BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

STEPS TOWARDS IMPROVEMENTS OF COMPUTER VISION METHODS FOR TRAFFIC ANALYSIS

KROKY KE ZLEPŠENÍ METOD POČÍTAČOVÉHO VIDĚNÍ PRO ANALÝZU DOPRAVY

PHD THESIS
DISERTAČNÍ PRÁCE

AUTHOR **Ing. JAKUB ŠPAŇHEL**
AUTOR PRÁCE

SUPERVISOR **prof. Ing. ADAM HEROUT, Ph.D.**
ŠKOLITEL

BRNO 2023

Abstract

The rapid urbanization and increasing number of vehicles on the roads have stretched traditional traffic management systems to their limits. Intelligent Transportation Systems (ITS) offer a solution, utilizing advanced technologies to enhance traffic flow and safety. The robustness of computer vision methods within ITS, essential for traffic analysis, remains a crucial area for improvement. This thesis substantially contributes to this field, specifically focusing on Vehicle Fine-Grained Recognition, Vehicle Re-Identification, License Plate Recognition, and Monocular Vehicle Speed Measurement. Several new datasets, highly appreciated by the research community, were introduced, enhancing the evaluation and exploration within each domain mentioned earlier.

The main contributions can be summarized as follows:

- Novel method for aggregation of visual features for vehicle re-identification & dataset.
- Innovative approach to license plate recognition using alignment of the license plate and holistic recognition & three published datasets.
- Novel augmentation techniques for vehicle fine-grained recognition & extension of previously published dataset.
- The biggest dataset for vehicle speed measurement & baseline evaluation with state-of-the-art methods.

The key findings of this work demonstrate a significant enhancement in the accuracy, efficiency, and robustness of computer vision methods applied to traffic analysis. This research's contributions have been recognized at top conferences and journals in ITS, setting new standards for future work.

By advancing the current state of ITS and contributing valuable resources for ongoing research, this thesis represents a step towards more sustainable and efficient intelligent transportation systems.

Keywords

traffic analysis, computer vision, vehicle fine-grained recognition, vehicle re-identification, vehicle speed measurement, license plate recognition, license plate alignment

Reference

ŠPAŇHEL, Jakub. *Steps Towards Improvements of Computer Vision Methods for Traffic Analysis*. Brno, 2023. PhD thesis. Brno University of Technology, Faculty of Information Technology. Supervisor prof. Ing. Adam Herout, Ph.D.

Rozšířený abstrakt

Rostoucí urbanizace a zvyšující se počet vozidel na silnicích přetěžují tradiční systémy řízení dopravy na hranici jejich možností. Řešení nabízejí inteligentní dopravní systémy (ITS), které využívají pokročilé technologie ke zvýšení plynulosti a bezpečnosti dopravy. Zásadní oblastí, kterou je třeba zlepšit a udělat robustnější, však zůstávají metody počítačového vidění v rámci ITS, které jsou nezbytné pro analýzu dopravy. Tato práce přispívá k této oblasti, konkrétně se zaměřuje na přesné (fine-grained) rozpoznávání vozidel, reidentifikaci vozidel, rozpoznávání registračních značek a měření rychlosti vozidel z jedné kamery. Bylo publikováno několik nových datových sad, ceněných výzkumnou komunitou, které slouží jako benchmark pro vyhodnocení a zkoumání v každé z výše uvedených oblastí.

Rozpoznávání typu vozidel Výzkum v této oblasti shrnuje práce *BoxCars: Improving fine-grained recognition of vehicles using 3-d bounding boxes in traffic surveillance* [202]. V rámci této publikace navrhujeme vylepšení stávající metody pro přesné (fine-grained) rozpoznávání typu vozidel a zveřejňujeme rozšířenou verzi datové sady *BoxCars*. Kromě datové sady patří mezi hlavní přínosy této práce návrh augmentačních technik pro učení neuronových sítí a nová metoda detekce 3D obalového tělesa vozidla ze statického snímku.

Vizuální re-identifikace vozidel Přínosy v této oblasti byly publikovány v článku *Learning feature aggregation in temporal domain for re-identification* [209]. V době publikace této práce metody sloužící pro re-identifikaci vozidel pracovaly převážně s jedním snímkem. Hlavním přínosem této práce je metoda agregace extrahovaných vektorů s vizuálními příznaky v rámci celé trajektorie vozidla. Tento princip dovoluje efektivně využívat jednotlivé části příznakových vektorů ve vztahu k jedinečnosti zachycené informace. Součástí této práce je také publikace datové sady *CarsReId74k*.

Automatické rozpoznávání registrační značky Publikace *Holistic recognition of low quality license plates by CNN using track annotated data* [208] a *Geometric alignment by deep learning for recognition of challenging license plates* [207] popisují inovativní přístup k rozpoznávání registračních značek pomocí zarovnání registrační značky a následného rozpoznání holistickým způsobem. Výzkumné komunitě jsme také přispěli třemi datovými sadami pro rozpoznání registračních značek ve ztížených podmínkách a pro kontrolu parkování.

Vizuální měření rychlosti vozidel Článek *Comprehensive data set for automatic single camera visual speed measurement* [200] potom prezentuje nejvíce využívanou datovou sadu *BrnoCompSpeed* a způsob jejího pořízení, včetně stanovení výchozích evaluačních metrik.

Přehled všech článků s počtem citací a hodnocením časopisů/konferencí je k dispozici v anglickém jazyce v Tabulce 1.1.

Klíčová zjištění této práce prokazují významné zvýšení přesnosti, účinnosti a robustnosti metod počítačového vidění aplikovaných na analýzu dopravy. Přínosy tohoto výzkumu byly oceněny na nejvýznamnějších konferencích a v časopisech v oblasti ITS a stanovují nové standardy pro budoucí práci.

Tím, že tato práce posunula současný stav ITS a přispěla cennými zdroji do stále probíhajícího výzkumu, představuje krok směrem k udržitelnějším, efektivnějším a inteligentnějším dopravním systémům.

Steps Towards Improvements of Computer Vision Methods for Traffic Analysis

Declaration

I hereby declare that this PhD's thesis was prepared as an original work by the author under the supervision of prof. Ing. Adam Herout, Ph.D. The supplementary information was provided by my former and current co-workers and co-authors of my research papers – mainly by Ing. Jakub Sochor, Ph.D., Ing. Roman Juránek, Ph.D. and Ing. Vojtěch Bartl. I have listed all the literary sources, publications, and other sources, used during the preparation of this thesis.

.....
Jakub Špaňhel
August 25, 2023

Acknowledgements

I would like to thank my supervisor, prof. Adam Herout, for his valuable advice and expert feedback. Many thanks goes also to my family for their support during my studies. Finally, the most significant and special acknowledgment goes to my wife, Regina, and my children, Jáchym and Robin, for their support and especially their patience during the time of writing this thesis, when the combination of work, thesis writing, and family was not easy in many cases.

Contents

Part I

Thesis Introduction and Related Work

Chapter 1

Introduction

Transportation has become an inextricable part of our modern existence, with thousands of vehicles hitting the roads daily. The resultant traffic congestion, particularly during rush hours, has become a universal urban experience, often leading to frustratingly long waits and inefficiencies. Frequent traffic accidents, road repairs, and closures further exacerbate this situation. For over a century, we have relied on traffic lights or semaphores to manage this urban traffic chaos. However, with the increasing number of vehicles, the adequacy of this system is being relentlessly tested.

The dawn of the fourth industrial revolution, characterized by a fusion of technologies blurring the lines between the physical, digital, and biological spheres, has brought about a paradigm shift in numerous sectors [189]. One of the critical areas experiencing this transformative effect is transportation, mainly through the development and implementation of Intelligent Transportation Systems (ITS). ITS are advanced applications that aim to provide innovative services relating to different modes of transport and traffic management [4], enhancing safety, efficiency, and sustainability.

The CIVITAS initiative, an organization of cities dedicated to cleaner and better transport in Europe, also emphasizes the importance of ITS for traffic monitoring, management, and enforcement. The organization argues that:

> “ *Intelligent transport systems (ITS) include traffic and congestion monitoring and management systems, with the integration of traffic control centers. Access control and route guidance systems offer a range of benefits for a city. Goods delivery companies often introduce ITS because they can optimize trips by combining global positioning system (GPS) technologies and existing logistics programs. Such traffic management and control systems have significant efficiency benefits for both public and private transport.*[1] ”
>
> CIVITAS, *Intelligent transport systems (ITS) for traffic monitoring, management and enforcement*, 2013

In line with this perspective, several European cities, including Monza, Malaga, Stuttgart, Tallinn, and Brno[1], have joined the CIVITAS initiative and embarked on developing and implementing ITS at a broader scale.

[1] http://www.civitas.eu/telematics/ITS

The crucial role of ITS in addressing the persistent and emerging challenges in transportation infrastructure cannot be overstated. These systems harness cutting-edge technologies to tackle issues like traffic congestion, environmental impacts, road safety, and efficient logistics management [99]. The urgency of these issues is escalating in the backdrop of rapid urbanization and population growth, making the research and development in ITS a necessity rather than a choice [153].

At the heart of ITS lies the power of Computer Vision. As a subfield of artificial intelligence, Computer Vision allows machines to interpret and understand the visual world. In the context of ITS, it enables vehicles and systems to perceive their environment, an essential capability for functionalities like traffic monitoring, vehicle detection, and autonomous driving [231].

This thesis, „Steps Towards Improvements of Computer Vision Methods for Traffic Analysis“ contributes to the growing body of ITS knowledge by focusing on the improvements of Computer Vision methods applied to traffic analysis and their robustness. This research extends through various sub-domains within ITS, specifically, Vehicle Fine-Grained Recognition, Vehicle Re-Identification, License Plate Recognition, and Monocular Vehicle Speed Measurement.

In the course of this research, several comprehensive datasets were introduced. Namely, *BoxCars116k* for Vehicle Fine-Grained Recognition (Chapter 5 on page 43), *CarsReId74k* for Vehicle Re-Identification task (Chapter 4), *ReId, HDR* and *CamCar6k* datasets for License Plate Alignment/Recognition (Chapter 3 on page 34), and finally *BrnoCompSpeed*, which is our frequently accessed dataset for Monocular Vehicle Speed Measurement task (Chapter 6 on page 54). Each dataset has been meticulously created to serve as a benchmark for advancing research in the respective areas. They are also proof of the rigorous work undertaken to address the unique complexities inherent in each field, serving as valuable resources for researchers and practitioners in ITS and Computer Vision.

These strides have been documented in a suite of articles (see Table 1.1 on the next page) which were published within ITS and Computer Vision research communities. These articles were accepted at major conferences and journals dedicated to ITS. This distinction points to the relevance of the research contributions to the field. Despite the core articles of this thesis, multiple Computer Vision and Pattern Recognition Workshops (IS 14.20) articles were also accepted, mainly at the AI City Challenge workshop over multiple years (2018-2022).

Each of listed research articles contributes to its field either by a novel method, a novel dataset or a combination of both. Here is summary of my contributions for each field specifically.

Vehicle Re-Identification Method for visual feature aggregation in the temporal domain for re-identification (called LFTD) and CarsReId74k dataset was published in journal paper *Špaňhel* et al.*, Learning feature aggregation in temporal domain for re-identification*, the first row in Table 1.1. The LFTD method helps aggregate extracted visual features from multiple observations (e.g., multiple video frames of vehicle trajectory) to create more robust and distinct visual features for re-identification tasks. More information about the proposed method is provided in Chapter 7 on page 62. Statistics and acquisition process of the CarsReId74k dataset are available in Chapter 4 on page 39.

Table 1.1: The following list of papers creates the core of this thesis. In addition to these articles, another 16 co-authored papers were accepted for publishing at various conferences or journals. The column *Cited* shows the number of citations in Scopus / Google Scholar. IF/IS stands for Impact Factor/Impact Score, respectively.

#	Authors	Title	Published at	Rank	Cited	CiteScore	IF/IS
1	Špaňhel, Sochor, Juránek, Dobeš, Bartl, Herout	Learning feature aggregation in temporal domain for re-identification	Computer Vision and Image Understanding, 2020	Q1	1 / 5	9.4	IF 5.526
2	Špaňhel, Sochor, Juránek, Herout, Maršík, Zemčík	Holistic recognition of low quality license plates by CNN using track annotated data	IEEE International Conference on Advanced Video and Signal Based Surveillance, 2017	B	78 / 111	-	IS 3.20
3	Špaňhel, Sochor, Juránek, Herout	Geometric alignment by deep learning for recognition of challenging license plates	IEEE International Conference on Intelligent Transportation Systems, 2018	-	2 / 4	-	IS 1.20
4	Sochor, Špaňhel, Herout	BoxCars: Improving fine-grained recognition of vehicles using 3-d bounding boxes in traffic surveillance	IEEE Transactions on Intelligent Transportation Systems, 2018	Q1	81 / 140	11.6	IF 7.801
5	Sochor, Juránek, Špaňhel, Maršík, Široký, Herout, Zemčík	Comprehensive data set for automatic single camera visual speed measurement	IEEE Transactions on Intelligent Transportation Systems, 2018	Q1	42 / 63	11.6	IF 7.801

License Plate Recognition Two conference papers, second and third row in Table 1.1, cover my contribution to the LPR field. Article *Špaňhel* et al., *Holistic recognition of low-quality license plates by CNN using track annotated data* proposes a method for recognition of low-quality license plates in a holistic way by machine learning model and presents two LPR datasets. Work *Špaňhel* et al., *Geometric alignment by deep learning for recognition of challenging license plates* further extends the findings from the previous article. It suggests to align/rectify the license plate by a neural network as a pre-processing step for LPR. Also, the license plate dataset for parking law enforcement was collected and published in this work. More information about the proposed methods is provided in Part IV on page 76. Statistics and acquisition process of the CarsReId74k dataset are available in Chapter 3 on page 34.

Vehicle Fine-Grained Recognition Contribution to this area was presented in journal paper *Sochor* et al., *BoxCars: Improving fine-grained recognition of vehicles using 3-d bounding boxes in traffic surveillance*, fourth row in Table 1.1. We have extended the previously published approach for vehicle fine-grained recognition and the original version of *BoxCars* dataset [198]. My most significant contribution to this article was the proposition

of multiple augmentation techniques for neural network training, share in the redesign of 3D bounding box estimation from a single image, and finally, the curation process and further expansion of vehicle fine-grained recognition dataset in an unconstrained environment. This article is cited by other researchers either for the BoxCars116k dataset or the presented method. For this reason, its main parts are covered in Chapter 5 on page 43.

This work was awarded the second-highest possible mark in the 2019 National Quality Assessment *(SKV Biblio 2019)* organized by Research, Development and Innovation Council, Government of the Czech Republic.

Monocular Vehicle Speed Measurement This research field is covered by paper *Sochor* et al.*, Comprehensive data set for automatic single camera visual speed measurement*, fifth row in Table 1.1. As the third author of this paper, my role was to organize, provide and control the data acquisition process and post-process collected data. I was also involved in the evaluation process. The dataset acquisition process and statistics are included in this thesis in Chapter 6 on page 54.

By augmenting the understanding and practical application of ITS, this thesis aims to contribute to the ongoing evolution of transportation systems, emphasizing robustness, efficiency, and sustainability. The broader objective is to catalyze advancements that will redefine the landscape of urban mobility, shaping a future where transportation is more intelligent, more responsive, and more inclusive.

Chapter 2

Existing Methods in Traffic Surveillance

This chapter compiles and further extends related work for all ITS fields, that were covered in published This chapter compiles and further extends related work for all ITS fields covered in published articles, forming the core of this thesis. It describes state of the art in part of the automatic traffic video surveillance systems, namely automatic camera calibration (Section 2.1), object detection (Section 2.2), object fine-grained recognition (Section 2.4), object re-identification (Section 2.6) and license plate recognition (2.8). Methods focused on detection, fine-grained recognition, and re-identification of vehicles are described in Sections 2.3, 2.5 and 2.7, respectively.

2.1 Camera Calibration for Speed Measurement of Vehicles

One of the most important parts of speed measurement of vehicles from a single monocular camera is calibration of the camera. In a general case, this includes dealing with perspective projection and different rotations of the camera; it is also necessary to deal with unknown distance from the camera to the ground plane of the road and possibly with radial and tangential distortion. It is usually necessary to obtain intrinsic and extrinsic camera parameters together with the scene scale (or the distance of the camera from the road/ground plane). Therefore, we include also a brief overview of the typical solutions of camera calibration for speed measurement of vehicles. However, the definition of traffic surveillance camera calibration is included in the first place. This section contains compiled version of related work for Chapter 6 on page 54 compared to original paper.

2.1.1 Traffic Surveillance Camera Calibration

General mathematical model for camera calibration is represented by a projection matrix $\mathbf{P} = \mathbf{K}\ [\mathbf{R}\ \mathbf{T}]$, where $\mathbf{K}$ denotes intrinsic camera parameters, $\mathbf{R}$ stands for camera rotation, and $\mathbf{T}$ represent camera translation. The extrinsic parameters (rotation and translation) are relative to defined world coordinate system (see Figure 2.1). Since the calibration for traffic surveillance is specific, we describe all these aspects with **application to vehicle speed measurement** in mind.

Goal The essential goal of traffic surveillance camera calibration is to measure speed of vehicles. For the speed measurement, it is required to be able measure time and distances

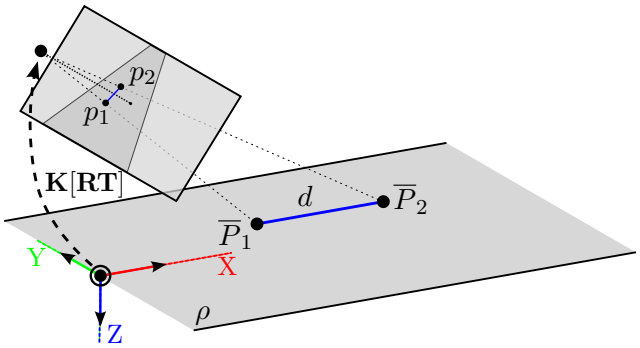


Figure 2.1: The essential goal of traffic surveillance camera calibration is to be able measure real world distance d between two points ($\overline{P}_1$, $\overline{P}_2$) on road plane given their projection to the image (p_1, p_2). X,Y, and Z axes represent a real world coordinate system and **K** represent intrinsic camera parameters, while **R** and **T** are extrinsic camera parameters.

on the road plane. The time measurement part is rather trivial. However, for the distance measurement it is necessary to measure the distance between two points on the road plane (or any other plane parallel to the road plane and with known distance from the road plane) given their projection to the image. See Figure 2.1 for an example.

Input For fully automatic methods, the camera calibration input is usually a video of the observed traffic scene. However, for methods which include manual steps, part of the input are also usually distance measurements on the road plane.

Assumptions Zero pixel skew is generally used as an assumption about the camera model. Another widely used assumption is that the camera's principal point is in the center of the image. Also, there is usually the assumption that the road can be approximated by a plane. The authors usually assume that the observed road segment is approximately straight, that the vehicles move straight and their velocity is constant on the measured segment (no acceleration).

Mathematical model As the standard camera model with **K** [**R** **T**] matrices is sufficiently described in existing literature [71]; we refer the readers there. However, it is also possible to use a different formulation based on vanishing points of the road plane [41]. This formulation is easily convertible to the standard one. Finally, for computation of 3D real world coordinates on the road plane of a point in image space, it is necessary to compute intersection of the road plane (e.g. $z = 0$ as shown in Figure 2.1) and a ray defined by the camera optical center and the coordinates on the image plane.

Attributes One important attribute of the camera calibration algorithm, which should be kept in mind, is whether the algorithm works automatically in the sense that there is no manual input required per installed camera. The property of being automatic becomes more important as the number of installed cameras grows. A number of papers and approaches to solving this problem exist and they will be discussed in detail in the following text. Another important attribute is whether the algorithm works from arbitrary viewpoint, as it is a significant drawback of a method if it requires specific camera placement relative to the observed road.

2.1.2 Camera Calibration Methods

The calibration methods for surveillance cameras can be divided into four main research directions. A brief summary is given in this section. For more details follow our original paper [200] or individual works from this summary.

Methods Based on Acquired Line Markings These methods [78, 79, 19, 63, 254] are based mainly on the detection of vanishing points and the assumption that the camera is only tilted along Y axis in Figure 2.1. However, by definition, this class of methods based on observed line markings is usable only when the line markings are present, visible, and recognizable.

Methods Based on Vehicles' Movement Works [43, 41, 187, 49] are part of the second research direction. The methods based on vehicles' movement no longer need visible road markings; however, when used on small local roads, the calibration may take some time as it usually improves with more observed vehicles.

Methods Using Manual Measurements Measurements on the road [152, 164, 286, 195, 149] have the biggest disadvantage that it is necessary to do the manual measurements, which potentially can mean stopping traffic on the road. The advantage of the methods may be (in some cases) that they are more accurate than automatic or semi-automatic ones.

Uncategorised methods Dailey *et al.* [34] proposed a method for vehicles speed measurement based on tracking of vehicle blobs and constraining them to move along a line. The blobs are detected as inter-frame differences followed by Sobel edge detector. The authors assume that the vehicles are moving towards or from the camera and use mean length of vehicles to obtain the scene scale.

Do *et al.* [37] proposed a camera calibration method for speed measurement based on artificial markers drawn on the road. They assume that the camera has zero pan angle and that markers determining vertices of an equilateral triangle with a known distance between vertices which are visible on the road. They used the triangle to obtain the scale factor and the tilt angle.

Lan *et al.* [114] use optical flow to compute the speed of different points of a vehicle and they average this speed to get the speed of vehicle in image units. However, to convert them into kilometers per hour, the authors assume that there is no perspective projection effect and the width of the ROI (width of lanes) is known.

2.1.3 Automatic Calibration Method based On Statistics of Dimensions

This method was developed in our institution by Dubská *et al.* [42, 43]. This work fits into the group of *methods based on vehicles' movement*. Details about this method are given here, as it meets all our requirements (it is fully automatic and it is usable from arbitrary viewpoint) and we use it later in the experiments. In principle, the method relies on camera calibration from two automatically detected vanishing points. The authors use a simple foreground detection model to filter areas with movement. The first vanishing point (VP1, which is in the direction of vehicles' movement) is recovered from tracked feature points on the vehicles using min eigenvalue detector and KLT tracker. The tracked points' motion is transformed using a line-to-line Hough transformation parametrized by

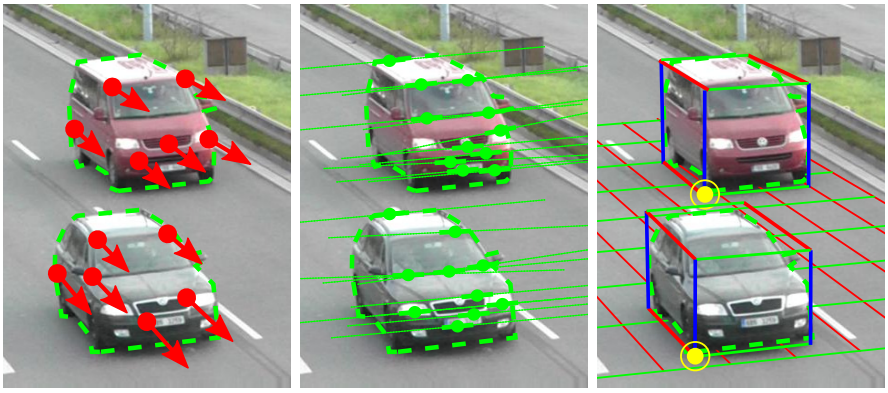


Figure 2.2: Automatic camera calibration according to Dubská [43]. From left to right: Tracked keypoints for VP1, oriented edges voting for VP2, and road plane with bounding boxes for the cars and reference points for tracking.

parallel coordinates [42] where the global maximum corresponds to the image of the first vanishing point. The second vanishing point (VP2) is extracted from strong edges present on the moving vehicles meeting some conditions given by the position of the VP1. The edges (and their orientations) are, again, transformed to the Hough space where the strongest maximum accounts for the vanishing point. From these two vanishing points, the camera intrinsics and extrinsics can be recovered (assuming principal point in the image center, square pixels and zero skew).

The authors propose an algorithm for computing the 3D bounding box around the vehicle blobs. Mean size of the bounding boxes and known mean dimensions of the vehicles for a given country accounts for the scene scale. Vehicle speed is measured simply by tracking 3D bounding boxes around the blobs using Kalman filter and measuring the travel distance in the real world.

The method was evaluated on several videos with several car passes with ground truth speed obtained from GPS.

2.1.4 Summary and Analysis of the Camera Calibration Methods

A summary of the presented camera calibration methods can be found in Table 2.1. As the table shows, some approaches have different limitations and do not work under all conditions. The reported mean error varies greatly – it should be noted that the error is not directly comparable, as it was evaluated by the authors on different datasets (generally not publicly available) and by different protocols.

To sum up the camera calibration methods, some of them [34, 114] do not take perspective projection into account, some algorithms [34, 63, 164, 114, 37] have limitations in camera placement. Quite a large number of approaches [152, 164, 195, 149] use measurements in the scene which enable direct camera calibration. Methods [78, 37] using a calibration pattern (virtual or drawn on the road) have been proposed. Another set of methods uses vanishing points to obtain camera calibration [187, 19, 63, 43].

