their features during the training. Wang et al. [WZMG19] develop a regularization technique called Stochastic Feature Reuse. Thereby, feature maps are randomly dropped during training and reused in new down sampled sub-networks. This results in decreased training costs, decreased overfitting and increased performance by training multiple sub-networks at once. Another approach is to handle missing annotations within a dataset. Pon et al. [PAHW18] state that public datasets contain only labeled traffic lights or traffic signs. They combine a Faster R-CNN with a mini-batch proposal selection mechanism. With their approach, they can detect classes in merged datasets where the class in one dataset is not annotated in other datasets from the merged one. These partly missing classes are correctly trained by not penalizing unlabeled objects if they are detected by the model. They use a background threshold for these detections, which results in a reduced likelihood for unlabeled objects of interest, which are considered as background.

2.2 Visual Recognition

Within this thesis, two main methods are processed for visual recognition in detail: Object classification ([KSH12], [HZRS16]) and object detection. Both are fundamental visual recognition problems. While the classification part typically predicts a label for the input image, the detection part deals with predicting instances of predefined objects to define which objects are present in the scene and where they are located in image space.

In the context of computer vision tasks, Gauen et al. [GDL⁺17] state that Machine Learning (ML) approaches made improvements within the last years. Furthermore, Xiao et al. [XTY⁺20] state that object detection techniques based on DL perform better compared to traditional feature-based methods due to the fact that learned features represent the underlying data better than handcrafted features. Recognition

and localization tasks are involved during the training of the detection network. As seen in Equation 2.1, this is indicated within the loss measurement, which is responsible that the model fits the training data. The loss function is divided into two parts: localization loss and classification loss.

$$Loss = L_{cls} + L_{loc} \quad (2.1)$$

L_{loc} trains the localization regression to distinguish the target object from the background and L_{cls} trains the classification head to determine the target object type [WSH20]. In Figure 2.1 the development process of different object detection algorithms is visualized. As shown in this figure, CNNs become a new standard for visual recognition approaches in the visual computing community and traditional detection algorithms are replaced by DL based approaches. Girshick et al. [GDDM14] present at first a CNN based detection approach called R-CNN. Within short temporal intervals, new detectors were presented.

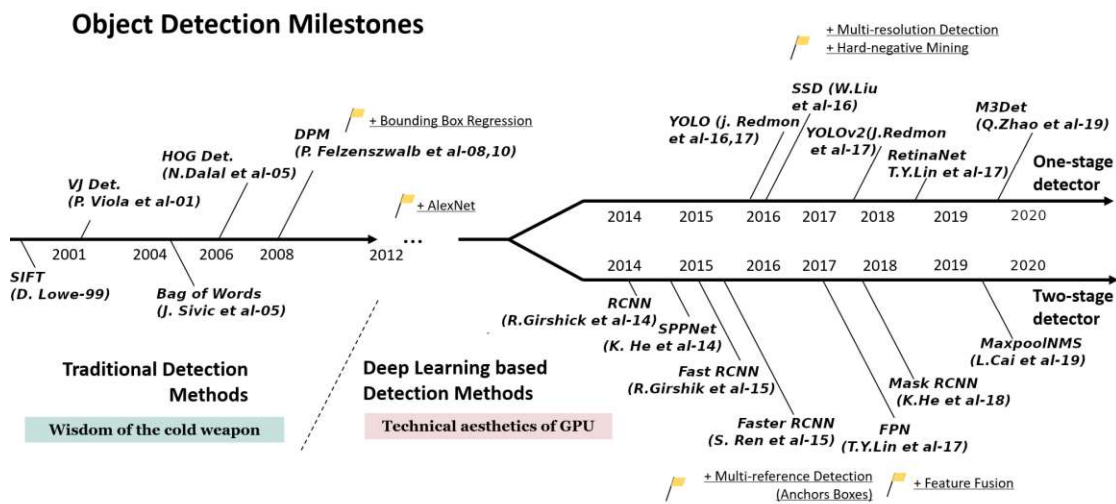


Figure 2.1: Object Detection Milestones based on [ZCS⁺23] and extended by [XTY⁺20]. At the beginning, traditional detection methods were predominant. Since 2012 the DL approaches outperform the traditional approaches. The DL based methods are further divided into one-stage and two-stage detectors. While the first one is usable for real-time applications, the second one provides a higher degree of accuracy compared with the one-stage detectors.

The learning-based object detection methods are split into one-stage and two-stage detectors. Two-stage detector models are based on a Region Proposal Network (RPN) to generate regions of interest. They are needed to make predictions for each region, which might contain an object. Within these regions bounding box regression and object classification are performed [SI18]. This detector type is represented by different Region-based Convolutional Neural Network (RCNN) approaches ([GDDM14], [Gir15], [RHGS15], [HGDG17]). By comparison, one-stage detectors ([RDGF16],[LAE⁺16],[LGG⁺17]) are

faster by skipping the region proposal stage. In addition, the localization task of the detection is treated as a simple regression problem instead of running detection and classification steps multiple times [SI18]. Thus, all predictions of an image are performed in a single pass through the network. Due to this characteristic, each detector is assigned to a specific position in the image and the predictions can be computed simultaneously. Due to the parallel computation, they are used for real-time applications.

A detector consists of multiple stages and within every step, hyper-parameters can be adapted to increase the model accuracy. Following, main basic components of each stage from a single-shot detector are described in detail.

Feature Pyramid Network (FPN). Objects from one class are presented in a wide range of scales within an image. To handle different scale stages, Lin et. al [LDG⁺17] presented FPN, which is built on an image pyramid. Processing images at different sizes allows to detect small objects depending on the scale stage, which leads to higher accuracy in terms of small object detection. While simple image pyramids are computationally relatively expensive and need high memory consumption [LDG⁺17], they construct custom feature pyramids. They are built on high-level semantic features based on further lateral connections as seen in Figure 2.2.

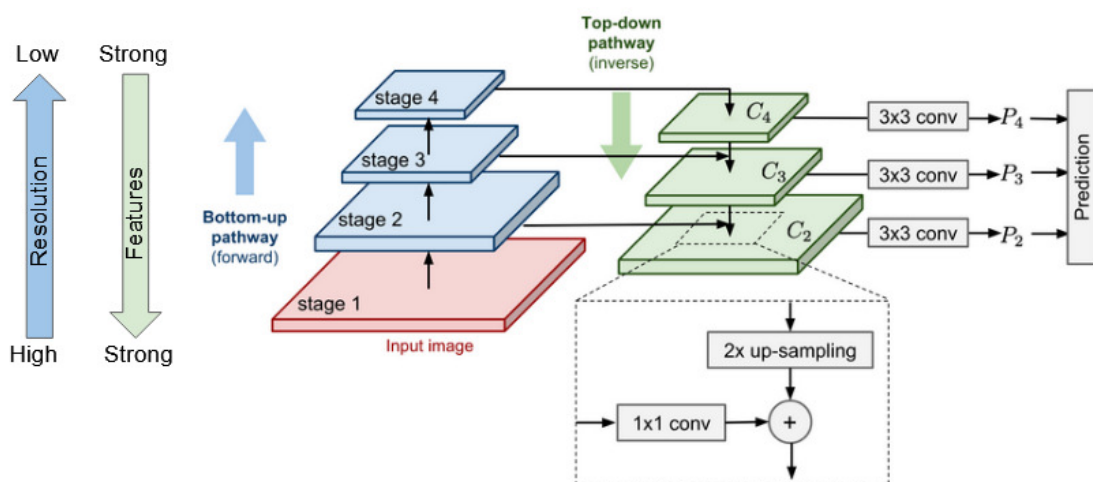


Figure 2.2: FPN with three prediction levels which is used for object detection at different resolutions [Wen18].

Anchor Boxes. Ren et al. [RHGS15] introduce anchor boxes. They are used to create a dense set of anchors by generating region proposals of various size and shape. The boxes represent the appearance of the most typical shapes, regarding object size and aspect ratio. In Figure 2.3 an example of nine predefined anchor boxes with three different scales and three different aspect ratios are shown. A bounding box is predicted if the trained model recognizes a defined object, which lies within the area of the box. A

disadvantage of this technique is, that the anchor appearance is uniformly sampled over the whole space. New techniques try to extend the anchors by eliminating the uniform distribution. Yang et al. [YZL⁺18] state that flexible anchor boxes increase robustness. Wang et al. [WCY⁺19] create a guided anchoring approach, which improves the model performance. Instead of using predefined scales and aspect ratios, they take non-uniform anchors. Hence, they predict the center of objects from arbitrary object scale and aspect ratio by using semantic features.

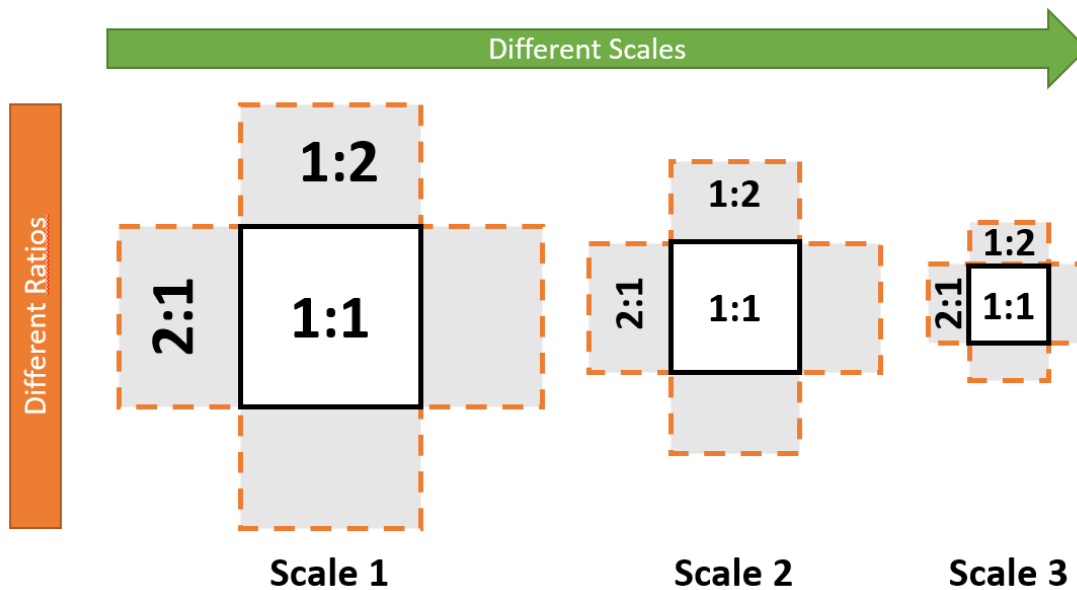


Figure 2.3: Concept of anchor boxes with three different scale levels and three different aspect ratios resulting in nine anchor boxes for each feature map location.

The output of the object detector are multiple predicted anchor boxes with different scores which results in overlapping detections of the same object. To finalize the detection process the redundant predictions are removed [ZCS⁺23]. For this step, Neubeck et al. [NVG06], for example, uses Non-Maximum Suppression (NMS) to filter the predictions so that only relevant detections remain. This is done by extracting only those predictions which reach a certain threshold. After selecting one, all other detections which have a lower score and a defined overlap measured by Intersection over Union (IoU) with the selected one are suppressed. This method has two disadvantages. It does not guarantee to take the best fitting bounding box and objects of the same class which are close to each other within the image tend to be suppressed. Bodla et al. [BSCD17] propose an extension which is called soft-NMS. Instead of suppressing a prediction by using IoU, its score is penalized by a continuous penalty function of their overlap. If a prediction still reaches the defined threshold after applying the penalty, the overlapping predictions are accepted as a potential detection.

2.3 Spatial and Temporal Propagation

While object detection and classification detects and categorizes specific objects from a single image, instance re-identification tracks object instances in different images to find corresponding bounding boxes of a specific object in an image sequence [BFM21]. Derived from image sequences, further features are extracted like temporal occlusion, object orientation or speed of the vehicle where the camera is mounted. The spatial factor extended by temporal dependencies leads to an extensive range of different approaches based on traditional computer vision techniques. To support scene interpretations based on DL datasets with additional tracking information are needed [COR⁺16]. By integrating the temporal factor sequence-based learning methods can be developed, as demonstrated in [CWW⁺17] and [CWM⁺17]. The authors introduce methods on relative pose estimation based on monocular video streams. The additional temporal information leads to a decreased pose estimation error.

With optical flow, which is a low-level computer vision problem, motion is analyzed for every pixel between images of sequential time [Mar10]. It creates a 2D vector field by evaluating the pixel movement in horizontal and vertical direction of consecutive frames. The resulting motion vectors are divided into two different granularity degrees: while sparse motion vectors are calculated only for specific points of an image, dense motion vectors are worked out for each pixel. In recent years, optical flow calculations are based on end-to-end trainable DL approaches, which improve them in terms of accuracy and performance [Pat19]. In Figure 2.4 flow vector techniques are compared. The learning-based approaches have a lower End-Point-Error (EPE) compared with the other ones.

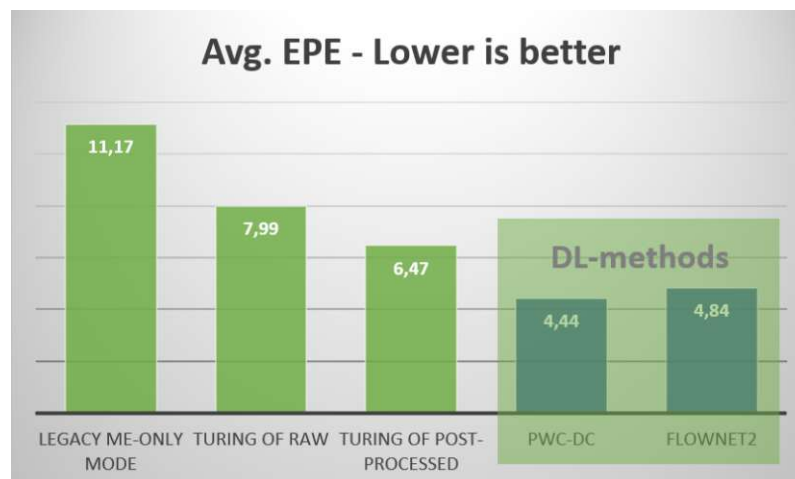


Figure 2.4: Average EPE from different optical flow approaches [Pat19]. The EPE compares the estimated optical flow vector with the ground truth optical flow vector. While traditional methods have an EPE over 5, the DL based methods are lower with an EPE of 4.44 and 4.84.

2.4 Localization

Keeping street maps manually up to date is traditionally slow and costly because of the manual effort. Within this specific field, Mapillary¹, which is a crowdsourcing-based platform, makes progress to increase the level of automation of the street map making process by using DL. The research group of Mapillary has investigated global signage systems and define features, which are most relevant for various applications, like robotic cars, derived by a database of street-view images [NORBK17], [EMO⁺20]. Their framework tries to create an inventory list of traffic signs within a specific area by using computer vision techniques. This approach tries to help cities to decrease time and costs when monitoring their street assets ^{2,3}.

When geotagging objects in the scene, a GPS signal with an appropriate accuracy is essential. Park et al. [PLCL14] estimate the camera direction of geotagged images. By combining images from Google Street View and Google Earth satellite, they are able to use them as reference images to determine the camera direction. Sharifi et al. [SNQP⁺20] develop a DL approach that takes storefronts from street-level imagery as input and provides the geolocation and type of commercial function as output. Zhang et al. [ZFL21] combine semantic segmentation, object detection and classification to identify traffic signs and lights. An attributed topological binary tree combined with six urban rules to place the objects into the world map is developed for identifying relations of road objects. The final localization is done by combining the binary tree with map features on OpenStreetMap. The object recognition rate is high with 97% completeness but they do not make any quantitative statements about their localization accuracy. Other scientific papers use depth information for the 3D triangulation in world space like [CJR⁺19]. Instead of using distance estimation by performing geometric relations of image points and real-world physical distances the following approaches try to predict object distances and coordinates by using CNNs. Krylov et al. [KKD18] geolocalizes traffic lights and telegraph poles from monocular image sequences by using two CNNs. While the first one performs object segmentation, the other one is used for distance estimation. Furthermore, they introduce a novel Markov Random Field model to perform the automatic object triangulation in areas containing multiple objects. With their approach, they reach a localization precision of about two meters. Zhu et al. [ZF19] present a trained model, which predicts distances of objects within an image. Furthermore, Nassar et al. [NLW19] predict the geolocations of objects by using two CNNs, one for projection and one for geo regression. Their approach reaches a mean absolute error of 4.36 meters at Mapillary Dataset. The mean absolute errors of these approaches depend on the predicted object types and the used methods and lie in the range of two meters up to ten meters like [ZWL⁺18].

¹<https://www.mapillary.com/>

²<https://www.geospatialworld.net/blogs/putting-traffic-signs-on-the-map/>

³<https://www.traffictechtoday.com/news/mapping/mapillary-adds-186-million-ai-generated-roadside-features-to-its-global-map.html>

2.5 Summary

This thesis cope with a combination of different disciplines. Each one is presented with explanations based on a selection of state-of-the-art papers. It starts with the challenges coming from training data to be able to train a robust model. Following by a comparison of one-stage and two-stage object detectors and their strengths and weaknesses. To make use of temporal propagation, methods for instance re-identification are researched. Concluding, geolocalization techniques are discussed.

Methodology

The goal of this thesis is to examine computer vision techniques which are used for a scene understanding of the driving environment including localization estimations and self-localization. Therefore, a framework is developed that consists of multiple components. An overview of the modular structure of the intended system is shown in Figure 3.1.

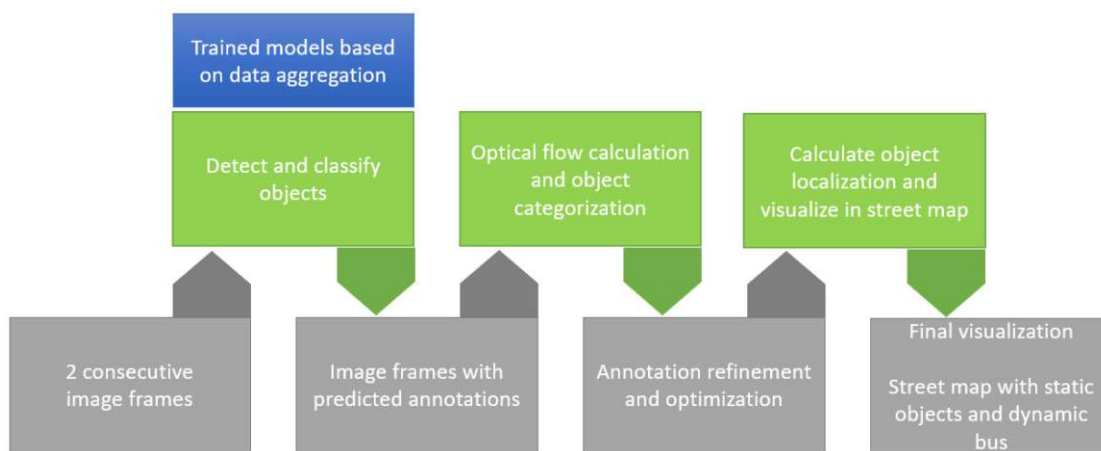


Figure 3.1: Overview of the system where the produced output in the lower row is generated by the operations in the upper row. It describes the process from two consecutive image frames up to the final output which consists of estimated localization and visualizations.

Within this chapter a detailed description of the developed framework is given and each step within the pipeline is discussed. At first the objects of interest are identified in Section 3.1. Section 3.2 describes the implementation of the image detection and classification tasks. While the former detects the specified objects within the scene, the

latter classifies the objects into fine-grained subcategories. A DL model is trained on a dataset with high object variability. This dataset is created by aggregating multiple open-source datasets which results in improved target datasets to train the models. Training a model from scratch is preferred instead of fine-tuning approaches in order to have full control and gain a thorough understanding of all parts of the processing pipeline. Following, refinement steps are used to optimize the results of the previous learning tasks which are discussed in Section 3.3. For example, two consecutive images are taken to calculate optical flow to readjusts the predicted bounding boxes and decrease bounding box oscillations within an image sequence. Furthermore, the objects are categorized according to their static, dynamic and transient properties. This is supported by prior knowledge dependent on the selected object classes and derived movement assumptions based on optical flow calculations. Finally, the positions of the defined objects are calculated 3.4 and subsequently visualized within a street map 3.5. Section 3.6 concludes the chapter by describing the used frameworks and setup to implement the presented pipeline.

3.1 Object Space

The object space characterizes the data and all relevant classes which are handled in this thesis. The developed approach is applied on objects in the ADAS area including objects on the driving area or close to it. Therefore, monocular RGB images are used. Furthermore, the data is extended by the GPS signal of the current camera capturing position to perform and evaluate object localization and GPS accuracy optimization. The training data is taken from available open-source datasets. For the evaluation process, camera recordings are taken from an autonomous bus during the auto.Bus project of AIT which are captured in Seestadt Aspern, Vienna. Hence, the driving environment is similar to a suburban area and contains different scenes and navigation priors. The driving area investigates chronological priors extended with drives of a road from both directions.

In Figure 3.2 the self-driving shuttle bus from the manufacturer Navya¹ is shown. In the course of the auto.Bus project, the sensor systems are developed further by AIT^{2,3}. Within this thesis the output of two sensors is used. A RGB camera marked with label (1) is collecting street view images within a predefined area. The camera is mounted inside the bus behind the windshield at an approximate height of 2.5 meters. Icon (2) shows the position of the second sensor which is a GNSS antenna. The bus antenna is mounted two meters behind the camera and measures one GPS signal per second with a maximum deviation of ten centimeters. The signals for the frames in between are interpolated which decreases the accuracy of the resulting outputs. Between GPS and image data there is a time delay of 17 seconds which must be taken into consideration during the dataset setup.

¹<https://navya.tech/en/>

²<https://www.ait.ac.at/themen/integrated-mobility-systems/projects/autobus-seestadt>

³<https://www.wienerlinien.at/web/wiener-linien/auto-bus-seestadt>



Figure 3.2: Autonomous bus from manufacturer Navya. A RGB camera is mounted on the upper side of the windshield (1). A GPS antenna is placed on top of the bus (2).

Objects are either moving or not moving in the real world. To examine the localization approach of this thesis, not moving objects with a fixed geographic position and explicit GPS coordinates are expected. Therefore, three categories are defined:

- **Static objects:** Navigation priors like *traffic signs* and *traffic lights* are essential for the driving and navigation process and have a fixed GPS location. Poles and street lighting systems are discarded due to inconsistencies between the different open-source datasets.
- **Dynamic objects:** This category contains moving objects which change their position over time. The bus needs to detect pedestrians as well as other human-driven vehicles to guarantee an optimized and safe road handling. Therefore, the classes *bus/truck*, *car*, *bicycle* and *pedestrian* are defined.
- **Transient objects:** The static and dynamic objects are able to switch into transient state. For example, parking cars on the roadside or temporal mounted traffic signs during roadworks. However, transient traffic signs are treated like static traffic signs and are recognized by the bus.

In Figure 3.3 the defined object space is visualized. On the left side samples of the static objects, traffic lights and traffic signs, as well as dynamic ones, like other busses, trucks, cars, pedestrians or bikers, are shown. Further examples of fine-grained subcategories for static objects are illustrated in the right. While static objects have a fixed position, dynamic objects are moving and temporarily occlude static objects. To model such

occlusions and limit the impact on the localization accuracy by decreasing missing data by occlusions, it is essential to divide the object space into dynamic, transient and static objects.

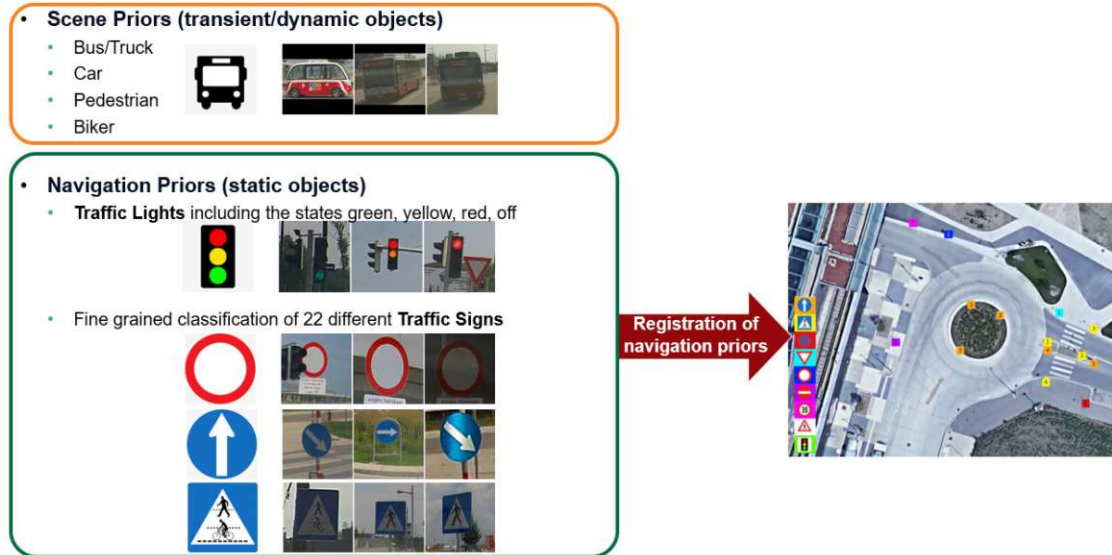


Figure 3.3: Illustration of the defined object space and the final registration of static objects within a street map. On the left side the static and dynamic objects of interest are shown whereby on the right side the static objects are visualized within the street map.

3.2 Object Detection and Classification

The object detection part identifies object instances of predefined classes from object space within a RGB image by putting a bounding box around the object. Object classification is performed to further divide the identified objects into subcategories. Both techniques are applied as a preprocessing step to find and identify the objects of interest in image space. Following, all relevant steps are discussed in detail which are used to setup, train and evaluate the used models of this thesis. At first the data aggregation stage is discussed. It is used to optimize the training progress. Afterwards, the used CNN architectures are described, and their custom adaptations are explained.