Several approaches to scale calibration have been proposed. Besides the multiple manual measurements on the road [152, 164, 195, 149] and calibration patterns [78, 37], two groups of methods exist. The algorithms from the first one [187, 19, 63, 114] use one known distance

Table 2.1: Summary of different camera calibration methods for speed measurement. It should be noted that the reported errors are only informative as all the methods are evaluated on different datasets and by different protocols. We consider a system to be automatic if it does not require any manual calibration for each individual camera. **auto** – denotes whether the system works fully automatically, **view** – denotes whether the system is usable from arbitrary viewpoint

	camera calibration method	**auto**	**view**	**mean error**
Dailey *et al.*, 2000 [34]	multiple assumptions on vehicle movements and known mean length of vehicles	✓	✗	6.5 km/h
Schoepflin *et al.*, 2003 [187]	detection of two vanishing points, one known length	✗	✓	N/A
Cathey *et al.*, 2005 [19]	vanishing point obtained from detected line markings, scale computed from lengths of stripes	✗	✓	N/A
Grammatik. *et al.*, 2005 [63]	one vanishing point obtained from detected line markings, the second one assumed in infinity, one known distance is required	✗	✗	3 km/h
He and Yung, 2007 [78]	calibration by pattern formed by lane markings	✗	✓	3.27 %
Maduro *et al.*, 2008 [152]	known angle of the ground plane, lengths of line markings' stripes	✗	✓	2 %
Nurhadiyatna *et al.*, 2013 [164]	known distances in the real world and in the scene, zero pan assumption	✗	✗	7.63 km/h
Sina *et al.*, 2013 [195]	manual measurements	✗	✓	3.3 km/h
Dubská *et al.*, 2014 [43]	detection of two vanishing points, scale computed by matching of statistics of vehicles' dimensions to mean dimensions of vehicles	✓	✓	1.99 %
Lan *et al.*, 2014 [114]	relaxation of perspective projection, known width of lanes	✗	✗	0.9 % – 2.5 %
Luvizon *et al.*, 2014 [149]	known real world measures	✗	✓	1.63 km/h
Do *et al.*, 2015 [37]	zero pan assumption, equilateral triangle drawn on the road	✗	✗	2.91 %
Filipiak *et al.*, 2016 [49]	constant speed assumption, evolutionary algorithm to recover intrinsic and extrinsic parameters from detected license plate sequences	✓	✗	2.3 km/h
You *et al.*, 2016 [254]	detection of vanishing point in the direction of vehicles' movements from lane markings and vanishing point perpendicular to road plane from poles and pedestrians, the scale is obtained from known height of camera above the road	✗	✓	N/A

in the scene (e.g., length of line marking stripe). The other methods use dimensions of vehicles [34, 43] to obtain a proper scale calibration.

A critical attribute of the calibration methods is whether they work fully automatically and do not require manual per-camera calibration input. Automation helps reduce the cost of camera installation, and automatic methods have better scaling properties. Only two approaches are fully automatic and do not require any manual camera calibration. Both of these methods [34, 43] use the mean dimension of vehicles to obtain a proper scaling factor for the given camera.

Methods [152, 164, 195, 149, 78, 37] which require measurements of physical dimensions on the road have even more significant drawbacks concerning the scaling properties. To perform the measurements, stopping (or limiting) traffic on the road is usually necessary, increasing installation time and costs.

Another critical attribute is whether the camera can be placed at an unconstrained position above the road, as some methods require that the camera has zero pan. This requirement can be hard to guarantee in real-world scenarios when the camera is not placed on a portal above the road. The only method that satisfies the conditions of automatic calibration from arbitrary view is [43], which we use later in the experiments.

Bartl *et al.* benefited from the prior knowledge published in works [43, 41, 200, 199] and further extends the idea of automatic camera calibration for traffic scenes either by optimization of the road plane [8, 9] or calibration based on landmark detection [10]. Deep neural networks were also adopted for the estimation of vehicle 3D location [222] or regression of camera calibration parameters from traffic scenes [268].

2.1.5 Evaluation Datasets Used in Existing Works

The described methods usually used different methods for evaluation of the speed measurement and ground truth speed acquisition. Some methods [34, 187, 149, 49] use inductive loops for ground truth acquisition, other methods [152, 43] GPS or RADAR [114]. Do *et al.* [37] used the speedometer on a motorbike, which should be considered very imprecise.

When it comes to the number of evaluated speed measurements, Lan *et al.* [114] used 2 010 ground truth speeds (only one video sequence), others [34, 187, 49] have hundreds of vehicles with known ground truth. And there are also works [63, 78, 152, 164, 195, 43, 114, 149, 37] that use at most tens of ground truth speeds with the lowest number in [37] (one ground truth speed) and the highest number of 75 measurements in [149]. Cathey at al. [19] have no evaluation at all. A summary of existing datasets can be found in Table 2.2. It should be noted that with the exception of [164, 43], the datasets are not publicly available which makes comparison of the methods impossible.

Almost every mentioned dataset (except [195] and a part of [78]) is recorded in daylight as the methods usually become unusable in the night when only headlights of vehicles are visible. Existing datasets usually evaluate only speed measurement error (with different statistics – mean, deviation etc.) and some exceptions (see Table 2.2) evaluate also other tasks.

The existing evaluation of algorithms should be considered insufficient as existing works use a small number of observed vehicles and scenes. Also, for GPS and speedometer, the ground truth is imprecise as in our evaluation GPS has mean error over 2 % and speedometer reports higher speed then the actual. Therefore, we created our novel dataset with precise ground truth and 20 865 of vehicles with ground truth speed. It is also possible to evaluate other camera calibration aspects such as calibration error and distance measurement on the road plane with the computed scale. These two metrics can provide interesting insights into properties of camera calibration algorithms as they are needed and harnessed in the intelligent transportation surveillance.

Besides evaluation datasets covered in Table 2.2 on the next page, multiple new datasets were for automatic camera calibration were published subsequently. SVLD-3D is combination of proposed BrnoCompSpeed dataset with dataset collected by Tang *et al.* [222]. The annotations were enriched by information related to 3D bounding boxes, such as centroids and vertices. On the other hand, *BrnoCarPark* by Bartl *et al.* [10] is dataset for automatic camera calibration based on vehicle landmarks.

Table 2.2: Summary of datasets used for evaluation of visual speed measurement methods.

dataset	videos	vehicles	source of gt	resolution	evaluation metrics
Dailey, 2000 [34]	1	532	induction loops	N/A	speed measurement error
Schoepflin, 2003 [187]	2	1 015	induction loops	320 × 240	speed measurement error
Grammatik., 2005 [63]	1	20	manual measurements	768 × 576	speed measurement error
He, 2007 [78]	1	64	RADAR	1280 × 1024	speed measurement error
Maduro, 2008 [152]	2	few	GPS	N/A	speed measurement error
Nurhadiyatna, 2013 [164]	10	15	GPS	320 × 240	speed measurement error
Sina, 2013 [195]	13	13	GPS	N/A	speed measurement error, vehicle counting
Dubská, 2014 [43]	6	29	GPS	864 × 480	speed measurement error, distance measurement error
Lan, 2014 [114]	1	2 010	RADAR	640 × 480	speed measurement error
Luvizon, 2014 [149]	1	75	induction loops	768 × 480	speed measurement error, license plate detection
Do, 2015 [37]	1	3	speedometer	N/A	speed measurement error
Filipiak, 2016 [49]	2	955	induction loops	1280 × 720	speed measurement error
BrnoCompSpeed	**18**	**20 865**	**LIDAR gates**	**1920 × 1080**	**calibration error, distance measurement error, speed measurement error, vehicle counting recall, false positives vehicles per minute**

2.2 Object Detection in General

Humans are perfectly capable of differing object in the scene from each other, but computers are not. Color information provided by an image is not sufficient and further image processing is necessary. For this reason, different types of features and descriptors were introduced into computer vision field for object detection or localization in image.

2.2.1 Feature Extraction

The most basic image feature extraction can be provided by simple image thresholding. However, binary information from thresholded images may be insufficient for more advanced detection task. Thus more robust and advanced features must be used.

Low-Level Features

The most basic low-level features for object detection are *corners* (also called *keypoint features* or *interest points*). Another class of low-level features are *edges*, which can be grouped into longer *curves* and *straight line segments*. This kind of features can be matched

based on their local appearance and localization in image and can also be good indicators of object boundaries. Both categories examine gradients in image for feature extraction or detection. Thus edges can be defined as the rapid change of intensities in image in one direction and corners can be defined as regions with rapid change of intensities in more than one direction. Edge and corner detectors are based on this approach and are using kernels for enhancement of gradient in image. In case of edge detection, the first and second order gradients are used. The most commonly used edge detectors are Canny edge detector [17], Sobel operator [197] (first-order methods) or Laplacian of Gaussian [154] (second-order method), and two commonly used corner detectors are Moravec [160] and Harris [70] corner detector.

Feature Descriptors

More advanced methods for image feature extraction/detection and matching these features in different images are image descriptors. Image descriptors usually combine more information to create richer and more robust description of image.

Scale invariant feature transform (**SIFT**) [143] aim to resolve many of the practical problems in low-level feature extraction and their use in image matching. SIFTs are formed by computing the gradient at each pixel in a 16×16 image sub-window around detected keypoint, using appropriate level of the Gaussian pyramid. In each 4×4 quadrant, a gradient orientation histogram is formed adding the weighted gradient value to one of eight orientation histogram bins. This results in 128 dimensional vector of non-negatives values for each detected keypoint.

Histogram of oriented gradients (**HOG**) [35] is feature descriptor based on local object appearance and shape within an image, which can be described by the distribution of intensity gradients or directions of edges. The image is divided into cells and for the pixels, in each cell, a histogram of oriented gradients is computed. HOG descriptor is then made by concatenation of this computed histograms.

Speeded up robust features (**SURF**) [11] approach employs approximations to second-order edge detection. SURF operator is computed on integral image by approximations of second-order differencing for Laplacian-of-Gaussian (LoG) operator.

Gradient location-orientation histogram (**GLOH**) [159] is a variant on SIFT that uses a log-polar binning structure instead of the four quadrants used in SIFT.

2.2.2 Object Detection Methods

The task of detection can be defined as localization of given pattern in image. Thus, detectors are detection methods or algorithms, which are capable of determining if the searched pattern is located in image or not, and give pattern position in the image. In this way, detection can be reformulated as image classification task and then classifiers can be used.

Boosted Classifiers are widely used by computer vision community because of their performance in image classification. The first boosted classifier was presented by Viola and Jones [228]. They developed an approach for visual object detection using cascade classifier and introduced AdaBoost algorithm for its training using Haar-like features. Cascade classifier is made from numerous smaller and weaker classifiers, which have together better performance than a single classifier. The main contribution of this approach is decision in multiple stages.

Decision trees are another approach to multistage decision-making and their impact on the pattern recognition task was discussed in numerous publications [103, 161, 119].

Neural Networks

Neural networks are widely used in machine learning nowadays. They are computational models, based on a collection of neural units (neurons), which try to solve problems like human brains. Each neural unit is connected with many others and transmits information to other neurons based on its activation state. Neural networks are self-learning and trained, and reach great performance in areas, where the solution of feature detection is difficult to express in a traditional computer vision. Many variants of neural networks were developed. However, only convolutional neural networks are covered in this part.

Convolutional neural networks (CNNs) play an important role in computer vision nowadays. They excel in many computer vision tasks like image classification, image super-resolution, text de-blurring and text recognition, object recognition, even object detection.

The proof that convolution neural networks can resolve as object detectors can be found in multiple publications [58, 182, 179, 134].

These days, convolution neural networks also thrive in other computer vision disciplines. They excel at various computer vision tasks, including object identification, text de-blurring, text recognition, and image super-resolution.

Due to their precision, CNNs are favored for object detection tasks. Numerous studies employing CNN to detect objects have been done, including [134, 180, 218, 216, 58, 57, 182, 76, 33].

Based on their behavior, these networks may be divided into three basic *meta-architectures*. The first component of the detection network, known as the feature extractor, is shared by all meta-architectures. Any currently available CNNs, such as VGG-16 [194], Inception-v2 [93], Inception-v3 [219], Inception-v4 [216] ResNet-101 [76], MobileNet [83], etc., can be used as a feature extractor.

The first meta-architecture covers the term *Single Shot Detector* (SSD). Despite the work by Liu et al. [134] using the term SSD as the detector's name, this context refers to a class of detectors that use a single feed-forward CNN to make direct class predictions without the need for proposed boxes for a second stage of classification. Apart from SSD detector, typical members of this group include YOLO [180], Multibox [218] or the Region Proposal Network (RPN) stage of Faster R-CNN [182], which are used to predict class-independent box proposals.

The second meta-architecture represents *Faster R-CNN* [182]. It is the evolution of R-CNN [58] and Fast R-CNN [57]. In this setting, detection is divided into two stages. In the first stage, called *Region Proposal Network* (RPN), features extracted from the image by an intermediate level of feature extractor are used to predict class-independent box proposals. In the second stage, box proposals are used to crop extracted features from the feature map, which are passed to the following levels of the feature extractor to predict classes and class-related boxes. Plenty of work is based on Faster R-CNN meta-architecture since 2015 [76, 256, 12, 32, 192] including SSD and R-FCN.

The last meta-architecture is called *Region-based Fully Convolutional Networks* (R-FCN) [33] follows-up the idea of the Faster R-CNN. However, crops are taken from the last layer preceding prediction instead of cropping features from the same level where region proposals were predicted. This step reduces per-region computation, leading to faster prediction than in the case of Faster R-CNN, while accuracies are comparable.

Table 2.3: A summary of vehicle detection methods with a brief description. **Mot.** – denotes motion-based methods. **App.** – denotes appearance-based methods.

	Mot.	App.	Features	Description
Cucchiara *et al.*, 2000 [31]	✓	✗	Multi-features: color, corners, edge maps and three-frame difference	Detect vehicles in urban traffic scenes by means of rule-based reasoning on visual data
Gupte *et al.*, 2002 [68]	✓	✗	Multi-features: color, corners, edge maps and background subtraction	Method is based on the correspondences between regions and vehicles. Experiment is performed on highway scenes.
Ma *et al.*, 2005 [151]	✗	✓	SIFT and edge features, Bayesian classification	Vehicle classification for mid-field video surveillance
Tsai *et al.* 2007 [226]	✗	✓	Multi-features: color, corners, edge maps, and wavelet coefficients, RBF network, Bayesian classifier	A novel color model and a multichannel classifier
Ottlik *et al.* 2008 [167]	✓	✓	Optical flow and edge features, 3D wireframes	Experiments on urban traffic videos with entire automatic initialization
Buch *et al.* 2009 [16]	✗	✓	3D histogram of oriented gradients (3DHOG)	Extension to HOG feature extraction by applying 3D spatial modeling
Feris *et al.* 2011 [48]	✗	✓	Haar-like, AdaBoost	A novel detection/tracking approach for capturing vehicles in challenging urban environments

2.3 Detection of Vehicles in Image and Video

Reliable and robust vehicle detection or localization in an image is one of the key parts of traffic monitoring and traffic analysis systems. The accuracy of vehicle detection has great impact on vehicle tracking, vehicle movement expression, and behavior understanding. There are two main categories in vehicle detection research:

- Methods based on **appearance** features.
- Methods based on **motion** features.

Appearance-based methods use the appearance features, e.g., color, texture and shape of vehicle to detect, or separate the vehicle from the background. Motion-based techniques use movement characteristics to distinguish vehicles from the stationary background images. Some methods can combine both principles. Table 2.3 presents methods described below.

2.3.1 Methods Based on Appearance Features

Ma *et al.* [151] designed framework for vehicle classification under a mid-field surveillance. SIFT features [143] with several important modifications were adopted as a descriptor for edge points. Firstly, polarities of SIFT features are discarded (gradient orientations with 180° differences are regarded as the same), that leads to a more robust solution against contrast differences and lighting changes. Secondly, gradient magnitudes are thresholded before forming a SIFT vector to reduce the influence of large specular reflections and non-uniform illumination changes. Thirdly, χ^2-distance between SIFT vectors is used. The final decision between pre-trained models is done by Bayesian classification.

Tsai *et al.* [226] proposed a method for detecting vehicles in static images using color and edges. This method introduces a new color transform model to locating possible vehicle

candidates by finding important vehicle color. In the beginning, color transformation is used to project all the colors of input pixels into a color space where vehicle pixels can be easily identified from backgrounds using Bayesian classifier or RBF network [15] (neural network with radial basis function as activation function). After color classification, a potential vehicle is verified using corner features and edge maps with wavelet coefficients.

Buch *et al.* [16] proposed a method for the detection and classification of individual vehicles and pedestrians in urban scenes with a modification of 2D features, derived from the histogram of oriented gradients (HOG), into 3D space. A calibrated camera allows them an affine transform of the observation into a normalized representation from which, using a combination of 3D interest points and HOG, the 3D-HOG features are defined. A variable set of interest points is used in the detection and classification processes, depending on which points in the 3D model are visible.

Feris *et al.* [48] presented a novel approach for vehicle detection in urban surveillance videos capable of handling large occlusions, different vehicle shapes and environmental condition changes (lighting changes, rain, shadows, and reflections). They created dataset of vehicles from surveillance cameras containing about 1,000,000 pictures from different surveillance cameras (different camera pose) and different lighting conditions with a user supervision. After that, they generate even more sample with partial occlusion pasting another vehicle on image I_B into image I_A using Poisson image reconstruction. Finally, they train the detector proposed by Viola and Jones [228] based on AdaBoost classifiers with Haar-like features. One disadvantage of this method is that one detector is needed for each camera view.

2.3.2 Methods Based on Motion Features

Cucchiara *et al.* [31] presented an approach for detecting vehicles in urban scenes with the help of rule-based reasoning on extracted features. The image processing (low-level) part extracts features from the scene by spatiotemporal analysis during daytime, and by morphological analysis of headlight at night. Then, vehicles are detected using different sets of rules in high-level module.

Gupte *et al.* [68] presented a method for vehicle detection and classification based on establishment of correspondences between regions and vehicles, as the vehicles move through the image sequence. First, a object mask is created using background subtraction. They are adapting the background image by addition and subtraction of specific parts of the object mask creating *instantaneous background*. Background image adapted in this way is more robust to illumination changes than the regular one. Regions extracted from the object mask are tracked through the video sequence for better traffic monitoring.

2.3.3 Methods Based on Appearance and Motion Features

Ottlik *et al.* [167] used Optical Flow (OF) with 3D wireframes for vehicle segmentation, estimated from line structure of the interlaced video recording to form OF Fields. OF Fields are estimated and segmented for each frame of the image sequence resulting into object image candidates (OICs). Confidences of OICs are accumulated by short-time tracking of OP fields. The final decision is made by predefined vehicle 3D models fitting on edge segments extracted from OICs. Compatibility of the 3D wireframe model and edge segments is checked using thresholds, one for the minimum score in the Hough space and a second one for the overlap between optical flow segment and object candidate. The disadvantage of this method is the need of 3D wireframes definition for each vehicle body type.

2.4 General Fine-Grained Object Recognition

We divide the fine-grained recognition methods from recent literature into several categories as they usually share some common traits. Methods exploiting annotated model parts (e.g. [92, 264]) are not discussed in detail as it is not common in fine-grained datasets of vehicles to have the parts annotated. This section contains updated related work for Chapter 5 on page 43 which was further elaborated from original paper.

2.4.1 Automatic Part Discovery

Parts of classified objects may be discriminatory and provide lots of information for the fine-grained classification task. However, it is not practical to assume that the location of such parts is known a priori as it requires significantly more annotation work. Therefore, several papers [250, 40, 251, 108, 193, 109, 269] have dealt with this problem and proposed methods how to automatically (during both training and test time) discover and localize such parts. The methods differ mainly in the ways in which they are used for the discovery of discriminative parts. The features extracted from the parts are usually classified by SVMs.

2.4.2 Methods using Bilinear Pooling

Lin *et al.* [128] use only convolutional layers from the net for extraction of features which are classified by a bilinear classifier [172]. Gao *et al.* [51] followed the path of bilinear pooling and proposed a method for Compact Bilinear Pooling getting the same accuracy as the full bilinear pooling with a significantly lower number of features.

2.4.3 Other Methods

Xie *et al.* [242] proposed to use a hyper-class for data augmentation and regularization of fine-grained deep learning. Zhou *et al.* [280] use CNN with Bipartite Graph Labeling to achieve better accuracy by exploiting the fine-grained annotations and coarse body type (e.g. Sedan, SUV). Lin *et al.* [126] use three neural networks for simultaneous localization, alignment and classification of images. Each of these three networks does one of the three tasks and they are connected into one bigger network. Yao *et al.* [251] proposed an approach which uses responses to random templates obtained from images and classifies merged representations of the response maps by SVM. Zhang *et al.* [265] use pose normalization kernels and their responses warped into a feature vector. Chai *et al.* [20] propose to use segmentation for fine-grained recognition to obtain the foreground parts of an image. A similar approach was also proposed by Li *et al.* [123]; however, the authors use a segmentation algorithm which is optimized and fine-tuned for the purpose of fine-grained recognition. Finally, Gavves *et al.* [52] propose to use object proposals to obtain the foreground mask and unsupervised alignment to improve fine-grained classification accuracy.

2.5 Fine-Grained Recognition of Vehicles

The goal of fine-grained recognition of vehicles is to identify the exact type of the vehicle, that is its make, model, submodel, and model year. The recognition system focused only on vehicles (in relation to general fine-grained classification of birds, dogs, etc.) can benefit

from that the vehicles are rigid, have some distinguishable landmarks (e.g. license plates), and rigorous models (e.g. 3D CAD models) can be available.

2.5.1 Methods Limited to Frontal/Rear Images of Vehicles

There is a multitude of papers [171, 36, 30, 170, 174, 117, 261, 141] using a common approach: they detect the license plate (as a common landmark) on the vehicle and extract features from the area around the license plate as the front/rear parts of vehicles are usually discriminative. There are also papers [262, 85, 88, 125, 7, 74] directly extracting features from frontal images of vehicles by different methods and optionally exploiting the standard structure of parts on the frontal mask of car (e.g. headlights).

2.5.2 Methods Based on 3D CAD Models

There were several approaches on how to deal with viewpoint variance using synthetic 3D models of vehicles. Lin *et al.* [129] propose to jointly optimize 3D model fitting and fine-grained classification, Hsiao *et al.* [84] use detected contour and align the 3D model using 3D chamfer matching. Krause *et al.* [110] propose to use synthetic data to train geometry and viewpoint classifiers for the 3D model and 2D image alignment. Prokaj *et al.* [173] propose to detect SIFT features on the vehicle image and on every 3D model seen from a set of discretized viewpoints.

2.5.3 Other Methods Based on Hand-crafted Features

Gu *et al.* [65] propose extracting the center of a vehicle and roughly estimate the viewpoint from the bounding box aspect ratio. Then, they use different Active Shape Models for alignment of data taken from different viewpoints and use segmentation for background removal. Stark *et al.* [210] propose using an extension of Deformable Parts Model (DPM) [47] to be able to handle multi-class recognition. The model is represented by latent linear multi-class SVM with HOG [35] features. The authors show that the system outperforms different methods based on Locally-constrained Linear Coding [232] and HOG. The recognized vehicles are used for eye-level camera calibration. Boonsim *et al.* [14] propose a method for fine-grained recognition of vehicles at night. The authors use relative position and shape of features visible at night (e.g. lights, license plates) to identify the make&model of a vehicle, which is visible from the rear side.

2.5.4 Machine Learning Based Methods

Liu *et al.* [132] use deep relative distance trained on a vehicle re-identification task and propose training the neural net with Coupled Clusters Loss instead of triplet loss.

Fang *et al.* [46] propose using an approach based on detected parts. The parts are obtained in an unsupervised manner as high activations in a mean response across channels of the last convolutional layer of used CNN. The authors in [89] introduce spatially weighted pooling of convolutional features in CNNs to extract important features from the image. Liu *et al.* [139] suggest using a part object relation provided by the CapsNet to solve problems that rely on a relation-based inference to detect visual saliency.

Since 2020, Lu *et al.* [145] propose using a part-level feature extraction method to enhance the discriminative ability of deep convolutional features for fine-grained vehicle recognition. More specifically, a basic feature grouping module is adopted to integrate

the feature maps of deep convolutional layers into groups in each of which the related discriminative parts are assembled. Then a fusion module follows to model the coarse-to-fine relationship of the part features and further ensure the integrity and effectiveness of the part features. In paper [146], authors further elaborated the idea, and they extract features from multiple ROIs from vehicles' frontal images. In 2023 feature aggregation part was added to the framework [144].

Hu *et al.* [87] proposed a method for fine-grained vehicle recognition in traffic surveillance videos. In each frame, vehicles are detected and tracked. 3D orientation within a frame is calculated based on pose estimation for each vehicle instance. RNN is used for the final decision combining visual, spatial, and temporal information from each video frame.

Besides traditional CNN architecture, the vision transformer (ViT) technique has shown outstanding performance in fine-grained object recognition tasks. Hu *et al.* [90] proposed a recurrent attention multi-scale transformer (RAMS-Trans), which recursively uses the self-attention mechanisms to learn discriminant region attention in a multi-scale manner. TransFG model by He *et al.* [75] integrates the raw weight of the transformers into an attention map, on which the network could select discriminative image patches and calculate their relation. This mechanism, named Part Selection Model, can be applied to most transformer architectures.

2.5.5 Summary of Existing Methods

Existing methods for the fine-grained classification of vehicles usually have significant limitations. They are either limited to frontal/rear viewpoints [171, 30, 174, 261, 141, 85, 88, 7] or require some knowledge about 3D models of the vehicles [173, 110, 84, 129] which can be impractical when new models of vehicles emerge. Our proposed method does not have such limitations. The method works with arbitrary viewpoints, and we require only 3D bounding boxes of vehicles. The 3D bounding boxes can be either automatically constructed from traffic video surveillance data [43, 41] or estimated from a single image during training/testing by our proposed method (see Section 5.2.4).

After the publication of our vehicle fine-grained recognition method, CNN models were broadly adopted [139, 145, 146, 144, 87] and similarly to other areas, vision transformers were applied as well [90, 75].

2.6 Object Re-Identification Methods

Object re-acquisition or re-identification is the process of matching identical objects between images taken from separate cameras. There is an enormous growth in requirements for object re-identification due to the expansion of security systems and increasing public security. There are two main areas of object re-identification topics. The first of them is focused on the re-identification of persons. The second is focused on the re-identification of vehicles.

The problem of person re-identification has been examined in numerous publications, and different approaches for this task were developed.

The first approach is the extraction of appearance-based features and their matching with already extracted features from different cameras. Oliveira *et al.* [165] presented a person re-identification solution based on local appearance features – color histograms in HSV space and SURF descriptors.

The second approach uses deep learning, such as neural networks, for person re-identification. Chopra *et al.* [29] adopted the Siamese neural network (SNN) for face verification and obtained excellent results. Zhang *et al.* [263] achieved outstanding performance in gait recognition for person identification using SNN. McLauglin *et al.* [158] presented a novel approach to person re-identification using recurrent convolutional network mixed with Siamese networks. Two inputs for this network were used – color image and optical flow computed from video sequence. This combination of inputs, network structure, and time recurrence have great results, and a promising way for future research in object re-identification was established. This section contains updated related work for Chapter 7 on page 62 which was further elaborated from original paper.

2.6.1 Person Re-Identification

Besides standard deep features learned by a Siamese network [158, 266, 245, 25, 284], other approaches to person re-identification have been proposed [274, 278, 130].

Several papers proposed to use body parts [27, 105, 120, 273]. Other papers went beyond Siamese networks and proposed triplet loss [27, 80] or quadruplet loss [26]. There were also attempts to learn a metric for the re-identification like KISSME [113], XQDA [125], You J. *et al.* [254] learn Mahalanobis distance on LBP and HOG3D features, and finally Shi *et al.* [191] learn Mahalanobis distance in an end-to-end manner. Sun *et al.* [214] proposed to use SVD for weight matrix orthogonalization to de-correlate feature vectors for person re-id.

Other authors exploit different types of features. For example, Wu *et al.* [239] propose to use deep features learned by a CNN together with hand-crafted features. The final representation for an image is obtained by fusing these features. Matsukawa *et al.* [156] use a novel descriptor based on hierarchical gaussians computed for patches in image. Chen *et al.* [24] propose to use compact binary hash codes as features for fast person re-identification. There were also attempts [133, 51] to recognize the walking cycle in image sequence and use the walking cycle to improve the accuracy of re-identification.

A group of works also propose to replace different parts of the re-identification pipeline by alternative solutions. Zhong *et al.* [279] use re-ranking based on k-reciprocal nearest neighbors to improve the performance. Zhou *et al.* [281] propose to use point-to-set distance instead of standard point-to-point. Lin *et al.* [127] take inter-camera consistencies of id assignment into account during training and inference to boost the results of

re-identification. Xia *et al.* [241] propose to use domain guided dropout to improve re-identification performance when trained on multiple datasets. Wang *et al.* [229] propose to add a network computing a cross-image representation for pairs of images. Cho *et al.* [28] estimate persons' poses and compare images with each person in an as similar as possible pose. Su *et al.* [213] use attributes (e.g "long sleeve") for person re-id. The attributes are first learned on a different dataset with attributes present and then fine-tuned for the target dataset. The attributes supervision for the target dataset comes from the assumption that same person has the same (unknown) attributes.

Luo *et al.* [147, 148] sets a strong baseline for deep person re-identification methods applicable to other domains as well.

2.7 Vehicle Re-Identification in Traffic Surveillance

Vehicle re-identification is the problem of identifying the same vehicle across different surveillance camera views. This problem consists of extraction of sufficient information from the detected vehicle in the image or video and the effectiveness usage of this information to find the same vehicle in a different set of detected vehicles. There are three major classes of vehicle re-identification methods.

- License plate recognition methods
- Appearance-based methods (hand-crafted)
- Appearance-based methods (ML based visual features)

Automatic license plate recognition (ALPR) plays an important role in numerous real-life applications, such as road traffic monitoring, automatic toll collection, and traffic law enforcement. When the license plate is a unique identification for each vehicle, license plate recognition (LPR) or verification (LPV) has been widely used in industry for vehicles identification. It is fulfilled by a combination of many techniques, like object detection, image processing, and pattern recognition. However, due to high demands on image quality and due to lack of LPR-ready surveillance cameras, existing methods for LPR can be used only in restricted conditions. Some methods are using license plate verification (LPV) instead of LPR. In difference with LPR, in LPV task the objective is to check if two license plates are the same.

Vehicle re-identification methods based on vehicle appearance are using visual features extracted from detected vehicles for re-identification. This includes texture features (Histogram of Oriented Gradients, SIFT, and their variances), color features (color histograms), possibly vehicle dimensions and movement information.

Methods based on hand-crafted appearance features are described in Section 2.7.2. License plate recognition based methods can be found in Section 2.7.1. The summary of methods described in this section can be found in Table 2.4. Combination of both principles and another appearance-based machine learning methods are discussed in Section 2.7.3.

2.7.1 Vehicle Re-Identification by License Plate Recognition

Wen *et al.* [237] developed an algorithm for LPR applied to ITS by a novel shadow removal technique and character recognition algorithms. Main contributions of this paper are the shadow removal method, which is based on the improved Bernsen algorithm [13] combined

Table 2.4: A summary of vehicle re-identification methods and their orientation (hand-crafted appearance-based, LPR-based) with a brief description. **App.** – denotes appearance-based methods, **LPR** – denotes methods based on license plate recognition. Appearance methods based on ML models are omitted from the table.

	App.	LPR	Description	Comment
Arth *et al.*, 2007 [3]	✓	✗	PCA-SIFT + vocabulary tree	Limited to 2D bouding boxes and without color information
Wen *et al.*, 2011 [237]	✗	✓	Elastic mesh for character segmentation + SVM	Novel shadow removal algorithms
Feris *et al.*, 2012 [48]	✓	✗	Statistical features: date, time, color, dimensions, speed	Vehicles classified into different types and colors
Zapletal *et al.*, 2016 [257]	✓	✗	HOG & color histogram models + linear regression	Models obtained from unwrapped 3D bounding boxes
Liu *et al.*, 2016 [137]	✓	✗	Low-level: SIFT + BOW, Color Name + BOW, High-level: learned by GoogLeNet	Mix of low-level and high-level features for vehicle re-identification
Kluwak *et al.*, 2016 [107]	✗	✓	ALPR + license plate tracking + In-Track Clustering	License plate clustering ensures better recognition performance
Liu *et al.*, 2016 [138]	✓	✓	Appearance-based model, license plate verification (Siamese CNN), spatiotemporal properties	Three step filtering of vehicles in database

with the Gaussian filter and support vector machine (SVM) integration for character recognition. Characters' features are extracted from the elastic mesh and the entire character string is taken for further processing in SVM integration.

Kluwak *et al.* [107] presented a new approach to ALPR methods on toll gates using video object tracking and single image ALPR method to improve the recognition rate. The proposed method of multi-frame LPR consists of three parts. First, a single plate is recognized. Second, the recognized license plate is tracked through frames. The first two parts can be done by standard ALPR and tracking methods.The last part is called In-Track Clustering Correction and can be done by clustering of each recognized character into the set of license plate characters through multiple positions in track. This principle ensures better recognition performance in case of incorrect character recognition on some of the track positions. Any of ALPR systems can be used for vehicle re-identification.

A survey of other automatic license plate recognition methods is discussed by Du *et al.* [39].

2.7.2 Appearance-Based Methods

General approach for *Appearance-based methods* is to extract features from images and classify extracted features by a classifier. Formerly, different types of *hand-crafted* features were used.

Arth *et al.* [3] use PCA-SIFT features for vehicle data extraction and a vocabulary tree for data representation. However, their approach does not take into account color information of the vehicles and only 2D bounding box is used.

Zapletal and Herout [257] presented a method based on linear regression on two models. One model is created by HOG descriptors and the other one is based on color histograms. Both models are extracted from unwrapped 3D bounding boxes of detected vehicles [43].

Atypical approach was proposed by Feris *et al.* [48], where vehicles are classified into different categories by date, time, color, vehicles' dimensions and speed.

Liu X. *et al.* [137] proposed a fusion model of low-level features and high-level semantic attributes for vehicle re-identification. *Texture features* are represented by the SIFT descriptors and are encoded by the bag-of-words (BOW) due to its accuracy and efficiency in image retrieval [196]. *Color features* are extracted by Color Name model [227] and quantized by the BOW model. *High-level attributes* are learned by a deep convolutional neural network (CNN) the GoogLeNet [217], which is fine-tuned on CompCars dataset [249].

Further improvements were made in work [138], where appearance-based model and license plate verification by Siamese Neural Network is used for filtering the target vehicles from database. These vehicles are then re-ranked exploiting spatiotemporal properties.

Zhang *et al.* [272] introduced new triplet sampling method for network training based on pairwise images, instead of random sampling of triplets. Proposed pairwise triplet sampling ensures simultaneous constraints both on intra-class similarity and inter-class dissimilarity.

Shen *et al.* [190] used *visual-spatio-temporal states* of the vehicle images and compute similarity scores between pair of these states for generating queries. Then final decision on similarity is make using Siamese CNN and path-LSTM network. Visual features are extracted by ResNet network[76].

Wang *et al.* [235] suggested to use orientation invariant feature vector composed of one global feature vector and four orientation-based region feature vectors, extracted from 20 vehicle key points. This aligned appearance-based feature representation is learned and spatiotemporal relations between probe and gallery images is adopted as a regularization strategy.

Liu H. *et al.* [132] proposed to use Coupled Cluster Loss (CCL) instead of Triplet Loss to make training phase more stable and accelerate the convergence speed. CCL function is defined over multiple samples rather than three and distances are measured between samples and cluster center instead of any randomly selected anchor samples. Also special variant of VGG neural network [194] is used to extract vehicle model information and the instance differences.

Yan *et al.* [244] searches visually-similar vehicles by exploiting multi-grain ranking constraints. They propose to use Generalized Pairwise Ranking or Multi-Grain based List Ranking for similar vehicle retrieval on features extracted by VGG neural network and performs better then CCL.

2.7.3 Combination of Multiple Methods, Supervised and Unsupervised Learning

Liu *et al.* [138] proposed the PROVID system for progressive vehicle re-identification, where three different approaches are used. Firstly, the most vehicles from database are filtered out due to different color, texture, shape or vehicle type using appearance-based model. Secondly, remaining vehicles are filtered using the license plate similarities between query and source vehicles, which are calculated by the Siamese neural network. Thirdly, the spatiotemporal properties are exploited to re-rank the vehicles to improve the vehicle search process.

Supervised Vehicle Re-Identification Vehicle Re-Id aims to retrieve a specific vehicle from huge galleries captured by different cameras. To achieve this goal, more and more works [138, 102, 223] use powerful convolutional neural networks (CNNs) to learn discrimi-

native representations of vehicle images instead of handcrafted features. To further improve performance, several works [138, 185, 112, 255] employ widely-used deep metric learning methods to pull vehicle images of the same identity close and push different vehicles far away.

However, these methods still suffer from the viewpoint variation problem. To address this issue, some works [285, 175] studied discriminative part-level features for better performance. Several recent works [66, 73, 225, 233] in vehicle ReID had stated that specific parts such as windscreen, lights, and vehicle brand tend to have much discriminative information. Other works [282, 142] attempt to synthesize more multi-view vehicle images by the popular generative adversarial networks (GANs).

Yao *et al.* [252] use a graphic engine to generate synthetic data with various viewpoints. Zhou *et al.* [283] propose to extract view-invariant features by directly learning a viewpoint-aware network. The authors designed a viewpoint-aware attentive multi-view inference (VAMI) model that only requires visual information to solve multi-view vehicle ReID problems.

He *et al.* [73] proposed a simple yet efficient part-regularized discriminative feature-preserving method, which enhances the perceptive capability of subtle discrepancies, and reported promising improvement.

Besides, a few works [190, 221, 277] make use of prior spatiotemporal knowledge to narrow down the possible search space to filter out masses of hard negative samples. Although achieving great success, these methods are difficult to distinguish subtle differences between similar vehicles. To alleviate the problem, as mentioned earlier, many part-based methods are proposed to mine fine-grained information. Liu X. *et al.* [136] divide the feature map into several stripes and learn discriminative features from these stripes individually. Zhang *et al.* [271] develop a part-guided attention network that adaptively locates import parts and combines global-local features for Vehicle Re-Id.

Recently, a pure-transformer method called TransReID [77] has shown that transformer-based methods achieve better performance than CNN-based methods on some vehicle ReID benchmarks.

Unsupervised Domain Adaptation Object Re-Identification When applied to unseen scenarios, supervised object re-id methods always suffer from dramatic performance drops. In recent years, unsupervised domain adaptation (UDA) methods have drawn increasing attention since they can mitigate the generalization problem by transferring re-id knowledge from labeled data to unlabeled data. Usually, UDA object re-id methods adopt a two-step pipeline for training. Firstly, training a re-id model by supervised learning on the labeled dataset. Secondly, to fine-tune the re-id model, predicting pseudo-labels for the unlabeled data by clustering algorithms and taking pseudo-labels as supervision signals. Song *et al.* [204] provide more theoretical analyses based on this popular fashion. For example, Lin *et al.* [130] proposes to leverage the attribute to help the knowledge transfer and reduce the pseudo label noise. Ge *et al.* [53] extend this idea and develops a mutual mean-teaching framework to ensemble parameters of different training state.

To enhance generalization ability, Zhai *et al.* [260] further extend the mutual learning strategy to multiple heterogeneous networks. The recent state-of-the-art performance has been achieved in [274] by using the mean teacher strategy and uncertainty [278].

Similarly, to resist label noise, Zhang *et al.* [270] gradually select reliable samples for training, and Yang *et al.* [246] repose an asymmetric co-teaching framework where two distinct modules generate pseudo labels for each other.

Jiang *et al.* [97] focuses on developing a robust part-aware structure-based vehicle re-id system against the massive appearance changes due to the pose and illumination variants. Visual features are extracted from detected vehicle images and further enhanced by vehicle parts detection, which explicitly introduces the prior knowledge on the structure of the rigid object. The authors also use the basic UDA object re-id pipeline to learn robust features for the unlabeled scenario.

2.7.4 Image Feature Pooling in Temporal Domain

In this section, we provide an overview of existing methods for feature pooling (aggregation) in temporal domain. Such pooling is usually used in the context of person re-identification (with the exception of Yang *et al.* [247] who used it for video face recognition). The methods are often trained by using a Siamese network [158, 266, 245, 25, 243, 247] with contrastive loss and optionally identification loss as well.

McLaughlin *et al.* [158] propose an approach for temporal domain pooling based on Recurrent Neural Networks (RNN). The authors extract features using a CNN and use a recurrent layer to compute the features for the whole track. The used RNN has an output for each time step and these outputs are averaged to obtain the final feature vector for re-identification. The authors further propose to use optical flow as an additional input to the network. A similar approach was proposed by Zhang *et al.* [266] with the exception that their method uses bi-directional RNN to get better re-id results. Also, the method proposed by Yan *et al.* [245] is similar with the exception that the image level features are not trained and LBP and color features are used instead.

Chen *et al.* [25] also follow the work of McLaughlin *et al.* [158]. However, they propose to merge the features extracted by RNN together with CNN spatial features averaged over the time steps. The authors use three such networks for different body parts and fuse their output features (by a weighted sum). One network is for full person images, while the second one is for upper body part, and the last network is used on lower body parts.

Another approach based on the work by [158] is proposed by [243], who introduce significant modifications to the method. First, image level features are extracted by spatial pyramid pooling; thus, spatial information is preserved in the feature vector. These features are then fed into a recurrent layer (similar to [158]). Finally, the recurrent features are pooled by an Attentive Temporal Pooling layer proposed by the authors. The layer takes features for all time steps for both, the probe and gallery sequences, and generate a matrix with weights. These weights are then used for weighting of features from different time steps in the sequence (see the original text for details). Although the approach benefits from using both the sequences which are being compared, it also has a significant drawback: efficiency. The comparison of two sequences becomes very complex because the attention matrix is computed as a product of 3 matrices and more importantly, it needs to be computed for every pair of sequences.

Zhangh *et al.* [267] adds a feature pooling layer into the CNN architecture before the first fully connected layer. This layer aggregates key information from different views of the person's trajectory (different time steps) in a single feature vector. They also incorporate two different learning distance metrics – minimum distance and average of minimum distance for comparing the query track with tracks in the database.

Unlike the other authors, [247] focus on video face recognition. The authors propose an approach to temporal pooling based on weighting of feature vectors from different time steps. The weight for a feature vector is obtained as a dot product with a template, which

is computed by a fully connected layer. The weights are then normalized to a probability distribution by softmax function. The weights for different time steps scale the contributions of images in the sequence according to their discriminative value.

Similarly, [284] propose to use a temporal attention model and generate weights for feature vectors in the track. However, in contrast to [247], the weights are generated at each time step for all the feature vectors in the sequence. Then, at all time steps, all feature vectors (from the given sequence) are weighted by a set of (different) weights; thus, at each time step, differently weighted input feature vectors are produced. The weights at each time step are obtained by a RNN layer. Furthermore, similarly to [158], the weighted feature vectors are fed into another RNN layer with output at each time step and then averaged to obtain the final track representation. The authors also use spatial RNNs to further improve the re-identification results.

Generally, for temporal pooling, the authors use either recurrent neural networks [158, 266, 245, 25, 243], learned weighting of feature vectors [247], or a combination of these approaches [284]. In contrast to the described methods, the proposed method (Section 7.2 on page 64) produces a different weight for every element of the feature vectors. These methods might be beneficial for Multi-Target Multiple Camera (MTMC) tracking tasks, where averaging/mean pooling of extracted identification features is used nowadays [253, 131].

2.7.5 Object Re-Identification – Summary

Object re-identification in the past couple of years, mainly focusing on the re-identification of persons and vehicles, has become essential due to increasing security needs.

Methods in this field usually use *appearance-based* features. They could be handcrafted [3, 48, 257, 138] or extracted by machine-learning model [158, 266, 245, 25, 284, 274, 278, 130] for single image object re-identification. Some methods are focused to more distinctive parts of the object (e.g., body parts [27, 105, 120, 273, 113, 125, 26] or structured parts of vehicles [235, 66, 73, 225, 233, 271]). For a combination of multiple extracted visual features along the object's trajectory, pooling/aggregation methods in the temporal domain are used [158, 266, 245, 25, 243, 284, 247]. These methods have benefits even for Multi-Target Multiple Camera tracking tasks [253, 131]. Compared to these methods, our proposed solution for pooling visual features in the temporal domain (Chapter 7 on page 62) in a feature element-wise weighted manner and derives information from the vehicle's trajectory, which has the main benefit of vehicles' viewpoint changes and more distinctive parts of vehicles can be seen.

2.8 License Plate Recognition

The problem of automatic license plate recognition coincides also with license plate detection, which can be done in multiple ways in software [2, 56, 86, 121, 178], even in hardware [259] and GPUs [81]. However, the problematic of license plate detection in image is not part of this thesis and it is mentioned only for completeness of the chapter.

Previous works on ALPR are typically based on optical character recognition (OCR) techniques and they start the process by segmenting characters from the license plate. Formerly used algorithms can be divided into two groups: projection-based and connected component (CC) based techniques. The disadvantage of this type of techniques is their sensitivity to rotation and distortion of license plate. These methods can be reffered to as *segmentation-based* ALPR.

Formerly used algorithms for license plate character recognition are either based on template matching [178] or machine learning. Learning-based methods are more robust due to usage of more discriminant features for learning such as direction features [237] or image density [56, 276]. The commonly used learning techniques include SVM [237], probabilistic neural networks [2, 56], or artificial neural networks [276].

This section contains updated related work for Part IV on page 76 which was further elaborated from original paper.

Segmentation-based ALPR Rotation and distortion causes *segmentation-based* ALPR methods [67, 163, 176, 178, 237, 116, 155] to fail in most cases.

Masood *et al.* [155] proposed to first detect and segment out the characters and then recognize them by a convolutional neural network and they integrated their approach to *Sighthound Cloud API*. Even recent work from Laroca *et al.* [116] uses character segmentation combined with individual character recognition based on CNN. From this class of methods, only one group can be used for in-the-wild license plate recognition under limited conditions. These methods are based on connected component techniques (CC-based) and they can deal with rotation and distortion of the license plate [2, 21, 56, 276]. CC-based techniques label blocks of pixels from binarized LPs, depending on 4- or 8-neighborhood connectivity and they use these blocks to segment the license plate. Unfortunately, these methods cannot segment characters correctly if they are connected together or broken.

Another approach is to use OCR systems designed for reading multi-character text in the wild, which are used to recognize digits and characters in different types of applications (e.g. reading house numbers and other texts on facades), where character segmentation of the input data can be also difficult. Goodfellow *et al.* [62] proved that neural nets are capable of recognizing multi-character texts without segmenting the characters in unconstrained natural photographs, which was confirmed by Jaderberg *et al.* [94].

Segmenation-free LPR Finally, *segmentation-free* methods [86, 23, 95, 121, 208, 122, 115, 6, 5] have state-of-the-art results in license plate recognition in the past few years. In 2016, Li and Shen [121] developed the first segmentation-free license plate recognition method. Their method is based on CNN for feature extraction and bi-directional recurrent neural network with LSTM units and connectionist temporal classification (CTC) [64] for sequential data labeling. They achieved state-of-the-art results on AOLP dataset [86] and Caltech Cars 1999[1]. Similar approaches based on CNNs were also used by other authors [23,

[1] http://www.vision.caltech.edu/archive.html

95]. However, these methods are focused on standard, high-quality images of license plates. Our work focuses on low-quality license plate images (e.g., motion blurred, damaged).

Li *et al.* [122] improved his previous work in 2017 using a region proposal network for license plate detection, and the output of BRNNs is linearly transformed before CTC is involved.

For the LPR task, methods originally proposed for *scene text recognition* can also be used, as proved by Laroca *et al.* [115]. This includes models based on Connectionist Temporal Classification (CTC), Attention, or Vision Transformers.

SpaTial Attention Residue Network (STAR-Net) for recognizing scene texts was introduced by Liu W. *et al.* [135]. This model is equipped with a spatial attention mechanism that employs a spatial transformer to remove the distortions of texts in natural images. In this way, the feature extractor can focus on the rectified text region without further distraction caused by text distortions. Their feature extractor exploits residue convolutional blocks to extract more discriminative text features for scene text recognition tasks.

Baek *et al.* [6] developed a four-stage framework for scene text recognition. Each stage was derived from the STR models existing in 2019 – *(transformation, feature extraction, sequence modeling, prediction)* and allows the user to configure models to match his needs. For example, CTC or Attention mechanisms can be selected in the *prediction* stage. Laroca *et al.* [115] benefits from this framework to create a TRBA model (Thin-Plate Spline transformation, ResNet, BiLSTM, Attention) for the LPR task.

Two years later, the ViTSTR model was implemented by Atienza *et al.* [5]. ViTSTR is a simple single-stage model architecture built on a vision transformer (ViT). The authors stated that this model achieves competitive accuracy with fewer parameters and fewer computational resources than the TRBA model. In terms of ViTSTR model trade-offs, nearly all configurations are at or near the frontiers to maximize accuracy, speed, and computational efficiency simultaneously.

End-to-end LPR Benefiting from the rapid development of convolutional neural networks, the performance of vehicle License Plate Detection (LPD) and License Plate Recognition (LPR) has been largely improved. Nonetheless, most existing methods solve detection and recognition problems separately and focus on specific scenarios, hindering real-world application deployment.

In 2018, Gonçalves *et al.* [60] pointed out the problem of combining LPD with LPR methods as each task has its accuracy, and the error propagation between each task is detrimental to the final accuracy. They propose using two deep neural networks and combining LPD and LPR tasks. They confirm that segmentation-free techniques for LPR further reduce error propagation and improve results.

In 2021, Huang *et al.* [91], authors designed ALPRNet to detect and recognize mixed-style LPs. The ALPRNet consists of two one-stage object detectors to detect and classify license plates and characters simultaneously. After that, the assembly module outputs the recognized license plate text. License plates and characters are objects for each detector respectively, and directly outputs the bounding boxes and corresponding labels of the license plates and characters. This technique allows users to avoid recurrent neural network branches for LPR. The authors also claim that this method is independent of the layout of license plate characters and works well for different types of license plates.

A year later, Qin and Liu [177] proposed a light-weight end-to-end model based on the anchor-free method for LPR, which can work in real-time. This model predicts the bounding

box and four corners of the license plate, corrects the feature map after ROI Align, and feeds it into the recognition CNN. The recognition task is treated as sequence labeling problems, which are directly solved by Connectionist Temporal Classification (CTC).

Fan *et al.* [45] incorporates a robust license plate detection network (CA-CenterNet) together with a segmentation-free network (CNNG) in 2022. CA-CenterNet detects the center of the license plate together with four regressed vectors pointing to license plate corners which are further used for rectification of the license plate.

The most recent works [98, 104] focus on speed and application of ALPR methods in real-world scenarios, which differ in requirements on distance, illumination, angle, and other conditions to work reliably. Jiang *et al.* [98] aims to ALPR in the unconstrained environment using a combination of proposed LPDNet and CRNet for LPD and LPR tasks, respectively. On the other hand, Ke *et al.* [104] proposed a combination of YOLOv3-tiny detector and light-weight MRNet based on multi-scale features to achieve high accuracy, high detection speed (751 FPS) and fast license plate recognition (2.9 ms).