3.2.1 Data Aggregation

"The bottleneck for software engineers is about finding good-enough training data with enough variability" by Jan Erik Solem (Mapillary's CEO)⁴

⁴<https://www.citylab.com/transportation/2017/02/how-to-teach-a-car-a-traffic-sign/516030/>

Each dataset has weaknesses and contains a bias, for example a trained model for object detection based on a dataset consisting of camera recordings from highway results in a decreased performance on other scenarios with increased object variability like recordings from a city containing different traffic signs and an increased number of other moving obstacles. Furthermore, image variability is decreased if, for example, images are captured only on sunny days. In such a case, the dataset as well as the model trained on this dataset are biased towards specific weather conditions and other conditions are excluded like rainy days which leads to a decreased robustness of the trained model against varying weather conditions. The central idea of data aggregation is that the data variability is increased when combining different images from different sources. By using multiple open-source datasets, combining their classes to new datasets and aggregating the source datasets to new target datasets, the dataset bias is decreased and dataset variability is increased. Therefore, datasets are selected which contain objects that appear within the selected object space to evaluate the developed approach. The distribution of the target classes is balanced to avoid overfitting. Finally, the aggregated datasets have to be converted into a unified structured to combine them and use them for the training and the evaluation of a model. Thereafter, they are used to train the learning models for object detection and classification.

The following aspects have to be considered when combining multiple image datasets:

- Annotations of different open-source datasets are combining traffic signs with other signs into a single class and thus makes it impossible to combine them into a fine-grained target dataset.
- To focus on specific traffic signs, redundant signs have to be identified and discarded. Therefore, the static objects are categorized into sub-classes.
- Each label category has a different appearance, variability and quantity depending on the source dataset. For example, single classes are underrepresented or have a lack of variability within the recordings. To be able to generate a balanced and well-defined dataset, adaptive data aggregation is needed.

The data aggregation is performed by union of different open-source datasets. At first the labels for the target dataset are defined and the open-source datasets are selected based on the previous mentioned aspects. During the aggregation the labels of the source datasets are reorganized and finally combined and merged into the aggregated target dataset. Section 4 describes the used datasets and the data aggregation approach for creating new target datasets based on the selected open-source datasets in detail. The resulting aggregated datasets are used for this thesis and the final evaluation progress.

3.2.2 Object Detection

Object detection is used to identify objects within a monocular RGB image. The resulting predictions are further processed to interpret and understand the driving scenes. Object

detection models trained on the newest state-of-the-art architectures resulting in increased accuracy and decreased false positives as already mentioned in Section 2.2. Figure 2.1 shows the variety of available DL networks to perform object detection. RetinaNet [LGG⁺17] is selected for the training from scratch. It belongs to the one-stage detectors and performs predictions in real-time which is needed when integrating the models into the bus. Furthermore, the FPN allows to adapt the network to focus on small objects which is a use case when driving through the city. Compared to two-stage detectors, RetinaNet has the same detection accuracy as two-stage detectors with the speed of a one-stage detection algorithm [LGG⁺17]. The network is composed of a backbone network and two task-specific subnetworks. One is used for classification and the other one for regression. The backbone computes a convolutional feature map over the entire input image. The output of the backbone gets classified by the classification subnetwork. The regression subnetwork performs convolution bounding box regression. An adopted FPN builds the backbone network. It provides five feature levels in the pyramid. They are called P3 to P7 whereby P3 is the finest one up to P7 which is the coarsest feature map. Each level is built on top of a network like ResNet [HZRS16] in a fully convolutional fashion. This allows the network to handle images of arbitrary size and outputs proportionally sized feature maps. On the bottom-up pathway, the last feature maps of each group of consecutive layers are taken. With the top-down pathway and their added lateral connections, nearest neighbor upsampling is used so that the last feature map is expanded to the same scale as the second-to-last feature map [LDG⁺17]. In RetinaNet a focal loss calculation is introduced, which down-weight easier classifications than harder ones. This reduces negative predictions within background regions and lead to decrease the imbalance of examples with varying degrees of complexity [LGG⁺17].

The methodology proposed adapts the RetinaNet to decrease the inference time of the trained network. The NMS calculations are moved to Graphics Processing Unit (GPU). Inspired by the mmdetection framework[CWP⁺19] the origin RetinaNet project is divided into 3 independent parts. They are backbone, neck and head. The backbone consists of different scale levels from the FPN whereby the used network forms image features at different granularity levels which pass their results to the neck. The neck combines the image features from the backbone and in the head the anchor head handling take place to perform the final bounding box and class prediction steps. This results in a framework with modular design. For example, for mobile or embedded applications a smaller and faster backbone network can be used like mobileNetv2 instead of ResNet101.

3.2.3 Object Classification

As seen in Figure 3.4 there are multiple signs detected in a single image. Not all of them are relevant for the driving process as they are advertising signs that resemble traffic signs and signs for other road users like signs close to the sidewalk with further information for pedestrians. By applying object classification, the objects are categorized and not relevant objects which do not belong to the defined subclasses are ignored.

Two classification models are trained for the categorization approach. One to classify

detected traffic signs whereby the other one is used to classify traffic lights. Furthermore, a negative class is integrated for each model. Both models are based on a DRN-C-26 neural network architecture[YKF17]. Further data aggregation techniques like class balance are used to avoid overfitting and to create a trained classification model with high precision. During the training, regularization techniques are used like dropout which set random layer activation to zero and data augmentation techniques like image transformations which are explained in detail in Section 5 for each experiment.

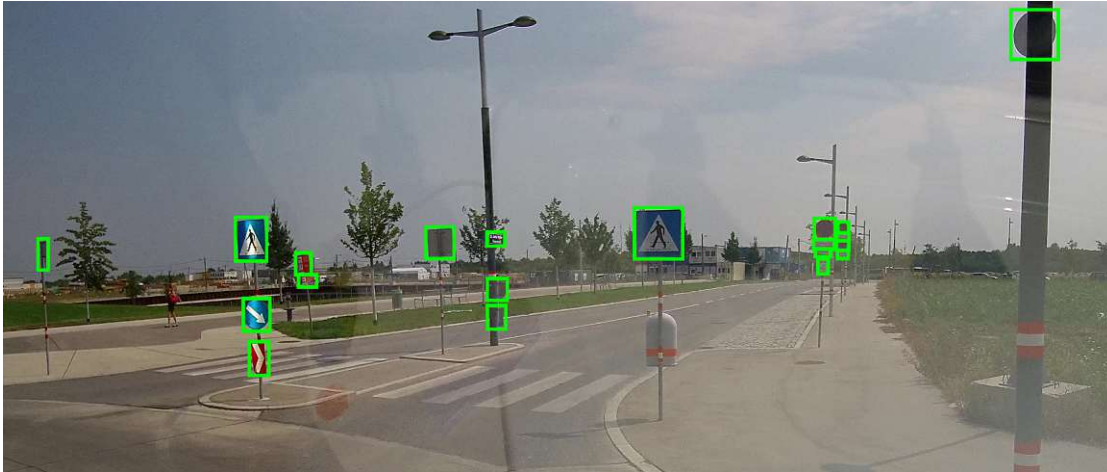


Figure 3.4: Example image from Seestadt Aspern of various signs marked with green bounding boxes. In addition to regular traffic signs, the image contains traffic signs captured from backface where the type of the object is not visible and further signs which are not applicable when driving a motorized vehicle like the sign in the right upper corner which is an information sign for construction workers at the construction site mounted approximately 5m above the ground.

3.3 Refinement by Optical Flow and Object Categorization

An object needs to be detected over time with correct bounding box size and bounding box position to calculate its position in world space over time from a monocular image stream. This requirement is needed to minimize uncertainties because uncertainties in the detection part leads to further uncertainties at the final localization. The central idea is to minimize missing detections and to make the predictions consistent over temporal propagation by using a postprocessing temporal refinement step which is applied by integrating optical flow. The resulting detections have an increased accuracy than without this postprocessing step to localize objects within a street map with increased precision. By using the temporal factor, the robustness is increased which is described in Figure 3.5. At first, a dense flow field is calculated by optical flow based on a previous and the current RGB image. Based on the generated flow field and the previous model predictions, object tracking is performed to get object instances over time. Afterwards, the following

refinements are progressed to increase the robustness of the developed approach:

- Based on the object tracks missing detections are added by motion-guided propagation (3.3.3) which can occur if the detector was not able to detect the object due to motion blur or changing lightning conditions which produce over- or underexposure effects.
- Object tracks are analyzed and wrong classifications are fixed (3.3.3).
- When the object tracking method is applied, not all predictions can be assigned to an object track. These predictions have to be discarded (3.3.4).
- Increase the tracking robustness by occlusion handling (3.3.5).
- Increase the localization robustness by minimizing bounding box oscillations over time (3.3.6). This is performed by smoothing their bounding box size over time.

Each part of the refinement stage is discussed in detail in the following subsections.

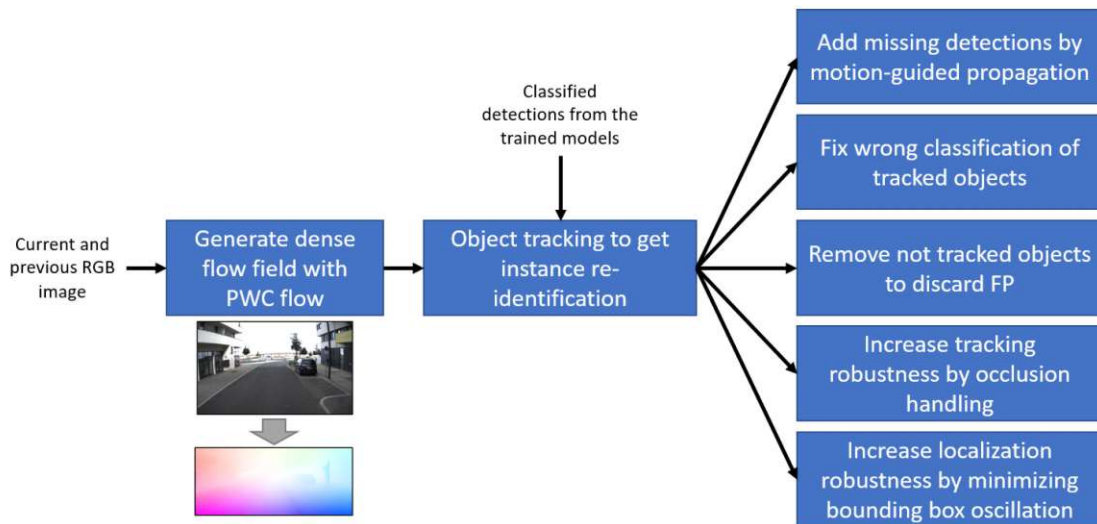


Figure 3.5: Workflow overview of the postprocessing refinement steps: at first the classified predictions of the object detector are used to generate

3.3.1 Optical Flow

A pretrained PWC-Net [SYLK18] from a python re-implementation of the network [Nik18] is used within this thesis. It is a CNN based optical flow approach. The criterion for choosing this network is that it is a small network with state-of-the-art performance. The network architecture is based on image pyramidal processing, warping, and a cost volume calculation which are classical optical flow estimation techniques. PWC-Net calculates a dense flow field by computing a motion vector for every pixel. Compared with FlowNet2

[IMS⁺17], the PWC-Net model is 17 times smaller. The runtime performance is quite high at about 35 frames per second (FPS) on images with a resolution of 1024x436px [SYLK18]. As seen in Figure 3.6 the optical flow is calculated each i^{th} frame depending on FPS or the video sequence and vehicle speed.

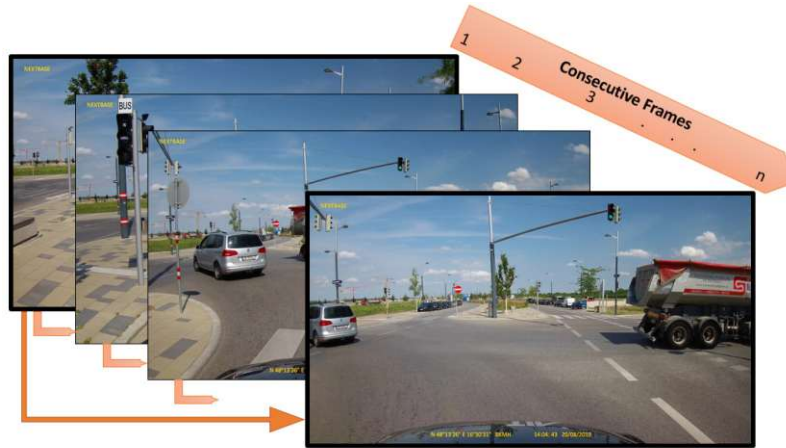


Figure 3.6: Example of an image sequence with calculation steps over multiple frames.

The dense flow field is calculated for the whole input image. However, to reduce the runtime cost, only areas with predicted bounding boxes which contain the traffic signs are used for the motion vector calculation. As shown in Figure 3.7 the bounding box areas are further scaled down to reduce background clutter.

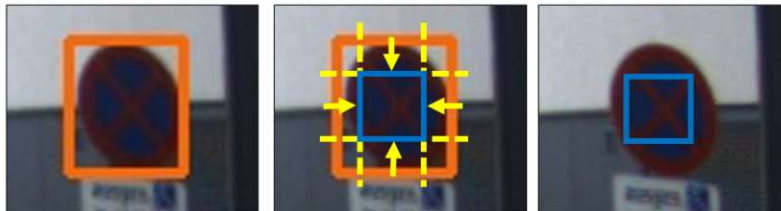


Figure 3.7: The image composition shows the narrowing of the bounding box region for calculating the resulting motion vector based on the calculated flow field. The image on the left shows a close-up of a traffic sign prediction marked as orange bounding box. The image in the middle demonstrates the narrowing of the used area for further motion vector calculations. The bounding box size is reduced by 25% for each direction indicated by the yellow arrows and dashed lines to exclude background. The resulting calculation area is shown as blue rectangle on the image on the right side.

For each narrowed region a mean vector is calculated which is the current motion vector of the object. This leads to further bounding boxes derived from previous bounding boxes and the calculated motion vectors in the case if the detector missed specific objects in the next frames over time. A visualization of this step is shown in Figure 3.8.

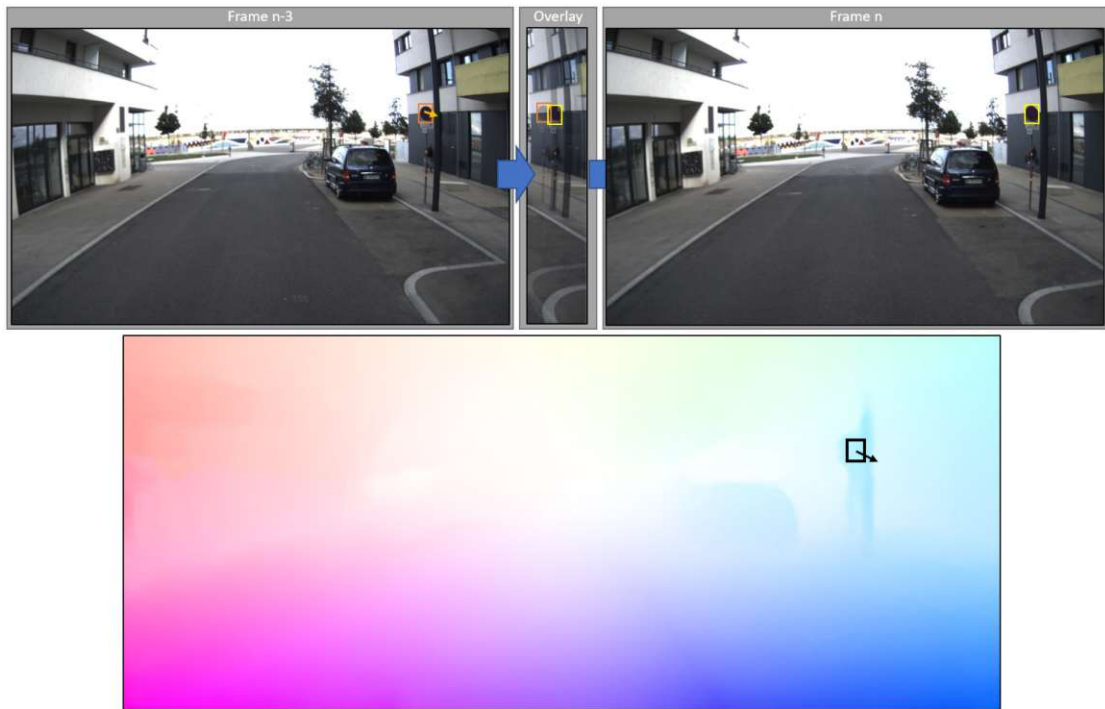


Figure 3.8: Visualization of the progress when calculating a bounding box position based on the previous frame and the calculated motion vector. The image on the bottom shows a visualization of the flow image from the RGB images of frame $n-3$ and frame n with a rectangle on the calculation area to get the resulting motion vector (black arrow). The images in the top row describe the progress: the left image shows a detected traffic sign (orange bounding box) and the calculated motion vector (orange arrow). The image in the middle shows an overlay of image n on image $n-3$ with the previous prediction (orange bounding box) and the current one (yellow bounding box). The image on the right shows the resulting prediction (yellow bounding box).

By using the motion-guided propagation approach based on optical flow, inaccurate detections are compensated and False Negative (FN)s are decreased over the whole sequence which is discussed in detail in the evaluation section. As an advantage the motion vector calculation increases robustness against noise. The derived refinement processes produced by applying optical flow leads to improved predictions of the resulting detections which are presented in the following subsections.

3.3.2 Multi Object Tracking

To calculate a spatial position the object must be known and identified over time. However, the selected detector is not able to recognize objects over time within consecutive image frames due to the fact that the model takes only the current image as input and doesn't have further knowledge about previous or next frames. The predictions of an object

instance must be connected by another method. Furthermore, tracking multiple objects of the same object type is a complex topic due to object occlusions or rapid changes in the appearance of the tracked objects [LXM⁺21]. While the occlusion handling is already covered by other methods within this thesis, the challenge of appearance changes does not have to be observed due to rigid physical objects which do not have abrupt appearance changes. Therefore, a lightweight method for object tracking⁵ is integrated to merge the detection and motion results to get all bounding box predictions over time of each physical traffic sign within the image sequence. It allows simple tracking by using object distance over consecutive frames and matching categories. The bounding box within different frames are connected by using the calculated a motion vector based on optical flow for each detected object. By using the motion vector, the successive object can be estimated. The nearest detection is identified within the next frame by calculating the distances of bounding boxes from different frames in image space. Predictions with the smallest distance difference between two consecutive frames within a 40-pixel range are defined as an object instance. As a result, each detected object gets an own tracking ID. The tracker tries to find object connections between the frames for an active tracked object until it is not detected 15 times in a row.

The tracking approach results in reduced flickering of detections of an object instance over a sequence of consecutive frames by adding missing detections through extrapolation. Consequently, the robustness of the detection results is increased as well as the accuracy of the optimized detection results. As a disadvantage, objects are only noticed if they are detected within at least two consecutive images within a defined subsequence otherwise, they are ignored.

3.3.3 Add Missing Detections and Fix Wrong Classification

Within this section a method for handling missing and wrong predictions is presented and shown in Figure 3.9. While the first and third frame show correct detections and classifications, the trained models fail in the second and fourth image. The second image shows a missing prediction and in the last frame the predicted label is set to *no parking sign* instead of *direction sign* which is not correct. To overcome objects which are not recognized by the object detector in a single frame, the estimated annotations are added in the current frame by using the predictions and further estimations of the previous frames. It is performed by extrapolating missing bounding boxes from previous predictions of an object instance and add the calculated bounding box into the current frame. For example, frame n-2 and frame n-1 have predictions of a specific sign which is not predicted in frame n which is the current frame. By extrapolating the bounding box based on frame n-2 and n-1, a new bounding box is calculated which is the estimated bounding box of the missed sign in frame n. To guarantee the same class of a tracked object over time, wrong predictions of the classification model need to be corrected. This is ensured by analyzing the label of the tracked object over time, finding the wrong classifications and fix them. For that purpose, the previous tracked predictions are evaluated and the most

⁵<https://www.pyimagesearch.com/2018/07/23/simple-object-tracking-with-opencv/>

common traffic sign subclass for this sign over all frames is selected as the resulting class of the object.



Figure 3.9: Example for possible errors during object detection and classification for a specific traffic sign instance over time. Frame one and three show correct predictions. Frame two has a missing annotation during object detection. In frame four a wrong object classification is visualized.

3.3.4 Discard False Positives

Wrong detections of static objects appear in a single frame instead of the whole sequence. Position calculations of wrong detected objects or weakly detected objects over time decrease the final localization results. Thus, predictions are only connected if they appear over a period of time which means that the object have to be detected at least two times within the last 20 frames. Annotations of static objects which are not connected within this period of 20 frames are discarded. This decreases the False Positive (FP) rate by over 50% which is shown in Section 5.3.2 in detail. Additionally, the position of a tracked object is only calculated if the object is at least three times detected within the last 20 frames and two of these detections are within the last five frames.

3.3.5 Occlusion Handling

During the tracking problems can occur which leads to a lost tracking ID. This is caused, for example, from occlusions by other objects. A lost tracking ID results in at least two tracks of an object instance or in a lost track of an object within an image sequence. To minimize the tracking errors, it is possible to deal with occlusions caused by dynamic objects moving in front of static ones. By calculating the mean vector of dynamic objects based on the optical flow helps to distinguish the moving direction of these objects and further to derive whether there is an occlusion of a traffic sign by another road user or not. Figure 3.10 visualizes an example of such an occlusion scenario. A truck is driving in front of a detected traffic sign so that the traffic sign is occluded on the image. By analyzing the motion vector of the driving object, it is obvious that the truck is driving in front of the sign and is set as occluded until the occlude leaves the area where the traffic sign was predicted earlier in the sequence so that the tracking ID is not getting lost.



Figure 3.10: Example of occlusion handling: A detected moving object occludes a static object. Both object instances are detected correct.

3.3.6 Minimize Bounding Box Oscillation

As seen in Figure 3.11 the bounding box predictions from the detection model are oscillating if the predictions are shown in consecutive frames within an image sequence which is a result of the changing visual appearance of objects as well as the changing environmental conditions during the recordings and the used training data of the detection model which have varying annotation policies depending on the source dataset. To increase the accuracy of the bounding box position and appearance over time, the predictions are readjusted. To do this, the object tracks are examined over time. It is assumed that the bus is driving forward so that the bounding box size increases the closer the camera gets. Outliers are detected by an inappropriate bounding box size, and they are readjusted by extrapolating a new bounding box size based on the previous frames and the optical flow refinement. It is assumed that the readjustment of the bounding boxes of an instance decreases bounding box oscillation and increases the final object localization calculation.

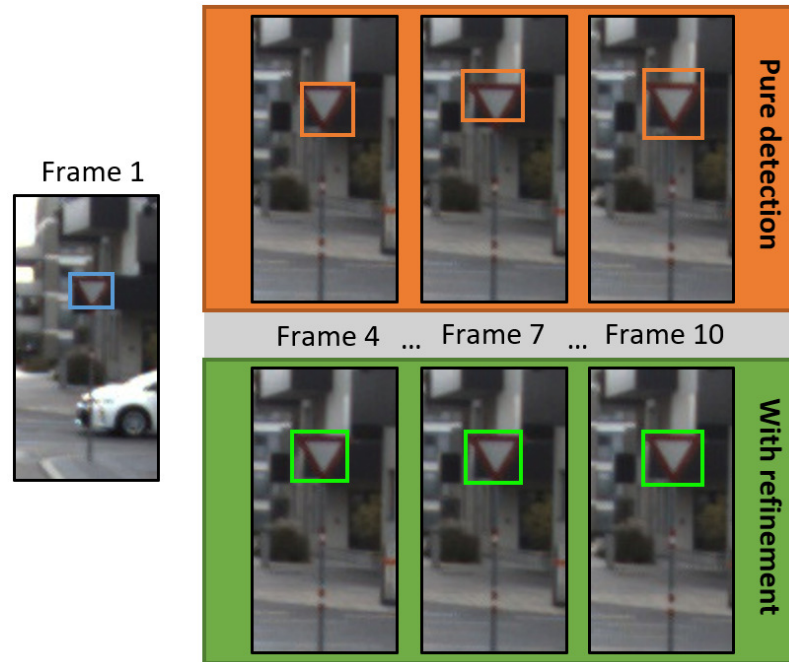


Figure 3.11: Example for bounding box oscillation from pure object detection result compared with bounding box readjustment by using optical flow refinement and interpolation.

3.4 Calculate Object Localization

In the previous sections techniques are used to increase the robustness of the developed approach and to generate accurate predictions. Therefore, it is expected that the physical object instances are consistently noticed over time. Now the concept for the final calculation is presented to generate the GPS localizations. The applied approach in the scope of this thesis is based on object size assumptions of the static objects and distance calculations between the temporal object movement in image space. Consequently, the accuracy and actuality of the street maps are increased by improving orientation, position and direction in world space.

Calculations are needed to get an estimation of the final object position. At first the distance between the camera and the object of interest is calculated for each frame. Hence, the focus of this thesis is set on traffic signs based on the assumption that they are standardized by road traffic regulations so that the real object size is known. Knowing the real physical object size enables the following object to GPS sensor distance calculation which are based on basic lens optics equations[Ful15]:

$$d_{obj} = \frac{f * h_{realObj} * h_{image}}{h_{virtualObj} * h_{cam}} + d_{cam2sensor} \quad (3.1)$$

which is based on the magnification m which is the ratio between image and object

height:

$$m = \frac{h_{image}}{h_{obj}} \quad (3.2)$$

f is the focal length, $h_{realObj}$ the real object height and h_{cam} the height of the mounted camera. All three values are in centimeter as well as the $d_{cam2sensor}$ parameter which is the distance between the camera and the GPS sensor mounted on the vehicle. h_{image} is the image height in pixel and $h_{virtualObj}$ the object height in pixel. d_{obj} is the resulting object to camera distance in centimeter. The parameter $h_{realObj}$ differs between the different classification types between 42cm and 67cm dependent on the classification label.

After the distance between the object and the sensor is known, the previous and current position of the vehicle have to be in the same format for further calculations. The GPS coordinates of the vehicle are in degrees so that they have to be converted into meters. To calculate the distance of two GPS signals from two specific camera positions the Haversine formula [DSGL07] is used:

$$d = 2R \arcsin \left(\sqrt{\sin^2 \left(\frac{\alpha_{lat} - \beta_{lat}}{2} \right) + \cos(\alpha_{lat}) \cos(\beta_{lat}) \sin^2 \left(\frac{\alpha_{lng} - \beta_{lng}}{2} \right)} \right) \quad (3.3)$$

Where d is the distance from two positions on Earth's surface in the format of latitude and longitude coordinates in degrees. The radius of the earth is defined as R and the two individual positions are described with latitude (lat) and longitude (lng) coordinates of α and β . By using this formula, the shortest geographic distance between two defined GPS points on earth is calculated which is given by the great-circle distance of these points. As an advantage, the Haversine formula is simple to process and the accuracy is higher compared to other distance measurement formulas like the law of cosines formula or equirectangular formula when combining with speed [MMH17].

Since the object to camera distance d_{obj} is already calculated, the final object position can be identified with triangulation. Therefore, multiple camera positions of the bus trail over time are taken to generate derived triangulation of view-rays as already done by Krylov et al. [KKD18].

While precision improvements are done by the previous pipeline parts, it is essential to further decrease inaccuracies during the final localization process and to eliminate apparent errors in object localization. Therefore, rules are established when performing localization calculations. After an object prediction over time is available, the detected object must be drawn into the street map. Therefore, the angle in relation to the origin bus position and its driving vector is calculated. Figure 3.12 shows that the angle between the bus direction vector (1) and the bus to object vector (2) is calculated. Depending on the calculated x angle, the general object position regarding the current bus position and the driving direction is derived:

- $x < 0$: The object is on the left side of the bus in driving direction.

- $x > 0$: The object is on the right side of the bus in driving direction.
- $x == 0$: The object is in front of the bus because both vectors are pointing in the same direction.

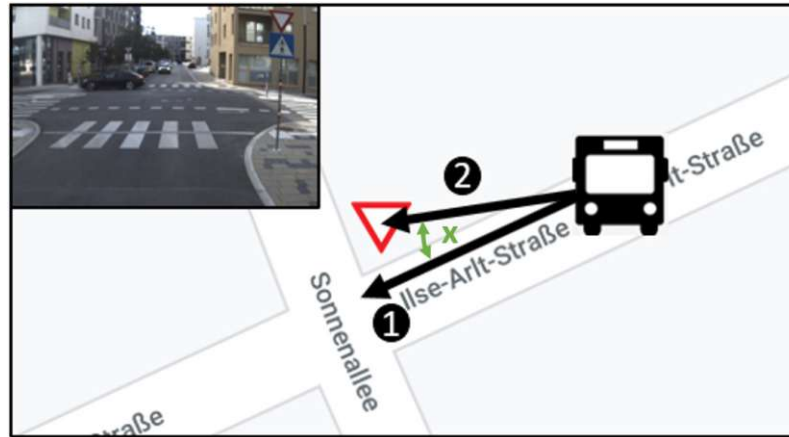


Figure 3.12: Calculation on which side of the bus the detected object occurred. (1) is the current bus vector while (2) shows the current vector from the bus to the detected object.

If the autonomous vehicle stays on the same position in at least ten consecutive frames for example stopping on a bus station, it is not feasible to perform triangulation. Therefore, a minimal distance of an object instance between consecutive selected frames must consist otherwise the frames are discarded, and the localization is not performed. This is assured that only predictions of frames are combined if they have a minimum camera distance of at least 150 centimeters apart from each other for calculating the static object locations.

3.5 Visualization

In the context of this study, the map making process is practically implemented as a custom street map representation by drawing static objects within the evaluated environment in real-time. The visualization part further helps to create and improve the framework during the whole development progress. Therefore, a web user interface is developed that facilitates a visualization of the current bus location and its camera view. As seen in Figure 3.13 it is possible to select a subsequence with their GPS bus positions. Current images with their detections are visualized on the left side whereby the current bus position and estimated traffic sign locations are visualized on the right side within the street map. The blue line within the map shows the covered distance by the bus. The label bars at the bottom are activated if an instance of their class is visible within the current image. Furthermore, interactivity during the running progress is integrated. Detection scores and traffic sign ground truth within the street is visible when switching them on. The detection threshold is adaptable for each class during run-time by the user.

A slider is available for each class and set default to 50% detection score. The user is able to activate or deactivate main parts of the pipeline to show the effects and impacts of each stage. As a result, a street map with ground truth and predictions of geotagged objects are visualized as well as a bus to object distance diagram is shown.

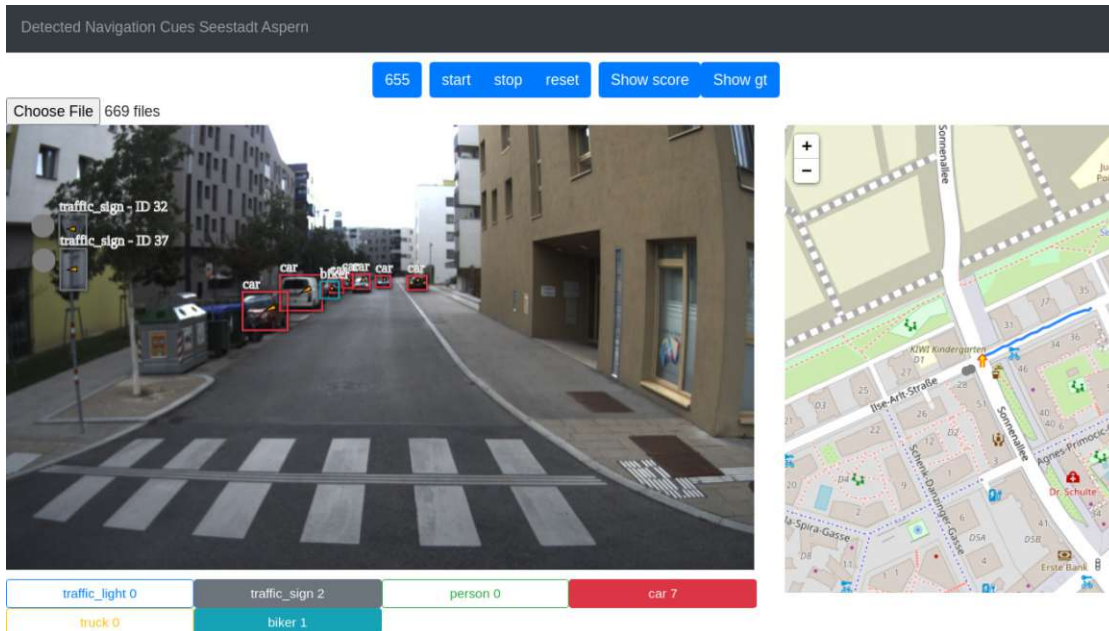


Figure 3.13: Visualization framework to load, estimate and visualize improved detections. On the left side the current image is shown. Predicted objects are marked with colored bounding boxes and the labels on the bottom are filled if an object of the label is predicted within the current image. The street map on the right side shows the estimated location of predicted static objects as well as the current bus location.

3.6 Implementation Details

The framework is split into backend and frontend. While the data processing is done in the backend, the visualizations are handled in the frontend. For the backend the programming language Python version 3.7 [VRDJ95] is used. To handle the data aggregation and the further learning approaches like object detection, object classification and optical flow estimation, the machine learning library PyTorch 1.8.1 [PGC⁺17] is used. The User Interface (UI) is created with React.js⁶. React-konva⁷ is a JavaScript drawing library and is used for the graphical animations like drawing the detections within the loaded images. It combines the HTML5 canvas library with React.js. To reserve all web requests between frontend and backend a Flask⁸ server is implemented which uses internally the

⁶<https://reactjs.org/>

⁷<https://github.com/konvajs/react-konva>

⁸<https://www.flaskapi.org/>

Jinja template. It is a light-weight method to combine the data processing of Python and the interactive visualization ability of React.js. For the street map rendering, React Leaflet⁹ is used which is derived from Leaflet.

3.7 Summary

Object detection and classification networks are selected, and the training strategies are discussed. The trained models are not faultless and to increase the robustness of the final object localization, refinement steps by optical flow and object categorizations are introduced. Using temporal propagation in combination with categorization helps to avoid lost detections and decreases FPs. After adjusting the predictions, the object localization is calculated by using the Haversine formula [DSGL07] and triangulation over time. Finally, the visualization strategy and the implementation details are discussed.

⁹<https://react-leaflet.js.org/>

CHAPTER 4

Datasets

Object detection is used to find the objects *car*, *bus/truck*, *bicycle*, *pedestrian*, *traffic sign* and *traffic light* within the image space. To further integrate a fine-grained label granularity for traffic lights and traffic signs, object classification is performed on the previous predicted static objects as aforementioned in Section 3.2. Within this chapter, the used open-source datasets and the applied aggregation steps are described to train the selected CNNs. This contains the label definitions and selections as well as the final configurations for the target datasets.

As mentioned in Section 3.1 the scope of this thesis is set to objects on or near by the driving areas especially from regions similar to that one in Seestadt Aspern. Therefore, image datasets captured in Europe are preferred for object detection as well as for object classification learning tasks due to the variety of different traffic sign appearances with the same meaning. As an example Figure 4.1 shows the variety of the traffic sign *pedestrian crossing* from seven specific countries all over the world. This demonstrates the diversity of signs which can have a completely different appearance depending on the country of origin. While the sign in Austria and Laos are quite similar except of the pedestrian icon which is male in Austria and female in Laos, the sign of Australia looks completely different which varies in shape, color and the drawings on the sign. Due to the fact that the appearance of such navigation priors can strongly vary around the world, it is essential to select datasets which contain objects with similar appearance to the final use cases.

In this section the investigated datasets for object detection 4.1 and classification 4.2 are described. Following, the data aggregation concept is presented in Section 4.3. The chapter closes with a description of the test data in Section 4.4.

¹<https://www.frontsigns.com/blog/the-difference-of-world-traffic-signs/>



Figure 4.1: Different appearance of traffic sign *pedestrian crossing* from 7 specific countries. The image is taken and adapted from ¹.

4.1 Datasets for Urban Object Detection

The datasets for the object detection tasks of the developed approach must contain dynamic objects which are used for refinement purposes as well as static objects which are needed for the resulting geo-localization. The target dataset classes for object detection are *car*, *bus/truck*, *bicycle*, *pedestrian*, *traffic sign* and *traffic light*.

To select usable datasets for the current learning approach, two rules are defined. These rules are:

- **Completeness:** All selected labels have to be annotated in the images of the dataset.
- **Balance:** Select datasets with high object variety and appearance. Further, reduce over-represented labels by removing images containing mainly these labels.

Completeness is essential to decrease the risk of missing annotations. If datasets are aggregated with missing object annotations, the model gets wrong FPs because these objects are still not annotated in the training dataset. By following this rule, datasets like Pascal VOC 2012 [EVGW⁺10] and MS COCO [LMB⁺14] are discarded due to the fact that both datasets do not contain all labels which are *car*, *bus/truck*, *bicycle*, *pedestrian*, **traffic sign** and **traffic light**. Balance is required to avoid overfitting during model training. For example, if 90% of the annotations are cars and the other 10% are annotations of the other 5 labels, it is assumed that the trained model is able to detect cars but fails by detecting objects of the other underrepresented classes. By following these rules, the Cityscapes[COR⁺16] and the Berkley Deep Drive 100.000 (BDD100k) [YCW⁺20] datasets are chosen.

Cityscapes dataset consists of 5.000 images mainly captured in cities in Germany with semantic masks on instance level divided into 35 different classes. The dataset was published in 2015. Its focus is on German cities which are comparable to Austrian cities. The similar appearance of objects and their surrounding makes this dataset valuable for the used approach. Furthermore, it contains GPS meta data and vehicle speed. The image size within the dataset is 2048x1024px. This dataset is commonly used for pixel- and instance-level learning tasks to achieve a semantic understanding[COR⁺16]. To make it usable for training and evaluation of a detection model, the pixel-wise ground truth

is converted into bounding boxes. As a disadvantage, the dataset is small compared to BDD100k with a scope of 5.000 images. The images were only captured on sunny days which leads to a low variability due to a lack of varying weather and lighting conditions. Another problem is Bayer demosaicing artefacts. The dataset contains images where even humans have problems to select the correct color state of a traffic light. Due to the pixel-level annotations of this dataset, the annotations are converted into bounding boxes to make them usable for object detection learning tasks which is used in this thesis. This results in partly inaccurate bounding boxes due to different interpretations of an object instance. For example, as seen in Figure 4.2 there are traffic light annotations which are too far away from the borders of the annotated object instances. This is a result of interpreting two traffic light objects as a single instance defined by the pixel-level annotation policy of the source dataset. This can lead to detection miss-matches if the overlap between ground truth and prediction is insufficient.

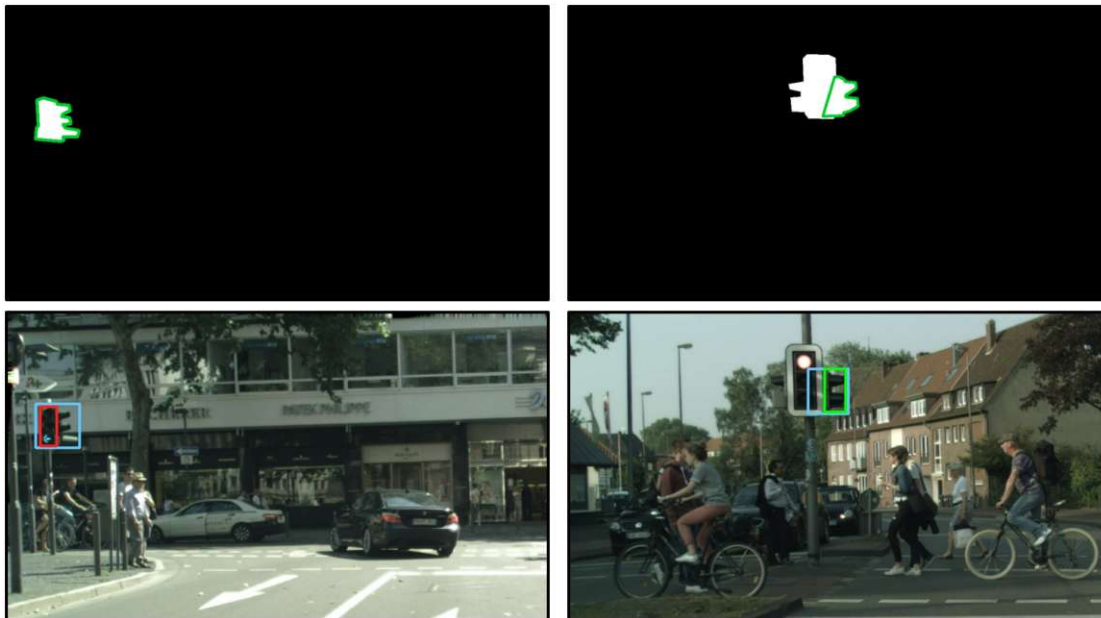


Figure 4.2: Two images of the Cityscapes dataset with their semantic masks of the traffic light objects and polyline annotations of the dataset are visualized as green lines which are used to create the bounding boxes of the object instances in the upper row showing inaccurate bounding box annotations which are created for this thesis. On the left side an image crop from Frankfurt is shown while the right image crop shows an image from Munster. The blue boxes are the derived ground truth annotations whereby the red and green box show annotations closer to the detectable object. The green box symbolizes that the IoU lies over 50% while the red box has an IoU lower than 50% compared with the ground truth bounding box.

Figure 4.3 shows that even a popular dataset like Cityscapes dataset is not faultless. It shows the semantic segmentation of the traffic lights and their associated polylines for

each object instance on the top and on the bottom the RGB image. An orange arrow is pointing on the missing object. Nevertheless, the images and objects are similar in appearance to the test area and the small errors are negligible.

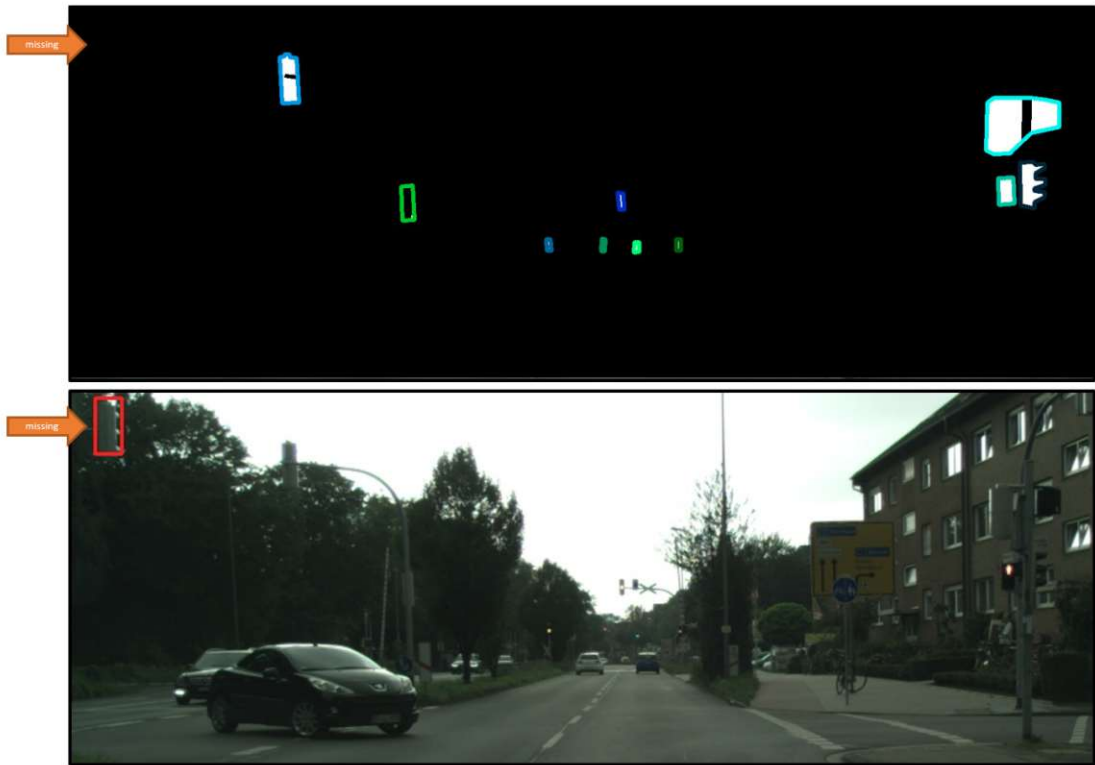


Figure 4.3: Example for missing annotation. The first image shows the semantic masks of the traffic light objects and the objects poly-line annotations of the dataset which are visualized as colored lines. The second image shows the RGB image. The missing traffic light annotation is marked with an orange arrow which point on the object which is missing.

BDD100k dataset contains images with a wide variety in geographic, environmental and weather diversity which is partly shown in Figure 4.4. The image size within the dataset is 1280x720px. As the dataset name suggests, it contains 100.000 keyframe images with 41 different classes and provides annotations for different learning tasks like semantic segmentation, pose estimation or object detection. An exceptional feature of the dataset is that it includes additionally to the other labels two different labels for traffic sign objects: the labels *traffic sign* as well as *traffic sign frame*[YCW⁺20]. However, the second label does not fit in the aggregated *traffic sign* class. The source dataset contains the labels *car*, *bus*, *bike*, *motorcycle*, *traffic light*, *traffic sign*, *train*, *truck*, *person* and *rider* which are mapped and merged into the six categories for the object detection task.

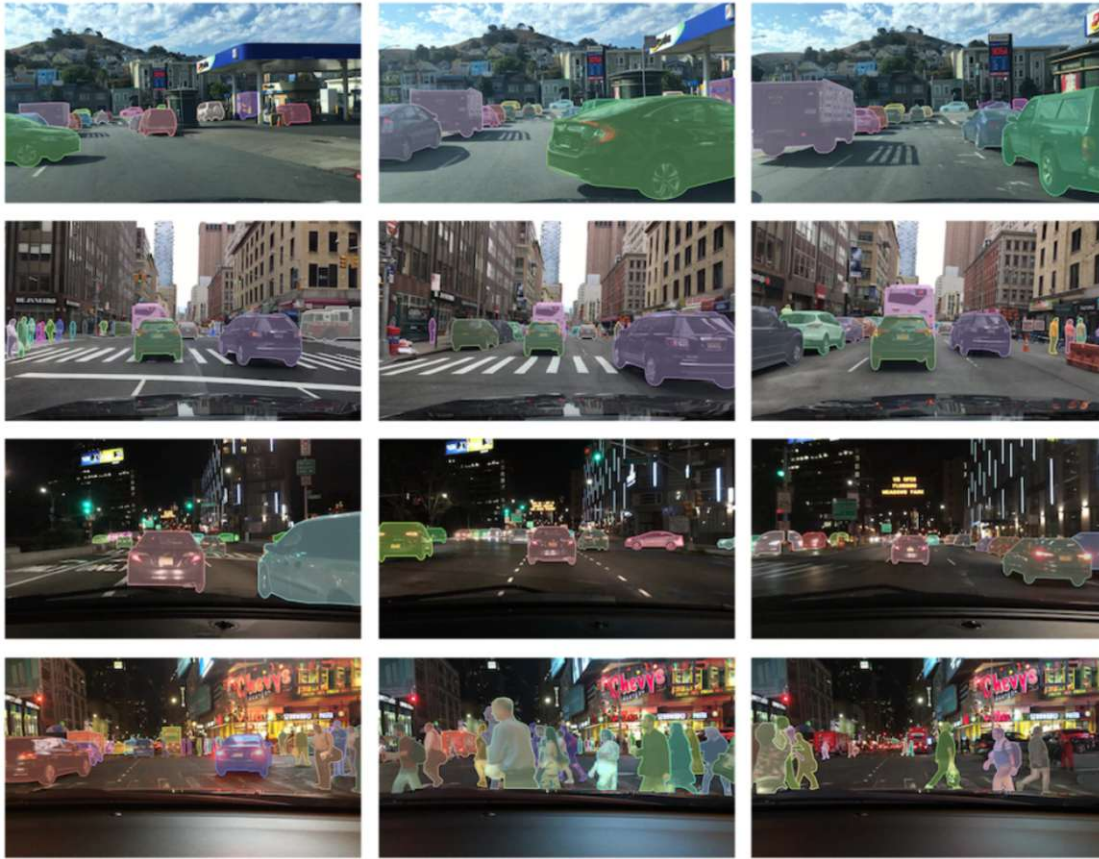


Figure 4.4: Example images of the BDD100k dataset. The figure is a revised illustration from the paper of the dataset [YCW⁺20] showing recordings from different cities and at different daytimes with visualized annotations.

4.2 Traffic Sign and Traffic Light Classification

The resulting object detections of traffic signs and traffic lights are taken as input for the object classification task to generate fine-grained subcategories of the detected static objects which are needed for the final localization approach. Derived from the detection labels, the object classification learning task consists of two independent models: one for traffic sign and one for traffic light classification. The model for traffic lights distinguish between *TL green*, *TL yellow*, *TL red* and *TL off*. While the first three are derived from the current traffic light conditions, the last one is set if the current traffic light status is not visible like traffic lights which are captured from backface or traffic lights which are switched off. The traffic sign model must cope with different traffic sign types which are commonly found in the test area of this thesis. Furthermore, it must handle other signs and traffic signs captured from backface. As seen in Figure 4.5, the defined labels for traffic sign classification are listed. The chosen signs are similar in shape, color, and

appearance. The label *unknown* is added as negative class to avoid wrong predictions within other labels. It contains different street signs which are uncommon or not included within the other 20 classes.

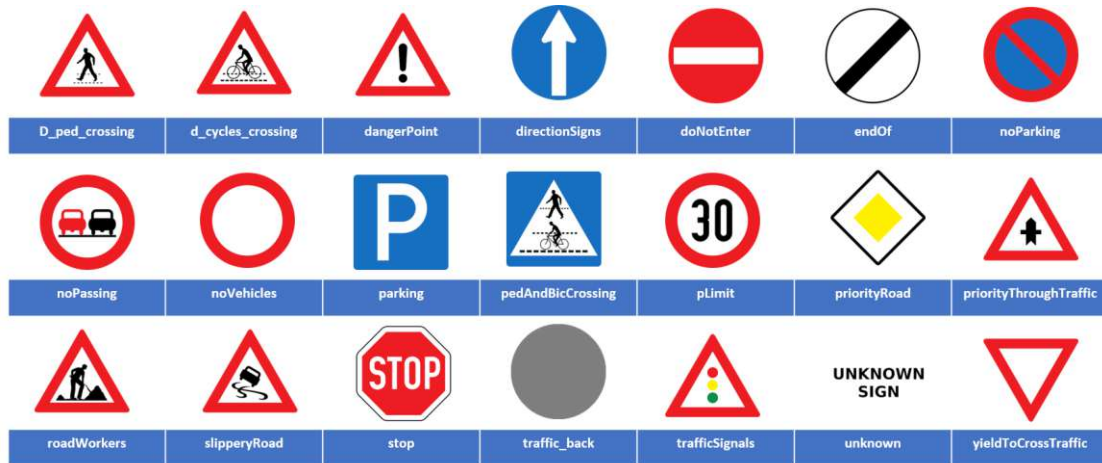


Figure 4.5: Illustration of the labels for object classification. 19 specific traffic signs are selected as well as a label for *unknown* signs and one for traffic signs captured from the backface which includes circular, rectangular and square-shaped signs from the back.