License Plate Recognition – Summary The field of License Plate Recognition has evolved to encompass a range of approaches, from traditional segmentation-based methods to modern segmentation-free and end-to-end techniques. These advancements have enabled improved accuracy, speed, and adaptability in various real-world scenarios, while also addressing challenges like rotation, distortion, and error propagation. The research landscape continues to innovate to meet the demands of practical LPR applications.

Part II

Datasets for Traffic Analysis

The field of traffic analysis has witnessed remarkable advancements in recent years, largely due to the availability of large-scale datasets that facilitate the development and evaluation of novel algorithms and methodologies. Although significant progress has been made in several traffic analysis domains, including license plate recognition, vehicle re-identification, vehicle fine-grained recognition, and monocular vehicle speed measurement, the absence of publicly available datasets has significantly hampered further developments in these domains.

Part II of this thesis focuses on addressing this crucial gap by presenting a collection of diverse datasets, each catering to a specific aspect of traffic analysis. These datasets were meticulously curated, annotated, and made publicly available for researchers and practitioners in the traffic analysis community. The development and release of these datasets mark a significant contribution to the field, providing new opportunities for analysis, benchmarking, and performance testing of algorithms.

The **Re-Id**, **HDR** and **CamCar6k** datasets (Chapter 3 on the next page) constitute a comprehensive collection of annotated license plate images captured under various environmental conditions, including different lighting conditions, weather conditions, and camera angles. This dataset plays a crucial role in advancing the accuracy and robustness of license plate recognition systems. Researchers can leverage this dataset to develop and evaluate algorithms that address challenges such as low lighting, occlusions, and variations in license plate styles, ultimately contributing to the development of more reliable and efficient license plate recognition systems.

The difficulty of re-identifying vehicles across many non-overlapping cameras is addressed by the **CarsReId74k** dataset (Chapter 4 on page 39), in any case. This dataset enables the development and testing of re-identification algorithms for identification of vehicles throughout a network of surveillance cameras by collecting vehicle images from various angles and in various situations. Such advancements in vehicle re-identification techniques hold tremendous potential for enhancing traffic monitoring, security, and surveillance systems.

To further enrich the understanding of vehicle characteristics, the **BoxCars116k** dataset (Chapter 5 on page 43) offers a comprehensive collection of images showcasing different vehicle makes, models, types and model years. This dataset is an invaluable resource for vehicle fine-grained recognition tasks, allowing researchers to create algorithms that can precisely identify vehicle characteristics. The existence of such a dataset promotes improvements in the identification of vehicle attributes, which enhances vehicle profiling and identification capabilities in traffic analysis systems.

As a crucial component of traffic analysis, monocular vehicle speed measurement is the final emphasis of the **BrnoCompSpeed** dataset (Chapter 6 on page 54). This dataset provides ground truth annotations for vehicle speed estimation tasks by capturing vehicle images at known distances. This dataset's accessibility enables researchers to create and test algorithms that precisely gauge vehicle speed based solely on monocular vision, which has profound implications for traffic management, enforcement, and safety.

The availability of these datasets to the traffic analysis community satisfies a critical need and offers various advantages. Firstly, these datasets can be used by academic researchers and industry professionals to evaluate and compare the effectiveness of their algorithms to cutting-edge techniques, encouraging healthy competition and advancing the field. Additionally, these datasets can be used as a foundation for training and fine-tuning machine learning models, enabling the development of more accurate and robust traffic analysis systems.

Chapter 3

Datasets for License Plate Recognition

This chapter discusses the creation of datasets for recognizing low-quality vehicle license plates. The characteristics of the data correspond to vehicle re-identification in traffic surveillance and parking law enforcement using controlling vehicles equipped with cameras.

The datasets of the first type were introduced in paper *Špaňhel* et al., *Holistic Recognition of Low Quality License Plates by CNN using Track Annotated Data, AVSS 2017* [208]. This paper focuses on an alternative approach to license plate recognition in a holistic, segmentation-free way. The novel user-annotated dataset was collected for this work and made publicly available for non-commercial use[1]. Images in the dataset are annotated by whole license plate tracks (sequences of observations of a single vehicle). Therefore, it provides accurate ground truth labels even for naturally blurred, partially occluded, and hardly readable license plate images, allowing a robust license plate recognizer to be trained. An example of such a license plate track can be seen in Figure 3.1.

The acquisition process for this dataset is described in Section 3.1 on the next page, and it is hereafter referred to as **ReId**. Additionally, a hand-crafted **HDR** dataset with license plates captured by a high-dynamic range camera was also introduced.

The second type dataset, referred to in the following text as **CamCar6k**, was published in paper *Špaňhel* et al., *Geometric Alignment by Deep Learning for Recognition of Challenging License Plates, ITSC 2018* [207].

As is generally known, Automatic License Plate Recognition (ALPR) is the backbone for many applications in traffic surveillance and intelligent transportation systems (automatic parking systems, security surveillance systems, toll gates, etc.). In many such applications, the cameras are fixed and positioned so that the license plates share a typical size (image resolution) and orientation, and they are not skewed. In such scenarios, the existing recognizers of license plates achieve almost perfect results. However, mobile monitoring platforms are used more frequently for parking enforcement and other applications. Also, the availability and properties of PTZ (pan-tilt-zoom) cameras offer much less restricted scenarios. Existing solutions of automatic license plate recognition (an overview is available in Section 2.8 on page 29) are not designed for these unconstrained cases and tend to achieve poor results. The first mentioned ALPR works proved that holistic (i.e., refraining from segmenting the characters) recognition outperforms other available recognition methods. However, that work does not deal with challenging license plates captured from

[1] https://medusa.fit.vutbr.cz/traffic/

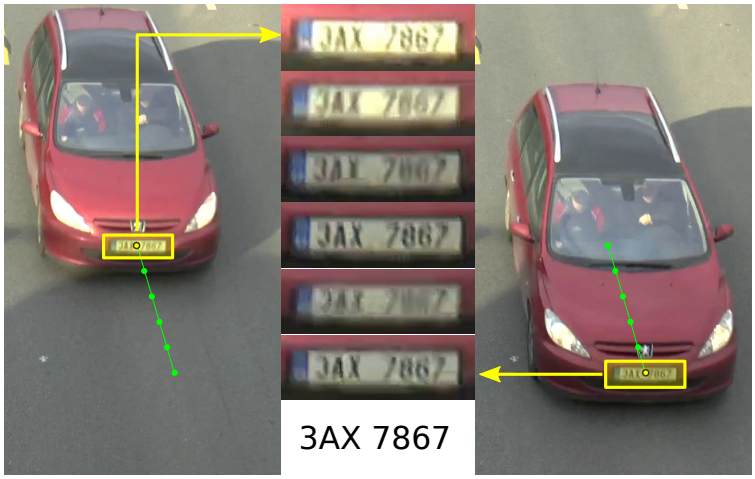


Figure 3.1: During data collection, license plates are detected and tracked. All images were manualy labeled in a per-track manner so even when text in some images cannot be recognized correctly, the ground truth can be assigned as long as the track contains at least one readable image.

different viewpoints. In this work, we propose a solution how to overcome that limitation. We designed a new convolutional neural network (CNN) to predict four corner points of the license plate in the unaligned image. These points define the transformation which rectifies the image for subsequent processing by the holistic license plate recognizer. Therefore, a novel dataset had to be collected.

The dataset sample, specifics, and acquisition process are described in Section 3.2 on the next page.

3.1 Low-quality License Plate Recognition Dataset

Traffic surveillance systems in global target mainly images with sufficient quality. However, it is important to be able to deal with low-quality images as the conditions might not be perfect all the time. To deal with low-quality license plate images, we collected a new dataset called **ReId**. Multiple videos from various locations and under different conditions were recorded for this dataset. The data was captured by Full-HD video cameras placed on bridges above the highway, simulating surveillance cameras on toll gates. We collected 9.5 hours of license plate recordings from 8 distinct locations on various daytimes. Examples of the recorded data are shown in Figure 3.2.

License Plates Detection For each image in the input sequence, we detect license plates using a boosted soft cascade classifier with LBP image features trained by a constant soft cascade algorithm [38]. Each detected license plate is tracked with Kalman filter. Our detector implementation takes under 5 ms per Full-HD frame on GTX 1080 GPU. We trained the detector on the dataset of 5,000 frontal license plate samples dimensioned 55×22 pixels and 4,000 negative samples using 4 rounds of hard negative mining [38], loading additional 4,000 negatives in each round. The final detector is composed from 512 LBP features serving as the weak classifiers.

Figure 3.2: Samples of recordings for different camera locations.

Figure 3.3: Examples of images from the ReId dataset. Each column shows license plates with a different text length.

Ground Truth Assignment Finally, the collected license plates were labeled by their ground truth texts. Ground truth label for each license plate was assigned by users using a newly created web-based annotation tool. License plates were recognized by the user from the whole track (a sequence of the same license plate in the video) and each license plate in the track was labeled at once after that. Annotating the whole track has one major advantage (putting the speed-up in annotation aside): it is possible to precisely annotate even almost un-readable (e.g. blurred, small, partially covered, ...) license plates if only one of the instances of the license plate is clear enough to be human-readable. The length of the license plate texts varies from 5 to 8 characters. Example images from the dataset are shown in Figure 3.3.

The dataset was split to training and test parts per videos, therefore training and test samples come from mutually exclusive sets of videos. The training part contains 7,393 tracks (105,924 images) and the test part contains 6,967 tracks (76,412 images). In total, there is 14,360 tracks, containing 182,336 images of license plates.

As Laroca *et al.* [115] pointed out, there could be overlap in a few identities (LP texts) as dataset splits were created based distinctiveness of individual recording sessions, not based on identities.

3.2 ALPR Dataset for Parking Law Enforcement

Publicly available datasets of license plate images are a scarce commodity globally. Two datasets designed for license plate recognition – **ReId** dataset (182,336 images), and **HDR** dataset (657 images) were described in previous section. Unfortunately, none of these

Figure 3.4: **Left:** Randomly selected samples from the HDR dataset. **Center:** random original blurred samples from the dataset published by Svoboda *et al.* [215]. **Right:** Deblurred images by the method proposed by Svoboda *et al.* [215].

Figure 3.5: Samples of license plate with annotated keypoints used for training aligner and recognizer. **Left:** Samples from *CamCar6k* dataset. **Right:** Samples from synthetic data (Sec. 3.2.2).

datasets contains ground truth annotations of license plate corners' positions. They are therefore not suitable neither for training or evaluation of license plate alignment.

In 2018, Laroca *et al.* [116] published the *UFPR-ALPR* dataset with 4,500 images FullHD images with annotated license plates. However, the data was captured from a camera fixed at a vehicle's windshield, thus with minimal distortions of the LPs. The same is true for the dataset created by Gonçalves *et al.* [61] which is even smaller (only 2,000 images).

3.2.1 CamCar6k – Public Dataset of License Plates in the Wild

We recorded 7.5 hours of video of license plates from different viewpoints taken by four cameras mounted on a vehicle passing among vehicles parked on streets and/or parking lots. The recordings cover almost all possible styles of parking (parallel, angle, perpendicular) both outside (streets, outdoor parking lots) and inside (parking garages).

In each frame of the video, license plates were detected by a Boosted Soft Cascade detector [38]. In order to deal with the range of rotations and perspective projections, we used three detectors, each tuned to a different range of deformations. The outputs of these detectors were merged and formed the final set of detections. Each detection was processed with a preliminary version of the recognizer in order to filter out false detections.

Original image frames with at least one detection were stored and 5,000 frames with detected license plates were chosen randomly and further processed by users. Their task was to annotate the *inner corners* of each license plate and transcribe the ground truth text for each image. Thus created dataset **CamCar6k** contains 6,064 images of license plates.

The locations of annotated keypoints allow us to generate more training samples by rotation, translation, shear and zoom. The distribution of the rotations of the original license plates in the collected data is shown in Figure 3.6. It can be seen that majority of license plates are rotated in an interval of $\pm 20°$.

The **CamCar6k** dataset was divided into a *training* and a *test* split, containing 2,750 and 3,314 images of license plates, respectively. In our experiments (Section 9.3 on page 87),

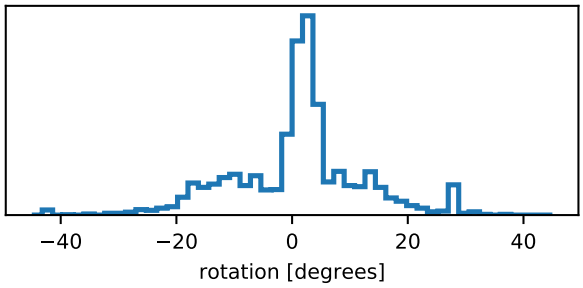


Figure 3.6: Distribution of rotation of license plates in the newly collected **CamCar6k** dataset. The majority of LPs are almost horizontal, but the dataset also contains a good coverage of angles between ±20°.

we use **ReId**, **HDR** and **CamCar6k** datasets mixed together for learning the LP recognizer. The **CamCar6k** dataset was made publicly available for non-commercial purposes[2].

3.2.2 Dataset of Synthetic Images of Cars with License Plates

With image transformations, the texts of the annotated license plates remain unchanged. It is impossible to collect all valid variants of license plates, or combinations of all the allowed characters. To avoid misclassification of license plates with previously unseen characters and their combinations, we developed a tool for generating synthetic data. Given real images with annotated corner points of the license plates, a template of the country-specific license plate layout, and an appropriate font for characters, the tool is able to generate any requested number of license plates with different text and rotation/distortion of the source image. We generated 100 k synthetic license plates of multiple countries (denoted by *synth* in the following text), which are used only in the training phase (we consider it appropriate to only evaluate on real-world data). Sample of synthetic license plate can be found in Figure 3.5 – **right** image.

[2] https://medusa.fit.vutbr.cz/traffic

Chapter 4

Dataset for Vehicle Re-identification

The availability of datasets for vehicle re-identification was limited for a longer time.

There are datasets of vehicles [110, 249, 202], which are created for fine-grained recognition with annotations on several attributes such as type, make, and color. However, the identities of the vehicles in the datasets are unknown; thus, the datasets are not directly applicable for vehicle re-identification, especially for evaluation.

Regarding genuine vehicle re-identification, Liu X. *et al.* [138] constructed a relatively small VeRi-776 dataset containing 50,000 images of 776 vehicles. Liu H. *et al.* [132] collected a VehicleID dataset containing 26,267 vehicles in 220k images taken from a frontal/rear viewpoint above the road. In 2017, Yan *et al.* [244] published two datasets, VD1 and VD2, for vehicle re-identification and fine-grained classification with over 220k of vehicles in total, with make, model, and year annotation. However, both datasets are limited to frontal viewpoints only.

Following the publication of our *CarsReId74k dataset*, Tang *et al.* published CityFlow dataset for *NVIDIA AI City Challenge* (AIC). From a single data source, authors created datasets for multiple tasks – dataset for multi-target multi-camera (MTMC) for traffic surveillance, made by multiple video sequences, and a vehicle re-identification dataset consisting of cropped frames from the MTMC dataset. The CityFlow-ReID dataset is used as a single probe for gallery search and does not contains annotations of image sequences (vehicle trajectories). Additionally, images are tightly cropped around cars and may not be suitable for every re-identification scenario. However, this dataset was not available during the publishing process of *CarsReId74k dataset*. Thus it is not included in the following text.

This chapter provides detailed information about the **CarsReId74k** dataset. This dataset was proposed in *Computer Vision and Image Understanding* journal in article Špaňhel *et al.*, *Learning Feature Aggregation in Temporal Domain for Re-Identification*, CVIU 2020 [209]. For more details about our vehicle re-identification approach methodology and results of the experiments, please follow the text in Chapter 7 on page 62.

4.1 CarsReId74k – Novel Vehicle Re-Identification Dataset

We focus on vehicle re-identification and we want to differentiate even vehicles with the same fine-grained type but different identities (different license plates). Therefore, we cannot use fine-grained vehicle recognition datasets [198, 202, 249, 110] for this task. As other existing

Table 4.1: The comparison of various vehicle re-id datasets.
* – Tracks are not guaranteed for each unique vehicle.
† – Unique vehicles from each dataset part can overlap. The total number of unique vehicles is probably lower.

	CarsReId74k	VehicleID	VeRi-776	PKU-VDs
# unique vehicles	17,681	26,267	776	†**221,519**
# tracks	**73,976**	—	6,822	N/A
# images	**3,242,713**	221,763	51,035	1,354,876
viewpoints	**various**	front/rear	**various**	front
multiple images in track	**yes**	no	***yes**	**yes**

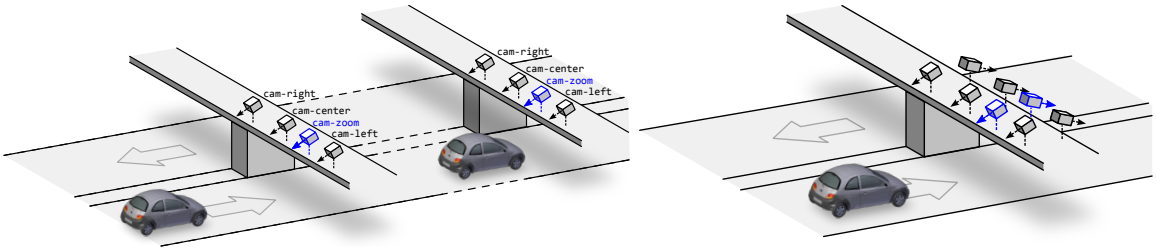


Figure 4.1: Recording setup for acquisition of novel CarsReId74k dataset. We simultaneously recorded data on two bridges by multiple cameras. One camera on each bridge was zoomed in so that it is possible to automatically recognize license plates and use them for the construction of the ground truth labeling (left image). For part of dataset vehicles were captured from single bridge on both sides, which yields to capture observed vehicles from frontal and rear viewpoints (right image).

vehicle re-identification datasets VeRi-776 [138], VehicleID [132] and PKU-VDs [244] are either small (VeRi-776) or limited to frontal/rear viewpoints (VehicleID, PKU-VDs). We collected a novel dataset **CarsReId74k** which does not have these limitations. The data were collected by 66 cameras from various angles and the dataset contains almost 74,000 of vehicle tracks with precise identity annotation (acquired from license plates). More detailed comparison of different available vehicle re-identification datasets can be found in Table 4.1.

4.1.1 Dataset Acquisition

The dataset was collected in multiple sessions. In each session, we placed four cameras on a bridge overlooking a freeway and four cameras on another bridge in vehicles' traveling direction. Figure 4.1 illustrates the recording setup and Figure 4.2 shows example frames from one such session. The videos were recorded for $\sim$ 1 hour and synchronized. One of the cameras was zoomed in enough to be able to read the license plates of all the passing vehicles (Figure 4.2 left). The other three cameras were placed so that they observed the road from left, center, and right position.

We used the zoomed-in videos to identify the passing vehicles. We detected the license plates by an ACF detector [38], tracked them, recognized by a recent method [208] and manually verified in order to eliminate any recognition errors. We also assigned a lane to each license plate track for easier matching. On all the other videos (left, center, right),

Figure 4.2: Frames from all cameras in one session. The license plates acquired from the *zoom* camera (left) were used for ground truth re-identification (silver car). Each row shows frames from one location within the session.

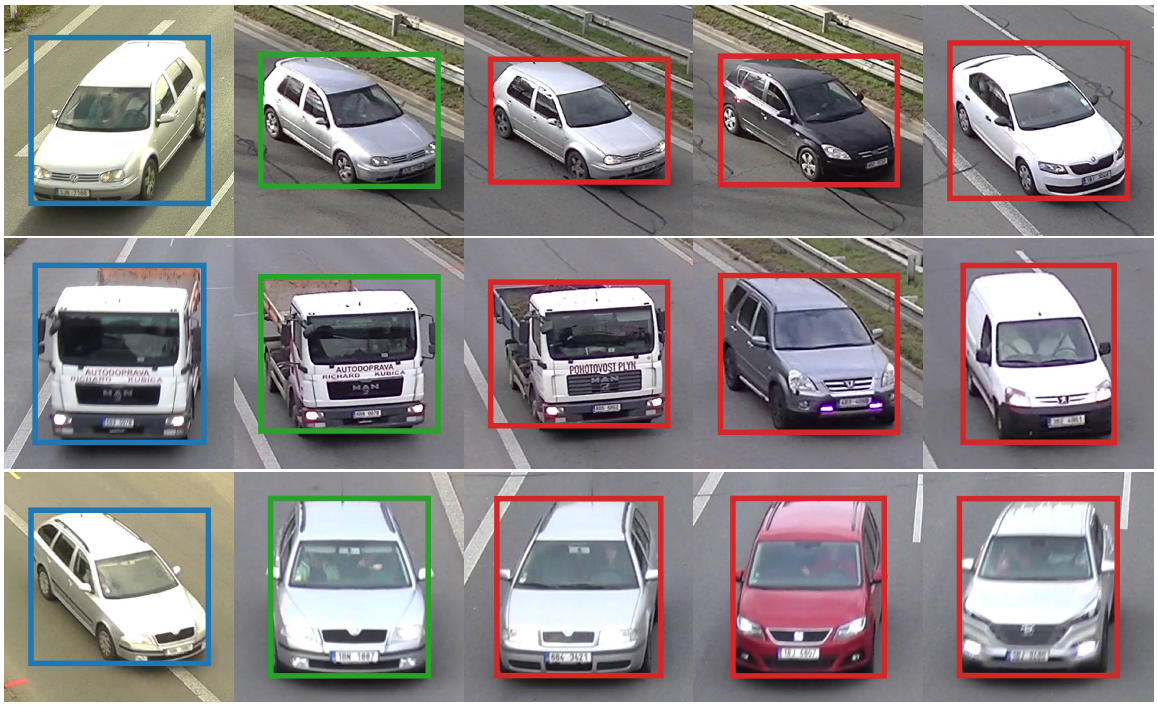


Figure 4.3: Examples of queries, positive, and negative samples. The negatives are sorted by difficulty from left to right (hard to easy) based on distances obtained from our re-identification feature vectors. It should be noted that the hardest negative sample has usually subtle differences (e.g. missing a small spoiler in the first row).

we detected and tracked the vehicles. We also constructed 3D bounding boxes [43] around the vehicles as [198] showed that the 3D bounding boxes were beneficial for fine-grained recognition. We also assigned the lane for each of these vehicles [43]. Finally, we matched the vehicles from the zoomed-in cameras (with known identities) to vehicles from the other cameras. We omitted all the vehicles which were not matched. It should be noted that the vehicles in the dataset from non-zoomed-in cameras have almost unreadable license plates; therefore, the dataset is suitable for **appearance-based** vehicle re-identification, preserving the anonymity of the vehicles.

Table 4.2: CarsReId74k dataset statistics. *The total number of unique vehicles is lower than the sum of unique vehicles from training, test and validation set because a small number of vehicles appear in all sets (same car present at two or more recording sessions by accident). †Number of negative pairs = mean number of negative pairs per positive pair.

	training	test	validation	total
# cameras	30	30	6	66
# unique vehicles*	7,658	9,678	1,100	17,681
# tracks	32,163	36,535	5,278	73,976
# images	1,469,494	1,467,680	305,539	3,242,713
# positive pairs	125,086	129,774	22,376	277,236
# negative pairs†	1,149	1,459	881	1,283

4.1.2 Dataset Statistics

The dataset was recorded in 11 sessions at different locations. We divided the dataset into the training, the testing and the validation part by sessions (five sessions for training, five sessions for testing and one validation). The total dataset statistics can be found in Table 4.2. The Table 4.1 on page 40 shows that our dataset is significantly larger than VeRi-776 [138] dataset with only 776 unique vehicles. And compared to VehicleID, VD1 and VD2 datasets [132, 244], our dataset is not limited to frontal/rear viewpoints. Compared to VehicleID dataset, CarsReId74k dataset has fewer unique vehicles (17,681 vs. 26,267), however far more image (3,242,713 vs. 221,763) as vehicles are seen from more viewpoints.

4.1.3 Proposed Evaluation Protocol

For each part (training, testing and validation), we collected all the pairs of tracks with the same vehicle identity (marked as *query*, *positive*). The query and positive tracks are always from different videos; however, they can come from the same session and location (e.g. left – right), from the same session and different location, or (in rare cases) also from different sessions within the training (or testing) set. This yields a significant number of positive pairs (277,236 in total). As the negative pairs, we use all other vehicle tracks in the same video as the positive track with the exception of vehicle tracks with the same identity as the positive track (a vehicle could be observed multiple times in one video). This yields a mean number of 1,283 negative vehicle tracks per positive pair. See Figure 4.3 for examples of positive and negative pairs.

Following other papers [132, 244, 138, 82, 234] on re-identification, we use **mAP** and **hit at rank** as the metrics for evaluation on the dataset. We encourage others to report hit rates at ranks 1, 5, 10, and 20 together with Cumulative Matching Curve for ranks 1 to 20.

Chapter 5

BoxCars116k – Dataset for Fine-grained Vehicle Recognition

There is a large number of datasets of vehicles (e.g., [184, 1, 169, 44, 240, 18, 166, 118, 59, 186, 55, 168, 157]) which are usable mainly for vehicle detection, pose estimation, and other tasks. However, these datasets do not contain annotations of the precise vehicles' make and model.

Regarding the fine-grained recognition datasets, there are some [210, 110, 129, 125] which are relatively small in number of samples or classes.

Lin *et al.* [129] published FG3DCar dataset (300 images, 30 classes), Stark *et al.* [210] made another dataset containing 1,904 vehicles from 14 classes. Krause *et al.* [110] published two datasets. One of them called *Car Types*, contains 16k images and 196 classes. The other one, *BMW 10*, comprises ten models of BMW vehicles and 500 images. Finally, Liao *et al.* [125] created a dataset of 1,482 vehicles from 8 classes. All these datasets are relatively small. Therefore, they are impractical for the training of CNN and deployment of real-world traffic surveillance applications.