In Table 4.1 the selected datasets for the classification training are shown. In total nine open-source datasets are selected for the data aggregation stage. The five datasets in the upper rows contain traffic signs while the four datasets in the lower ones are used for traffic lights classification. In the first column the dataset names are listed, the following two columns list the whole amount of object instances and available labels for each dataset and the last two columns show the selected and used annotations and labels for the data aggregation stage to create the target datasets for the training. The datasets GTSRB[SSSI12], Mapillary[NORBK17] and DTLT[FMKD18] are large and overrepresented compared to the other selected datasets. Nevertheless, the other datasets are valuable too. They are used to increase label variability and to cover object types which are not included or underrepresented within the others. The UdacitySDC dataset[Gon16] contains bounding box annotations of recorded image data during driving. Due to the fact that this dataset contains images with unlabeled traffic signs, it is excluded from the object detection part. Nevertheless, the annotated traffic lights are cropped and used for the traffic light classification task. The Mapillary dataset is challenging too when including it into a final target dataset for object detection. It contains semantic masks without any instance segmentation. The annotations provide pixel-wise labeling of the images. As an advantage the dataset contains annotations for the label *traffic sign back* which is only included in this dataset. By using the data of the Mapillary dataset, backfaces of traffic sign detections can be classified correctly. To extract the wanted objects from the Mapillary dataset, the *traffic sign back* segmentations are converted into bounding boxes to make them usable for the object classification task. This has

been implemented by extracting the object instances based on their instance IDs which results in 53.617 extracted image crops of traffic signs captured from backface. The traffic signs from frontface are skipped because traffic signs which are belonging to at least two fine-grained classification labels are included within the traffic sign frontface class.

Table 4.1: Overview of open-source datasets used for object classification data aggregation. The dataset name (**Dataset**), the amount of annotated objects (**Annotations**) and their number of defined labels (**Lbls**) with the selected annotations (**Sel.Annotations**) and the amount of selected labels (**Sel.Lbls**) from the whole dataset labels to aggregate the resulting dataset are listed. The first 5 datasets are containing annotations for traffic sign classification and the last 4 datasets are used to create a target dataset for traffic light classification.

Dataset	Annotations	Lbls	Sel.Annotations	Sel.Lbls
BelgiumTS [MTBVG13]	7.125	62	6.218	56
GTSRB [SSSI12]	39.209	43	38.789	42
MAPILLARY [NORBK17]	-	-	53.617	1
MASTIF [ŠBK ⁺ 10]	3.213	66	2.774	57
SwedishTS [LF11]	6.651	20	3.669	7
BSTL [BNB17]	24.242	15	24.241	13
LISA [JPM ⁺ 16]	51.826	7	37.810	6
UdacitySDC [Gon16]	93.086	11	14.693	6
DTLD [FMKD18]	292.245	620	117.674	7

There are signs which are found in Vienna and do not appear in open-source dataset which containing recordings of other European cities. In Figure 4.6 a small extract of various signs captured in the investigated area of Seestadt Aspern are shown which are common signs for Vienna.

To further increase the intraclass variety of the label *unkown*, signs which are specific for Vienna with the restriction that they are not captured in the area of Seestadt Aspern are added manually. The restriction is needed in order not to distort the evaluation on the test data which are recorded in Seestadt Aspern. By adding new samples like street signs and bus signs of Vienna which are not contained in any open-source dataset the negative class variety is increased. The images are harvested from the internet and are added to the final dataset. Due to the manual effort, a manageable amount of 315 images are selected and processed for image classification which is a contribution of less than 0.3% of new unseen objects. Nevertheless, adding this small subset has an influence on the results which is examined within the evaluation.

4.3 Data Aggregation

After finishing the data selection process, the target datasets are created. The aggregation of different datasets is used to increase dataset variety and to decrease the bias of



Figure 4.6: Example of area specific traffic signs and other uncommon signs from Seestadt Aspern.

single datasets. As mentioned in the previous sections, the object appearance and the environmental conditions are varying between the datasets. As seen in Figure 4.7 different heterogeneous datasets are combined and aggregated to a new target dataset. At first the open-source datasets which are created for another learning task are converted into the annotation type format of the resulting target dataset in order to ensure a uniform annotation format in the aggregated target dataset. Afterwards, the convert data are merged into a unified target dataset.

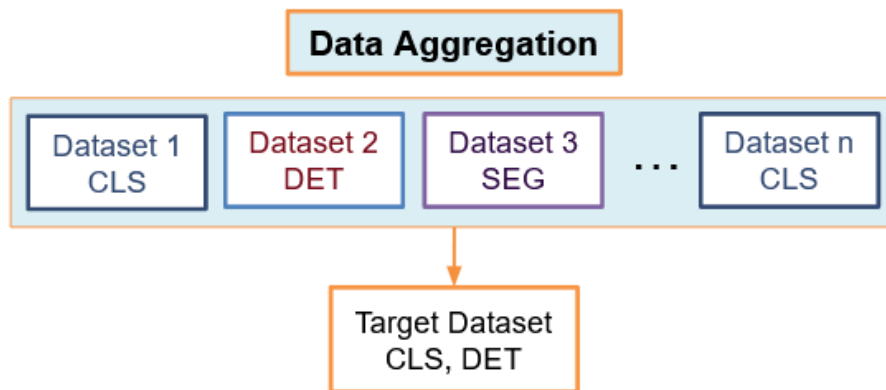


Figure 4.7: Aggregating different datasets by adapting their annotations to the needed learning task and reorganize the label settings to an unique target dataset.

To avoid loss of quality during the aggregation stage, the merging and reductions of different datasets have to be well-designed. This is supported by descriptive dataset statistics like label distribution, number of annotations per class and bounding box

appearances like width to height ratios of the annotated bounding boxes for each label. They are used as indicators to get a quick overview of the data and to examine the dataset during the creation. This is helpful for further adaptations and aggregations which leads to an increased degree of generalization. For example, by avoiding class imbalance within a generated dataset. By performing dataset aggregation based on dataset analyzes, it is able to increase the quality and quantity of the target dataset. Following, two used techniques which are used in this thesis are described:

- keep underrepresented labels: All images containing underrepresented labels are collected and taken for the new aggregated dataset.
- reduce overrepresented labels: Remove images containing only overrepresented labels if they are over a defined threshold.

By applying the aggregation steps leads to a balanced dataset. This means that the instances are uniform distributed over all classes so that there does not exist any underrepresented label anymore. This subsequently results in an improved precision for each label when performing the learning task. In Table 4.2 the imbalance between under- and overrepresented classes are examined. Therefore, the datasets Berkley Deep Drive (BDD) and CityScapes (CS) are compared with the resulting dataset which is called *keepUR*. The new dataset has a reduced class imbalance from ratio 1:48 to a ratio of 1:20. This is reached by discarding over 50% of the source datasets. In detail, the class *car* is highly overrepresented in the dataset BDD which is decreased to create the *keepUR* dataset.

The aggregation of the discussed open-source datasets for classification results in two main datasets which are called *Traffic Lights Merged (TLM)* and *Traffic Signs Merged (TSM)*. As implied by the name, *TLM* is used for traffic light classification and *TSM* for traffic sign classification.

Table 4.2: Dataset aggregation for object detection: It compares the selected datasets BDD and CS with the aggregated and optimized dataset *keepUR* which is based on the others. This leads to a maximum label ratio of 1:20 which means that each class has at least 5% of object instances if the current class is merged with the class with the most instances. The last four columns show the instances of the smallest and the largest class of each dataset and the ratio in % between their own classes per label. The unbalance of the dataset decreases while the under-sampling of the frequent labels is performed in a reasonable scope which is approved by the experiments in Section 5.

Dataset	Images	Labels	min Instances	ratio(%)	max Instances	ratio(%)
BDD	79.847	1.442.297	16.813	2	801.655	98
CS	2.974	83.711	1.182	4	27.155	96
keepUR	30.438	653.575	15.874	5	309.794	95

4. DATASETS

A detailed overview of the label distribution for object detection datasets is shown in Table 4.3. The label instances are categorized into small, medium and large. This subdivision is defined by the authors of the MS COCO dataset. The bounding box area for the 3 categories are defined as follow:

- Small: 0x0 up to 32x32 results in an area up to 1.024px
- Medium: 32x32 up to 96x96 results in an area from 1.025px up to 9.216px
- Large: greater than 96x96 results in an area greater than 9.216px

While dynamic objects are increased in the medium size visible, static objects tend to appear small within the images. Therefore, a detector is needed which can cope with small objects. Finally, the aggregated datasets are divided into training set and validation set with a ratio of 9:1.

Table 4.3: Overview of the label distribution within each dataset divided into small(s), medium(m) and large(l) on the basis of their bounding box areas, respectively 32x32px, 96x96px and greater than 96x96px. For each category the values are shown for object instances and percentage of the whole class.

Label	Dataset	s	s(%)	m	m(%)	l	l(%)	sum
traffic Light	CS	6.218	60%	3.731	36%	288	2%	10.237
	BDD	156.775	86%	23.993	13%	121	0%	180.889
	keepUR	75.420	73%	13.859	25%	280	1%	89559
traffic Sign	CS	13.037	62%	6.997	33%	834	3%	20.868
	BDD	173.236	73%	59.688	25%	4.125	1%	237.049
	keepUR	85.493	68%	30.185	29%	2.115	3%	117.793
person	CS	5.293	29%	9.150	50%	3.551	19%	17.994
	BDD	41.598	45%	43.805	48%	5.516	6%	90.919
	keepUR	31.663	38%	35.023	49%	5.963	12%	72.649
car	CS	4.278	15%	13.083	48%	9.794	36%	27.155
	BDD	299.279	42%	276.653	38%	135.716	19%	711.648
	keepUR	127.710	29%	121.092	43%	60.992	26%	309.794
truck	CS	75	6%	445	37%	662	56%	1.182
	BDD	6.560	15%	17.953	43%	17.229	41%	41.742
	keepUR	6.635	11%	18.394	40%	17.882	48%	42.911
biker	CS	1.113	17%	3.428	54%	1.734	27%	6.275
	BDD	4.032	27%	8.100	55%	2.563	17%	14.695
	keepUR	5.139	22%	11.474	55%	4.256	22%	20.869

4.4 Test Data

The developed pipeline of this thesis copes with different computer vision topics. The generated framework containing the whole pipeline consists of different stages whereby each stage needs its own metric to evaluate the performance and accuracy of the developed approach. While in Section 4.1 and in Section 4.2 the training and validation data are described to train and validate the learning model, this section describes the process of creating custom test datasets for each stage within the pipeline. To guarantee evaluation over the whole framework and of each stage, test data for each specific task is developed. These tasks are object detection and the impact of data aggregation as well as object localization. For object detection evaluation the CS validation set is used. The dataset contains 500 images and their semantic masks are transformed into bounding box annotations which results in 15.970 annotated objects within 497 images. To further evaluate the trained models and the impact of data aggregation, a custom test dataset for object detection is created. The custom datasets are independent from the training data to guarantee no data bias from the source datasets during the final evaluation of the developed approach. The images for the test dataset are taken in Seestadt Aspern by AIT. Sequences are extracted of the auto.Bus project recordings. The self-captured images are manually annotated and classified. For object localization evaluation only self-captured images with their GPS data are taken due to high accuracy reasons of the captured signals. In the following subsections, the self-captured images and signal properties are described. Furthermore, to enable a seamless evaluation the different test data compositions of the whole framework are discussed in detail.

4.4.1 Self-Captured Images

The recordings for the auto.Bus project are captured with 2 different vehicles: a car and a self-driving bus. On both vehicles a monocular RGB camera is mounted on the windshield on top of the vehicle to capture the whole scene in driving direction. In the early stage of the project, a car is used for generating the first test data. During the work of this master thesis, an autonomous bus in Seestadt Aspern started its test operation and collects test data too. A comparison of the two setups is shown in Table 4.4. While the car recordings have a higher resolution and an increased vehicle velocity, the bus recording has an accurate GPS signal but contains stopping scenes within the bus drive which are discarded. Compared with the bus sequences, the images captured by the car provide an increased driving velocity and an increased driving area. This results in a higher motion blur due to higher car velocity and driving vibrations. The GPS signal of the camera mounted on the car has an accuracy of up to 10m and the signal is temporally lost or wrong if the car drives through an underpass or nearby high-rise buildings. So, the car recordings are used to proof hypotheses belonging to the object detection and classification stage due to the fact that the driving area contains the route of the autonomous bus containing all relevant traffic signs while the bus recordings are used to validate the GPS position estimations.

Table 4.4: Comparison of the recordings from Seestadt Aspern.

	Car	Bus
recording date	30.07.2019	10.10.2019
vehicle speed	up to 40km/h	up to 15km/h
FPS	30	15
GPS deviation	over 5m	up to 10cm
total number of frames	1.620	18.500
number of selected frames	336	5.958
total number of labels	3.292	14.638

Relevant frames and sequences are annotated and the GPS positions of static objects within the driving area are manually measured by using different online sources containing satellite images and GPS data which is the same annotation approach as already done by Zhang et. al [ZFL21]. The custom datasets are challenging due to light changes and various dynamic road users as well as over- and underexposure which is present in the captured data too. Figure 4.8 shows two images of the test data containing overexposure, underexposure, windshield effects and low resolution. Furthermore, the image resolution of the bus recording is decreased compared with the images of the open-source datasets used for training.



Figure 4.8: Two images of the test data. On the left image underexposure and windshield artefacts are visible, while the right image shows overexposure and low image resolution.

To manually annotate the selected sequences, 2 free available annotation tools are explored and used to add the bounding boxes to the custom data:

- Scalabel²
- Computer Vision Annotation Tool (CVAT)³

Both frameworks have a clear UI, an extensive documentation and are still maintained by the authors. Scalabel is used for single image frame annotations which are mainly

²<https://github.com/scalabel/scalabel>

³<https://github.com/opencv/cvat>

used for object detection and classification purposes. The tool was developed by BDD research group [YXC⁺18] and supports a comprehensive range of functionalities to produce annotations for computer vision models. CVAT is needed to annotate whole video sequences or subsequences. While Scalabel gets performance problems and other errors during the annotation progress when annotating image sequences, CVAT handles the task without any problems.

4.4.2 Data used for Object Detection Evaluation

To guarantee a correct and unbiased evaluation, the test data has to be independent from the training data. While the trained models are based on the open-source datasets BDD and CS, different subsets are used for performance measurements. Taking test data from various sources helps to increase the complexity of the test data. For example, the data contains varying driving environments and object appearances as well as different capturing conditions. Evaluating models on different test data shows the robustness of the trained models based on objects from the test area compared with other open-source datasets. The three resulting test datasets consist of 833 images for object detection evaluation:

- CS validation set consists of 497 images with 15.970 annotated objects [COR⁺16]
- **Seestadt1** is the smallest subset with focus on challenging scenes from the testing areas. Therefore, 20 different recordings are explored and 54 images are extracted and bounding box annotations are manually created.
- **Seestadt2** consists of an annotated image sequence. One FPS is extracted which results in 282 extracted and manually annotated images.

The images for the custom test subsets **Seestadt1** and **Seestadt2** are collected within a specific region in Seestadt Aspern and subsequently annotated. Figure 4.9 describes the selected regions. Area (*a*) contains an increased appearance of significant traffic signs and other dynamic objects and area (*c*) concentrates on traffic lights for different road users like drivers and pedestrians. Area (*b*) places an emphasis on dynamic objects. This area includes a bus station which tends to a higher number of pedestrians waiting for the next bus within this region. To further increase the degree of difficulty the selected regions contain unseen traffic signs and unexpected vehicles.

While area (*a*) is visualized in Figure 4.10, in Figure 4.11 area (*c*) is shown in detail. Both pictures are satellite images from Google Maps⁴. The static objects are drawn into the satellite images to show their distribution and to visualize the scene complexity. Figure 4.10 indicates an increased local accumulation of direction signs and signs for pedestrian crossing. These signs are close to each other which is challenging for object detection as well as for object tracking in the case of sign overlapping or occlusion.

⁴<https://www.google.at/maps/@48.2260153,16.5057103,16.96z?hl=de>



Figure 4.9: Evaluation areas in Seestadt Aspern. The zoomed-out map shows the official driving route of the bus. The displayed detail of the map figures essential evaluation areas. Each area is focused on a specific scenario: a) roundabout with 14 traffic signs, b) bus station with number of dynamic obstacles and c) traffic lights with different states.

In Figure 4.11 traffic signs and traffic lights are visualized. This scene is complex due to the high number of detectable objects. There are up to six different traffic lights on a single GPS location due to different driving and walking directions and different traffic light types for different road users.

In Table 4.5 the label distributions are listed in detail. The CS subset is the biggest one. It is well distributed with an annotation ratio less than 1:3 between the classes except the label truck which is underrepresented with 235 annotated instances. During the creation process of **Seestadt1** and **Seestadt2** care has been taken to create test datasets with a high variance in object appearance within the images and over the whole subset by taking images with sufficient time delay like one FPS or selecting frames manually by avoiding selecting multiple images which contain the same physical object in it. Furthermore, the custom subsets contain diverse traffic signs of different types. As a disadvantage, bikers are underrepresented due to a lack of captured biker during recording time. Figure 4.12 shows this circumstance too. In **Seestadt2** traffic signs are highly present as well as in **Seestadt1**.



Figure 4.10: Visualized street map of evaluation area (a). The signs are shown on the left side whereby the locations are marked in the map by colored numbers.

Table 4.5: Overview of size and label distribution for each dataset.

	CS	Seestadt1	Seestadt2
biker	1.880	9	22
car	4.667	300	245
person	3.419	71	368
truck	235	56	271
traffic light	1.661	281	150
traffic sign	4.108	425	1.094
sum of labels	15.970	1.142	2.150
images	497	54	282

4.4.3 Data used for Localization Evaluation

The GPS coordinates of the traffic signs as well as the GPS signal of the bus during the bus ride are taken as ground truth to evaluate the calculated locations of traffic signs and the bus itself based on the previous estimations. To decrease inference time the data captured by the bus is cropped from resolution 1280x1056px to 1280x760px by assuming that on the lower area of the images are no traffic signs visible. As seen in Figure 4.13

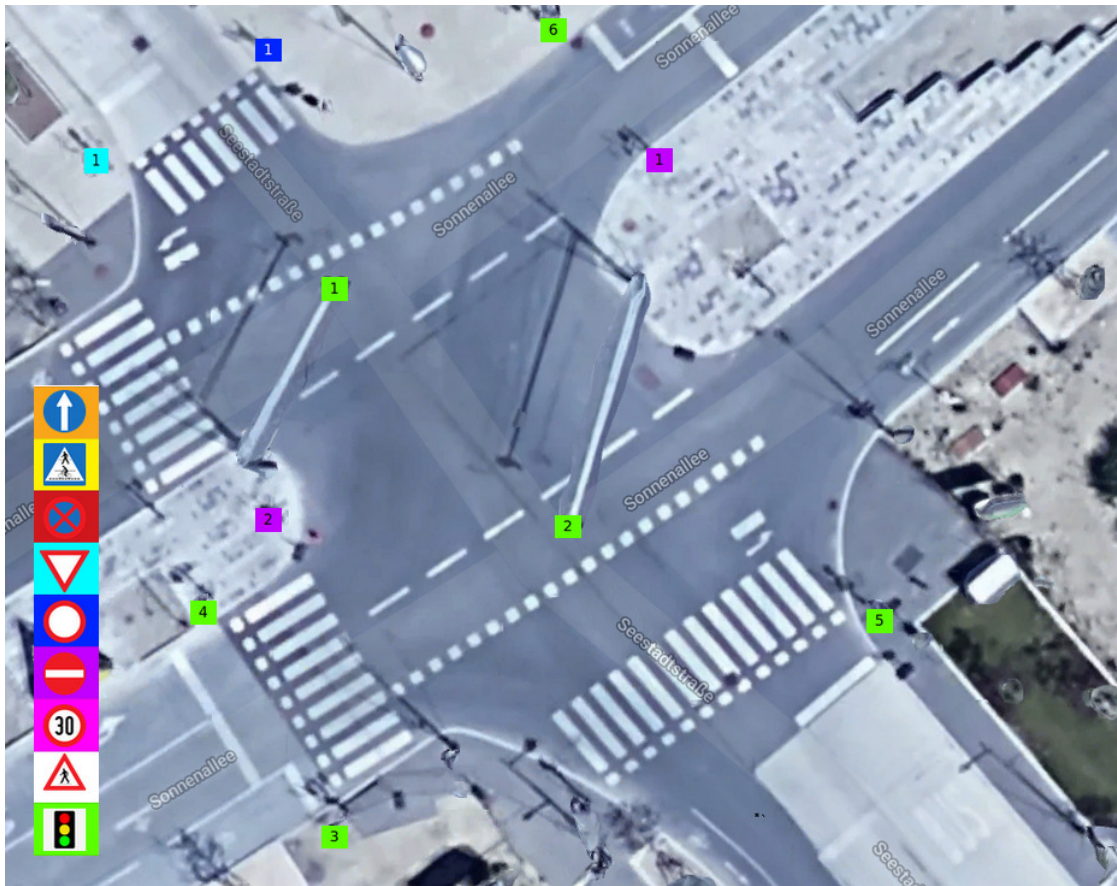


Figure 4.11: Visualized street map of evaluation area (c). The signs are shown on the left side whereby the locations are signed in the map by colored numbers.

not relevant image regions are removed which containing mainly road segments.

During the drive 1.219 GPS signals are captured and logged for the whole sequence which contains 18.500 frames in total. The maximum distance between two consecutive GPS locations of the bus is four meters and the average distance is 1.8 meters. For the sake of completeness, the minimum distance is zero meter in the case when the bus stops. This results in a maximum speed of 14.4 km/h which is slow compared to other common motorized road users which are allowed to drive up to 50 km/h in the current driving area. To increase the amount of data usable for evaluation, each frame gets its own current bus position. Therefore, the latitude and longitude values are interpolated between two consecutive GPS signals. For each synchronized frame which was recorded within the time delay of the two GPS signals, an interpolated GPS position is calculated and set. Figure 4.14 shows the capturing route of the bus. The blue line is the interpolated GPS signals over the whole driving distance which fit the drivable area. Start and end point are on the right upper corner. The bus stations are labeled with blue circles and are

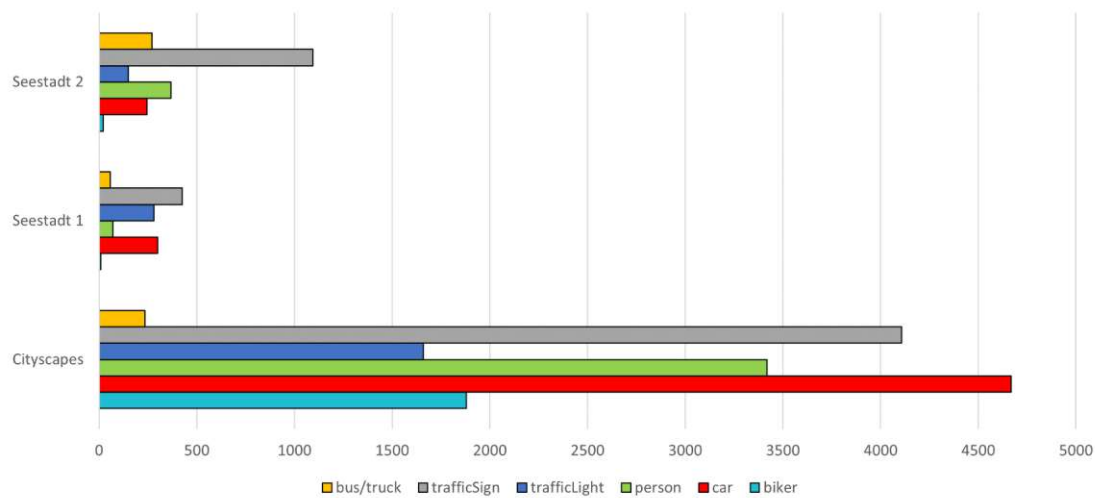


Figure 4.12: Label distribution of the final evaluation datasets for object detection.



Figure 4.13: Example of an image recorded by the bus. On the left side the raw image is shown with the planned image cut and on the right side the cropped image is visualized.

visited by the bus from both road directions except for the station on the left bottom corner. Furthermore, the bus stops for up to one minute at these positions. There are 30 red and purple dots beside the driving road. These marks show the GPS positions of all predefined static objects within the evaluation space. The red dots present a single sign whereby the purple dots show two signs at the same position. This results in 33 traffic sign instances captured from different driving sides along the streets. The GPS positions of the signs are identified and collected manually. This is done by manually extracting the sign positions from Google Maps satellite images.

Due to comparison purposes and to show strengths and weaknesses in different driving scenes of the developed approach, the recorded sequence is divided into subsequences. Therefore, regions are selected containing specific traffic signs combined with other



Figure 4.14: Seestadt Aspern GPS evaluation area. The blue circles indicate areas of bus stops whereby the red and purple dots are traffic sign locations. The red point indicates one and the pink two traffic signs on the same position.

challenging situations. In Figure 4.15 the identified subsequences are visualized. In total there are 19 scenes defined and extracted from the bus trail. Different scenarios are captured like traffic intersections, curvy roads or a high volume of different traffic signs. The subsequences which are drawn and connected in a grey box indicate that they are captured on the same road area but from different street sides. The sequences are coming closer to the essential traffic sign position from different viewpoints and the signs are captured from frontface and backface. The red and purple circles show the traffic sign positions. Whereby red icons contain one traffic sign at a specific position and purple icons mean that there are two traffic signs at the same position. The red lines are the interpolated latitude and longitude coordinates from the selected subsequence. A sequence contains one traffic sign as seen in *SEQ02* up to 8 traffic signs which is shown in *SEQ09*. Further challenges are that the signs are occluded like in *SEQ03* or the driving area is curvy like in *SEQ02* or *SEQ08*.

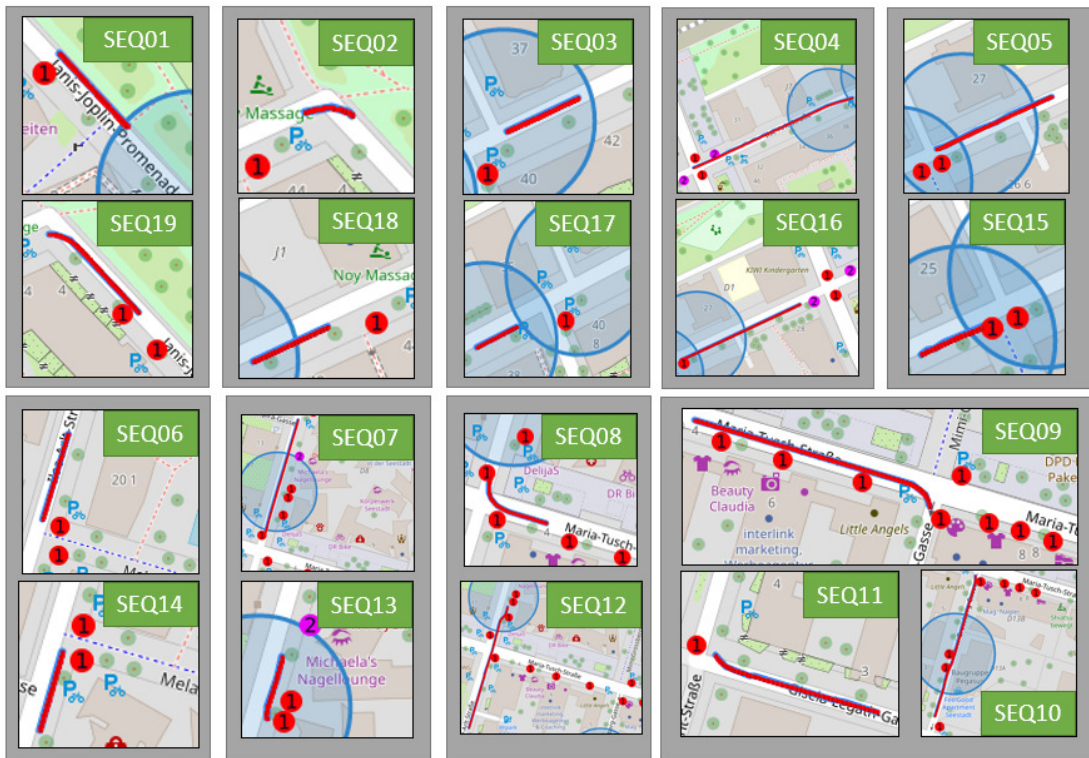


Figure 4.15: Sequences visualized with traffic sign position and bus locations. The different grey boxes represent sequences in the same street map area with the same traffic signs but captured from different driving directions on the road.

In Table 4.6 the extracted subsequences are listed and analyzed in detail. Whereby the longest sequence *SEQ12* contains 839 images and the shortest one *SEQ17* 74 images. The subsequences have up to 3.004 annotated traffic signs and a maximum bus driving distance of approximately 106 meters. This is essential to make assumptions of bus to sign distances for the final evaluation.

4.5 Summary

In the scope of this thesis, multiple open-source datasets are aggregated to construct custom target datasets for training object detection and classification models. Furthermore, test datasets are defined to evaluate the developed approach. One part of the test subsets is focused on the evaluation of the object detection parts by using open-source data and custom datasets. The other test subsets are used for evaluating the localization parts.

Table 4.6: Extracted subsequences of the bus trail recordings. Each sequence has a unique name and the contained images and visible object instances (Signs) are listed. Furthermore, the bounding box annotations (Annotations) for object detection evaluation is shown. Essential metrics for object localization are the whole driving distance (Driving Dist.(m)) and the camera to object distance (Sign Dist.(m)).

Sequence	Images	Signs	Annotations	Driving Dist.(m)	Sign Dist.(m)
SEQ01	180	1	177	24.43	15.48 - 39.03
SEQ02	76	1	64	9.82	17.81 - 25.13
SEQ03	121	1	115	15.17	17.14 - 31.31
SEQ04	667	6	3.004	106.21	5.72 - 114.53
SEQ05	324	2	598	41.45	8.99 - 49.38
SEQ06	136	2	214	24.43	9.61 - 34.89
SEQ07	562	7	1.790	76.24	3.96 - 85.94
SEQ08	186	2	197	29.46	6.53 - 31.78
SEQ09	479	7	1.237	78.09	4.67 - 62.50
SEQ10	635	3	1.132	90.59	4.67 - 59.68
SEQ11	279	1	273	51.68	6.87 - 57.92
SEQ12	839	5	2.431	92.72	7.32 - 73.27
SEQ13	117	2	221	17.30	11.64 - 28.14
SEQ14	101	2	107	20.06	8.58 - 20.06
SEQ15	238	2	433	18.63	5.58 - 26.85
SEQ16	617	6	2.183	62.36	7.66 - 93.58
SEQ17	74	1	69	14.01	16.09 - 29.64
SEQ18	106	1	106	21.68	12.66 - 33.90
SEQ19	221	2	287	34.72	11.72 - 36.34

Evaluation

This chapter investigates the performance and accuracy of the developed approach up to the final localization. To identify the strengths and weaknesses of individual parts within the defined pipeline, different datasets and metrics are used for the evaluation in multiple experiments. As already mentioned in Section 4 to cope all aspects of the pipeline, the evaluation data consists of an existing open-source test dataset and a manually annotated image data which increases the data variety. The accuracy of the developed pipeline is the result of three major tasks, namely: The detection accuracy of the trained models, the accuracy after applying refinement and optimization approaches on the detections results, and the accuracy of the final localization technique used to estimate objects in world space. The used evaluation metrics are divided into two categories: one to measure the detection performance and the other one to measure the final localization error. At first in Section 5.1, the evaluation metrics and their properties are introduced. Afterwards, in the Sections 5.2, 5.3.1 and 5.4 the selected experiments are evaluated and the results for detection and localization are presented. Finally, in Section 5.6 the results for the different tasks are discussed and limitations are pointed out in Section 5.5.

5.1 Evaluation Metrics

Since the proposed methodology involves multiple steps to produce the final output, it is essential to use evaluation metrics that are specific to the use case in each step to analyze preliminary as well as the final results of the developed approach. Thus, each part of the developed framework is evaluated by its own. A special focus is set to the data aggregation and the final static object localization in the scope of this thesis. Within this Section, the metrics are described and restrictions with respect to location measurement, ground truth completeness and other environmental constraints are discussed and taken into account which are essential during the evaluation study.

Accuracy of Detection

To obtain the accuracy of a detection model, a confidence score for each prediction and an associated classification probability is calculated and combined. The confidence score indicates the probability of a bounding box containing an object or not and the classification probability shows the correct label classification of a bounding box. The IoU metric is based on the Jaccard index [Jac01]. As shown in Figure 5.1 this metric indicates the overlap between predicted and ground truth bounding boxes [PNDS20]. The higher the index, the better the areas are matching.

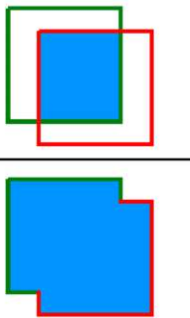
$$IOU = \frac{\text{area of overlap}}{\text{area of union}} = \frac{\text{area of overlap}}{\text{area of union}}$$


Figure 5.1: Illustration of the IoU metric [PNDS20].

For the evaluation of the object detection parts the standard metrics of the popular PASCAL VOC benchmark [EEVG⁺15] are used which are $\text{IoU} > 50\%$ to consider a correct prediction and the mean Average Precision (mAP) to measure the performance of an object detector. The mAP is based on the Average Precision (AP) which is the interpolated average precision by summarizing the shape of the Precision x Recall curve. As seen in Equation 5.1 a set of equally spaced recall levels r starting from zero up to one with 0.1 step size which results in eleven recall levels is summed up. To get the average the sum is divided by eleven:

$$AP = \frac{1}{11} \sum_{r \in [0, 0.1, \dots, 1]} p_{\text{interp}}(r) \quad (5.1)$$

$p_{\text{interp}}(r)$ is the maximum precision of the interpolated recall level r with $p(\tilde{r})$ as the measured precision at recall \tilde{r} :

$$p_{\text{interp}}(r) = \max_{\tilde{r}: \tilde{r} \geq r} p(\tilde{r}) \quad (5.2)$$

So the precision is not observed at each point of the curve but approximated by interpolating the precision at eleven recall levels. The mAP in Equation 5.3 averages over all classes the AP:

$$mAP = \frac{1}{N} \sum_{i=1}^N AP_i \quad (5.3)$$

whereby i is the index of the current class and N the total number of classes [PNDS20]. Further essential metrics are precision (Equation 5.4), recall (Equation 5.5) and F1-Score (Equation 5.6) described by Goutte et al. [GG05]:

$$\text{Precision}(P) = \frac{TP}{TP + FP} \quad (5.4)$$

$$\text{Recall}(R) = \frac{TP}{TP + FN} \quad (5.5)$$

$$\text{F1-Score} = 2 \cdot \frac{P \cdot R}{P + R} \quad (5.6)$$

While the *Precision* shows how many of the predictions are correct predicted objects, the *Recall* provides how many of the relevant objects are predicted. The *F1-Score* uses both, *Precision* and *Recall*, and is defined as the average of precision and recall [GG05]. F1-score, precision, recall and mAP are common metrics to evaluate object detection models. Within this thesis they are used to measure the quality of the model predictions and their further adaptations by comparing the results with the given annotations of the test datasets.

5.1.1 Accuracy of Localization

The localization accuracy is examined by evaluating the error of the relative distance between the bus and the tracked object in cm as well as the absolute error of the predicted location and the real object location. The first one is the distance error whereby a relative error regarding the current bus location is elaborated. In detail it is the estimated distance between camera and located objects in cm. The second one is the longitudinal distance error or GPS error in longitudinal and lateral position. It is used to calculate an absolute spatial error which is performed by calculating the distance between the real object location to the estimated object location in cm. This shows the displacement between the estimated and the ground truth values which is an established accuracy metric in mapping science [Chr91].

5.2 Evaluation of Object Detection

To investigate the efficiency of training from scratch based on data aggregation instead of using a pretrained model trained on a single dataset the object detection and classification results are examined and discussed. At first the data aggregation itself is explored by aggregating two specific open-source datasets to test the assumptions that data aggregation reduces dataset bias and leads to a balanced dataset. Furthermore, advantages and disadvantages of this technique are established. Afterwards, the detection approach combined with object classification models is analyzed and compared to a pretrained YOLOv3 model. It is expected that the models from scratch result in an increased accuracy due to increased dataset variability and decreased dataset bias.

5.2.1 Data Aggregation

Data aggregation is used to combine different datasets and reduce them to a decreased target subset with similar variability of the source dataset but containing less images which have low impact on the resulting prediction accuracy due to object similarities and over-representations. This hypothesis is tested by aggregating target datasets from different open-source datasets and subsequently compare the results of the models with each other.

Experiment Setup

The experiments compare the prediction results of models trained on aggregated datasets to make statements about effectiveness of data aggregation compared with random image selection. The models are trained on target datasets which are based on *BDD_CS* dataset described in Section 4.3. The *BDD_CS* dataset is a combination of the open-source datasets BDD and CS which are introduced in Section 4.1. It is expected that if the data is aggregated by applying specific rules that it results in an increased detection accuracy compared to randomly reducing a dataset. Furthermore, extending a dataset by another one increases the dataset variety and consequently results in an improved detection model trained on the adapted data. Based on these assumptions the following four target datasets are generated:

- $D1_{rdm}$: Images are randomly selected.
- $D2_{rdm_bal}$: Images are randomly selected but with label balancing which means an increased selection priority for images containing objects from underrepresented classes.
- $D3_{agg}$: Images selected by aggregation techniques discussed in the following paragraph.
- $D4_{agg}$: The same as $D3_{agg}$ but only containing images from BDD dataset to examine the impact of the CS dataset.

The first two datasets are randomly created whereby the last two datasets are aggregated with the following aggregation steps:

1. Label balancing: The overrepresented classes are reduced up to 150.000 object instances. This is produced by removing images where only overrepresented objects appear. For example, the class *car* is overrepresented thus all images containing only annotations of the label *car* are discarded until the class *car* has reached 150.000 annotations which are remaining within the dataset.
2. Minimum bounding box width and height: Afterwards, bounding boxes are ignored which have less than 10px width or height because the focus is set to near or medium viewing range and tiny objects far away are out of scope.
3. Certain bounding box shape: Due to the fact that the dataset annotations are partly derived from semantic masks, cropped objects are discarded if less than 20% of the object is visible. For example, a car moving out of the scene where less than 10% of the car is visible in image space is discarded. Therefore, it is assumed that the bounding box appearance must follow a maximum width to height ratio of 1:12 to ensure that the object is truncated and less than 20% visible. If this assumption is true for an annotation, the annotation is removed from the dataset.

To show the impact of the specialization on underrepresented classes by data aggregation and not by adapting parameters at model training, the datasets are reduced to a small amount of 3.000 images per subset which is less than 5% of the images from the source datasets. This demonstrates strength and weaknesses when selecting specific input data based on aggregation strategies. Subsets which have after the aggregation stage over 3.000 samples are random reduced to the specified amount of 3.000 images. In Table 5.1 the distribution of the bounding box size is shown. For all object sizes $D1_{rdm}$ and $D2_{rdm_bal}$ as well as $D3_{agg}$ and $D4_{agg}$ are similar distributed. Both random subsets have a higher amount on annotated bounding boxes compared with the other two subsets. While $D2_{rdm_bal}$ contains the most annotated objects, $D4_{agg}$ has the lowest amount on annotated objects.

Table 5.1: Aggregated detection datasets of 3.000 images with varying bounding box distribution.

Dataset	Images	Labels	Small(%)	Medium(%)	Large(%)
$D1_{rdm}$	3.000	65.287	52%	35%	13%
$D2_{rdm_bal}$	3.000	69.188	50%	36%	14%
$D3_{agg}$	3.000	61.757	40%	43%	17%
$D4_{agg}$	3.000	56.684	44%	41%	15%

Figure 5.2 shows the relative label distributions by class for each subset. The number of static objects is increased within the random datasets whereby the underrepresented classes *truck* and *biker* are increased in $D3_{agg}$. Cars are highly represented in all four

subsets with over 40% but compared with the source datasets the annotations of cars are decreased for all target datasets. While $D1_{rdm}$ has a label ratio of 1:14, the $D3_{agg}$ subset has a label ratio of 1:7. Thus, the latter one has an increased balance of the label appearance compared with the other ones.

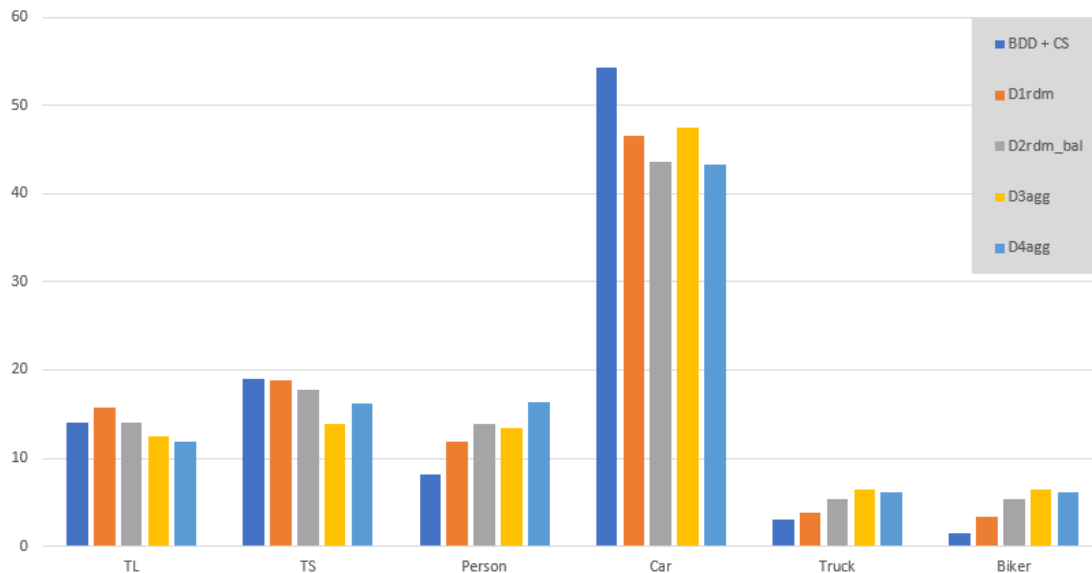


Figure 5.2: Relative label distribution of the aggregated detection datasets by class. On the x axis the labels are listed while the y axis shows the label percentage of annotated objects from label in relation to the total annotations of a dataset. The $BDD+CS$ dataset is the combination of the source datasets BDD and CS and was added for comparison purposes.

Model Training

After finishing the data preprocessing step, the trained models from scratch based on the aggregated detection training data are created and evaluated. Therefore, each dataset ($D1_{rdm}$, $D2_{rdm_bal}$, $D3_{agg}$ and $D4_{agg}$) are split into 90% train set and 10% validation set for observing the training progress. The used data augmentation techniques are fixed to random horizontal flip with $p=0.5$ and normalization with mean and standard deviation of the training set. The input image is resized by scaling it to 1024x608px and bicubic interpolation and batch size of 10. The image is resized to allow real-time performance and relatively similar to the final test setup of the bus recordings. Within the RetinaNet architecture resnet101 is set as backbone which has twice as many parameter than the default backbone resnet50. This results in an increased accuracy and in an increased inference time compared with resnet50. The models are trained up to 40 epochs and the checkpoints are saved after each epoch. After the training the model from the checkpoint is taken which has the highest mAP on the validation set.

Results

To compare the trained models and to get essential findings from the underlying experiments the models are applied on the defined test sets CS test set, *Seestadt1* and *Seestadt2* which are presented in Section 4.4.2. The detection results of the trained models are shown in Table 5.2 for each defined test set. The mAP is for all experiments low with 25% up to 32%. It can be inferred that this is an effect of the reduced number of training samples per subset. However, the evaluation results support the assumption that aggregated datasets have a decreased bias and an increased class balance which results in increased mAP because the mAP takes the average over all classes. Over all test sets the class *traffic light* is hard to predict correctly for all trained models. This can be due to the complexity of finding small static objects within the scene. Another reason can be that all datasets have problems with this class because the underlying source datasets do not cope with the traffic light appearance and variability or the object annotation itself of the test sets.

Table 5.2: Detection results from object detection models which are trained on the generated datasets. The first two columns show the test and training dataset while the following columns present the AP for each label. In the last column the mAP over all labels is shown. The highest values of each test set (Test Dataset) and each evaluation metric are highlighted in bold type.

Test Dataset	Training Dataset	TL	TS	person	car	truck	biker	mAP
CS	D1 _{rdm}	12.3	18.7	28.6	56.9	38.2	24.1	29.8
CS	D2 _{rdm_bal}	12.6	20.4	29.5	59.0	40.8	28.3	31.7
CS	D3 _{agg}	14.8	21.6	30.7	58.2	40.5	30.2	32.7
CS	D4 _{agg}	13.0	17.2	26.2	57.6	40.6	28.5	30.5
Seestadt1	D1 _{rdm}	15.3	19.9	22.5	50.3	31.7	16.7	26.1
Seestadt1	D2 _{rdm_bal}	13.4	16.3	16.9	54.4	40.1	35.8	29.5
Seestadt1	D3 _{agg}	13.6	17.5	24.6	49.0	42.5	41.3	31.4
Seestadt1	D4 _{agg}	16.5	18.6	19.4	53.1	39.6	28.4	29.3
Seestadt2	D1 _{rdm}	6.6	28.8	46.8	46.2	23.8	1.1	25.6
Seestadt2	D2 _{rdm_bal}	4.7	25.4	37.7	55.5	42.5	0.7	27.7
Seestadt2	D3 _{agg}	7.4	25.8	45.8	60.8	46.0	0.3	31.0
Seestadt2	D4 _{agg}	9.6	28.0	44.5	48.7	31.5	4.6	27.8

Figure 5.3 shows the complexity of traffic light detection where objects of the same class are close together as seen by the four object instances within the image crops. While the bigger two traffic lights are clearly distinguishable, errors occur when distinguishing the two objects on the lower half of the image crop which are smaller with a resolution of 10x25px per instance and close together. Furthermore, the incorrect prediction on the left side from D3_{agg} shows that the object appearance of small and long bounding boxes tends to be considered as FP if the bounding box is not overlapping the object with an IoU less than 50%. It shows that the ground truth of the bounding box in the test set is

drawn close to the object boundaries whereby the annotations of the source dataset are not that accurate. This indicates that the annotations from the source datasets have to be as accurate to pass the evaluation metrics on the test data. In this case accurate means that the bounding box annotations are set close to the object boundaries.

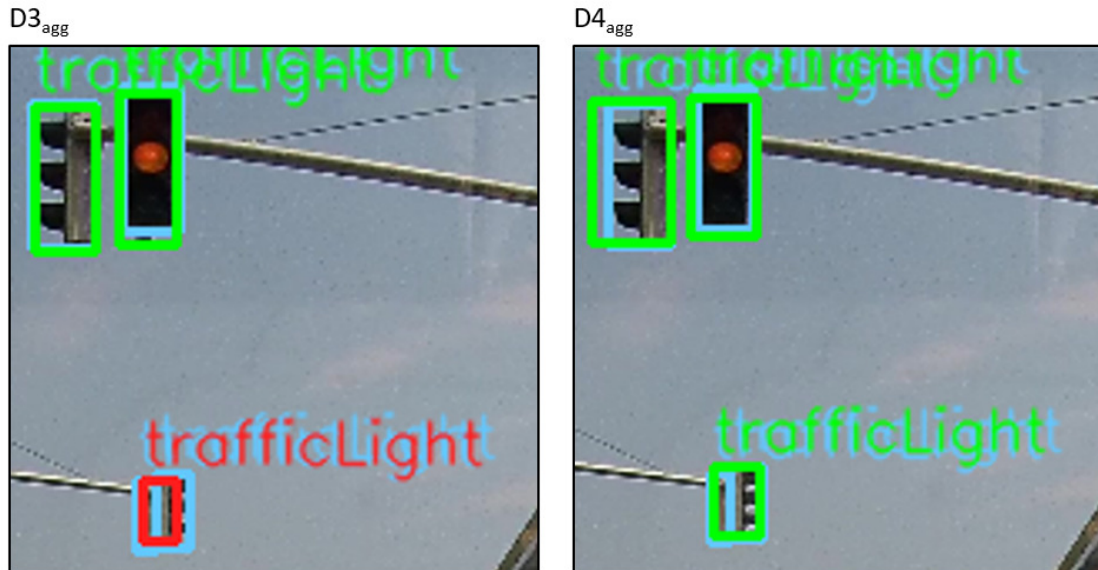


Figure 5.3: Example of a cropped region containing detection results of the class *traffic light*. A result from $D3_{agg}$ on the left side is compared with the result of $D4_{agg}$ on the right side. The light blue lines show the ground truth. Green bounding boxes are correct predictions whereby red ones are wrong predictions. The objects have a varying resolution between 10x25px and 20x40px.

Furthermore, the class *biker* in the *Seestadt2* test set has a low accuracy of under 5% over all detection results. It can be inferred that the trained model based on $D3_{agg}$ performs best for all three test sets due to the highest mAP for all three test sets. As in the experiment, creating a dataset by specific aggregating techniques has a positive impact on the target dataset quality which results in an increased performance of the resulting model compared with a dataset based on random selection. The second-best dataset is $D2_{rdm_bal}$. It shows that it has an impact if the selection process is fine-tuned on selecting underrepresented classes. The underrepresented class *truck* has an increased accuracy when trained with $D3_{agg}$, $D4_{agg}$ or $D2_{rdm_bal}$ compared with the random selected dataset $D1_{rdm}$ which has a lower accuracy of 2.6% up to 22.2% compared with the other ones. The final assessment is about the impact of using different dataset sources to increase variability. A finding is that the mAP of $D3_{agg}$ is increased compared with $D4_{agg}$ over all three test sets which supports the assumption of increased dataset quality by adding annotations from the CS dataset. By investigating the single classes traffic light and traffic sign, $D4_{agg}$ has a higher accuracy compared to $D3_{agg}$ for test set *Seestadt1* and *Seestadt2* and a lower one for test set *CS*. This can be due to the number of missing

annotations of static objects within the CS source dataset so that the model trained on $D3_{agg}$ gets penalized during training when detecting a not annotated static object correct.

5.2.2 Pretrained Model vs Adapted Model from Scratch

After showing the efficiency of data aggregation on small subsets and their findings are discussed, models are trained on enlarged datasets based on data aggregation and compared with each other. This tests the assumption that the impact still exists when using a larger target dataset for training. Therefore, the same test sets are used like in the section before. Furthermore, aggregation steps and parameter optimizations are evaluated. To point out the importance of understanding all parts of the network as well as the data and having full control during the training process, the evaluation results are compared with another popular network, a pretrained YOLOv3 model. The performance of YOLOv3 is as accurate as RetinaNet [RF18]. The fully convolutional network predicts the bounding boxes and probabilities for each image region at once. Given the huge popularity of the network, Redmon et al. [RDGF16] make further improvements to increase detection accuracy and inference time. Later versions are YOLOv5 published in 2020 [Joc20] and the newest ones YOLOv6 [LLJ⁺22] and YOLOv7 [WBL22]. Other variations followed in 2021 like YOLOX [MLWX22]. An implementation¹ is selected based on the official code of YOLOv3². It was trained on MS COCO and allows detections across different input image scales.

Experiment Setup

Four models are trained whereby each one is trained on one of the following datasets:

- M_{CS} : The model is trained on the CS dataset.
- M_{BDD} : The model is trained on the BDD dataset.
- M_{BDD+CS} : The model is trained on both datasets, the BDD and the CS.
- M_{keepUR} : The model is trained on the CS and BDD datasets aggregated with the defined aggregation techniques which are label balancing, minimum bounding box with and height and certain bounding box shape discussed in the previous section. This results in a target dataset that is focused on keeping annotations of underrepresented classes.

While M_{CS} , M_{BDD} and M_{BDD+CS} take the whole open-source datasets for the training, M_{keepUR} is based on an aggregated dataset of both which is smaller than the combining of the datasets without the data aggregation steps performed. In Figure 5.4 the label

¹<https://github.com/ayooshkathuria/pytorch-yolo-v3>

²<https://github.com/pjreddie/darknet>

5. EVALUATION

distribution over all used datasets is shown. Cars are overrepresented in the open-source BDD dataset. This class is halved in the aggregated dataset *keepUR* to avoid overfitting on a specific class during training.