Yang *et al.* [249] published a large dataset *CompCars*. The dataset consists of a web-nature part, made of 136k of vehicles from 1 600 classes taken from different viewpoints. It also contains a surveillance-nature part with 50k frontal images of vehicles taken from surveillance cameras. Liu *et al.* [137] published dataset *VeRi-776* for the vehicle re-identification task. The dataset contains over 50k images of 776 vehicles captured by 20 cameras covering a 1.0 km^2 area in 24 hours. Each vehicle is captured by $2 \sim 18$ cameras under different viewpoints, illuminations, resolutions, and occlusions. The dataset also provides additional attributes such as bounding boxes, vehicle types, and colors.

BoxCars116k dataset described in the following lines focuses on fine-grained vehicle recognition task under an unconstrained environment, and it is an expansion of the original *BoxCars21k* dataset [198], which was published in 2016. Images were captured from various viewpoints in wild scenarios, which makes this dataset very challenging.

The following text in this chapter contains detailed information about our contribution to vehicle fine-grained recognition and acquisition of *BoxCars116k* dataset. This work was published in *Transactions on Intelligent Transportation Systems* journal in article Sochor *et al.*, *BoxCars: Improving fine-grained recognition of vehicles using 3-d bounding boxes in traffic surveillance*, T-ITS 2018 [202]. Related work for this publication can be found in Section 2.4 on page 19. For evaluations of our experiments, please see Section *V. Experiments* in the original paper.

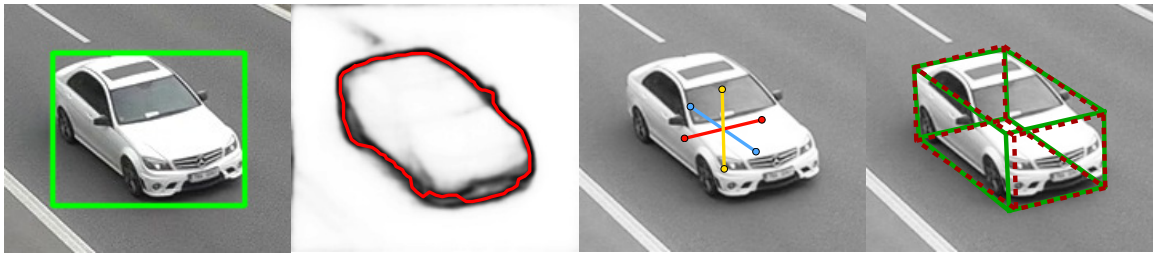


Figure 5.1: Example of automatically obtained 3D bounding box used for fine-grained vehicle classification. **Top left:** vehicle with 2D bounding box annotation, **top right:** estimated contour, **bottom left:** estimated directions to vanishing points, **bottom right:** 3D bounding box automatically obtained from surveillance video (green) and our estimated 3D bounding box (red).

Abstract In this paper, we focus on fine-grained recognition of vehicles mainly in traffic surveillance applications. We propose an approach that is orthogonal to recent advancements in fine-grained recognition (automatic part discovery, bilinear pooling). Also, in contrast to other methods focused on fine-grained recognition of vehicles, we do not limit ourselves to a frontal/rear viewpoint, but allow the vehicles to be seen from any viewpoint. Our approach is based on 3D bounding boxes built around the vehicles. The bounding box can be automatically constructed from traffic surveillance data. For scenarios where it is not possible to use precise construction, we propose a method for an estimation of the 3D bounding box. The 3D bounding box is used to normalize the image viewpoint by "unpacking" the image into a plane. We also propose to randomly alter the color of the image and add a rectangle with random noise to a random position in the image during the training of Convolutional Neural Networks. We have collected a large fine-grained vehicle dataset BoxCars116k, with 116k images of vehicles from various viewpoints taken by numerous surveillance cameras. We performed a number of experiments which show that our proposed method significantly improves CNN classification accuracy (the accuracy is increased by up to 12 percentage points and the error is reduced by up to 50 % compared to CNNs without the proposed modifications). We also show that our method outperforms state-of-the-art methods for fine-grained recognition.

5.1 Fine-grained Vehicle Recognition

Fine-grained recognition of vehicles is interesting, both from the application point of view (surveillance, data retrieval, etc.) and from the point of view of general fine-grained recognition research applicable in other fields. For example, Gebru *et al.* [54] proposed an estimation of demographic statistics based on fine-grained recognition of vehicles. In this article, we are presenting methodology which considerably increases the performance of multiple state-of-the-art CNN architectures in the task of fine-grained vehicle recognition. We target the traffic surveillance context, namely images of vehicles taken from an **arbitrary viewpoint** – we do not limit ourselves to frontal/rear viewpoints. As the images are obtained from surveillance cameras, they have challenging properties – they are often small and taken from very general viewpoints (high elevation). We also construct the training and testing sets from images from different cameras as it is common for surveillance appli-

cations that it is not known a priori under which viewpoint the camera will be observing the road.

Methods focused on the fine-grained recognition of vehicles usually have some limitations – they can be limited to frontal/rear viewpoints or use 3D CAD models of all the vehicles. Both these limitations are rather impractical for large scale deployment. There are also methods for fine-grained recognition in general which were applied on vehicles. The methods recently follow several main directions – automatic discovery of parts [109, 193], bilinear pooling [128, 51], or exploiting structure of fine-grained labels [242, 280]. Our method is not limited to any particular viewpoint and it does not require 3D models of vehicles at all.

We propose an orthogonal approach to these methods and use CNNs with a modified input to achieve better image normalization and data augmentation (therefore, our approach can be combined with other methods). We use 3D bounding boxes around vehicles to normalize vehicle image (see Figure 5.4 for examples). This work is based on our previous conference paper [198]; it pushes the performance further and we mainly propose a new method on how to build the 3D bounding box without any prior knowledge (see Figure 5.1). Our input modifications are able to significantly increase the classification accuracy (up to **12 percentage points**, classification error is reduced by up to **50 %**).

The key contributions of the paper are:

- Complex and thorough evaluation of our previous method [198].
- Our novel data augmentation techniques further improve the results of the fine-grained recognition of vehicles relative both to our previous method and other state-of-the-art methods (Section 5.2.3).
- We remove the requirement of the previous method [198] to know the 3D bounding box by estimating the bounding box both at training and test time (Section 5.2.4).
- We collected more samples to the BoxCars dataset, increasing the dataset size almost twice (Section 5.3).

We will make the collected dataset and source codes for the proposed algorithm publicly available[1] for future reference and comparison.

5.2 Proposed Methodology for Fine-Grained Recognition of Vehicles

In agreement with recent progress in the Convolutional Neural Networks [220, 111, 22], we use CNN for both classification and verification (determining whether a pair of vehicles has the same type). However, we propose to use several data normalization and augmentation techniques to significantly boost the classification performance (up to 50 % error reduction compared to base net). We utilize information about 3D bounding boxes obtained from traffic surveillance camera [43]. Finally, in order to increase the applicability of our method to scenarios where the 3D bounding box is not known, we propose an algorithm for bounding box estimation both at training and test time.

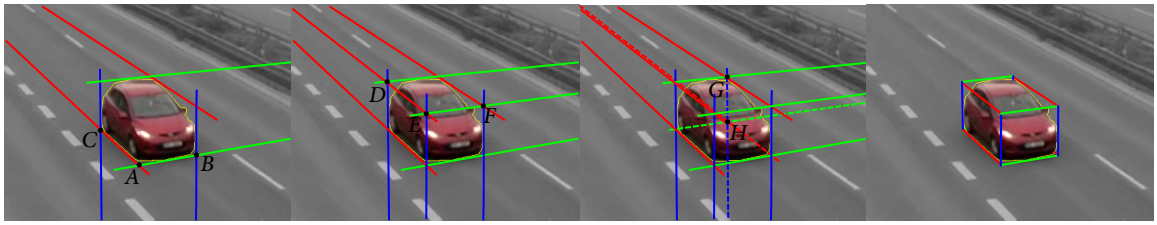


Figure 5.2: 3D bounding box construction process. Each set of lines with the same color intersects in one vanishing point. See the original paper for full details [43]. The image was adopted from the paper with the authors' permission.

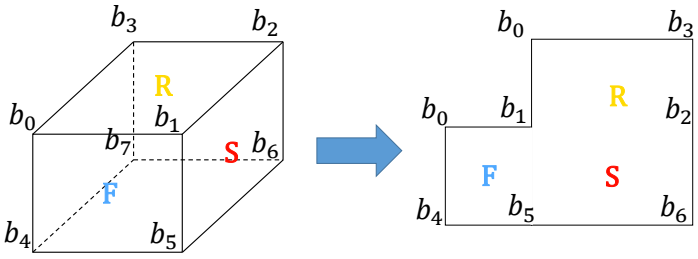


Figure 5.3: 3D bounding box and its unpacked version.

5.2.1 Image Normalization by Unpacking the 3D Bounding Box

We based our work on 3D bounding boxes proposed by [43] (Fig. 5.4) which can be automatically obtained for each vehicle seen by a surveillance camera (see Figure 5.2 for schematic 3D bounding box construction process or the original paper [43] for further details). These boxes allow us to identify the side, roof, and front (or rear) side of vehicles in addition to other information about the vehicles. We use these localized segments to normalize the image of the observed vehicles (considerably boosting the recognition performance).

The normalization is done by unpacking the image into a plane. The plane contains rectified versions of the front/rear (**F**), side (**S**), and roof (**R**). These parts are adjacent to each other (Fig. 5.3) and they are organized into the final matrix **U**:

$$\mathbf{U} = \begin{pmatrix} \mathbf{0} & \mathbf{R} \\ \mathbf{F} & \mathbf{S} \end{pmatrix} \tag{5.1}$$

The unpacking itself is done by obtaining homography between points b_i (Fig. 5.3) and perspective warping parts of the original image. The left top submatrix is filled with zeros. This unpacked version of the vehicle is used instead of the original image to feed the net. The unpacking is beneficial as it localizes parts of the vehicles, normalizes their position in the image and it does all that without the necessity of using DPM or other algorithms for part localization. Later in the text, we will refer to this normalization method as **Unpack**.

5.2.2 Extended Input to the Neural Nets

It it possible to infer additional information about the vehicle from the 3D bounding box and we found out that these data slightly improve the classification and verification performance. One piece of this auxiliary information is the encoded viewpoint (direction from which the

[1] https://medusa.fit.vutbr.cz/traffic

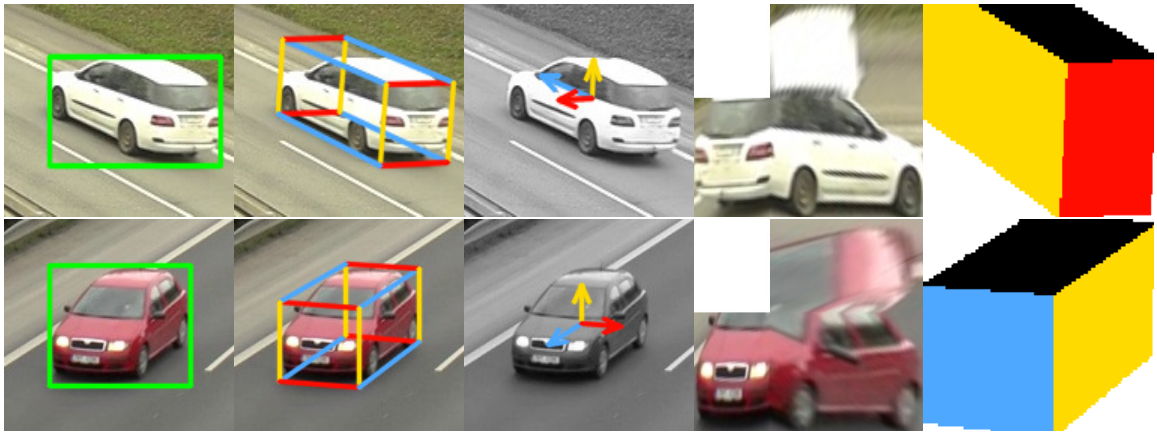


Figure 5.4: Examples of data normalization and auxiliary data fed to nets. **Left to right:** vehicle with 2D bounding box, computed 3D bounding box, vectors encoding viewpoints on the vehicle (**View**), unpacked image of the vehicle (**Unpack**), and rasterized 3D bounding box fed to the net (**Rast**).

Figure 5.5: Examples of proposed data augmentation techniques. Left most image contains the original cropped image of the vehicle and other images contains augmented versions of the image (**Top** – **Color**, **Bottom** – **ImageDrop**).

vehicle is observed). We also add a rasterized 3D bounding box as an additional input to the CNNs. Compared to our previously proposed auxiliary data fed to the net [198], we handle frontal and rear vehicle sides differently.

View. The viewpoint is extracted from the orientation of the 3D bounding box – Fig. 5.4. We encode the viewpoint as three 2D vectors v_i, where $i \in \{f, s, r\}$ (*front/rear*, *side*, *roof*) and pass them to the net. Vectors v_i are connecting the center of the bounding box with the centers of the box's faces. Therefore, it can be computed as $v_i = \overrightarrow{C_c C_i}$. Point C_c is the center of the bounding box and it can be obtained as the intersection of diagonals $\overleftrightarrow{b_2 b_4}$ and $\overleftrightarrow{b_5 b_3}$. Points C_i for $i \in \{f, s, r\}$ denote the centers of each face, again computed as intersections of face diagonals. In contrast to our previous approach [198], which did not take the direction of the vehicle into account; instead, we encode the information about the vehicle direction ($d = 1$ for vehicles going to camera, $d = 0$ for vehicles going from the camera), in order to determine which side of the bounding box is the frontal one. The

vectors are normalized to have a unit size; storing them with a different normalization (e.g. the front one normalized, the other in the proper ratio) did not improve the results.

Rast. Another way of encoding the viewpoint and also the relative dimensions of vehicles is to rasterize the 3D bounding box and use it as an additional input to the net. The rasterization is done separately for all sides, each filled by one color. The final rasterized bounding box is then a four-channel image containing each visible face rasterized in a different channel. Formally, point p of the rasterized bounding box $\mathbf{T}$ is obtained as

$$\mathbf{T}_p = \begin{cases} (1,0,0,0) & p \in \square b_0 b_1 b_4 b_5 \text{ and d} = 1 \\ (0,1,0,0) & p \in \square b_0 b_1 b_4 b_5 \text{ and d} = 0 \\ (0,0,1,0) & p \in \square b_1 b_2 b_5 b_6 \\ (0,0,0,1) & p \in \square b_0 b_1 b_2 b_3 \\ (0,0,0,0) & \text{otherwise} \end{cases} \tag{5.2}$$

where $\square b_0 b_1 b_4 b_5$ denotes the quadrilateral defined by points b_0, b_1, b_4 and b_5 in Figure 5.3.

Finally, the 3D rasterized bounding box is cropped by the 2D bounding box of the vehicle. For an example, see Figure 5.4, showing rasterized bounding boxes for different vehicles taken from different viewpoints.

5.2.3 Additional Training Data Augmentation

In order to increase the diversity of the training data, we propose additional data augmentation techniques. The first one (denoted as **Color**) deals with the fact that for fine-grained recognition of vehicles (and some other objects), their color is irrelevant. The other method (**ImageDrop**) deals with some potentially missing parts of the vehicle. Examples of the data augmentation are shown in Figure 5.5. Both these augmentation techniques are done only with predefined probability during training, otherwise they are not modified. During testing, we do not modify the images at all.

The results show that both these modifications improve the classification accuracy both in combination with other presented techniques or by themselves.

Color. In order to increase training samples color variability, we propose to randomly alternate the color of the image. The alternation is done in the HSV color space by adding the same random values to each pixel in the image (each HSV channel is processed separately).

ImageDrop. Inspired by Zeiler *et al.* [258], who evaluated the influence of covering a part of the input image on the probability of the ground truth class, we take this a step further and in order to deal with missing parts on the vehicles, we take a random rectangle in the image and fill it with random noise, effectively dropping any information contained in that part of the image.

5.2.4 Estimation of 3D Bounding Box from a Single Image

As the results show, the most important part of the proposed algorithm is **Unpack** followed by **Color** and **ImageDrop**. However, the 3D bounding box is required for unpacking the vehicles and we acknowledge that there may be scenarios when such information is not available. For these cases, we propose a method on how to estimate the 3D bounding box for both training and test time when only limited information is available.

As proposed by [43], the vehicle's contour and vanishing points are required for the bounding box construction. Therefore, it is necessary to estimate the contour and vanishing

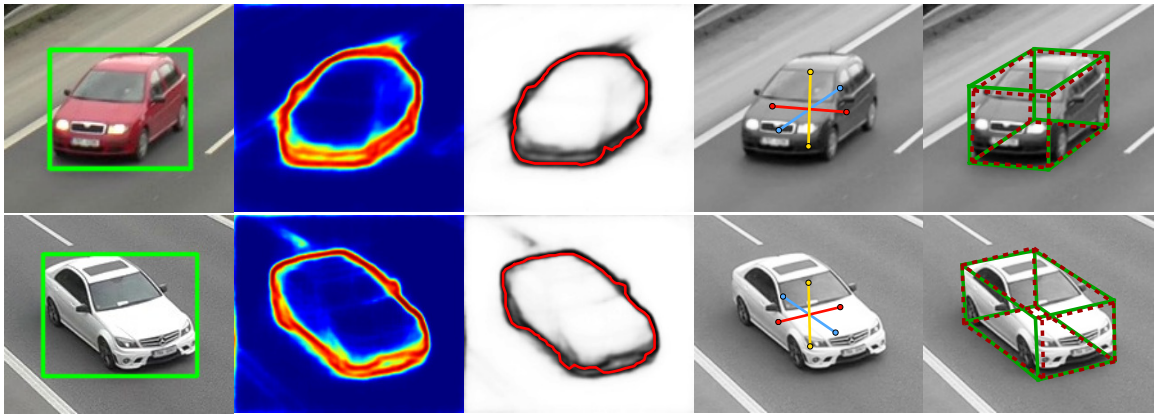


Figure 5.6: Estimation of 3D bounding box. **Left to right:** image with vehicle 2D bounding box, output of contour object detector [248], our constructed contour, estimated directions towards vanishing points, ground truth (**green**) and estimated (**red**) 3D bounding box.

points for the vehicle. For estimating the vehicle contour, we use Fully Convolutional Encoder-Decoder network designed by Yang *et al.* [248] for general object contour detection and masks with probabilities of vehicles contours for each image pixel. To obtain the final contour, we search for global maxima along line segments from 2D bounding box centers to edge points of the 2D bounding box (see Figure 5.6 for examples).

We found out that the exact position of the vanishing point is not required for 3D bounding box construction, but the directions to the vanishing points are much more important. Therefore, we use regression to obtain the directions towards the vanishing points and then assume that the vanishing points are in infinity.

Following the work by Rothe *et al.* [183], we formulated the regression of the direction towards the vanishing points as a classification task into bins corresponding to angles and we used ResNet50 [76] with three classification outputs. We found this approach more robust than a direct regression. We added three separate fully connected layers with softmax activation (one for each vanishing point) after the last average pooling in the ResNet50 (see Figure 5.7). Each of these layers generates probabilities for each vanishing point belonging to the specific direction bin (represented as angles). We quantized the angle space by bins of 3° from −90° to 90° (60 bins per vanishing point in total).

As the training data for the regression we used BoxCars116k dataset (Section 5.3) with the test samples omitted. The direction to vanishing points were obtained by method [43, 41]; however, the quality of the ground truth bounding boxes was manually verified during annotation of the dataset and imprecise samples were removed by the annotators. To construct the lines on which the vanishing points are, we use the center of the 2D bounding box. Even though there is bias in the direction of the training data (some bins have very low number of samples), it is highly unlikely that for example, the first vanishing point direction will be close to horizontal.

With all this estimated information it is then possible to construct the 3D bounding box in both training and test time. It is important to note that by using this 3D bounding box estimation, it is possible to use this method outside the scope of traffic surveillance. It is only necessary to train the regressor of vanishing points directions. For the training of such a regressor, it is possible to use either the directions themselves or viewpoints on the vehicle and focal lengths of the images.

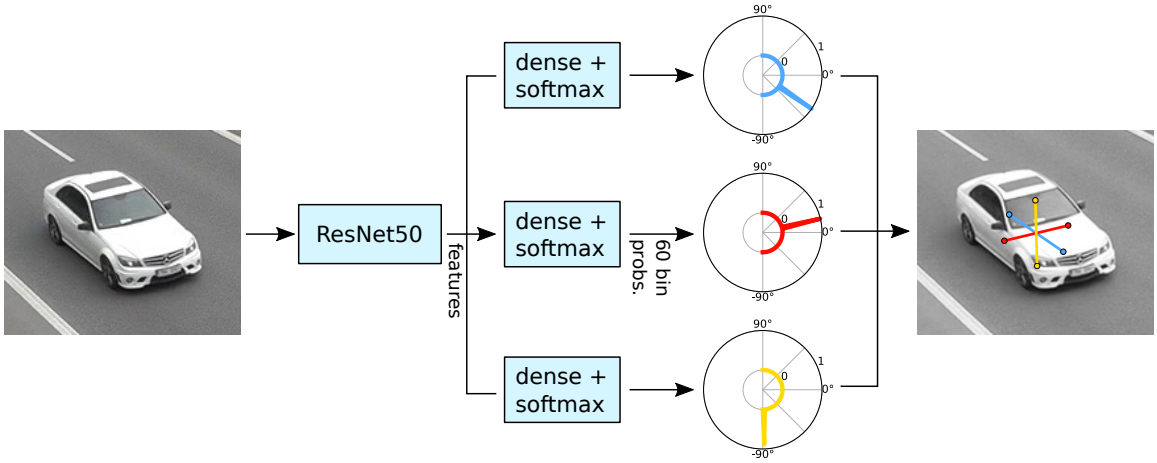


Figure 5.7: The CNN used for estimation of directions towards vanishing points. The vehicle image is fed to ResNet50 with 3 separate outputs which predict probabilities for directions of vanishing points as probabilities in a quantized angle space (60 bins from $-90°$ to $90°$).

Figure 5.8: Collate of random samples from the BoxCars116k dataset.

Using this estimated bounding box, it is possible to unpack the vehicle image in test time without any additional information required. This enables the usage of the method when the traffic surveillance data are not available. The results show that by using this estimated 3D bounding boxes, our method still significantly outperforms other convolutional neural networks without input modification.

5.3 BoxCars116k Dataset

We collected and annotated a new dataset *BoxCars116k*. The dataset is focused on images taken from surveillance cameras as it is meant to be useful for traffic surveillance applications. We do not restrict that the vehicles are taken from the frontal side (Fig. 5.8). We used surveillance cameras mounted near streets and tracked passing vehicles. The cameras were placed on various locations around Brno, Czech Republic and recorded the passing traffic from an arbitrary (reasonable) surveillance viewpoint. Each correctly detected vehicle (by Faster-RCNN [182] trained on COD20k dataset [100]) is captured in multiple images, as it passes by the camera; therefore, we have more visual information about each vehicle.

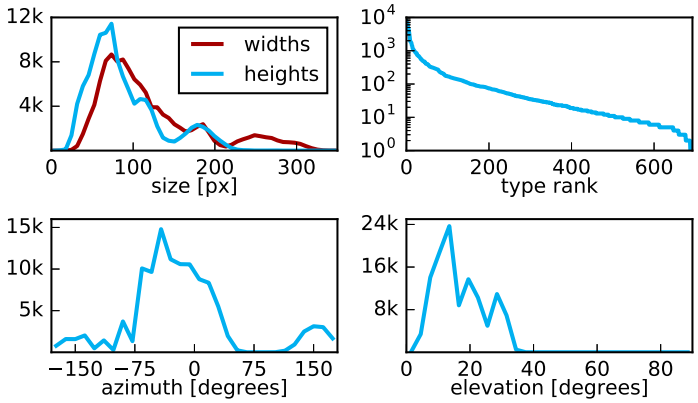


Figure 5.9: BoxCars116k dataset statistics – **top left:** 2D bounding box dimensions, **top right:** number of fine-grained types samples, **bottom left:** azimuth distribution (0° denotes frontal viewpoint), **bottom right:** elevation distribution.

5.3.1 Dataset Acquisition

The dataset is formed by two parts. The first part consists of data from *BoxCars21k* dataset [198] which were cleaned up and some imprecise annotations were then corrected (e.g. missing model years for some uncommon vehicle types).

We also collected other data from videos relevant to our previous work [43, 41, 201]. We detected all vehicles, tracked them and for each track collected images of the respective vehicle. We downsampled the framerate to $\sim$ 12.5 FPS to avoid collecting multiple and almost identical images of the same vehicle.

The new dataset was annotated by multiple human annotators with an interest in vehicles and sufficient knowledge about vehicle types and models. The annotators were assigned to clean up the processed data from invalid detections and assign exact vehicle type (make, model, submodel, year) for each obtained track. While preparing the dataset for annotation, 3D bounding boxes were constructed for each detected vehicle using the method proposed by [43]. Invalid detections were then distinguished by the annotators based on these constructed 3D bounding boxes. In the cases when all 3D bounding boxes were not constructed precisely, the whole track was invalidated.

Vehicle type annotation reliability is guaranteed by providing multiple annotations for each valid track ($\sim$ 4 annotations per vehicle). The annotation of a vehicle type is considered as correct in the case of at least three identical annotations. Uncertain cases were authoritatively annotated by the authors.

The tracks in *BoxCars21k* dataset consist of exactly 3 images per track. In the new part of the dataset, we collect an arbitrary number of images per track (usually more than 3).