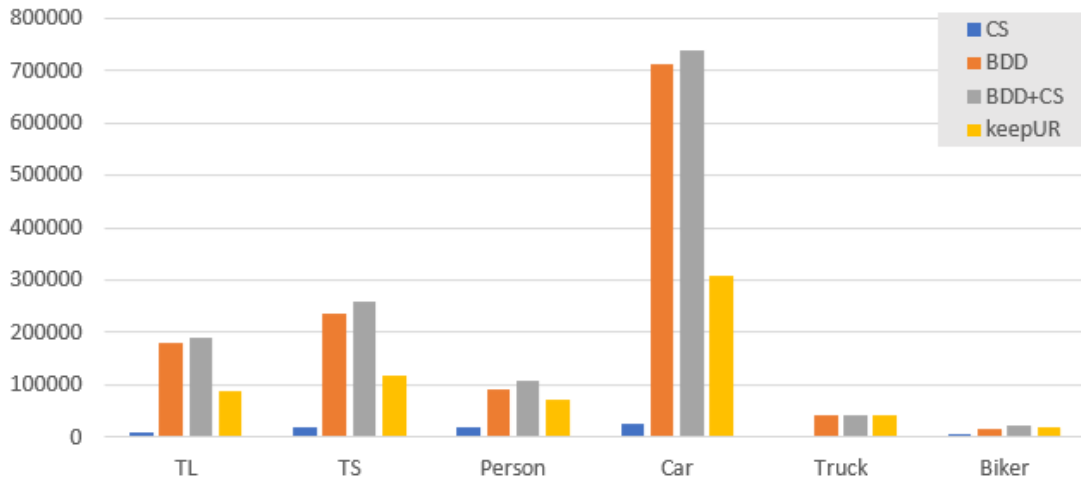


Figure 5.4: The label distribution of two open-source datasets and their combined datasets *BDD+CS* *keepUR* are compared with each other.

The class composition of each label from the *keepUR* dataset is shown in Figure 5.5. Compared with the BDD dataset, the CS dataset is just a small extension for the other dataset over all classes.

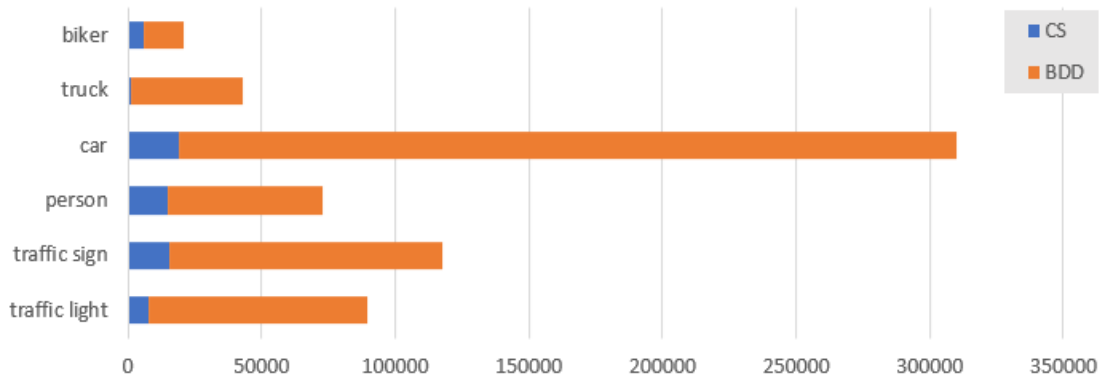


Figure 5.5: Label distribution of M_{keepUR} which is a combination of the two datasets CS and BDD.

Model Training

The training setup for the models M_{CS} , M_{BDD} , M_{BDD+CS} and M_{keepUR} is the same as in the previous experiment. The YOLOv3 model (M_{YOLOv3}) has a different setup due to its training configurations. Both architectures, RetinaNet and YOLOv3 need an input image size which is a multiple of 32 and the trained YOLOv3 model needs a squared image as input. To make the output of the trained model comparable to the models from scratch, the input resolution is set to 800x800px which lies midway between the values 1024px and 608px by downscaling the image width and increasing the height which results in an output image which looks vertically stretched and corresponds to the training data. As a disadvantage no pretrained model was found containing both traffic sign and traffic light predictions as well as the other selected classes. The selected model (M_{YOLOv3}) includes all necessary classes except the class traffic sign.

Results

It is assumed that aggregating different datasets has an impact to the trained model and leads to an increased accuracy. The CS dataset constitutes only 3.75% of the aggregated dataset M_{keepUR} . To test this assumption, the impact of using just the training set of the CS dataset gets evaluated by comparing the model M_{keepUR} with the other models. In Table 5.3 the results of each model for each test set is shown. The last two columns show the mAPs for the predictions. While the first one takes the mAP over all classes, the second one leaves off the traffic sign class to calculate the mAP. This is performed due to the missing traffic sign class in the pretrained YOLOv3 model based on the MS COCO dataset.

It is expected that M_{CS} performs best on the CS test set due to the fact that the data from train and test are from the same source and have equal recording conditions in similar environmental and urban areas. Nevertheless, M_{keepUR} performs best on the CS test subset. This is a result of the aggregation due to selecting samples for the training set which fit the objects in the test set best compared with the other ones. An essential finding is that the models M_{CS} and M_{BDD} have a lower mAP than M_{BDD+CS} and M_{keepUR} . This shows the positive impact of the CS data combined with the BDD data. The pretrained M_{YOLOv3} outperforms all other models when analyzing Seestadt1 (43.4%) and Seestadt2 (45.5%) shown in column mAP (-TS). Compared with the other models, it has a high accuracy for the class *traffic light* which is the main impact on the performance improvement compared with all the other models. This leads to the result that the datasets which are used for training from scratch has a lack on traffic light annotations. Taking a closer look at TS accuracy shows that the aggregated dataset M_{keepUR} performs best over all three test sets. This effect is based on the aggregation which produces a balanced dataset so that underrepresented classes result in increased performance. As a negative effect of the aggregation approach, the performance of the overrepresented class *car* is decreased in M_{keepUR} over all test sets compared with the M_{BDD+CS} model. As already shown in the previous results, the class *biker* in the *Seestadt2* test set performs

Table 5.3: Detection results from YOLOv3 model (M_{YOLOv3}) and models from scratch (M_{CS} , M_{BDD} , M_{BDD+CS} and M_{keepUR}). The best results are highlighted in bold type. The values present the AP for each label, the second last column shows the mAP over all labels and the last column the mAP without the AP of the label *traffic sign* which is not available at M_{YOLOv3} .

Test Set	Model	TL	TS	person	car	truck	biker	mAP	-TS
CS	M_{CS}	15.6	21.0	32.1	58.8	33.4	29.7	31.8	33.9
CS	M_{BDD}	15.1	18.1	31.7	61.6	47.2	27.0	33.5	36.5
CS	M_{BDD+CS}	16.0	17.2	34.0	63.7	50.0	30.3	35.2	38.8
CS	M_{keepUR}	18.8	23.1	34.8	61.6	53.5	37.1	38.2	41.2
CS	M_{YOLOv3}	19.2	-	34.1	53.4	33.5	22.3	27.1	32.5
Seestadt1	M_{CS}	7.6	14.5	17.2	35.1	17.2	37.0	21.4	22.8
Seestadt1	M_{BDD}	20.5	23.5	23.2	62.8	33.9	15.6	29.9	31.2
Seestadt1	M_{BDD+CS}	19.0	20.0	34.8	69.1	40.3	48.4	38.6	42.30
Seestadt1	M_{keepUR}	17.4	21.0	37.2	66.2	45.8	38.8	37.7	41.1
Seestadt1	M_{YOLOv3}	29.7	-	47.5	59.3	49.9	30.6	36.3	43.4
Seestadt2	M_{CS}	4.0	16.8	45.4	40.2	26.4	0.8	22.3	23.4
Seestadt2	M_{BDD}	7.4	30.7	48.6	62.5	38.5	0.6	31.4	31.5
Seestadt2	M_{BDD+CS}	10.3	27.9	63.5	72.2	40.0	0.0	35.7	37.2
Seestadt2	M_{keepUR}	9.6	30.7	73.6	64.0	43.2	2.0	37.2	38.5
Seestadt2	M_{YOLOv3}	22.7	-	77.4	56.2	71.2	0.0	38.0	45.5

quite bad. The performance further decreases with an average precision between 0% up to 2% over all detection results.

5.3 Evaluation of Refinement

After the object detection part is explored, the temporal factor is taken into account to validate detections over time and furthermore evaluate the possible improvements of the refinement steps. Due to the highest AP of the class traffic signs with improvements of up to 13.9% compared with the other models, the M_{keepUR} model is used for the further experiments. The test data for this step within the pipeline is introduced in Section 4.4.3 consisting of 19 video sequences and their relevant GPS locations as ground truth. Furthermore, manual generated ground truth is provided for the relevant traffic signs which consist of bounding box annotations, the object occlusion state as boolean parameter and their unique GPS localizations for each of the defined sequences. To examine each aspect of the current pipeline stage, this evaluation part is split into three consecutive experiments which consists of raw traffic sign detection over time, refinement by classification, and optical flow optimization. It is expected by analyzing each stage after the other that the resulting detection accuracy should be increased. The input images are resized from 1280x760px to 1024x608px for experiments within this section which is equal to the previous experiments.

Table 5.4: Detection results generated by the model M_{keepUR} . Sequence ID, FP, TP, Ground Truth (GT), precision (Prec), recall and f1-score are listed for each sequence.

Sequence	FP	TP	GT	Precision(%)	Recall(%)	f1-score(%)
SEQ01	66	1	177	1.5	0.6	0.8
SEQ02	7	28	64	80.0	43.8	56.6
SEQ03	385	3	115	0.8	2.6	1.2
SEQ04	1.006	1.220	3.004	54.8	40.6	46.7
SEQ05	695	220	598	24.0	36.8	29.1
SEQ06	289	65	214	18.4	30.4	22.9
SEQ07	695	345	1.790	33.2	19.3	24.4
SEQ08	324	12	197	3.6	6.1	4.5
SEQ09	1.005	159	1.237	13.7	12.9	13.2
SEQ10	683	360	1.132	34.5	31.8	33.1
SEQ11	363	204	273	36.0	74.7	48.6
SEQ12	1.595	740	2.431	31.7	30.4	31.1
SEQ13	282	96	221	25.4	43.4	32.1
SEQ14	185	26	107	12.3	24.3	16.4
SEQ15	815	17	433	2.0	3.9	2.7
SEQ16	1.152	856	2.183	42.6	39.2	40.9
SEQ17	204	0	69	0.0	0.0	0.0
SEQ18	89	57	106	39.0	53.8	45.2
SEQ19	444	224	287	33.5	78.1	46.9
ALL	10.284	4.633	14.638	31.1	31.7	31.4

5.3.1 Experiment for Raw Detection Over Time

During this experiment it is desired to predict True Positive (TP)s of static objects to be able to perform multi object tracking by combining the object detection results with an optical flow approach. The previous experiments rate the current performance of the object detectors over time by inspecting the traffic sign detections without any refinements or optimizations. Nevertheless, these experiments show that the performance is too low for further localization calculations with an average precision of 21.0% up to 30.7% from the best performing model M_{keepUR} . Thus, refinement steps and optimizations are added and evaluated. At first the TPs must be increased which is achieved by decreasing the detection threshold from the commonly used value of 50% [EEVG⁺15] to 25% for accepting a predicted bounding box from the detection model. It is expected that FPs are increased too which must be eliminated in the further stages.

In Table 5.4 the detection results for traffic signs of the defined sequences are shown without any refinements. The precision is always under 50% excepted for the sequences *SEQ02* and *SEQ04*. *SEQ01*, *SEQ03* and *SEQ17* have less than 5 correct predictions over the sequence which makes it impossible to localize the objects with the developed approach. It is necessary to have predictions of an object instance over time for the

further steps. Therefore, the three sequences are excluded in the following evaluation sections. In Figure 5.6 the detection problems are analyzed. In *SEQ01* the sign is underexposed and far away from the camera with a minimum object height of 12px up to 43px which is less than 2% up to 6% in relation to the original image height. The images of *SEQ03* contains underexposure too and the traffic sign is mostly occluded by other static objects like a tree. In *SEQ17* as well as *SEQ15* the traffic signs are captured sideways when driving along the streets. The objects are small and long which increases the complexity for the trained model to identify these objects. The sign *noVehicle* from *SEQ15* has light reflections which can further decrease the prediction performance. This means that there are missed signs which are not detected by the model as well as many FPs.



Figure 5.6: Traffic signs which are not detected by the object detection model. Five traffic signs within four subsequences listed with the traffic sign class and its height over the whole subsequence.

5.3.2 Experiment for Refinement by Classification

The experiment in Section 5.3.1 increases the TPs by allowing detections with a low detection score. Table 5.4 shows that the 10.284 FPs are over twice as many as the TPs which are 4.633 instances. It is expected that the FPs are decreased by adding a fine-grained classification model which divides the predictions into relevant and irrelevant detections. For example, during the detection process, there are objects identified as traffic signs which are no signs or signs which are out of scope. Figure 5.7 shows extracts from *SEQ04* of traffic sign predictions which are out of scope for example white arrow signs and signs of street names. In the third column a special sign of Seestadt Aspern is pictured which is the bus stop sign of the autonomous bus. The other FP detection examples are wrongly interpreted regions and other signs which are not used for driving.

It is expected that a fine-grained classification of the detection label *traffic sign* tends to decreased FPs by discarding predicted bounding boxes which are belonging to not relevant object classes. Thus, each detection patch is used as input for the fine-grained

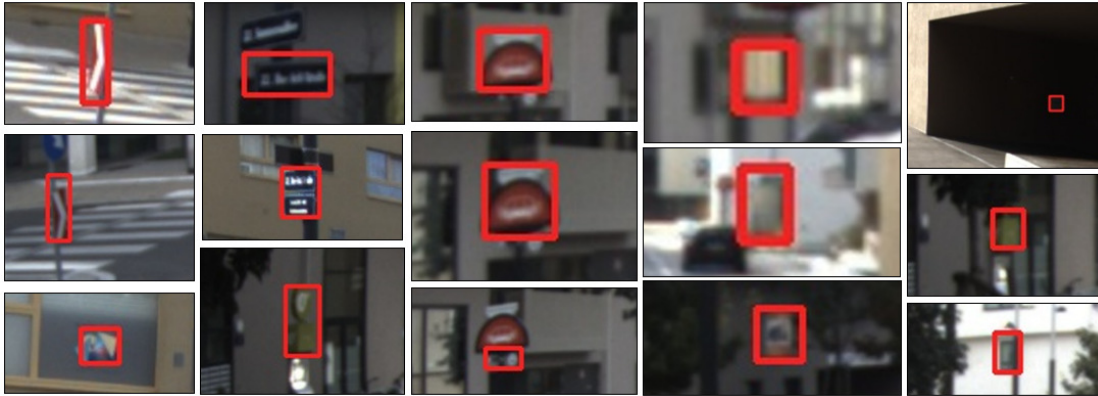


Figure 5.7: Examples of detected traffic signs in SEQ04 which are no regular traffic signs.

classification model presented in Section 4.2 to estimate if a prediction of the object detection model is correct or must be discarded.

To examine the relevance of the classification refinement, classification models are trained. The used dataset for the object classification training contains from 350 up to 3.500 samples per class to ensure that the dataset is balanced which further decreases overfitting on individual classes. The model trained on this dataset is called *CLS1*. During evaluation the dataset samples of the labels *traffic sign back* and *unknown* are reworked manually so that inappropriate samples are discarded and replaced with other ones. For example, *traffic sign back* samples which do not match the appearance of signs from class *traffic sign back* like huge traffic signs from the back mounted on highways or truncated images with over 75% occlusion are replaced. Furthermore, samples are moved from *traffic sign back* to the class *unknown* like white signs and other special signs which are not showing the back side of a sign. The model which is trained on the adapted dataset is called *CLS2*. Both models are trained with an input image size of 256x256px. The used data augmentations are random crop, horizontal flip and Gaussian blur to increase the training variety.

In Table 5.5 the results of both classification models combined with the detection results from Section 5.3.1 are shown. While *CLS1* has a higher recall, *CLS2* mostly has a higher precision and f1-score. This demonstrates that *CLS1* discards less predictions than *CLS2* because there are less signs classified as *unknown* signs. Furthermore, there are over 5.000 FPs at *CLS1* and over 8.000 FPs at *CLS2* rejected compared to less than 400 rejected TPs.

The summarized results of the current and the previous experiment are compared in Table 5.6. By combining fine-grained classification with object detection reduces FP detections while TP are hardly ever rejected. When *CLS1* is applied, 5.301 wrong predictions are removed which are over 50% of the FP predictions and less than 100 instances of correct traffic sign estimations are rejected. The *CLS2* results show the positive impact of the adapted split of *unknown* and *traffic sign back* labels used for the training. The

model is able to identify nearly 8.500 FPs while keeping nearly all correct predicted signs compared to the wrong estimations but less than *CLS1*. By using this refinement step, FPs are drastically reduced. This has a positive impact on the multi object tracking approach due to less wrong predictions which can be mixed with other objects.

5.3.3 Experiment for Optimization

After filtering out wrong predictions using object classification shown in Section 5.3.2, this subsection should increase the detections again. While errors are minimized in the previous section by removing FPs, the optimization experiment aims to increase the detection accuracy by adding TPs. Therefore, not detected traffic signs on single image frames within the whole sequence are estimated and added by using optical flow combined with multi object tracking. Therefore, the fine-grained detection results are used as input. To show the impact of FPs the results of *CLS1* and *CLS2* are taken as input and the final results after the optimization are compared for each evaluation sequence. Figure 5.8 shows the results after performing optical flow by pretrained PWCNet [Nik18] by using the default settings with input image size of 1024x436px for two horizontally appended images. After the optical flow, a simple tracking approach³ is performed. The orange regions are incorrect predictions and yellow regions are missed predictions. The green bars show the correct predictions displayed in relative values. In *SEQ02* the *CLS1* based model has 70% TPs while the *CLS2* based model fails. This could be explained by too sparse object detections over time which makes it impossible to estimate missing predictions. This can be compensated by adjusting the parameters of the tracking approach but will further lead to an increased FP rate for other sequences. This decreases the advantages of the previous refinement step. *SEQ19* performs best by having no FN and less than 15% FP.

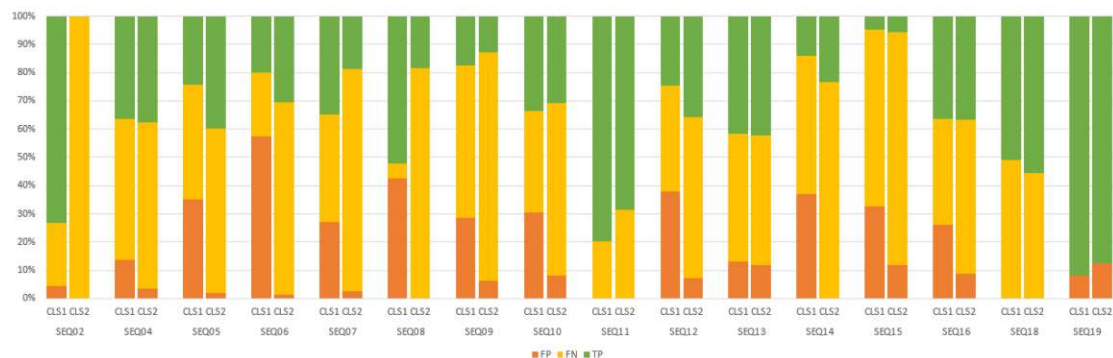


Figure 5.8: Fine-grained detection results after detection optimization by temporal adaptations are visualized. For each sequence and each classification result the FP, FN and TP for model M_{keepUR} with *CLS1* on the left side compared to M_{keepUR} with *CLS2* on the right side per sequence.

³<https://www.pyimagesearch.com/2018/07/23/simple-object-tracking-with-opencv/>

During the experiment it was observed that sparse detections of correct signs can be discarded too if it is not possible by the simple tracker to identify an object instance over time. As a result, objects which are not recognized over consecutive frames are further unable to perform the localization estimation. In Figure 5.9 examples of ten different traffic signs within four evaluation sequences are shown which have too less detections to track them over time. By inspecting the visual results in detail, it is deduced that the target datasets have a lack on "yield to cross traffic" signs recorded from traffic signs back face due to the fact that the object detection model is not able to detect them frequently. The first two image crops of *SEQ07* show three traffic signs from class *traffic sign back*. They are not detected due to underexposure and low image resolution. The other three image crops from this sequence showing the complexity of traffic signs occluded by a tree which are missed by the detector. The parking signs extracted from *SEQ09* are not detected because they are too far away to detect them precisely over time. When creating this sequence, the minimum bus to sign distance of the visualized signs are 30.1m up to 59.7m.



Figure 5.9: Missed localization signs due to missing predictions when performing object detection. The signs are mainly triangular-shaped signs from the back.

The experiment demonstrates that by using PWCNet to perform optical flow and combining it with a simple tracking approach increases the TP detection results. Nevertheless, negative signs which are not relevant for the localization part and should be rejected are not always discarded and furthermore detected over multiple frames which increases the FP rates.

5.3.4 Summary

After the experiments are discussed within each section, a comparison of the results based on each experiment are established within this section and the findings are summarized. In Table 5.7 the precision(P_n), the recall(R_n) and the F1-Score($F1_n$) are listed whereby n defines the experiment ID: Raw detection results (1), classification refinement (2) and optimization by optical flow combined with tracking (3). Experiment (1) offers the lowest precision and F1-score values. This is an expected result due to the high amount of FP detections produced by the low detection threshold. The lowest recall over all sequences is produced by experiment (2) due to decrease of correct predictions. Precision, recall and

F1-score are increased in experiment (3) in nearly all sequences and model combinations. While the pure classification improvement leads to a F1-score of up to 81.8%, the further improvement increases the score up to 95.8%. As seen in *SEQ18* and *SEQ11* single wrong predictions are efficiently discarded while correct predictions are increased within this optimization stage of experiment (3).

In Figure 5.10 the FPs, FNs and TPs are visualized as line chart over all experiments. As expected, the FN rate is high in experiment (1) and (2) and decreases in experiment (3). Furthermore, the FP rate strongly decreases from experiment (1) to experiment (2). Therefore, insights are acquired that the refinement part is working and they are more than halved in experiment (2) compared with experiment (1). Another finding is that the adaptations in *CLS2* in experiment (2) are working due to the stronger decrease of FP rate of *CLS2* compared with *CLS1*. Nevertheless, the FN rate is lower and the TP rate is higher for results based on *CLS1* data than based on *CLS2*. It can be inferred that the tracking approach is effective in *CLS1* by increasing correct annotations and rejecting wrong predictions.

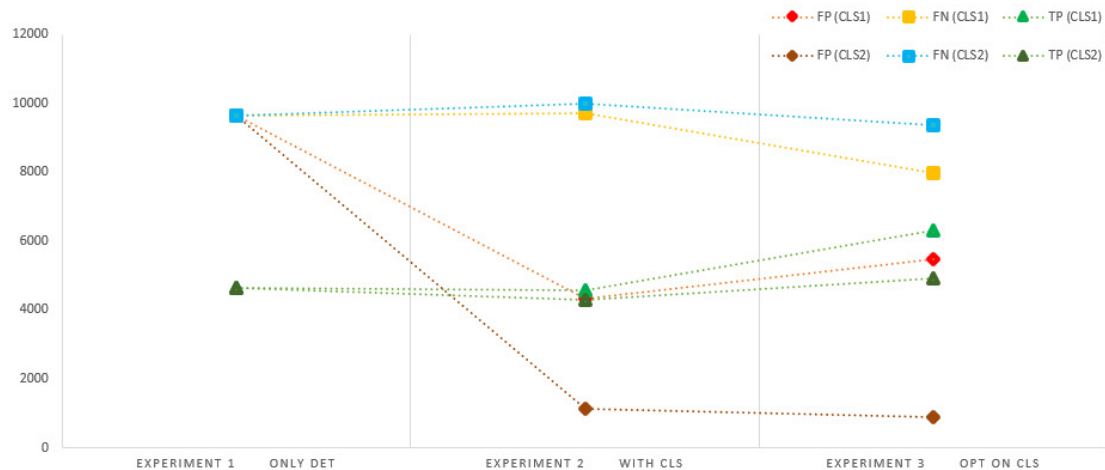


Figure 5.10: Line chart over all experiments showing the adjustments of the FPs, FNs and TPs between each experiment. The line starts with the raw detection results (ONLY DET), followed by the classification refinements (with CLS) and closing with the optimization stage by optical flow combined with tracking (OPT ON CLS).

5.4 Evaluation of Localization

After the detection models and the following refinement steps are tested, the last part of the pipeline is evaluated which is the final localization calculation. While distance calculations are performed from a single image frame, the used approach for GPS position estimation needs multiple frames to get the object position over time. In the first step the distance between the current bus GPS position to the detected traffic sign localization

within single frames are examined. Afterwards, the traffic sign localizations are evaluated. Both experiments are based on the latest detection results improved by refinement and optimization.

5.4.1 Experiment for Distance Evaluation

Within this section two aspects are taken into consideration: the time on which the single traffic sign instances are detected the first time and how accurate the distance can be calculated. While the first one shows the maximum bus to object distance for small objects, the second issue provides insights about the reachable accuracy based on the produced input data. It is expected that the distances over time of a specific physical object are getting smaller due to the fact that the driving bus is getting closer to the objects in driving direction.

Results - Time of Detection

In Figure 5.11 the maximum distance which is the point at which the traffic sign is visible within the image sequence is visualized as green dot while the blue bar shows the predictions from zero up to the prediction with highest distance of the traffic sign. The signs are detected for the first time between 8.2m and 54.9m. This results in an average distance of 28.2m over all examined traffic sign instances. Objects which are further away than 55m from the camera are smaller than 10x10px in image space. In *SEQ04* too small objects are not detected which is a result of the used dataset where tiny objects are excluded. The distances of the first object recognition show strong variations within a test sequence as well as between the sequences. Traffic signs captured from backside are harder to detect than traffic signs from front-side. While front-side objects are recognized at an object size of 12x12px, the backside objects are noticed starting from a size of 20x20px. In *SEQ05* the traffic signs are oriented to the crossroad which results in increased FNs compared to traffic signs oriented in the direction of the camera. In *SEQ09* the first recognition of the traffic signs are delayed due to increased occlusion by other objects like trees or poles. Summarized, the late visual recognitions by the detector are evoked by too small object size, uncommon traffic sign viewpoints and occlusions by other other obstacles like trees. The signs "yield to cross traffic" are early detected with the exception of *SEQ12* where the sign is occluded by a tree.

Results - Camera to Object Distance

The distance error is visualized in Figure 5.12 whereby zero means that there is no difference between calculated distance and the ground truth distance. An orange bar shows the range of the distance for a specific sign over the whole time where it appears within the test sequence. Short bars symbolize that there is a constant distance error over time which means that the bounding boxes appearance is constant too over time. A long bar indicates that the bounding boxes belonging to the examined object are inaccurate and oscillating over time. This leads to wrong distance calculations and subsequently

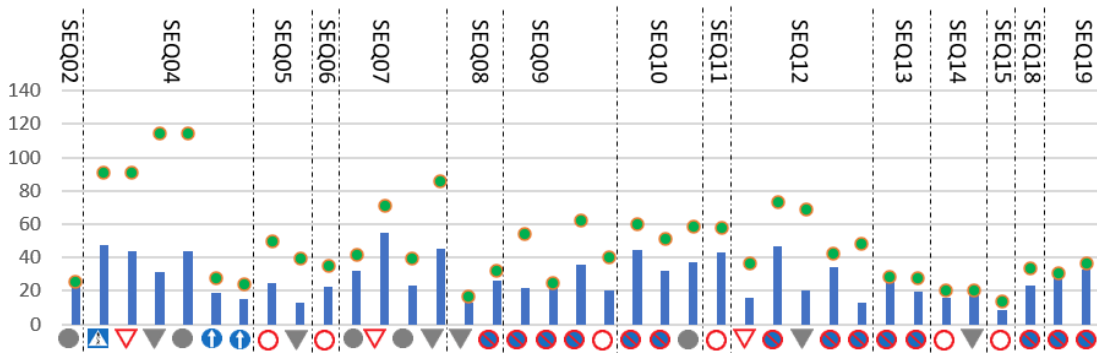


Figure 5.11: Distance of traffic sign to current bus location when first time recognized by the detector (blue bar) in relation with the maximum distance measured from ground truth annotation (green dot) when the traffic sign is first visible in the recorded sequence. The dashed vertical lines group traffic signs instances by their test sequences. The distances vary due to street conditions like other objects occluding the traffic signs.

increased inaccuracies in the final localization part. Especially the traffic sign back objects of "yield to cross traffic" signs have a higher error rate compared with the other object instances. All facts imply that the bounding box shapes of this specific object type are incorrect which leads to the wrong distance results. The traffic signs of class *No vehicles allowed* have low distance errors of a maximum of nine meters over all instances and a mean value over all signs of this type of 2.11m.

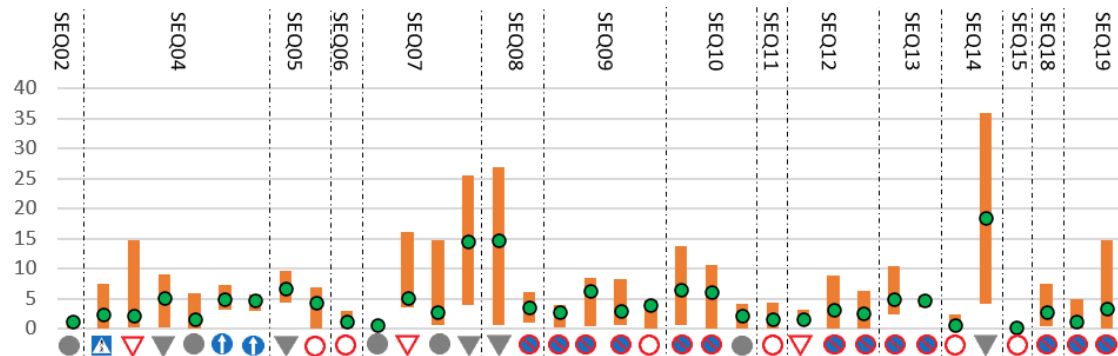


Figure 5.12: Distance error of traffic signs over time (orange bars) in combination with the median distance error (green circles). The closer the objects, the lower the error between predicted distance and ground truth. Most of the traffic signs have a distance error lower than five meters. But the model has still problems with traffic signs from back-side resulting in a minimum distance error over five meters.

5.4.2 Experiment for Position Evaluation

The position accuracy compared with the distance error over all sequences is shown in Figure 5.13. Zero indicates that there is no error between ground truth and estimated

position. The orange bars show the distance error and the dark blue bars the position error over time. The green dots are the average errors over time calculated for each object and error metric. It can be assumed that the errors which are occurred in the previous sections influence the final position results. Thus, the same errors which occur in the previous Subsection 5.4.1 are further appear in the current experiment. The figure shows that the location error for traffic sign instance is equal or higher compared with the camera to object distance error. Furthermore, the minimum position error is higher than the minimum distance error. This shows that previous errors accumulate at the end and have a negative influence on the final result. *SEQ02* has a gap of nearly eleven meter between the median distance error and the median position error. This sequence is quite short with 77 images where the bus drives along a single curve of the road and the detections of the sign are not accurate along the object boundaries which results in an increased position error.

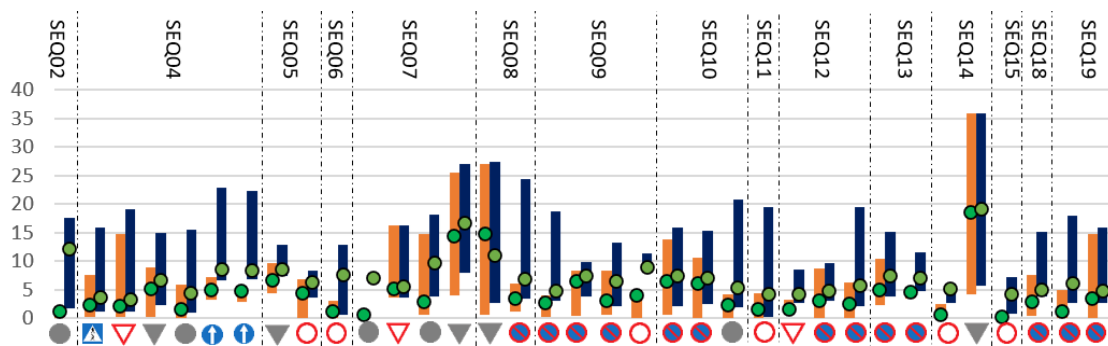


Figure 5.13: Distance error (orange bars) compared with the location error (dark blue bars) and their avg errors (green dots on top of each bar).

SEQ08 and *SEQ09* contain 'no Parking' signs which have a distance error of under five meters but show a position error of over 15m. As seen in Figure 5.14 it can be inferred that the high position error is a result of the specific appearance of this sign. It contains an additional explanation under the round traffic sign which results in a rectangular sign which is uncommon within the training data and results in detections of the whole physical sign as well as predictions of the drawn sign over time.

5.5 Limitations

During the work of this thesis limitations have appeared which are discussed within this section. Firstly, the test data of the bus contains an accurate GPS signal per second which is used as ground truth for distance and localization experiments. However, as the sequences are recorded with 15 FPS there is only an accurate signal for each 15th frame. Subsequently, the signals are interpolated over time which leads to an inaccuracy of up to 50cm from the real bus GPS coordinates. Further critical aspects of the test data are the image quality as well as the resolution which makes them challenging. Compared to other related works the knowledge of the individual physical object size of traffic signs and the

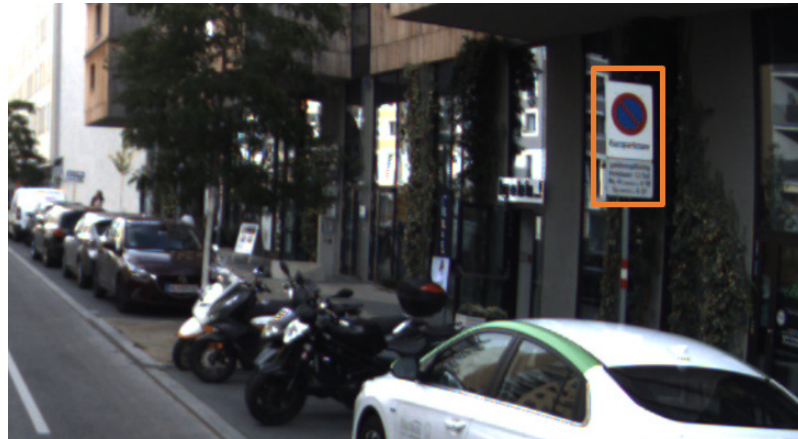


Figure 5.14: Example of 'no Parking' sign from Seestadt Aspern, Vienna.

height of the camera mounted on the bus must be known which is a disadvantage of this approach. Occlusions of traffic signs by other static objects results in corrupted data and incorrect distance and localization measurements due to an incorrect assumed object height based on the partly occluded object. This leads to wrong distance estimations due to wrong assumptions of the original object size. Assuming a wrong object height makes the bus-to-object distance longer or shorter than it is. Despite these restrictions, the experiments show presentable results which are summarized and discussed in the following section.

5.6 Summary

After evaluating different aspects within the developed pipeline, the key findings of the experiments are summarized to finalize this chapter.

The underlying ground truth data for training the detection models are not pixel accurate and differ between the datasets which leads to inaccuracies. While the open-source datasets are focused on the object detection part by itself, the performance of the developed approach benefits from pixel-accurate bounding boxes. Nevertheless, the inaccuracies of the object detection part are compensated by the introduced optimization steps. While missing detections within an image sequence increase the localization complexity, optical flow decreases detection fails by interpolating missing detections and handle occlusions by other objects. This results in increased coherent localization results than depending on results from each pipeline stage.

By performing data aggregation and train a model based on the aggregated dataset can result in increased mAP than without using data aggregation. Nevertheless, the dataset quality and its object variety have an increased effect on the model performance shown when comparing the pretrained model with the models from scratch. The experiments are pointing out that traffic sign captured from backface of the class *yield to cross traffic*

tends to be not recognized within the image sequences or too inaccurate predicted. This implies that this sign type is underrepresented in the training set of the object detection model. Furthermore, the samples show low image quality and uncommon viewpoints of the signs to the camera direction. However, the labels *no vehicle* and *no parking* are well represented by the training data which results in an average distance error of 1.3m for the first one and 3.9m for the second label. The average localization error 6.5m for the label *no vehicle* and 6.2m for the label *no parking*. The increased average localization error results from outliers which are decreasing the average localization error. For example, the parking sign in *SEQ18* has an localization error of 5m with a range between 3.9m up to 15.2m. In Figure 5.15 three signs (green dots) and their estimated localizations (orange dots) are visualized. The blue lines are the bus line and the blue circle symbolize bus stations. The left image shows a no parking sign from backface where most of the predictions are on the same level but slightly displaced. The images on the middle and on the right side show predictions of signs belonging to the label *no vehicle*. While the localization points in *SEQ06* scattered around the sign, the calculated localizations in *SEQ15* are close to the sign over the whole sequence.

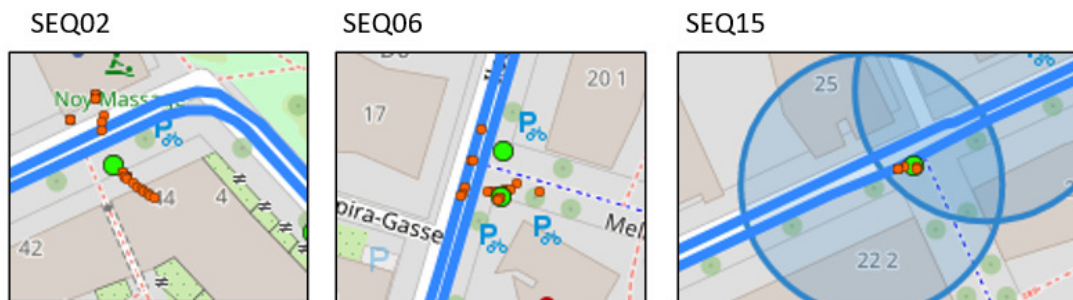


Figure 5.15: Visualization of the resulting GPS positions. The green dot is the ground truth while the orange dots are the calculated positions over time.

To conclude, the object detection results combined with additional optimization techniques increases the final detection performance and allows to perform traffic sign localization calculations. As a disadvantage when calculating the final results based on a prior object detection module are incorrect bounding box shapes which lead to increased localization errors. Nevertheless, the error is compensated and minimized by the used refinement and optimization techniques. Concluding, the final localization results are comparable with current state-of-the-art approaches which are listed in Section and have localization errors from one meter up to 10 meters.

Table 5.5: Fine-grained classification refinement of detection results. For each sequence the two classification models *CLS1* and *CLS2* are used for evaluation. Furthermore, the evaluation parameter FP, TP, GT, precision, recall and f1-score are listed.

Sequ	Model	FP	TP	GT	Prec(%)	Recall(%)	f1-score(%)
SEQ02	CLS1	6	28	64	82.4	43.8	57.1
	CLS2	4	17	64	81.0	26.6	40.0
SEQ04	CLS1	426	1.215	3.004	74.0	40.5	52.3
	CLS2	132	1.121	3.004	89.5	37.3	52.7
SEQ05	CLS1	300	218	598	42.1	36.5	39.1
	CLS2	28	216	598	88.5	36.1	51.3
SEQ06	CLS1	154	65	214	29.7	30.4	30.0
	CLS2	9	60	214	87.0	28.0	42.4
SEQ07	CLS1	559	345	1.790	38.2	19.3	25.6
	CLS2	66	293	1.790	81.6	16.4	27.3
SEQ08	CLS1	170	12	197	6.6	6.1	6.3
	CLS2	15	12	197	44.4	6.1	10.7
SEQ09	CLS1	544	133	1.237	19.7	10.8	13.9
	CLS2	72	116	1.237	61.7	9.4	16.3
SEQ10	CLS1	378	362	1.132	48.9	32.0	38.7
	CLS2	121	354	1.132	74.5	31.3	44.1
SEQ11	CLS1	0	189	273	100.0	72.5	81.8
	CLS2	1	169	273	99.4	69.2	76.3
SEQ12	CLS1	819	747	2.431	47.7	30.7	37.4
	CLS2	358	730	2.431	67.1	30.0	41.5
SEQ13	CLS1	20	95	221	82.6	43.0	56.6
	CLS2	19	95	221	83.3	43.0	56.7
SEQ14	CLS1	33	24	107	42.1	22.4	29.3
	CLS2	7	23	107	76.7	21.5	33.6
SEQ15	CLS1	128	17	433	11.7	3.9	5.9
	CLS2	43	16	433	27.1	3.7	6.5
SEQ16	CLS1	722	841	2.183	53.8	38.5	44.9
	CLS2	184	781	2.183	80.9	35.8	49.6
SEQ18	CLS1	1	57	106	98.3	53.8	69.5
	CLS2	1	57	106	98.3	53.8	69.5
SEQ19	CLS1	68	223	287	76.6	77.7	77.2
	CLS2	76	224	287	74.7	78.1	76.3
ALL	CLS1	4.328	4.571	14.277	51.4	32.0	39.4
ALL	CLS2	1.136	4.284	14.277	79.0	30.0	43.5

Table 5.6: Summarized predictions and ground truth over all sequences.

Combination	FP	TP	GT	Prec(%)	Recall(%)	f1-Score(%)
only DET	9.629	4.629	14.277	32.5	32.4	32.4
with CLS1	4.328	4.571	14.277	51.4	32.0	39.4
with CLS2	1.136	4.284	14.277	79.0	30.0	43.5

Table 5.7: Comparison of detection metrics for all detection evaluation stages expressed as a percentage. Whereby the columns show P_n stands for precision, R_n for recall and $F1_n$ for F1-Score. The n defines the detection stage. The highest value over the stages is printed in bold type.

Sequ	Data	P_1	P_2	P_3	R_1	R_2	R_3	$F1_1$	$F1_2$	$F1_3$
SEQ02	CLS1	80.0	82.4	94.2	43.8	43.8	76.6	56.6	57.1	84.5
	CLS2	80.0	81.0	0.0	43.8	26.6	0.0	56.6	40.0	0.0
SEQ04	CLS1	54.8	74.0	72.7	40.6	40.5	42.3	46.7	52.3	53.5
	CLS2	54.8	89.5	91.8	40.6	37.3	38.9	46.7	52.7	54.7
SEQ05	CLS1	24.0	42.1	40.8	36.8	36.5	37.5	29.1	39.1	39.1
	CLS2	24.0	88.5	95.3	36.8	36.1	40.5	29.1	51.3	56.8
SEQ06	CLS1	18.4	29.7	25.9	30.4	30.4	47.2	22.9	30.0	33.4
	CLS2	18.4	87.0	95.7	30.4	28.0	30.8	22.9	42.4	46.6
SEQ07	CLS1	33.2	38.2	56.4	19.3	19.3	47.8	24.4	25.6	51.7
	CLS2	33.2	81.6	87.7	19.3	16.4	19.1	24.4	27.3	31.4
SEQ08	CLS1	3.6	6.6	55.1	6.1	6.1	90.9	4.5	6.3	68.6
	CLS2	3.6	44.4	100.0	6.1	6.1	18.3	4.5	10.7	30.9
SEQ09	CLS1	13.7	19.7	38.1	12.9	10.8	24.7	13.2	13.9	29.9
	CLS2	13.7	61.7	66.8	12.9	9.4	13.7	13.2	16.3	22.7
SEQ10	CLS1	34.5	48.9	52.6	31.8	32.0	48.4	33.1	38.7	50.4
	CLS2	34.5	74.5	79.7	31.8	31.3	33.5	33.1	44.1	47.1
SEQ11	CLS1	36.0	100.0	100.0	74.7	72.5	79.9	48.6	81.8	88.8
	CLS2	36.0	99.4	100.0	74.7	69.2	68.5	48.6	76.3	81.3
SEQ12	CLS1	31.7	47.7	39.4	30.4	30.7	39.7	31.1	37.4	39.6
	CLS2	31.7	67.1	83.2	30.4	30.0	38.5	31.1	41.5	52.6
SEQ13	CLS1	25.4	82.6	76.3	43.4	43.0	48.0	32.1	56.6	58.9
	CLS2	25.4	83.3	77.9	43.4	43.0	48.0	32.1	56.6	59.4
SEQ14	CLS1	12.3	42.1	27.6	24.3	22.4	22.4	16.4	29.3	24.7
	CLS2	12.3	76.7	100.0	24.3	21.5	23.4	16.4	29.3	37.9
SEQ15	CLS1	2.0	11.7	12.5	3.9	3.9	6.9	2.7	5.9	8.9
	CLS2	2.0	27.1	32.6	3.9	3.7	6.5	2.7	6.5	10.8
SEQ16	CLS1	42.6	53.8	58.0	39.2	38.5	49.2	40.9	44.9	53.2
	CLS2	42.6	80.9	80.6	39.2	35.8	40.1	40.9	49.6	53.6
SEQ18	CLS1	39.0	98.3	100.0	53.8	53.8	50.9	45.2	69.5	67.5
	CLS2	39.0	98.3	100.0	53.8	53.8	55.7	54.2	69.5	71.5
SEQ19	CLS1	33.5	76.6	92.0	78.1	77.7	100.0	46.9	77.2	95.8
	CLS2	33.5	74.7	87.5	78.1	78.1	100.0	46.9	76.3	93.3
ALL	CLS1	31.1	51.4	53.4	78.0	32.0	44.1	46.9	39.4	48.3
	CLS2	31.1	79.0	84.8	31.7	30.0	34.4	31.4	43.5	48.9

Conclusion

An autonomous vehicle needs to know its position for orientation and navigation purposes. For the use of autonomous vehicles, the street maps need to be highly detailed and kept up to date to allow autonomous driving maneuvers. Getting useful street maps on traditional manner is time consuming and expensive due to manual efforts. Furthermore, accurate self-localization is essential for the steering and navigation process. Thus, accurate self-localization as well as keeping track of navigation priors like traffic signs and traffic lights on the street is an essential research topic.

This thesis proposes a novel strategy to improve geolocalization by using data aggregation and further optimizations for robust detection over time which results in object localization estimations. This is enabled by using object detection and classification based on data aggregation. It is an adaptive approach to aggregate multiple data sources which increases the generalization degree by iteratively applying dataset statistics. This results in increased label balancing and decreased dataset bias. After the trained model generates predictions on the input data, multi-object tracking based on optical flow estimations is performed to localize the tracked objects. By using optical flow, the robustness of the previous detection model is increased by adding missing detections and discard wrong single predictions over time. By deriving features from temporal and spatial knowledge, the search-range for object detection is narrowed.

Depending on the physical object type, the experiments show a median localization error between 3.24m and 19.05m. The localization error strongly varies depending on the traffic sign class. The worst signs of type *traffic sign back* which is shown by the fact that all errors over nine meter are from type *traffic sign back*. Signs which have a unified appearance and are well depicted by the selected training datasets perform best. For example, signs of type *pedestrian crossing* and *yield to cross traffic* have a median localization error between 3.24m and 5.53m. The sign from label *no vehicle* in recording sequence 15 has stable bounding box detections which are close to the object borders over time. This results in a mean location error of 1.8m and a minimum location error

6. CONCLUSION

of 0.3m. This shows that a precise detection over time generates competitive results compared to the state of the art.

In a future work the developed approach can be extended by increasing the collected data to make use of information from multiple drives of the same route. This can improve the developed approach and decrease the current weaknesses of the trained models by extending the data variability. This gives the ability to handle objects for the localization part as well as to cope with different weather and light conditions. Another interesting future topic is to foresee decisions of other road users by extending the current approach with pose estimation of the predicted objects. Derived from this, assumptions can be made if specific objects are further relevant for the autonomous vehicle or not. For example, if traffic lights are oriented in relation to the driving direction. Finally, scene understanding could be another essential extension based on this thesis. By deriving each scene depending on the appearance of specific objects within the scene like bus stations or roundabout helps to localize the autonomous vehicle and estimating the traffic situation. Furthermore, the level of traffic can be distinguished and chronological knowledge can be derived like at the end of school the probability is increased that an increased number of children are waiting at the bus stations.