5.3.2 Dataset Statistics

The dataset contains 27 496 vehicles (116 286 images) of 45 different makes with 693 fine-grained classes (make & model & submodel & model year) collected from 137 different cameras with a large variation of viewpoints. Detailed statistics about the dataset can be found in Figure 5.9 and the supplementary material. The distribution of types in the dataset is shown in Figure 5.9 (**top right**) and samples from the dataset are in Figure 5.8.

The dataset also includes information about the 3D bounding box [43] for each vehicle and an image with a foreground mask extracted by background subtraction [211, 286]. The dataset has been made publicly available[2] for future reference and evaluation.

Compared to "web-based" datasets, the new *BoxCars116k* dataset contains images of vehicles relevant to traffic surveillance which have specific viewpoints (high elevation), usually small images, etc. Compared to other fine-grained surveillance datasets, our dataset provides data with a high variation of viewpoints (see Figure 5.9 and 3D plots in the supplementary material).

5.3.3 Training & Test Splits

Our task is to provide a dataset for fine-grained recognition in traffic surveillance without any viewpoint constraint. Therefore, we have constructed the splits for training and evaluation in a way which reflects the fact that it is not usually known beforehand from which viewpoints the vehicles will be seen by the surveillance camera.

Thus, for the construction of the splits, we randomly selected cameras and used all tracks from these cameras for training and vehicles from the rest of the cameras for testing. In this way, we are testing the classification algorithms on images of vehicles from previously unseen cameras (viewpoints). This splits selection process implies that some of the vehicles from the test set may be taken under slightly different viewpoints from the ones that are in the training set.

We constructed two splits. In the first one (**hard**), we are interested in recognizing the precise type, including the model year. In the other one (**medium**), we omit the difference in model years and all vehicles of the same subtype (and potentially different model years) are present in the same class. We selected only types which have at least 15 tracks in the training set and at least one track in the testing set. The hard split contains 107 fine-grained classes with 11 653 tracks (51 691 images) for training and 11 125 tracks (39 149 images) for testing. Detailed split statistics can be found in the supplementary material.

5.4 Conclusion

This article presents and sums up multiple algorithmic modifications suitable for CNN-based fine-grained recognition of vehicles. Some of the modifications were originally proposed in a conference paper [198], while others are results of the ongoing research. We also propose a method for obtaining the 3D bounding boxes necessary for the image unpacking (which has the largest impact on performance improvement) without observing a surveillance video, but only working with the individual input image. This considerably increases the application potential of the proposed methodology (and the performance for such estimated 3D boxes is only somewhat lower than when "proper" bounding boxes are used). We focused on a thorough evaluation of the methods: we coupled them with multiple state-of-the-art CNN architectures [194, 76], and measured the contribution/influence of individual modifications.

Our method significantly improves the classification accuracy (up to **+12 percentage points**) and reduces the classification error (up to **50 % error reduction**) compared to the base CNNs. Also, our method outperforms other state-of-the-art methods [128, 193, 51] by **9 percentage points** in single image accuracy and by **7 percentage points** in whole track accuracy.

[2] https://medusa.fit.vutbr.cz/traffic

We collected, processed, and annotated a dataset *BoxCars116k* targeted to fine-grained recognition of vehicles in the surveillance domain. Contrary to a majority of existing vehicle recognition datasets, the viewpoints are greatly varying and correspond to surveillance scenarios; the existing datasets are mostly collected from web images and the vehicles are typically captured from eye-level positions. This dataset has been made publicly available for future research and evaluation.

Chapter 6

BrnoCompSpeed – Dataset for Monocular Vehicle Speed Measurement

The text in this chapter contains detailed information about acquiring the BrnoCompSpeed dataset and our contribution to monocular vehicle speed measurement. This work was published in *Transactions on Intelligent Transportation Systems* journal in article Sochor *et al.*, *Comprehensive dataset for automatic single camera visual speed measurement*, T-ITS 2018 [200]. Related work for vehicle speed measurement can be found in Section 2.1 on page 8. The comparison of the proposed **BrnoCompSpeed** dataset with other available datasets for vehicle speed measurement can be found in Table 2.2 on page 14, as this comparison is heavily connected to state-of-the-are literature. For evaluations of our experiments, please see Section *V. Experiments* in the original paper.

Abstract In this paper, we focus on traffic camera calibration and visual speed measurement from a single monocular camera, which is an important task of visual traffic surveillance. Existing methods addressing this problem are hard to compare due to a lack of a common dataset with reliable ground truth. Therefore, it is not clear how the methods compare in various aspects and what are the factors affecting their performance. We captured a new dataset of 18 full-HD videos, each around one hour long, captured at 6 different locations. Vehicles in the videos (20 865 instances in total) are annotated with precise speed measurements from optical gates using LIDAR and verified with several reference GPS tracks. We made the dataset available for download and it contains the videos and metadata (calibration, lengths of features in image, annotations, etc.) for future comparison and evaluation. Camera calibration is the most crucial part of the speed measurement; therefore, we provide a brief overview of the methods and analyze a recently published method for fully automatic camera calibration and vehicle speed measurement and report the results on this dataset in detail.

6.1 Monocular Vehicle Speed Measurement

Speed measurement is one of the crucial problems in traffic surveillance. So far, the field is dominated by radar and section speed measurements because they meet tight methodological requirements and standards. However, these methods are limited in the information

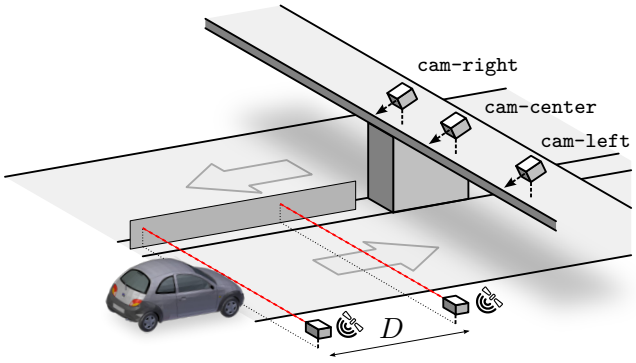


Figure 6.1: The data recording setup. We use two LIDARs synced by GPS time, and three cameras recording the highway from different surveillance viewpoints.

they provide and they may be expensive. For example, in radar measurement it is impossible to recognize the fine-grained models of passing cars and the radar antenna must be placed at a specific position regarding the traffic. Section speed measurement requires two cameras for each position and a complex infrastructure for processing the data. Speed measurement from a single monocular camera is not typically used for surveillance; however it can be beneficial – one camera can be used for surveillance on multiple lanes, it is possible to use the data for fine-grained make & model recognition of the vehicles [74, 7, 88, 85] and other tasks. Another interesting aspect is that it is possible to use already installed monitoring/security cameras for speed measurement and other traffic analysis tasks.

A number of works dealing with monocular speed measurement can be found in the literature [187, 34, 63, 78, 152, 164, 195, 43, 114, 149, 37] (detailed individually below). Such systems are on the rise especially recently, with the growing number of IP cameras, with increase of their resolution, and with the development of computer vision algorithms used for their processing. Our aim is to provide an important missing piece: a dataset which would allow for reliable comparison between the approaches. These systems are described in detail in the following section.

We captured a new benchmark dataset of 18 full-HD videos taken from surveillance viewpoints on the traffic (see Figure 6.1). Each of the videos is around one hour long to allow for even lengthy calibration procedures and self-adjustment of the surveillance system. Triplets of videos are observing the same time interval at the same location from different angles. These shots were captured at 6 different locations. Vehicles in the videos (20 865 instances in total) are annotated with precise speed measurements from optical gates using LIDAR and verified with several reference GPS tracks. We provide[1] the videos and metadata (calibration, distances measured on the road plane, annotations, etc.) for future comparison and evaluation. To illustrate the properties of the dataset and to establish a first baseline, we analyze the data by a recently published method for fully automatic camera calibration and vehicle speed measurement [43] and we report the quantitative results.

Although the dataset is focused on speed measurement, it can be used also for different traffic surveillance tasks, for example vehicle counting, tracking, vehicle classification and other.

[1] https://medusa.fit.vutbr.cz/traffic

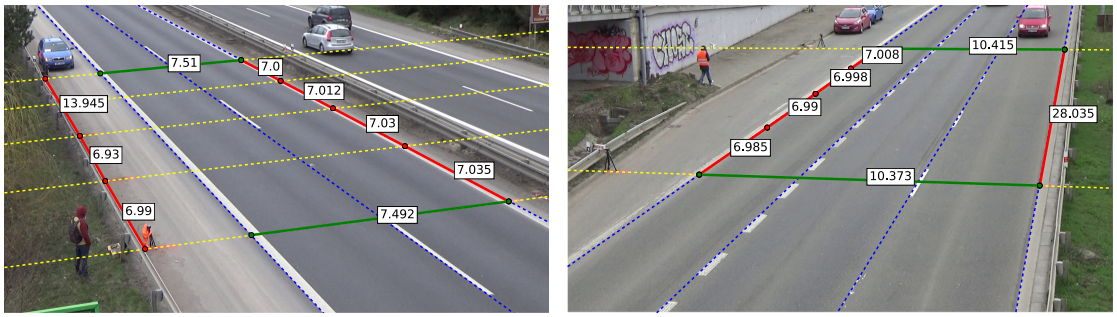


Figure 6.2: Markings and measured distances on the road plane. **Blue dashed lines** – annotated lane dividing lines, **yellow dashed lines** – measurement lines, **red line segments** – measured distances towards the first vanishing point, **green line segments** – measured distances towards the second vanishing point. Images with these annotations for all videos can be found in the supplementary material. Best viewed on screen.

We consider the camera calibration algorithm to be the most crucial part of speed measurement. It defines how well the speed measurement is done as it is impossible to measure speed accurately with a poorly calibrated camera. The used algorithm also defines whether it is usable with a camera observing the road from arbitrary viewpoint and it determines whether the method can be used fully automatically which is important for large scale deployment. Therefore, we include a brief overview of existing camera calibration algorithms for traffic surveillance applications.

The key contributions of this paper are: **a)** Novel, publicly available dataset for evaluation of camera calibration in traffic surveillance and speed measurement. The dataset contains 18 videos and 20 865 vehicles with known precise ground truth. **b)** Thorough and complex evaluation of a recent fully automatic method for traffic camera calibration [43].

6.2 Dataset Acquisition Methodology

We performed six recording sessions at different locations with free flow traffic. For each session, we obtained three videos (approximately one hour long) from different positions by different video cameras (Panasonic HC-X920, Panasonic HDC-SD90, Sony Handycam HDR-PJ410). The videos were recorded in full-HD resolution and with 50 frames per second progressive scan. The recording setup is schematically shown in Figure 6.1 and an example of the scene is in Figure 6.2.

Reference speed values of passing vehicles were obtained from a pair of experimental setups, containing a LIDAR (LaserAce® IM HR 300), a GPS module (Leadtek LR9540D), and a PC. These were placed on the side of the road perpendicular to the direction of traffic flow at a defined distance D between them. It was important to place the lasers to the same height and parallel in the vertical and horizontal axes (see Figure 6.1). This requirement guarantees that an incoming vehicle always disturbs the laser beams at the same point.

The LIDAR works in the single shot mode (one laser pulse per range measurement). The sampling rate is 1 kHz and maximal measurement range is 300 m. GPS receiver synchronizes times on PC using TIMEMARK signal (1 pulse per second with 1 μs precision). The data from each LIDAR and GPS module were recorded by the computer and each measurement was assigned with a high resolution timestamp obtained from the operating system.

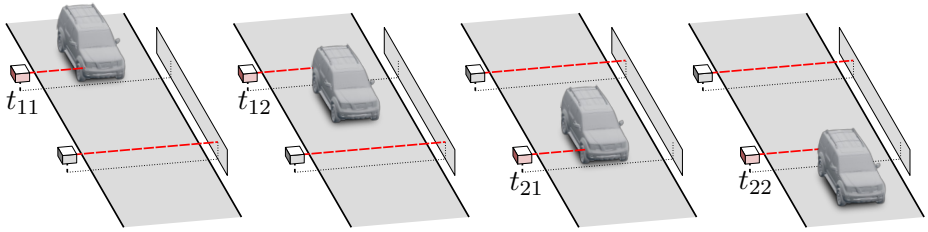


Figure 6.3: The four phases of a passing car which are used for ground truth speed annotation. Best viewed on screen.

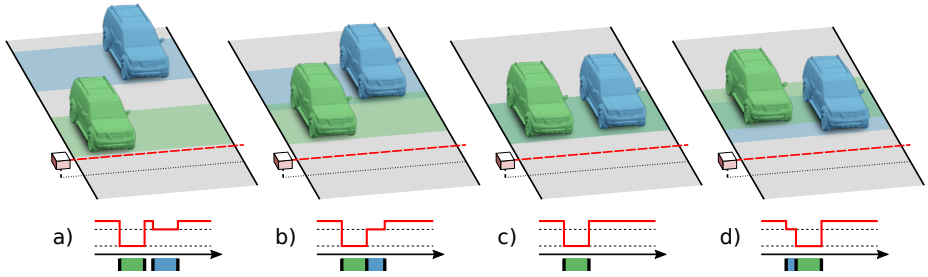


Figure 6.4: Possible variants of occlusion. **a)** the vehicles are not occluded at all, **b)** the closer vehicle is occluding the frontal part of the farther vehicle **c)** the farther vehicle is fully covered, **d)** the farther vehicle's rear part is covered. The graphs below represent LIDAR responses with different levels for empty road, fast lane (top dashed line), and slower lane (bottom dashed lane). See text for description of how all these situations are handled. Best viewed on screen.

Distance logs from both LIDARs are pre-processed individually. We search for timestamps t_{xy} (see Figure 6.3) which correspond to car entering/leaving first/second laser beam. And each excitation is assigned with the lane based on the measured distance from the LIDAR. Excitations generated by the same car on the first and second LIDAR need to be matched. The matching is based on the correspondence of lane with limits on speed and acceleration of cars. We calculate immediate speed when entering the first laser v_{11} (at the time t_{11}), length of the vehicle L, and its average acceleration a over measured span of known length D by the following set of equations:

$$v_{11} + \frac{1}{2}a(t_{12} - t_{11}) = \frac{L}{t_{12} - t_{11}} \tag{6.1}$$

$$v_{21} + \frac{1}{2}a(t_{22} - t_{21}) = \frac{L}{t_{22} - t_{21}} \tag{6.2}$$

$$v_{11} + \frac{1}{2}a(t_{22} - t_{11}) = \frac{D + L}{t_{22} - t_{11}} \tag{6.3}$$

Then, it is possible to compute immediate speed at any point of the measured span. Unfortunately, when a car is partially occluded by another vehicle, the equations above cannot be used for the calculation (as some timestamps are unknown). If at least timestamps t_{11} and t_{21} are known, the average speed can be computed as

$$v_{avg} = \frac{D}{t_{21} - t_{11}}. \tag{6.4}$$

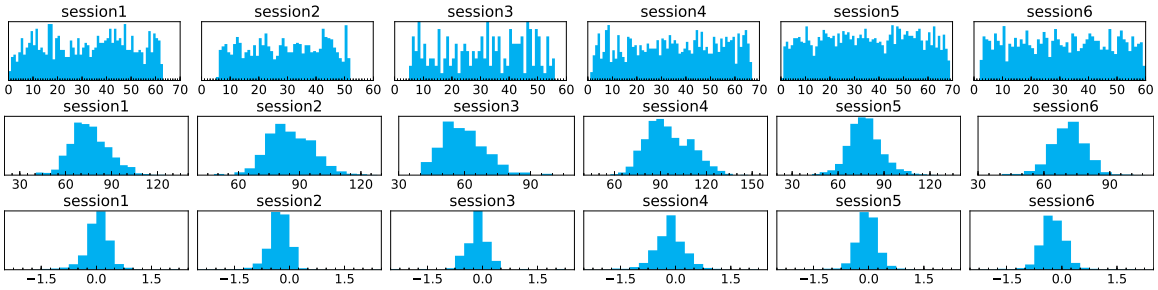


Figure 6.5: **top:** Histograms of density of vehicles for each session; one minute granularity in x-axis. **middle:** Ground truth speeds measured by the LIDAR setup (Section 6.2); speed in km/h on x-axis. **bottom:** Ground truth accelerations; m/s^2 on the x-axis.

As we are using LIDARs (instead of e.g. optical gates), we are able to detect situations when a vehicle is partially occluded by a closer vehicle using the measured distances by the LIDARs. See Figure 6.4 for examples of all possible occlusion types. There are several possibilities of occlusion on the pair of LIDARs:

1. Occlusion situations on both the LIDARs are either **a)** or **d)** – in these situations we are able to detect that there is a occluded vehicle and measure their speed.

2. Occlusion situations on at least one LIDAR is of type **b)** or **c)** – we are able to detect that there is a second "shadowed" vehicle; the speed measurement is not reliable and the second vehicle is omitted from the dataset and evaluation.

3. Occlusion situations on **both** LIDARs are **c)** – the second "shadowed" vehicle cannot even be detected. This situation is very unlikely, as the vehicle in the fast lane would have to be smaller, precisely aligned, and maintain the same speed as the closer vehicle.

In summary, we either measure the speed accurately, or we know that the speed measurement is not precise and we ignore such a measurement. Therefore, besides the 20 865 vehicles with precise ground truth speed, the dataset contains 2 779 instances of vehicles which are marked as invalid for speed measurement evaluation.

We also performed manual verification of the matched timestamps t_{11} and t_{21} by checking that they correspond to the same vehicle in the video.

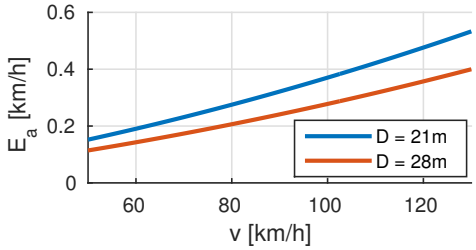
6.2.1 Accuracy of the Acquired Dataset

The distance D between LIDARs is 28 meters (21 meters in one case), and the LIDARs have 1 kHz sampling rate. The actual value of D for every recording session was measured by handheld laser distance meter, and we assume that upper bound of the distance measurement error is $e_d = 0.05\,\text{m}$. Time measurement error caused by improper synchronization of LIDARs is at most $e_t = 1\,\text{ms}$. Both, e_d and e_t are exaggerated and in reality they are lower. The upper bound of speed measurement error E_r (relative) and E_a (absolute) for the given speed v can be computed as:

Table 6.1: Numbers of vehicle passes with known ground truth for each video.

	left	center	right
session1	854	848	849
session2	1 163	1 258	1 583
session3	193	193	193
session4	1 188	1 192	1 177
session5	2 021	2 027	2 030
session6	1 358	1 353	1 358
TOTAL		20 865	

$$E_r = \frac{e_d + e_t \cdot v}{D}$$
$$E_a = E_r \cdot v$$



For a vehicle going $v = 20\,\mathrm{m/s}$ (72 km/h), the resulting maximum possible errors are $E_r = 0.25\,\%$ and $E_a = 0.05\,\mathrm{m/s}$ (0.18 km/h). We consider these errors to be small enough as the errors of the methods presented in experiments are much higher than this error of measurement.

6.3 Dataset Statistics and Evaluation Protocol

The dataset consists of 18 videos (6 sessions on different locations, 3 videos from different angles for each location) and there is totally 20 865 vehicles with known ground truth speed.

To provide statistics about the dataset we report the total number of cars with ground truth speed for each video in Table 6.1. We also report histograms of speeds, accelerations, and traffic density in Figure 6.5.

The dataset is (to our knowledge) by far larger than other datasets serving similar purpose reported in the literature. It covers views typical for traffic surveillance from arbitrary cameras. It provides high quality videos with various traffic conditions (low traffic in Session 3, high traffic in Sessions 5 and 6). However, it is quite limited in lighting and weather conditions. Almost all videos were taken in cloudy weather (except some parts of Session 3) with no distracting phenomena (fog, rain, etc.).

6.3.1 Evaluation Protocol

For future comparison of methods, we provide an evaluation script[2] which automatically evaluates all the used metrics. It requires two vanishing points of the road plane, principal point of the camera and scale of the scene as the calibration parameters. Then, the systems are supposed to report for each observed vehicle a track of one arbitrary reference point on the road plane (frame numbers + image coordinates). In our case, the point is obtained by the constructed 3D bounding boxes (see Figure 2.2). The point must be on the road plane

[2]The evaluation code is available together with the dataset at https://medusa.fit.vutbr.cz/traffic

for proper projection; however it can be any point on the road plane which the authors are able to localize – it is not necessary to use the 3D bounding boxes.

To compare vehicles with the ground truth, we match the time when a vehicle passed the measurement line to the time reported by LIDAR and the lane in which the vehicle is. As the vehicles are sometimes not tracked correctly and the tracking can be lost, we extrapolate the vehicle trajectory in order to get the correct time.

For each vehicle, we calculate tentative speeds between the positions K frames apart (approximately 0.1 s, $K = 5$ for 50 fps video) by projecting the image point coordinates to the road plane using the provided calibration. The resulting speed is then median of the tentative speeds. We found out that this method is more robust than measuring the full section speed due to possible tracking errors.

The computation of distance between two points p_1 and p_2 is schematically shown in Figure 2.1 with general model for traffic surveillance camera and it is described in detail in the supplementary material of BrnoCompSpeed paper [200].

As methods may require different training sets we define three train/test splits. Split **A** uses all videos for testing, split **B** has Session 1 and Session 2 reserved for training, and finally, split **C** has Session 1, Session 2, and Session 3 for training. Whenever it is possible, the results should be reported on the splitting with the lowest number of training sessions.

6.4 Vehicle Speed Measurement – Summary

We collected and processed a dataset for evaluation of purely visual speed measurement by a single monocular camera. Cameras are becoming ubiquitous and a considerable portion of them observe traffic. By providing this dataset we intend to encourage research of fully automatic traffic camera calibration methods, which could be used for mining valuable automatic traffic surveillance data from existing and new camera infrastructure.

On the collected data, we evaluated an approach which is both fully automatic and can process virtually arbitrary views (see Section *V. Experiments* in original paper [200]). The evaluation shows its weaknesses (localization of the VP2 and scale inference), which can encourage further research in this area, which we will focus on. The measurements also established a first baseline to be outperformed by future works.

Part III

Re-Identification of Vehicles from Image and Video

Chapter 7

Learning Feature Aggregation in Temporal Domain for Re-Identification

This part of the thesis contains detailed information about our contribution to vehicle re-identification methods using feature aggregation in the temporal domain. This work was published in *Computer Vision and Image Understanding* journal in article Špaňhel *et al.*, *Learning Feature Aggregation in Temporal Domain for Re-Identification*, CVIU 2020 [209]. Related work for this paper is elaborated in Section 2.6 on page 22. Statistics and information about the acquisition process of *CarsReId74k* dataset, introduced in this work, is presented in Chapter 4 on page 39.

Abstract Person re-identification is a standard and established problem in the computer vision community. In recent years, vehicle re-identification is also getting more attention. In this paper, we focus on both these tasks and propose a method for aggregation of features in temporal domain as it is common to have multiple observations of the same object. The aggregation is based on weighting different elements of the feature vectors by different weights and it is trained in an end-to-end manner by a Siamese network. The experimental results show that our method outperforms other existing methods for feature aggregation in temporal domain on both vehicle and person re-identification tasks. Furthermore, to push research in vehicle re-identification further, we introduce a novel dataset CarsReId74k. The dataset is not limited to frontal/rear viewpoints. It contains 17,681 unique vehicles, 73,976 observed tracks, and 277,236 positive pairs. The dataset was captured by 66 cameras from various angles.

7.1 Vehicle Re-Identification Introduction

We consider the problem of re-identification of individuals observed by different cameras at different locations and times. Our work applies to the fairly standard person re-identification [234, 82, 243, 266, 25, 284], and to the rather emerging vehicle re-id [132, 138, 190, 235, 244, 272], but it can be used for other similar tasks as well.

The re-id system is given a query track of images and a database of pre-stored tracks, one of which is assumed to share the same identity with the query. The system is supposed to output a small subset of the best matching database samples along with their similarity

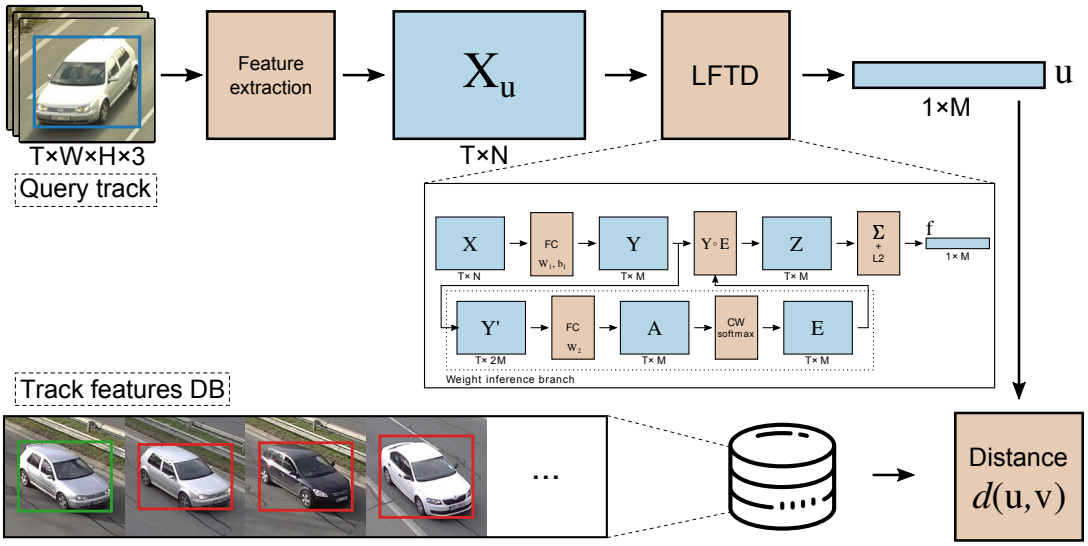


Figure 7.1: We propose a new method **LFTD** for aggregation of features in temporal domain. The method generates one feature vector per track of observed objects (e.g. vehicles, persons). See Section 7.2.2 for details.

scores. Some solutions process the images in the tracks directly (comparing images in the query track versus images in the database – e.g. [257]). However, fast and real-time processing requires the system to extract a short feature vector for each of the database tracks and to match them to the feature vector extracted from the query track by computing a cheap pairwise metric. Our work is targeted on the second, generally more efficient, mode of processing, that is extraction of a single fixed-size feature vector for a track of variable length by aggregating the feature vectors extracted from individual observations (images).