List of Figures

1.1	Scenery of a traffic square containing different dynamic objects [Mad19] which cope different essential research topics like object detection, multi object tracking, scene understanding up to behavior prediction.	2
1.2	Three main aspects to cope within this thesis: (a) Objects must be detected and classified within the traffic scene. (b) The detections are improved by temporal propagation which is achieved by robust classification and occlusion handling over consecutive frames. (c) Finally, the relevant detected objects are geolocalized within the scene to estimate the main camera position in space.	4
2.1	Object Detection Milestones based on [ZCS ⁺ 23] and extended by [XTY ⁺ 20]. At the beginning, traditional detection methods were predominant. Since 2012 the DL approaches outperform the traditional approaches. The DL based methods are further divided into one-stage and two-stage detectors. While the first one is usable for real-time applications, the second one provides a higher degree of accuracy compared with the one-stage detectors.	9
2.2	FPN with three prediction levels which is used for object detection at different resolutions [Wen18].	10
2.3	Concept of anchor boxes with three different scale levels and three different aspect ratios resulting in nine anchor boxes for each feature map location. .	11
2.4	Average EPE from different optical flow approaches [Pat19]. The EPE compares the estimated optical flow vector with the ground truth optical flow vector. While traditional methods have an EPE over 5, the DL based methods are lower with an EPE of 4.44 and 4.84.	12
3.1	Overview of the system where the produced output in the lower row is generated by the operations in the upper row. It describes the process from two consecutive image frames up to the final output which consists of estimated localization and visualizations.	15
3.2	Autonomous bus from manufacturer Navya. A RGB camera is mounted on the upper side of the windshield (1). A GPS antenna is placed on top of the bus (2).	17
		81

3.3	Illustration of the defined object space and the final registration of static objects within a street map. On the left side the static and dynamic objects of interest are shown whereby on the right side the static objects are visualized within the street map.	18
3.4	Example image from Seestadt Aspern of various signs marked with green bounding boxes. In addition to regular traffic signs, the image contains traffic signs captured from backface where the type of the object is not visible and further signs which are not applicable when driving a motorized vehicle like the sign in the right upper corner which is an information sign for construction workers at the construction site mounted approximately 5m above the ground.	21
3.5	Workflow overview of the postprocessing refinement steps: at first the classified predictions of the object detector are used to generate	22
3.6	Example of an image sequence with calculation steps over multiple frames.	23
3.7	The image composition shows the narrowing of the bounding box region for calculating the resulting motion vector based on the calculated flow field. The image on the left shows a close-up of a traffic sign prediction marked as orange bounding box. The image in the middle demonstrates the narrowing of the used area for further motion vector calculations. The bounding box size is reduced by 25% for each direction indicated by the yellow arrows and dashed lines to exclude background. The resulting calculation area is shown as blue rectangle on the image on the right side.	23
3.8	Visualization of the progress when calculating a bounding box position based on the previous frame and the calculated motion vector. The image on the bottom shows a visualization of the flow image from the RGB images of frame n-3 and frame n with a rectangle on the calculation area to get the resulting motion vector (black arrow). The images in the top row describe the progress: the left image shows a detected traffic sign (orange bounding box) and the calculated motion vector (orange arrow). The image in the middle shows an overlay of image n on image n-3 with the previous prediction (orange bounding box) and the current one (yellow bounding box). The image on the right shows the resulting prediction (yellow bounding box).	24
3.9	Example for possible errors during object detection and classification for a specific traffic sign instance over time. Frame one and three show correct predictions. Frame two has a missing annotation during object detection. In frame four a wrong object classification is visualized.	26
3.10	Example of occlusion handling: A detected moving object occludes a static object. Both object instances are detected correct.	27
3.11	Example for bounding box oscillation from pure object detection result compared with bounding box readjustment by using optical flow refinement and interpolation.	28
3.12	Calculation on which side of the bus the detected object occurred. (1) is the current bus vector while (2) shows the current vector from the bus to the detected object.	30

3.13	Visualization framework to load, estimate and visualize improved detections. On the left side the current image is shown. Predicted objects are marked with colored bounding boxes and the labels on the bottom are filled if an object of the label is predicted within the current image. The street map on the right side shows the estimated location of predicted static objects as well as the current bus location.	31
4.1	Different appearance of traffic sign <i>pedestrian crossing</i> from 7 specific countries. The image is taken and adapted from ¹	34
4.2	Two images of the Cityscapes dataset with their semantic masks of the traffic light objects and polyline annotations of the dataset are visualized as green lines which are used to create the bounding boxes of the object instances in the upper row showing inaccurate bounding box annotations which are created for this thesis. On the left side an image crop from Frankfurt is shown while the right image crop shows an image from Munster. The blue boxes are the derived ground truth annotations whereby the red and green box show annotations closer to the detectable object. The green box symbolizes that the IoU lies over 50% while the red box has an IoU lower than 50% compared with the ground truth bounding box.	35
4.3	Example for missing annotation. The first image shows the semantic masks of the traffic light objects and the objects polyline annotations of the dataset which are visualized as colored lines. The second image shows the RGB image. The missing traffic light annotation is marked with an orange arrow which point on the object which is missing.	36
4.4	Example images of the BDD100k dataset. The figure is a revised illustration from the paper of the dataset [YCW ⁺ 20] showing recordings from different cities and at different daytimes with visualized annotations.	37
4.5	Illustration of the labels for object classification. 19 specific traffic signs are selected as well as a label for <i>unknown</i> signs and one for traffic signs captured from the backface which includes circular, rectangular and square-shaped signs from the back.	38
4.6	Example of area specific traffic signs and other uncommon signs from Seestadt Aspern.	40
4.7	Aggregating different datasets by adapting their annotations to the needed learning task and reorganize the label settings to an unique target dataset.	40
4.8	Two images of the test data. On the left image underexposure and windshield artefacts are visible, while the right image shows overexposure and low image resolution.	44
4.9	Evaluation areas in Seestadt Aspern. The zoomed-out map shows the official driving route of the bus. The displayed detail of the map figures essential evaluation areas. Each area is focused on a specific scenario: a) roundabout with 14 traffic signs, b) bus station with number of dynamic obstacles and c) traffic lights with different states.	46
		83

4.10	Visualized street map of evaluation area (a). The signs are shown on the left side whereby the locations are marked in the map by colored numbers. . .	47
4.11	Visualized street map of evaluation area (c). The signs are shown on the left side whereby the locations are signed in the map by colored numbers. . .	48
4.12	Label distribution of the final evaluation datasets for object detection. . .	49
4.13	Example of an image recorded by the bus. On the left side the raw image is shown with the planned image cut and on the right side the cropped image is visualized.	49
4.14	Seestadt Aspern GPS evaluation area. The blue circles indicate areas of bus stops whereby the red and purple dots are traffic sign locations. The red point indicates one and the pink two traffic signs on the same position.	50
4.15	Sequences visualized with traffic sign position and bus locations. The different grey boxes represent sequences in the same street map area with the same traffic signs but captured from different driving directions on the road. . . .	51
5.1	Illustration of the IoU metric [PNDS20].	54
5.2	Relative label distribution of the aggregated detection datasets by class. On the x axis the labels are listed while the y axis shows the label percentage of annotated objects from label in relation to the total annotations of a dataset. The <i>BDD+CS</i> dataset is the combination of the source datasets BDD and CS and was added for comparison purposes.	58
5.3	Example of a cropped region containing detection results of the class <i>traffic light</i> . A result from $D3_{agg}$ on the left side is compared with the result of $D4_{agg}$ on the right side. The light blue lines show the ground truth. Green bounding boxes are correct predictions whereby red ones are wrong predictions. The objects have a varying resolution between 10x25px and 20x40px.	60
5.4	The label distribution of two open-source datasets and their combined datasets <i>BDD+CS keepUR</i> are compared with each other.	62
5.5	Label distribution of M_{keepUR} which is a combination of the two datasets CS and BDD.	62
5.6	Traffic signs which are not detected by the object detection model. Five traffic signs within four subsequences listed with the traffic sign class and its height over the whole subsequence.	66
5.7	Examples of detected traffic signs in SEQ04 which are no regular traffic signs.	67
5.8	Fine-grained detection results after detection optimization by temporal adaptations are visualized. For each sequence and each classification result the FP, FN and TP for model M_{keepUR} with <i>CLS1</i> on the left side compared to M_{keepUR} with <i>CLS2</i> on the right side per sequence.	68
5.9	Missed localization signs due to missing predictions when performing object detection. The signs are mainly triangular-shaped signs from the back. . .	69
84		

5.10	Line chart over all experiments showing the adjustments of the FPs, FNs and TPs between each experiment. The line starts with the raw detection results (ONLY DET), followed by the classification refinements (with CLS) and closing with the optimization stage by optical flow combined with tracking (OPT ON CLS).	70
5.11	Distance of traffic sign to current bus location when first time recognized by the detector (blue bar) in relation with the maximum distance measured from ground truth annotation (green dot) when the traffic sign is first visible in the recorded sequence. The dashed vertical lines group traffic signs instances by their test sequences. The distances vary due to street conditions like other objects occluding the traffic signs.	72
5.12	Distance error of traffic signs over time (orange bars) in combination with the median distance error (green circles). The closer the objects, the lower the error between predicted distance and ground truth. Most of the traffic signs have a distance error lower than five meters. But the model has still problems with traffic signs from back-side resulting in a minimum distance error over five meters.	72
5.13	Distance error (orange bars) compared with the location error (dark blue bars) and their avg errors (green dots on top of each bar).	73
5.14	Example of 'no Parking' sign from Seestadt Aspern, Vienna.	74
5.15	Visualization of the resulting GPS positions. The green dot is the ground truth while the orange dots are the calculated positions over time.	75

List of Tables

4.1	Overview of open-source datasets used for object classification data aggregation. The dataset name (Dataset), the amount of annotated objects (Annotations) and their number of defined labels (Lbls) with the selected annotations (Sel.Annotations) and the amount of selected labels (Sel.Lbls) from the whole dataset labels to aggregate the resulting dataset are listed. The first 5 datasets are containing annotations for traffic sign classification and the last 4 datasets are used to create a target dataset for traffic light classification.	39
4.2	Dataset aggregation for object detection: It compares the selected datasets BDD and CS with the aggregated and optimized dataset <i>keepUR</i> which is based on the others. This leads to a maximum label ratio of 1:20 which means that each class has at least 5% of object instances if the current class is merged with the class with the most instances. The last four columns show the instances of the smallest and the largest class of each dataset and the ratio in % between their own classes per label. The unbalance of the dataset decreases while the under-sampling of the frequent labels is performed in a reasonable scope which is approved by the experiments in Section 5.	41
4.3	Overview of the label distribution within each dataset divided into small(s), medium(m) and large(l) on the basis of their bounding box areas, respectively 32x32px, 96x96px and greater than 96x96px. For each category the values are shown for object instances and percentage of the whole class.	42
4.4	Comparison of the recordings from Seestadt Aspern.	44
4.5	Overview of size and label distribution for each dataset.	47
4.6	Extracted subsequences of the bus trail recordings. Each sequence has a unique name and the contained images and visible object instances (Signs) are listed. Furthermore, the bounding box annotations (Annotations) for object detection evaluation is shown. Essential metrics for object localization are the whole driving distance (Driving Dist.(m)) and the camera to object distance (Sign Dist.(m)).	52
5.1	Aggregated detection datasets of 3.000 images with varying bounding box distribution.	57
		87

5.2	Detection results from object detection models which are trained on the generated datasets. The first two columns show the test and training dataset while the following columns present the AP for each label. In the last column the mAP over all labels is shown. The highest values of each test set (Test Dataset) and each evaluation metric are highlighted in bold type.	59
5.3	Detection results from YOLOv3 model (M_{YOLOv3}) and models from scratch (M_{CS} , M_{BDD} , M_{BDD+CS} and M_{keepUR}). The best results are highlighted in bold type. The values present the AP for each label, the second last column shows the mAP over all labels and the last column the mAP without the AP of the label <i>traffic sign</i> which is not available at M_{YOLOv3}	64
5.4	Detection results generated by the model M_{keepUR} . Sequence ID, FP, TP, GT, precision (Prec), recall and f1-score are listed for each sequence.	65
5.5	Fine-grained classification refinement of detection results. For each sequence the two classification models <i>CLS1</i> and <i>CLS2</i> are used for evaluation. Furthermore, the evaluation parameter FP, TP, GT, precision, recall and f1-score are listed.	76
5.6	Summarized predictions and ground truth over all sequences.	77
5.7	Comparison of detection metrics for all detection evaluation stages expressed as a percentage. Whereby the columns show P_n stands for precision, R_n for recall and $F1_n$ for F1-Score. The n defines the detection stage. The highest value over the stages is printed in bold type.	78

Acronyms

- ADAS** Advanced Driver-Assistance Systems. 1, 16
- AP** Average Precision. 54, 59, 64, 88
- BDD** Berkley Deep Drive. 41, 42, 45, 56, 58, 61–63, 84, 87
- BDD100k** Berkley Deep Drive 100.000. 34–37, 83
- CNN** Convolutional Neural Network. 2, 3, 5, 9, 13, 18, 22, 33
- CS** CityScapes. 41–43, 45–47, 56, 58–63, 84, 87
- CVAT** Computer Vision Annotation Tool. 44, 45
- DL** Deep Learning. 4, 8, 9, 12, 13, 16, 20, 81
- EPE** End-Point-Error. 12, 81
- FN** False Negative. 24, 68, 70, 71, 84, 85
- FP** False Positive. 26, 32, 34, 59, 65–70, 76, 77, 84, 85, 88
- FPN** Feature Pyramid Network. 10, 20, 81
- FPS** frames per second. 23, 44–46, 73
- GPS** Global Positioning System. 3–5, 13, 16, 17, 28–30, 34, 43, 44, 46–50, 55, 64, 70, 73, 75, 81, 84, 85
- GPU** Graphics Processing Unit. 20
- GT** Ground Truth. 65, 76, 88
- IoU** Intersection over Union. 11, 35, 54, 59, 83, 84
- mAP** mean Average Precision. 54, 55, 58–60, 63, 64, 74, 88

ML Machine Learning. 8

NMS Non-Maximum Suppression. 11, 20

RCNN Region-based Convolutional Neural Network. 9

RPN Region Proposal Network. 9

TLM Traffic Lights Merged. 41

TP True Positive. 65–70, 76, 77, 84, 85, 88

TSM Traffic Signs Merged. 41

UI User Interface. 31, 44

Bibliography

- [Agg18] Charu C Aggarwal. *Neural networks and deep learning*, volume 10, chapter 1. Springer, 2018.
- [APS20] Fabio Arena, Giovanni Pau, and Alessandro Severino. An overview on the current status and future perspectives of smart cars. *Infrastructures*, 5(7):53, 2020.
- [BCA⁺16] Lamberto Ballan, Francesco Castaldo, Alexandre Alahi, Francesco Palmieri, and Silvio Savarese. Knowledge transfer for scene-specific motion prediction. In *European Conference on Computer Vision*, pages 697–713. Springer, 2016.
- [BFM21] Vaibhav Bansal, Gian Luca Foresti, and Niki Martinel. Where did i see it? object instance re-identification with attention. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, pages 298–306, 2021.
- [BNB17] Karsten Behrendt, Libor Novak, and Rami Botros. A deep learning approach to traffic lights: Detection, tracking, and classification. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1370–1377, 2017.
- [BSCD17] Navaneeth Bodla, Bharat Singh, Rama Chellappa, and Larry S Davis. Soft-nms—improving object detection with one line of code. In *Proceedings of the IEEE international conference on computer vision*, pages 5561–5569, 2017.
- [Chr91] Nicholas R Chrisman. The error component in spatial data. *Geographical information systems*, 1(12):165–174, 1991.
- [CJR⁺19] Jaesung Choe, Kyungdon Joo, Francois Rameau, Gyumin Shim, and In So Kweon. Segment2regress: Monocular 3d vehicle localization in two stages. In *Robotics: Science and Systems*, 2019.
- [COR⁺16] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele.

The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016.

- [CWM⁺17] Ronald Clark, Sen Wang, Andrew Markham, Niki Trigoni, and Hongkai Wen. Vidloc: A deep spatio-temporal model for 6-dof video-clip relocalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6856–6864, 2017.
- [CWP⁺19] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, et al. Mmdetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019.
- [CWW⁺17] Ronald Clark, Sen Wang, Hongkai Wen, Andrew Markham, and Niki Trigoni. Vinet: Visual-inertial odometry as a sequence-to-sequence learning problem. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.
- [DSGL07] Michael John De Smith, Michael F Goodchild, and Paul Longley. *Geospatial analysis: a comprehensive guide to principles, techniques and software tools*. Troubador publishing ltd, 2007.
- [DYC21] Mengxiao Du, Shiyao Yang, and Qun Chen. Impacts of vehicle-to-infrastructure communication on traffic flows with mixed connected vehicles and human-driven vehicles. *International Journal of Modern Physics B*, page 2150091, 2021.
- [EEVG⁺15] Mark Everingham, SM Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International journal of computer vision*, 111(1):98–136, 2015.
- [EMO⁺20] Christian Ertler, Jerneja Mislej, Tobias Ollmann, Lorenzo Porzi, Gerhard Neuhold, and Yubin Kuang. The mapillary traffic sign dataset for detection and classification on a global scale. In *European Conference on Computer Vision*, pages 68–84. Springer, 2020.
- [EVGW⁺10] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.
- [FMKD18] Andreas Fregin, Julian Muller, Ulrich Krebel, and Klaus Dietmayer. The driveu traffic light dataset: Introduction and comparison with existing datasets. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3376–3383, 2018.

- [Ful15] Wayne Fulton. Calculate distance or size of an object in a photo image. <https://www.scantips.com/lights/subjectdistance.html>, 2015. Accessed: 2023-03-20.
- [GDDM14] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.
- [GDL⁺17] Kent Gauen, Ryan Dailey, John Laiman, Yuxiang Zi, Nirmal Asokan, Yung-Hsiang Lu, George K Thiruvathukal, Mei-Ling Shyu, and Shu-Ching Chen. Comparison of visual datasets for machine learning. In *2017 IEEE International Conference on Information Reuse and Integration (IRI)*, pages 346–355. IEEE, 2017.
- [GG05] Cyril Goutte and Eric Gaussier. A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. In *European conference on information retrieval*, pages 345–359. Springer, 2005.
- [Gir15] Ross Girshick. Fast r-cnn. In *The IEEE International Conference on Computer Vision (ICCV)*, pages 1440–1448, 2015.
- [Gon16] Eric Gonzalez. Udacity self-driving-car. <https://github.com/udacity/self-driving-car>, 2016. Accessed: 2023-03-19.
- [gov22] Gps accuracy. <https://www.gps.gov/systems/gps/performance/accuracy/>, 2022. Accessed: 2022-07-19.
- [HGDG17] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.
- [HZRS16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [IMS⁺17] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2462–2470, 2017.
- [Int18] Sae International. Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles. https://saemobilus.sae.org/content/j3016_201609, 2018. Accessed: 2023-04-22.
- [Jac01] Paul Jaccard. Étude comparative de la distribution florale dans une portion des alpes et des jura. *Bull Soc Vaudoise Sci Nat*, 37:547–579, 1901.

- [Joc20] Glenn Jocher. Yolov5 by ultralytics. <https://github.com/ultralytics/yolov5>, 2020. Accessed: 2023-03-19.
- [JPM⁺16] Morten Bornø Jensen, Mark Philip Philipsen, Andreas Møgelmo, Thomas Baltzer Moeslund, and Mohan Manubhai Trivedi. Vision for looking at traffic lights: Issues, survey, and perspectives. *IEEE Transactions on Intelligent Transportation Systems*, 17(7):1800–1815, 2016.
- [Kir15] Richard Kirk. Cars of the future: the internet of things in the automotive industry. *Network Security*, 2015(9):16–18, 2015.
- [KJY18] Ye-Hoon Kim, Jun-Ik Jang, and Sojung Yun. End-to-end deep learning for autonomous navigation of mobile robot. In *2018 IEEE International Conference on Consumer Electronics (ICCE)*, pages 1–6. IEEE, 2018.
- [KKD18] Vladimir A Krylov, Eamonn Kenny, and Rozenn Dahyot. Automatic discovery and geotagging of objects from street view imagery. *Remote Sensing*, 10(5):661, 2018.
- [KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.
- [LAE⁺16] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.
- [LBD⁺89] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.
- [LDG⁺17] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2117–2125, 2017.
- [LF11] Fredrik Larsson and Michael Felsberg. Using fourier descriptors and spatial models for traffic sign recognition. In *Scandinavian conference on image analysis*, pages 238–249. Springer, 2011.
- [LGG⁺17] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

- [LLJ⁺22] Chuyi Li, Lulu Li, Hongliang Jiang, Kaiheng Weng, Yifei Geng, Liang Li, Zaidan Ke, Qingyuan Li, Meng Cheng, Weiqiang Nie, et al. Yolov6: A single-stage object detection framework for industrial applications. *arXiv preprint arXiv:2209.02976*, 2022.
- [LMB⁺14] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision—ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.
- [LXM⁺21] Wenhan Luo, Junliang Xing, Anton Milan, Xiaoqin Zhang, Wei Liu, and Tae-Kyun Kim. Multiple object tracking: A literature review. *Artificial intelligence*, 293:103448, 2021.
- [Mad19] Radhika Madhavan. How self-driving cars work - a simple overview. <https://emerj.com/ai-sector-overviews/how-self-driving-cars-work/>, 2019. Accessed: 2021-04-18.
- [Mar10] David Marr. *Vision: A computational investigation into the human representation and processing of visual information*. MIT press, 2010.
- [MGE21] Marilo Martin-Gasulla and Lily Elefteriadou. Traffic management with autonomous and connected vehicles at single-lane roundabouts. *Transportation Research Part C: Emerging Technologies*, 125:102964, 2021.
- [MLWX22] Songzhe Ma, Huimin Lu, Yifan Wang, and Han Xue. Yolox-mobile: A target detection algorithm more suitable for mobile devices. In *Journal of Physics: Conference Series*, volume 2203, page 012030. IOP Publishing, 2022.
- [MMH17] Tareq Monawar, Shafayat Bin Mahmud, and Avijit Hira. Anti-theft vehicle tracking and regaining system with automatic police notifying using haversine formula. In *2017 4th International Conference on Advances in Electrical Engineering (ICAEE)*, pages 775–779. IEEE, 2017.
- [MS15] Ian Model and Lior Shamir. Comparison of data set bias in object recognition benchmarks. *IEEE Access*, 3(1):1953–1962, 2015.
- [MTBVG13] Markus Mathias, Radu Timofte, Rodrigo Benenson, and Luc Van Gool. Traffic sign recognition—how far are we from the solution? In *The 2013 international joint conference on Neural networks (IJCNN)*, pages 1–8. IEEE, 2013.
- [Nik18] Simon Niklaus. A reimplement of PWC-Net using PyTorch. <https://github.com/sniklaus/pytorch-pwc>, 2018. Accessed: 2023-03-03.

- [NLW19] Ahmed Samy Nassar, Sébastien Lefèvre, and Jan Dirk Wegner. Simultaneous multi-view instance detection with learned geometric soft-constraints. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6559–6568, 2019.
- [NORBK17] Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulo, and Peter Kotschieder. The mapillary vistas dataset for semantic understanding of street scenes. In *Proceedings of the IEEE international conference on computer vision*, pages 4990–4999, 2017.
- [NVG06] Alexander Neubeck and Luc Van Gool. Efficient non-maximum suppression. In *18th International Conference on Pattern Recognition (ICPR'06)*, volume 3, pages 850–855. IEEE, 2006.
- [O⁺18] World Health Organization et al. Global status report on road safety 2018: Summary. *World Health Organization*, 2018.
- [PAHW18] Alex Pon, Oles Adrienko, Ali Harakeh, and Steven L Waslander. A hierarchical deep architecture and mini-batch selection method for joint traffic sign and light detection. In *2018 15th Conference on Computer and Robot Vision (CRV)*, pages 102–109, 2018.
- [Pat19] Abhijit Patait. An introduction to the nvidia optical flow sdk. <https://devblogs.nvidia.com/an-introduction-to-the-nvidia-optical-flow-sdk/>, 2019. Accessed: 2019-08-08.
- [PGC⁺17] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.
- [PLCL14] Minwoo Park, Jiebo Luo, Robert T Collins, and Yanxi Liu. Estimating the camera direction of a geotagged image using reference images. *Pattern recognition*, 47(9):2880–2893, 2014.
- [PNDS20] Rafael Padilla, Sergio L Netto, and Eduardo AB Da Silva. A survey on performance metrics for object-detection algorithms. In *2020 international conference on systems, signals and image processing (IWSSIP)*, pages 237–242, 2020.
- [QLL22] Rui Qian, Xin Lai, and Xirong Li. 3d object detection for autonomous driving: A survey. *Pattern Recognition*, 130:108796, 2022.
- [RDGF16] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

- [RF18] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.
- [RHGS15] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.
- [ŠBK⁺10] S Šegvić, K Brkić, Z Kalafatić, V Stanisavljević, M Ševrović, Damir Budimir, and I Dadić. A computer vision assisted geoinformation inventory for traffic infrastructure. In *13th International IEEE Conference on Intelligent Transportation Systems*, pages 66–73, 2010.
- [SBR21] Azim Shariff, Jean-François Bonnefon, and Iyad Rahwan. How safe is safe enough? psychological mechanisms underlying extreme safety demands for self-driving cars. *Transportation Research Part C: Emerging Technologies*, 126:103069, 2021.
- [SI18] Petru Soviany and Radu Tudor Ionescu. Optimizing the trade-off between single-stage and two-stage deep object detectors using image difficulty prediction. *2018 20th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)*, pages 209–214, 2018.
- [Sin15] Santokh Singh. Critical reasons for crashes investigated in the national motor vehicle crash causation survey. Technical report, 2015.
- [SM16] Luca Studer and Giovanna Marchionni. Evaluation of the impact of its. *Evaluation of Intelligent Road Transport Systems: Methods and Results*, pages 67–122, 2016.
- [SNQP⁺20] Shahin Sharifi Noorian, Sihang Qiu, Achilleas Psyllidis, Alessandro Bozzon, and Geert-Jan Houben. Detecting, classifying, and mapping retail store-fronts using street-level imagery. In *Proceedings of the 2020 International Conference on Multimedia Retrieval*, pages 495–501, 2020.
- [SSSI12] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks*, 32:323–332, 2012. Selected Papers from IJCNN 2011.
- [SYLK18] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8934–8943, 2018.
- [TE⁺11] Antonio Torralba, Alexei A Efros, et al. Unbiased look at dataset bias. In *CVPR*, volume 1, page 7, 2011.

- [TPCT17] Tatiana Tommasi, Novi Patricia, Barbara Caputo, and Tinne Tuytelaars. A deeper look at dataset bias. In *Domain adaptation in computer vision applications*, pages 37–55. Springer, 2017.
- [VDE15] Frank Van Diggelen and Per Enge. The world’s first gps mooc and worldwide laboratory using smartphones. In *Proceedings of the 28th international technical meeting of the satellite division of the institute of navigation (ION GNSS+ 2015)*, pages 361–369, 2015.
- [VRDJ95] Guido Van Rossum and Fred L Drake Jr. *Python tutorial*. Centrum voor Wiskunde en Informatica Amsterdam, The Netherlands, 1995.
- [WBL22] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *arXiv preprint arXiv:2207.02696*, 2022.
- [WCY⁺19] Jiaqi Wang, Kai Chen, Shuo Yang, Chen Change Loy, and Dahua Lin. Region proposal by guided anchoring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2965–2974, 2019.
- [Wen18] Lilian Weng. Object detection part 4: Fast detection models. <https://lilianweng.github.io/posts/2018-12-27-object-recognition-part-4>, 2018. Accessed: 2023-04-19.
- [WSH20] Xiongwei Wu, Doyen Sahoo, and Steven CH Hoi. Recent advances in deep learning for object detection. *Neurocomputing*, 396:39–64, 2020.
- [WZMG19] Mingjie Wang, Jun Zhou, Wendong Mao, and Minglun Gong. Multi-scale convolution aggregation and stochastic feature reuse for densenets. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 321–330, 2019.
- [XTY⁺20] Youzi Xiao, Zhiqiang Tian, Jiachen Yu, Yinshu Zhang, Shuai Liu, Shaoyi Du, and Xuguang Lan. A review of object detection based on deep learning. *Multimedia Tools and Applications*, 79(33):23729–23791, 2020.
- [YCW⁺20] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. pages 2633–2642, 06 2020.
- [YKF17] Fisher Yu, Vladlen Koltun, and Thomas Funkhouser. Dilated residual networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 472–480, 2017.
- [YXC⁺18] Fisher Yu, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving video database with scalable annotation tooling. *arXiv preprint arXiv:1805.04687*, 2018.

- [YZL⁺18] Tong Yang, Xiangyu Zhang, Zeming Li, Wenqiang Zhang, and Jian Sun. Metaanchor: Learning to detect objects with customized anchors. In *Advances in Neural Information Processing Systems*, pages 320–330, 2018.
- [ZCS⁺23] Zhengxia Zou, Keyan Chen, Zhenwei Shi, Yuhong Guo, and Jieping Ye. Object detection in 20 years: A survey. *Proceedings of the IEEE*, 111(3):257–276, 2023.
- [ZF19] Jing Zhu and Yi Fang. Learning object-specific distance from a monocular image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3839–3848, 2019.
- [ZFL21] Chaoquan Zhang, Hongchao Fan, and Wanzhi Li. Automated detecting and placing road objects from street-level images. *Computational urban science*, 1(1):1–18, 2021.
- [ZLK⁺17] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 40(6):1452–1464, 2017.
- [ZWL⁺18] Weixing Zhang, Chandi Witharana, Weidong Li, Chuanrong Zhang, Xiaojiang Li, and Jason Parent. Using deep learning to identify utility poles with crossarms and estimate their locations from google street view images. *Sensors*, 18(8):2484, 2018.