We propose a new method for **feature aggregation in temporal domain** LFTD (Learning Features in Temporal Domain) which takes feature vectors extracted from the individual observations (images) as its input, and it results in a single relatively low-dimensional time-pooled feature vector usable by the re-id system. Unlike other methods which use either RNN [158, 266, 245, 25, 243, 284, 267] or produce weights for feature vectors as a whole [247, 284, 243], our method produces a different weight for every element of the feature vectors which leads to an improved performance as different parts of feature vectors are weighted differently. The weights are generated by a neural network for each set (track) of feature vectors. The final feature vector for the track is obtained by computing element-wise product between the track's features and the weight matrix, and then reducing the matrix in temporal domain by summation. The results show that the proposed method outperforms other methods [245, 158, 51, 243, 266, 25, 284, 267] in both vehicle and person re-identification tasks. See Figure 7.1 for the full re-id pipeline.

Furthermore, we propose to use a different metric for comparing the feature vectors. Previous works [113, 125, 191] showed that it is beneficial to use Mahalanobis distance for feature comparison rather than Euclidean (or cosine) distance. However, the Mahalanobis distance has significant limitations, mainly its time complexity which is quadratic with respect to feature vector dimensionality. Therefore, we propose to use **Weighted Euclidean** distance, constructed by constraining the Mahalanobis distance learning to diagonal matrix. The experiments show that it outperforms both Mahalanobis [191] and Euclidean distance, while it keeps linear time and memory complexity.

To improve the availability of datasets for vehicle re-identification, we collected and annotated a new vehicle re-identification dataset called **CarsReId74k**. As it is common in traffic surveillance to have whole tracks of vehicles and not individual images, the dataset includes multiple observations for each vehicle as it is passing in front of the cameras (*left*, *center*, *right*). We focus on **appearance-based** vehicle re-identification: vehicles' license plates were only used for ground truth data acquisition (recorded by a *zoomed-in* camera). The images of vehicles taken by the other cameras are in most cases so small that it is not possible to recognize the license plates. The dataset contains 17,681 unique vehicles, 73,976 observed tracks, and 277,236 positive pairs, taken by 66 cameras from various angles in multiple sessions. We make the dataset publicly available[1] for future comparison and research.

7.2 Proposed Method for Learning Feature Aggregation in Temporal Domain

The standard baseline to aggregating features from multiple observations of the same object in temporal domain is to use averaging over time. However, existing literature [245, 158, 51, 243, 266, 25, 284] shows that the accuracy can be improved over the simple averaging by feature vector weighting or by using RNN. We propose a novel method for the aggregation in temporal domain, which is based on weighting different elements of the features vectors by different weights.

The proposed LFTD method aggregates arbitrary features from a sequence of images (of an arbitrary length), extracted by any feature extractor (it can be even some of newly presented spatial attention networks [230, 212]) into a single fixed-sized feature vector. It allows to create a database of previously seen objects (with multiple observations) with such fixed-sized feature vectors and then quickly search the database for objects similar to query objects. LFTD expands the feature dimensions by concatenating the average feature vector to features extracted in every time step. It allows the network to propagate global information form the track to each individual observation. Feature vectors are weighted by column-wise softmax (i.e. along time axis) which forces the network to pick important observation for every feature in the vector instead of weighting observations as a whole. This network design performed the best during our preliminary experiments, compared with user-based vector normalization (subtracting or dividing features by average feature vector), or different types of feature expansion (e.g. by max-pooled feature vector, etc.).

7.2.1 Image Feature Extraction

We are processing *the whole tracks* of objects of interest with labels corresponding to identities $\{(\mathcal{T}_i, l_i)\}$, where $\mathcal{T}_i$ is a sequence of images $(\mathbf{I}_1, \mathbf{I}_2, \ldots, \mathbf{I}_{T_i})$, i.e. observations of an object l_i in the track.

For each track (image sequence), features are extracted for each image independently by a feature extractor (a CNN-based or another one, the method is not limited by design to a particular type). The feature extractor yields a feature matrix $\mathbf{X}_i \in \mathbb{R}^{T_i \times N}$ for each track $\mathcal{T}_i$. T_i is number of time samples (images) for each track $\mathcal{T}_i$ and N is the length of an individual feature vector. In our experiments $N = 2048$, in case of ResNet-50, and $N = 1536$ for Inception-ResNet-v2.

[1] https://medusa.fit.vutbr.cz/traffic

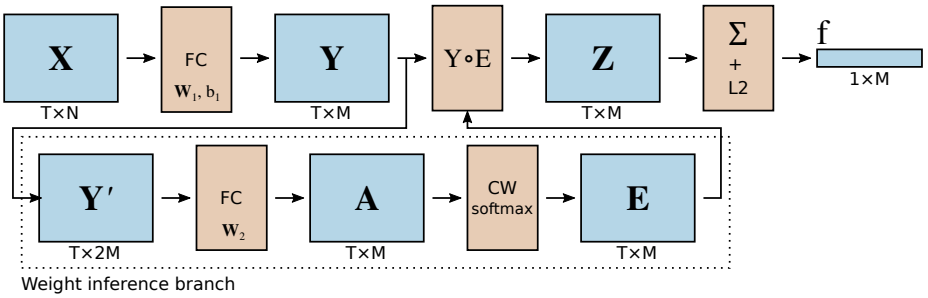


Figure 7.2: Schematic network design representing the proposed method for feature aggregation in temporal domain. See Section 7.2.2 for explanation of the symbols.

To make the notation uncluttered, we will omit the lower index i from now on. Therefore, we will refer to a individual track as $\mathcal{T}$, the number of time samples of the track as T, and its features as $\mathbf{X} \in \mathbb{R}^{T \times N}$.

7.2.2 Processing of Features in Temporal Domain

The schematic design of the feature aggregation network is illustrated in Figure 7.2 and the description follows. Aggregation of features $\mathbf{X} \in \mathbb{R}^{T \times N}$ in temporal domain is essentially a mapping $\varphi : \mathbb{R}^{T \times N} \mapsto \mathbb{R}^{M}$, where M is the dimensionality of feature vector $\mathbf{f}$ representing track $\mathcal{T}$.

First, the feature vector of each observation in the track is compressed from N to M dimensions ($M < N$) by

$$\mathbf{y}_\tau = \tanh\left(\mathbf{W}_1 \mathbf{x}_\tau + \mathbf{b}_1\right), \quad 1 \leq \tau \leq T, \tag{7.1}$$

where $\mathbf{W}_1 \in \mathbb{R}^{M \times N}$ are the parameters of the first fully connected layer (Figure 7.2), forming a compressed feature matrix $\mathbf{Y} \in \mathbb{R}^{T \times M}$.

In order to allow "communication" between the features across the track, we form a new feature matrix $\mathbf{Y}' \in \mathbb{R}^{T \times 2M}$, where each row contains the original feature vector in that row and an average feature vector for the whole track. Therefore $\mathbf{Y}' = [\mathbf{y}'_1, \mathbf{y}'_2, \ldots, \mathbf{y}'_T]^\top$, where

$$\mathbf{y}'_\tau = \begin{bmatrix} \mathbf{y}_\tau \\ \frac{1}{T} \sum_{i=1}^{T} \mathbf{y}_i \end{bmatrix}. \tag{7.2}$$

From these feature vectors concatenated with the average feature vector, we generate activations by another fully connected layer $\mathbf{a}_\tau = \mathbf{W}_2 \mathbf{y}'_\tau$, forming matrix $\mathbf{A} \in \mathbb{R}^{T \times M}$. These activations are then normalized by softmax; however, the normalization is not done by rows (as usually), but by columns to normalize the activation for each component of the feature vector. Therefore, the normalization yields matrix $\mathbf{E} \in \mathbb{R}^{T \times M}$, where

$$e_{\tau j} = \frac{\exp(a_{\tau j})}{\sum_{i=1}^{T} \exp(a_{ij})}. \tag{7.3}$$

The weight matrix $\mathbf{E}$ is then merged with the compressed feature matrix $\mathbf{Y}$ by Hadamard (element-wise) product into matrix $\mathbf{Z} = \mathbf{Y} \circ \mathbf{E}$. The final feature vector $\mathbf{f}$ is then obtained as a sum of feature vectors in rows of matrix $\mathbf{Z}$, normalized to a unit vector.

$$\mathbf{f} = \frac{\sum_{\tau=1}^{T} \mathbf{z}_\tau}{\left\| \sum_{\tau=1}^{T} \mathbf{z}_\tau \right\|_2} \tag{7.4}$$

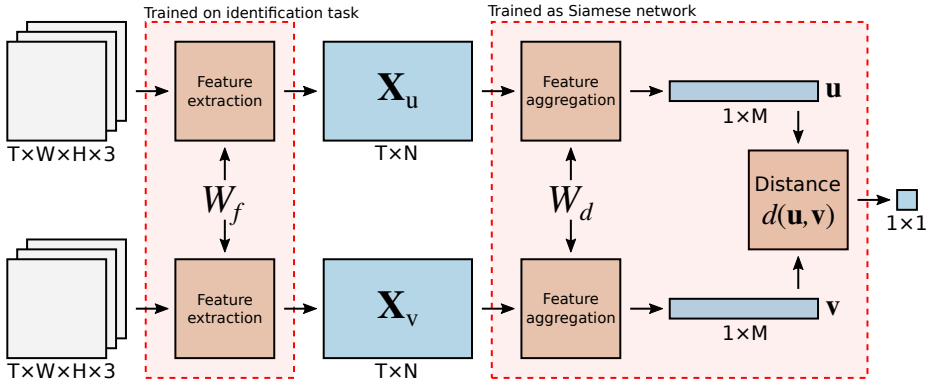


Figure 7.3: Schematic design representing the full training and inference pipeline. In our approach, we train the image feature extractor NN on the identification task on the given dataset; however, the proposed method for feature aggregation can work with an arbitrary image feature extractor. W_f and W_d refer to the shared weights of feature extractor part and feature aggregation part, respectively.

Therefore, if matrix $\mathbf{A}$ contained a single constant value, the aggregation would be reduced to one fully connected layer followed by average pooling. Instead, the weights $\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}_1$ are trained by back-propagation and the network is thus able to produce better features.

7.2.3 Metrics for Distance Computation

The re-identification task is defined by a query sample (track) and a gallery of samples (tracks), where one sample from the gallery is supposed to have the same identity as the query sample. It is common to use Euclidean (or cosine for unit feature vectors) distance $d_{\mathrm{E}}(\mathbf{u}, \mathbf{v}) = \sqrt{\sum_i (u_i - v_i)^2}$ to rank the gallery samples by their distance from the query feature vector.

Previous works have shown that other distance metrics can outperform the Euclidean one, and the Mahalanobis distance seems to be powerful [113, 125]. Mahalanobis distance between vectors $\mathbf{u}$ and $\mathbf{v}$ is computed as $\sqrt{(\mathbf{u}-\mathbf{v})^\top \mathbf{M}(\mathbf{u}-\mathbf{v})}$, requiring that matrix $\mathbf{M}$ is symmetric and positive semi-definite [191]. They claim that such a constraint is hard to enforce and propose to decompose the matrix $\mathbf{M} = \mathbf{W}\mathbf{W}^\top$ and learn $\mathbf{W}$ instead. Then, the Mahalanobis distance is computed by the following equation:

$$d_{\mathrm{M}}(\mathbf{u}, \mathbf{v}) = \sqrt{(\mathbf{u}-\mathbf{v})^\top \mathbf{W}\mathbf{W}^\top (\mathbf{u}-\mathbf{v})}. \tag{7.5}$$

Using Mahalanobis distance as proposed by [191] improves the re-identification accuracy, paying a high price in terms of its time complexity. Both time and memory asymptotic complexities are $\mathcal{O}(D^2)$ where D is the dimensionality of the feature vectors. This can cause significant problems in re-identification as the computational cost for quadratic time complexity is significantly larger even for $D = 128$. Therefore, we propose to learn suitable weights for Weighted Euclidean distance (equivalent to Mahalanobis distance when matrix $\mathbf{M}$ is diagonal), instead. We express the Weighted Euclidean distance by

$$d_{\mathrm{WE}}(\mathbf{u}, \mathbf{v}) = \sqrt{\sum_{i=1}^{D} w_i (u_i - v_i)^2}, \tag{7.6}$$

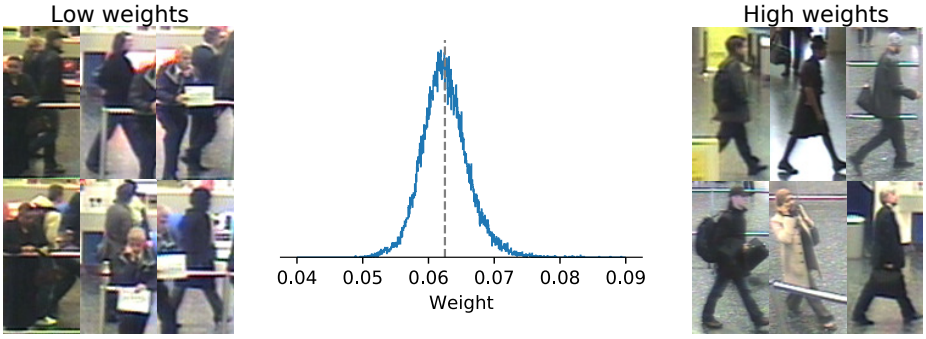


Figure 7.4: **Middle:** Distribution of mean weights for test images in iLIDS-VID dataset [234]. The dashed grey line denotes image weight for average pooling with $T = 16$. **Sides:** Images with the lowest and highest weights which show that low weight is usually assigned to images with occluding pedestrians.

where $\mathbf{w} = [w_1, w_2, \ldots, w_D]$ are learned weights. It should be noted that if all the weights w_i are equal to 1, the metric is reduced to standard Euclidean distance. Before learning, we initialize the weights by randomly sampling from normal distribution with $\mu = 1$ and $\sigma = 0.1$.

As the Weighted Euclidean distance can be interpreted as Mahalanobis distance with diagonal matrix $\mathbf{M}$, the same conditions must be kept. The symmetricity is satisfied trivially as it is a diagonal matrix. However, to ensure the positive semi-definite property, we ensure that all the weights w_i are non-negative by clipping values bellow zero after each update of the weights during learning.

The Weighted Euclidean distance has benefits when compared to both standard Euclidean and Mahalanobis distances. Compared to the Euclidean distance, it has a higher expressive power thanks to learned weights $\mathbf{w}$. On the other hand, compared to full Mahalanobis distance, it is much faster as both time and memory complexity of the Weighted Euclidean distance is $\mathcal{O}(D)$. At the same time, as the results in Section 7.3.1 show, our proposed Weighted Euclidean distance also outperforms both Euclidean and full Mahalanobis distance in terms of re-identification accuracy.

7.2.4 Full Training and Inference Network

Both the feature aggregation network (Section 7.2.2) and the Weighted Euclidean metric (Section 7.2.3) are trained by a Siamese network [69], see Figure 7.3. For speeding up the training, we pre-train the feature extractor (Inception-ResNet-v1 [216] for vehicle re-id and ResNet50 [76] for person in our case) for the identification task using the dataset training data and then we cache all features for the tracks and train the feature aggregation and distance metric with the cached features. Training the network end-to-end did not improve the results further. We use a standard contrastive loss [69]

$$L(\mathbf{u}, \mathbf{v}, y) = y \cdot d(\mathbf{u}, \mathbf{v})^2 + (1 - y) \cdot [m - d(\mathbf{u}, \mathbf{v})]_+^2 , \tag{7.7}$$

where $\mathbf{u}$ and $\mathbf{v}$ are feature vectors, m is the margin between negative samples, $[\ldots]_+$ denotes maximum value with zero, and $y = 1$, if $l_u = l_v$ or 0 otherwise (l_u and l_v are sample identities). Distance d is one of d_{E}, d_{M}, or d_{WE} from the previous section.

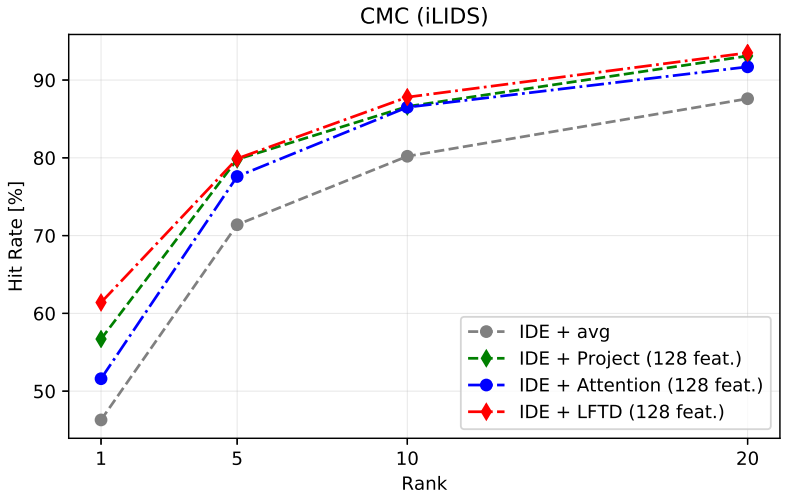


Figure 7.5: CMC curve on the iLIDS dataset for individual parts of the proposed network.

7.2.5 Design Choices

We analyzed several design choices we made. During preliminary experiments we used ReLU nonlinearity in Equation (7.1) and found out that the results are significantly better with tanh nonlinearity.

Furthermore, on iLIDS-VID dataset [234], we tested how important different parts of the network are. In these experiments, 128 dimensional features were used (except the average pooling, where the features had 2048 dimensions). When only average pooling was used, we got Hit@1 46.3 % and with the full network Hit@1 is 61.4 %. However, if we use only the weighting mechanism (omit feature projection by (7.1)), the Hit@1 is 51.6 %. And finally, if we use average pooling (omit the weighting mechanism) with the feature projection (7.1), we receive Hit@1 56.7 %. This shows that both parts of the network contribute to the accuracy and the contributions can be merged to obtain better results. A graphical comparison of design choices evaluation can be found in Figure 7.5. Full results of design choices evaluation for different Hit@Rank can be found in Table 7.1.

Finally, we analyzed the mean weights for different images and the distribution of mean weights together with images with lowest and highest weights can be found in Figure 7.4. The results show that the weights are centered around $1/T$ (i.e. average pooling weight) which was expected. Also, low weights are usually assigned to images with occluding objects or pedestrians.

Furthermore, we analyzed the homogeneity of the weights for individual observations (i.e. how much the weights differ within one observation). The mean relative standard deviation is 0.34; the weights therefore differ significantly.

7.3 LFTD – Experimental Results

We evaluate our method on the vehicle and person re-identification tasks on multiple public datasets to show that the aggregation performs well on various classes of data. Datasets for evaluation were chosen considering the availability of tracks (multiple observations) of

Table 7.1: Evaluation of individual parts of the proposed network on the iLIDS dataset.

Method	**Hit@R (iLIDS)**			
	1	5	10	20
IDE + avg	46.3	71.4	80.2	87.6
IDE + Project (128 feat.)	56.7	79.8	86.6	93.1
IDE + Attention (128 feat.)	51.6	77.6	86.5	91.7
IDE + LFTD (128 feat.)	61.4	79.9	87.8	93.5

Table 7.2: Results for different methods for vehicle re-identification on CarsReId74k dataset. The methods use 128 dimensional feature vectors with the exception of avg which uses 1,536 dimensional feature vectors. The methods use Euclidean distance with the exception of LFTD – M (full Mahalanobis [191]) and LFTD – WE (Weighted Euclidean as proposed in Section 7.2.3). *Input modifiers* – UNP, UNP+IM [202]. *Aggregation methods* – RNN [158], NAN [247].

			Hit@Rank			
Input Modif.	Aggregation	mAP	1	5	10	20
None	avg	0.608	55.3	66.4	71.3	76.5
UNP	avg	0.652	58.4	72.8	78.0	83.1
UNP+IM	avg	0.672	61.2	73.8	78.7	83.5
UNP+IM	RNN	0.678	59.0	78.2	84.5	89.7
UNP+IM	NAN	0.700	63.3	77.5	82.7	87.5
UNP+IM	LFTD	0.746	68.5	81.6	85.8	89.6
UNP+IM	LFTD – M	0.757	69.5	83.2	87.3	90.7
UNP+IM	LFTD – WE	**0.779**	**71.3**	**85.8**	**89.9**	**93.1**

each object's identity in the dataset because this work proposes a method for aggregation of features in the time domain and variable camera viewpoints.

7.3.1 Vehicle Re-Identification

Datasets available in 2019 does not fit conditions described before at least in one condition (see Sec. 4.1), thus vehicle re-identification task was evaluated on our novel CarsReId74k only.

For feature extraction from images we use **Inception-ResNet-v2** [216] with images resized to 331×331 yielding feature vectors with length 1,536 for each input image. [198, 202] showed that unpacking the input vehicle by 3D bounding box and alternating the input image colors is beneficial for fine-grained recognition of vehicles; we use these modifications for re-identification of vehicles as well.

The feature extractor was fine-tuned on the identification task using the training part of the CarsReId74k dataset. The fine-tuning was done with Adam optimizer, learning rate 0.0001, batch size 4 for 300 epochs with standard augmentation techniques (random flip and shift of the bounding box).

When it comes to feature aggregation in temporal domain, we compare several methods with the following naming conventions:

- **avg** – standard average pooling of feature vectors,
- **RNN** – method proposed by [158] based on recurrent neural network,

- **NAN** – **N**eural **A**ggregation **N**etwork proposed by [247],
- **LFTD** – our method (short for **L**earning **F**eatures in **T**emporal **D**omain).

To make the comparison fair, we always compare the methods with features of the same length (128 dimensional features by default). The only exception is average pooling where the final features are always 1,536 dimensional. As NAN [247] does not reduce the number of features, we added a trainable fully connected layer between the feature extractor and the aggregation network. As both RNN [158] and NAN [244] use Euclidean distance in the original design, we evaluate the networks with the Euclidean distance. Following other previous works [158, 266, 25, 243, 284], we fix the number of time samples to $T = 16$.

We also compare different metrics for comparison of the feature vectors. The standard Euclidean distance is used as the baseline. We also use the full Mahalanobis distance (as proposed by [191]) – shortened as **M**; and our Weighted Euclidean distance – shortened as **WE**. The full Mahalanobis distance was trained with regularization term $0.5\lambda\|\mathbf{W}\mathbf{W}^\top - \mathbf{I}\|_F^2$ as proposed by the authors [191] with $\lambda = 0.01$.

To increase the training speed, all the aggregation networks were trained on cached features extracted by the Inception-ResNet-v2 feature extractor. The networks were trained in Siamese settings for 30 epochs with batch size 32 on train and validation set. We employed hard negative mining during the training and all positive pairs and one hardest negative pair per positive pair were presented to the network in one epoch during the training. For the RNN [158], we used original hyperparameters as proposed in the paper (SGD, lr: 0.001, margin: 2); changing them did not improve the accuracy further. We were forced to change the hyperparemters for NAN [247] to different values than used in the paper as the network did not converge with the original ones. We used RMSprop optimizer with learning rate 1e-6, and margin 1; different hyperparameters did not improve the accuracy further. Our method LFTD was trained by Adam optimizer with learning rate 1e-5 (1e-4.4 in the case of Mahalanobis and Weighted Euclidean distance) and margin 2.

The vehicle re-identification results can be found in Table 7.2 and Cumulative Matching Curve (CMC) is shown in Figure 7.6. The results show several things. First, both the Unpack (UNP) [198] modification and image modifications (IM) [202] improve the accuracy of vehicle re-identification. Second, all feature aggregation methods in the temporal domain (RNN [158], NAN [247], LFTD) improve the accuracy when compared with the average pooling in the task of vehicle re-identification. Third, our method (LFTD) outperforms other methods for temporal aggregation (RNN [158], NAN [247]). Finally, using other metrics than Euclidean also improves the accuracy. Our proposed Weighted Euclidean distance significantly outperforms the full Mahalanobis distance (as proposed by [191]); and at the same time, our method has significantly lower time demands. It has time and memory complexity $\mathcal{O}(D)$ instead of $\mathcal{O}(D^2)$ for the full Mahalanobis distance, where D is the dimensionality of the feature vectors.

Our explanation of better performance of Weighted Euclidean distance instead of Mahalanobis distance is that there is not enough training data to train the full matrix M. This hypothesis is supported by Fig. 7.7 where the performance with Mahalanobis distance does not increase and by the fact that $\frac{\mathrm{tr}(|\mathbf{M}|)}{\sum|\mathbf{M}|} = 0.997$, i.e. almost all the information in the matrix is on its diagonal.

We were also curious how the accuracy changes with increasing the dimensionality of the feature vectors. As Figure 7.7 shows, all methods improve with increasing dimensionality; however, the results are still similar. Our method LFTD with our proposed Weighted

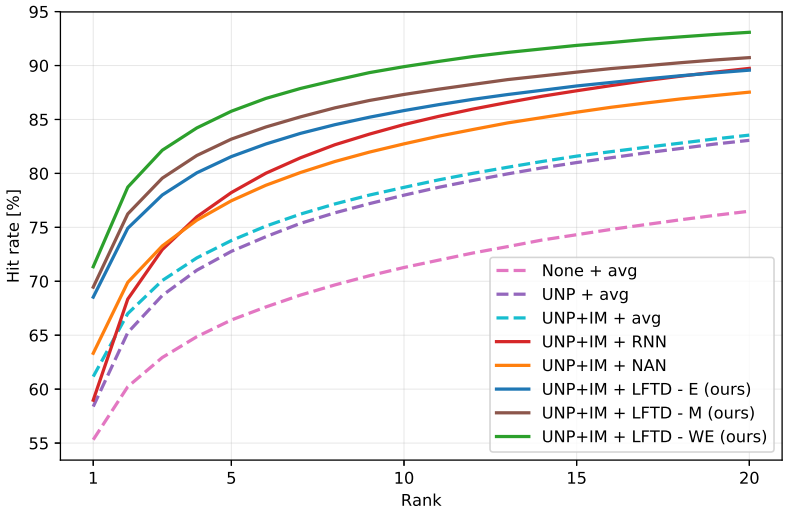


Figure 7.6: Cumulative Matching Curve for different methods for vehicle re-identification on CarsReId74k dataset. The methods use 128 dimensional feature vectors with the exception of avg which uses 1,536 dimensional feature vectors. The methods use Euclidean distance with the exception of LFTD – M (full Mahalanobis [191]) and LFTD – WE (Weighted Euclidean as proposed in Section 7.2.3).

Euclidean distance is outperforming all other methods for all of the tested feature vector dimensionalities.

7.3.2 Person Re-Identification

To show that our method is applicable also outside the scope of vehicle re-identification, we evaluated it on the person re-identification task. We use two common datasets: iLIDS-VID [234] and PRID-2011 [82] as they are usually used by other methods for feature aggregation in temporal domain [245, 158, 51, 243, 266, 25, 284]. Futrhermore, for fair comparison of proposed method, our work was also evaluated on the MARS dataset by [275].

It should be noted that the subject of our study is the aggregation of features extracted on images by an existing feature extractor. That is why we include in the comparison those methods that do the same, not methods which use a significantly different method of image **feature extraction**.

For the above reasons, we are not comparing our method with some of the published methods such as QAN [140] or SpaAtn (DRSTA) [124] as they are focusing on *spatiotemporal* attention pooling, and because of that, they provide enhanced feature extraction. In our work, we target fusion of existing feature extractors. Besides that, their evaluation is not following the standard evaluation protocol used in the previous works and with the used datasets, as they are pre-training the networks on different types of image-based person re-identification tasks, thus their results are hardly comparable.

iLIDS-VID and PRID-2011

We always used a half of the dataset for training and the other half for testing. Therefore, the evaluation is done on 100 tracks (150 tracks) with PRID-2011 (iLIDS-VID) dataset. We used 10 random splits in the case of the PRID-2011 dataset, and the 10 published splits

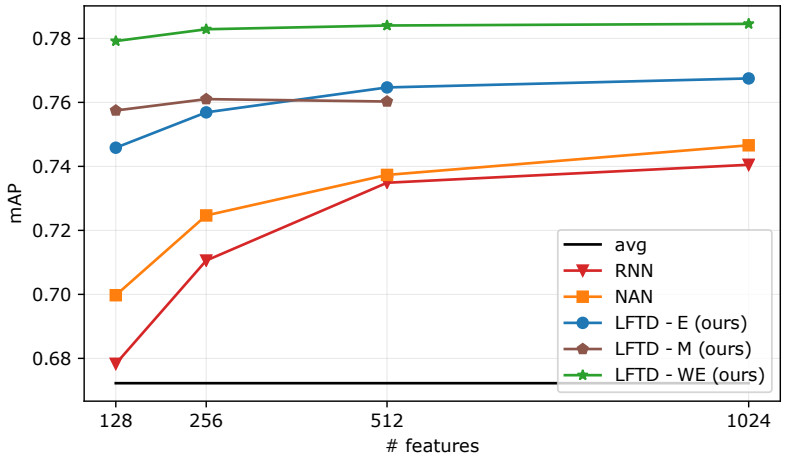


Figure 7.7: Performance analysis of different methods for feature vector aggregation in temporal domain with changing number of features on CarsReId74k dataset. The avg pooling is shown only for visual comparison and uses 1,536 dimensional feature vectors. We omitted LFTD – M with 1024 features from evaluation because of long evaluation time (months) and performance drop of version with 512 features. All the methods use Euclidean distance with the exception of LFTD – M (full Mahalanobis [191]) and LFTD – WE (Weighted Euclidean as proposed in Section 7.2.3).

in the case of iLIDS-VID. We used ResNet50 [76] as the feature extractor from the images and trained it on the identification task by Adam optimizer with learning rate 0.0001 for 60 epochs with batch size 8, using standard augmentation techniques (random flip, rotation, and shift). We trained our method (LFTD) in a Siamese network by Adam optimizer with cross-validated learning rate for 150 epochs with batch size 8. We always used 16 time samples per track and contrastive loss margin 2. We also evaluate the average pooling with KISSME [113] and XQDA [125] with cross-validated hyperparameters (regularization, and PCA reduction dimensionality in the case of KISSME).

We used standard Euclidean distance as the metric for our algorithm, because the number of training data in the datasets is rather low and the accuracy did not improve further with other distances. This is caused mainly by insufficient amount of training data because the network was able to re-identify the training tracks without any error already with the standard Euclidean distance. The results can be found in Table 7.3 and as the table shows, LFTD significantly increases the performance compared to average pooling or average pooling with other metric learning (KISSME, XQDA). The results also show that our method outperforms other methods for feature aggregation in temporal domain [245, 158, 51, 243, 266, 25, 284] in Hit@1.

Evaluation of KISSME or XQDA metrics together with the features produced by the proposed LFTD method is not included in the results because of lacking relevancy of such comparison. LFTD produces features dependent on the metric used during training and it generates different feature vector representations for different metrics involved (Euclidean, Weighted Euclidean, Mahalanobis).

Table 7.3: Person re-identification results on PRID-2011 and iLIDS-VID dataset. The top-3 results are highlighted in the following way: **first**, **second**, and **third**. KISSME – [113], XQDA – [125]. LFTD metric used for this experiment is the standard Euclidean distance because of insufficient amount of training data for the Weighted Euclidean.

Method	Hit@R (PRID)				Hit@R (iLIDS)			
	1	5	10	20	1	5	10	20
Yan *et al.*, 2016 [245]	58.2	85.8	93.4	97.9	49.3	76.8	85.3	90.0
McLaughlin *et al.*, 2016 [158]	70.0	90.0	95.0	97.0	58.0	84.0	91.0	96.0
Gao *et al.*, 2016 [51]	68.6	**94.6**	**97.4**	98.9	55.0	**87.5**	**93.8**	**97.2**
Xu *et al.*, 2017 [243]	77.0	**95.0**	**99.0**	**99.0**	62.0	**86.0**	**94.0**	**98.0**
Zhang *et al.*, 2017 [266]	72.8	92.0	95.1	97.6	55.3	85.0	**91.7**	95.1
Chen *et al.*, 2017 [25]	77.0	93.0	95.0	98.0	61.0	85.0	**94.0**	**97.0**
Zhou *et al.*, 2017 [284]	**79.4**	**94.4**	—	**99.3**	55.2	**86.5**	—	**97.0**
Zhang *et al.*, 2017 [267]	60.2	85.1	—	94.2	83.3	93.3	—	96.7
avg	69.4	90.5	95.0	97.6	46.3	71.4	80.2	87.6
avg + KISSME	70.5	91.0	95.1	97.7	56.1	79.0	87.9	93.9
avg + XQDA	75.6	94.3	**98.2**	**99.0**	59.5	83.7	90.3	96.2
LFTD (128)	79.2	92.4	95.8	98.4	61.4	79.9	87.8	93.5
LFTD (256)	**79.4**	93.7	96.8	98.6	**62.8**	82.1	88.1	94.1
LFTD (512)	**80.2**	**94.6**	97.3	98.9	**63.5**	83.3	89.5	94.9
LFTD (1024)	**80.0**	93.9	**97.4**	**99.2**	**63.7**	82.9	90.0	94.7

MARS dataset

Features published by the authors of the dataset were used in our experiments. The network was trained on the training part of the published features in the Siamese setting for 30 epochs with batch size 32 with Contrastive Loss and Adam optimizer. Hard negative mining was employed during the training, and all positive pairs and 20 hardest negative pairs were presented to the network in one epoch during the training. Values of learning rate and loss margin were fine-tuned for each variant individually. All variants of our method evaluated on the dataset can be found in Table 7.4. It should be noted that *Baseline* is the variant (*IDE, average pooling, Euclidean distance, single query*) reported by the authors [275].

7.4 Feature Aggregation in Temporal Domain – Summary

We proposed a new scheme for extracting feature vectors for the whole tracks of multiple observations of an object (vehicle, person) of interest in the re-identification task. Our method can work with arbitrary per-image features (e.g. feature vectors from ResNet50 or Inception-ResNet-v2). Based on such feature vectors we learn a considerably shorter (128 features) per-track feature vector by using the newly proposed LFTD (Learning Features in Temporal Domain). We also propose to use a different distance metric for comparing the feature vectors – **WE** (Weighted Euclidean). It is based on the Mahalanobis distance, whose learned matrix $\mathbf{M}$ is made diagonal. This proposed distance metric is much cheaper in terms of computational and memory resources ($\mathcal{O}(D)$ instead of $\mathcal{O}(D^2)$ in the case of the full Mahalanobis metric), but at the same time, it is better at solving the re-identification task. The results show that the increase of HIT@1 by using the LFTD was 7.3 percentage points for the vehicle re-identification task compared to average pooling, and 17.4 percentage points for the person re-identification with the iLIDS-VID dataset and up to 6.4 percentage points on the MARS dataset. The Weighted Euclidean metric further increased HIT@1 by

Table 7.4: Person re-identification results on MARS dataset. Baseline is the variant (*IDE, average pooling, Euclidean distance, single query*) reported by authors of the dataset [275]. * - RNN-CNN [158] trained by [243].

		Hit@Rank			
Variant	mAP	1	5	10	20
Baseline	0.424	60.0	77.9	-	87.9
RNN-CNN* [243]	-	40.0	64.0	70.0	77.0
ASTPN [243]	-	44.0	70.0	74.0	81.0
[267]	-	55.5	70.2	-	80.2
LFTD - E (512)	0.481	65.5	80.3	85.5	89.4
LFTD - E (1024)	0.483	65.9	80.7	84.8	89.2
LFTD - WE (512)	0.488	66.1	81.0	85.4	89.8
LFTD - WE (1024)	0.489	66.4	81.5	85.9	89.8

other 2.8 percentage points in case of vehicle re-identification. We collected and annotated a vehicle re-identification dataset CarsReId74k for development and evaluation of vehicle re-identification systems and we make it public. It contains 17,681 unique vehicles, 73,976 observed tracks, and 277,236 positive pairs, taken from various angles – not just from the front or rear.

Part IV

Improvements in License Plate Recognition

A key component of traffic analysis, law enforcement, and surveillance systems is licence plate recognition (LPR). However, the accurate recognition of license plates remains a challenging task, particularly when dealing with low-quality, degraded and distorted license plate images. Furthermore, the precise geometric alignment of license plates is essential for improving the overall recognition accuracy. In this thesis, we present two significant contributions that address these challenges: the proposed methods for holistic recognition of low-quality license plates (Chapter 8 on the following page) and geometric alignment of license plates using deep learning techniques (Chapter 9 on page 83).

The recognition of low-quality license plates is a crucial task for ensuring the robustness and effectiveness of license plate recognition systems in real-world scenarios. Traditional approaches often struggle with low-quality images that suffer from issues such as blur, noise, and poor lighting conditions. To overcome these challenges, we propose a novel deep learning-based method for holistic recognition of low-quality license plates. Our method leverages the power of convolutional neural networks (CNNs) to learn discriminative features directly from the license plate images, enabling accurate recognition even in challenging conditions. By considering the holistic representation of the license plates, our method exhibits superior performance compared to traditional methods, offering a promising solution for improving the overall recognition accuracy in real-world scenarios. This work was published in paper *Špaňhel* et al*., Holistic Recognition of Low Quality License Plates by CNN using Track Annotated Data* [208]. This conference article is included in Chapter 8 on the following page.

In addition to low-quality license plate recognition, the geometric alignment of license plates is a critical pre-processing step that significantly impacts the subsequent recognition performance. Misalignment of license plates due to varying camera perspectives, vehicle poses, and other factors can introduce challenges such as occlusions, distortions, and skewed characters, leading to decreased recognition accuracy. To address this issue, in paper *Špaňhel* et al*., Geometric Alignment by Deep Learning for Recognition of Challenging License Plates* [207], we propose a deep learning-based method for the geometric alignment of license plates. By exploiting the power of convolutional neural networks, our method learns to detect and correct the misalignment in license plate images, ensuring that the license plates are correctly oriented and aligned for accurate recognition. Integrating this geometric alignment step into the license plate recognition pipeline enhances the overall system performance, particularly in scenarios where license plates exhibit significant variations in alignment. The content of this article is available in Chapter 9 on page 83.

In conclusion, the contributions made by these two papers are significant in the field of license plate recognition. The proposed method for holistic recognition of low-quality license plates offers a robust solution to the challenges associated with poor image quality, enhancing the accuracy and reliability of license plate recognition systems in real-world scenarios. Additionally, the deep learning-based method for the geometric alignment of license plates addresses the crucial issue of misalignment, ensuring accurate recognition by correcting misaligned license plate images. By integrating these methods into license plate recognition systems, the overall performance is greatly improved, making them more effective in scenarios with varying image quality and alignment. These contributions have the potential to advance the field of license plate recognition and contribute to the development of more accurate, robust, and efficient systems.

Chapter 8

Holistic Recognition of Low Quality License Plates by CNN using Track Annotated Data

Abstract This work is focused on recognition of license plates in low resolution and low quality images. We present a methodology for collection of real world (non-synthetic) dataset of low quality license plate images with ground truth transcriptions. Our approach to the license plate recognition is based on a Convolutional Neural Network which holistically processes the whole image, avoiding segmentation of the license plate characters. Evaluation results on multiple datasets show that our method significantly outperforms other free and commercial solutions to license plate recognition on the low quality data. To enable further research of low quality license plate recognition, we make the datasets publicly available.

8.1 Low-quality License Plate Recognition

Automatic license plate recognition (ALPR) is a common task nowadays and it is used by many applications of intelligent transportation systems for security and traffic control. Traffic flow analysis, automatic vehicle speed measurement, parking violation enforcement are only a few examples where ALPR is commonly used. A large number of previous approaches to ALPR have been proposed [2, 23, 56, 95, 121, 178, 237, 276]. Most of the existing algorithms (and commercial solutions) use character segmentation as one step in their algorithm. However, proper character segmentation has a significant influence on the recognition rate of such an ALPR system. If the segmentation is improper, the license plate will be recognized incorrectly even if the recognizer itself is robust and it can deal with different characters font, rotation and size. Other problems that affect the character segmentation process and that are hard to handle include image blur, uneven lighting, shadows and noise.

This paper focuses on an alternative approach to the license plate recognition in a holistic, segmentation-free way. The results confirm that this approach is applicable even under challenging conditions (e.g. low-quality images, blurred images, uneven lighting, image noise), where proper character segmentation is nearly impossible. The novel user-annotated dataset was collected for the purpose of this work and will be publicly available for non-

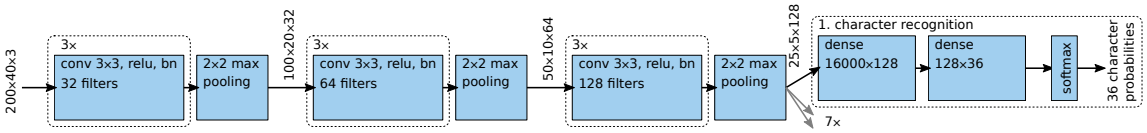


Figure 8.1: Schematic figure of used Convolutional Neural Network. The network contains three sequences of three convolutional layers with ReLU and Batch Normalization. The features are then fed into 8 branches of the net with fully connected layers predicting characters on respective positions.

commercial use[1]. Images in the dataset are annotated by whole license plate tracks (sequences of observations of a single vehicle), therefore it provides real ground truth labels even for naturally blurred, partially occluded, and hardly readable license plate images, which allows to train a very robust license plate recognizer. An example of such a license plate track can be seen in Figure 3.1 on page 35.

A single convolutional neural network (CNN) is used in the proposed method for holistic license plate recognition. Unlike [23, 121], our method handles character localization and recognition by a single CNN, without the need of additional recurrent neural network for correct data labeling. In our case, the model is trained on European license plates only and it is evaluated on the same types of license plates. It is safe to assume that the proposed approach will be applicable to different styles of license plates (e.g. US, China, etc.) by providing a sufficient amount of training data.

8.2 Holistic-CNN – Methodology

We use Convolutional Neural Networks for recognition of the license plate text. In order to avoid using character segmentation, the whole RGB image of the license plate is fed into the net. The net is designed to predict 8 characters of the license plate. The image is processed by a convolutional part of the CNN and then 8 branches of fully connected layers predict 8 characters while each branch always predicts one character on the same position in the license plate text. Scheme of the CNN is shown in Figure 8.1. As it can be seen in Figure 8.1, the convolutional part contains three sequences of three convolutional layers with ReLU and Batch Normalization.

We also tested a **smaller network** which contains three sequences of **two** convolutional (instead of three) layers with a smaller number of filters (16/32/64 instead of 32/64/128).

The smaller network contains $\sim$ 8M parameters while the bigger one has $\sim$ 17M parameters. The results in Section 8.3 show that the smaller CNN achieves similar accuracy as the deeper one. However, the smaller network is able to process an image approximately three times faster.

License plates of variable lengths. One of the challenging tasks in segmentation-free license plate recognition are license plates with variable length (frequently, but not exclusively license plates from different countries). The method proposed in this work is trained for $n_{\max}$ number of outputs, which equals to license plate with maximal length in our ReId dataset (8 characters in our case). In the case that license plate has lower than $n_{\max}$ number of characters, than $n_{\text{add}} = n_{\max} - n_{\text{lp}}$ blank fill characters ("#") is added to

[1] https://medusa.fit.vutbr.cz/traffic/

Figure 8.2: Attention of fully connected layers for different letters in the license plate. Left to right, top to bottom: 1st to 8th character. The 4th letter in most cases contains the blank fill character.

the license plate text during the training phase, where n_{lp} is the length of current license plate. This blind text of n_{add} characters is inserted to the ground truth license plate text before last 4 characters. Location of this blind text outcomes from the most common layout of license plates in our country. Examples of license plates with various character layouts can be found in Figure 3.3.

Training details. Both variants of the CNNs were trained using Tensorflow with Adam optimizer. Initial learning rate was set to 0.001 and the nets were trained for 80 epochs. We also tested larger number of features in fully connected layers, however the accuracy did not improve further.

Attention span of fully connected layers. During experimenting with the learned CNN, we visualized the locations where the individual fully connected parts of the CNN "look" for their data. Figure 8.2 shows images with mean of weights in the first fully connected layer corresponding to different spatial locations. The mean is done over all 128 channels and all 128 outputs of the fully connected layer. The figure shows that the network properly learned to use correct spatial locations for different letters in the license plates. The 4th letter does not have a clear blob with a distinct maximum, as majority of the license plates in the datasets (Sec. 8.3) contain 7 characters, and the 4th letter was designated as the blank fill character.

8.3 Experiments with Holistic LPR

The proposed method was evaluated on different datasets together with two other methods for appropriate evaluation. The proposed method was trained on the training part of the ReId dataset only and evaluated on the test part the of ReId dataset. We also evaluated whether it is possible to use the trained model with license plates of different type from other datasets to analyze transferability of the model.

8.3.1 Existing solutions

OpenALPR. OpenALPR[2] is an open source library for Automatic License Plate Recognition written in C++. This software is based on OpenCV[3] computer vision library and Tesseract OCR[4] and it is a classical representative of a method based on character segmentation.

[2] https://github.com/openalpr/openalpr
[3] http://www.opencv.org
[4] https://github.com/tesseract-ocr/tesseract

Table 8.1: Evaluation of different license plate recognition systems on ReId, HDR, and Svoboda *et al.* [215] (blurred + deblurred) datasets. The datasets were evaluated as character/license plate error rate for Top 1 results including processing time of most of the methods. The processing speed was measured on a PC with i5-6500@3.2GHz and GTX 1080 and PC with i7-3770@3.4GHz in the case of UnicamLPR.

	error rate [%] (character/license plate)				**speed [ms]**	
system	ReId	HDR	[215]	deblur. [215]	CPU	GPU
OpenALPR	41.9/66.0	28.6/60.0	93.1/99.0	12.3/30.4	22.64	—
UnicamLPR	19.8/30.0	65.0/68.1	38.8/61.7	12.9/16.7	13.70	—
ours	**0.4**/**1.4**	**3.5**/**9.7**	**16.4**/**44.6**	**2.7**/**9.0**	36.64	0.82
ours (small)	**0.4**/1.7	4.5/12.1	20.0/54.1	3.1/11.3	**9.59**	**0.31**

In our experiments, we use the original settings for EU license plates. Detection phase is skipped and license plate recognition is evaluated on already cropped images, where top 10 recognized license plates are obtained for each input image for better comparison.

UnicamLPR. Similarly to the evaluation by Svoboda *et al.* [215], a commercial license plate OCR was also evaluated on testing datasets. The UnicamLPR[5] is a software for detection and recognition of license plates optimized for real-time and low latency processing of license plates captured by traffic surveillance cameras. The software is robust and allows an angle between the camera and the license plate up to ±30°. It should be noted that the UnicamLPR is designed for license plates of standard image quality.

8.3.2 Evaluation Datasets

ReId. The testing part of our ReId dataset (described in Section 3.1 on page 35) was used in the experiments. It contains 76,412 color license plate images of different lengths, image blur and slight occlusion. The dataset samples are shown in Figure 3.3 on page 36.

HDR. The HDR dataset (described in Section 3.1 on page 35) used in evaluation was captured by DSLR camera with three different exposures, from which *medium* and *high* exposure images were used in experiments. License plates were hand-cropped from images and annotated by users resulting in 652 images. Examples from the dataset can be found in Figure 3.4 on page 37. It should be noted that the images contain rotated license plates which are of different type than the ones used for training.

Svoboda et al [215]. The dataset used by Svoboda *et al.* [215] was captured by surveillance cameras in a production-use traffic monitoring system and were set to capture license plates with motion blur. The task of Svoboda *et al.* [215] was to deblur these license plates by CNN. License plates were labeled with their ground truth texts by humans from deblurred images. We evaluate the method on two sets from this dataset. The first one contain the original blurred images, and the second one contain images deblurred by the authors of the deblurring method [215]. Each set consist of 711 greyscale images with IR flashlight. Samples from both parts of the dataset are shown in Figure 3.4 on page 37. It

[5] http://www.camea.cz

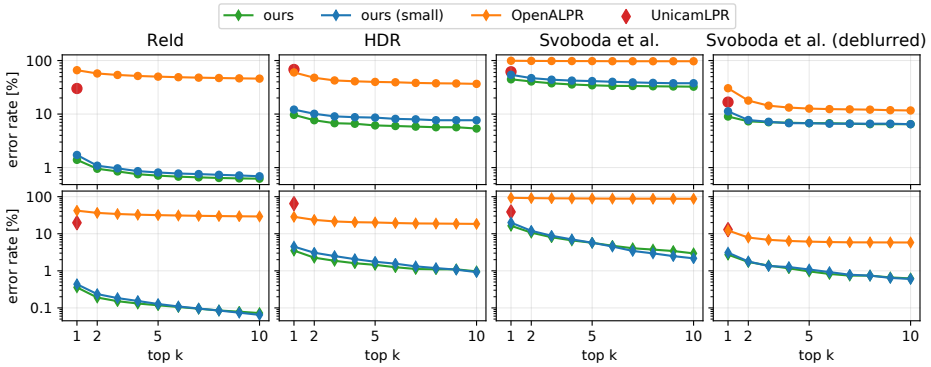


Figure 8.3: Error rates for different license plate recognition methods on different datasets. The **top** row shows the error rates for full license plate recognition and the **bottom** row shows the error rates for single character recognition. The figures show error rates for different number of most probable license plates. The results show that our method outperforms both OpenALPR and commercial solution UnicamLPR used by Svoboda *et al.* [215]. Also, a small variant of our CNN achieves a similar accuracy as the larger used CNN.

Figure 8.4: Examples of correctly (**top row**) and incorrectly (**bottom row**) recognized license plates from ReId (**left part**) and HDR (**right part**) datasets with top-2 recognition results for each image. It can be seen that license plates are very challenging and almost unreadable for human in some cases.

should be noted that those license plate images which were unreadable from the deblurred version by humans were removed from both parts of the dataset to provide reliable ground truth labels, so the results are slightly different from those that were published before.

8.3.3 Results

As it can be seen from Table 8.1 and Figure 8.3, the proposed CNNs outperform other evaluated solutions on all used datasets. The results show slight increase of error rates on HDR and Svoboda *et al.* [215] datasets, but it should be noted that our method was trained only on the ReId dataset, therefore the error rates on the datasets show transferability of the trained CNNs to other datasets with different types of images (e.g. long shutter time, greyscale, rotated license plates). Figure 8.3 also shows the progress of error rates of the proposed method and OpenALPR from Top-1 to Top-10 accuracy results.

The results also show that a smaller version of the network (see Section 8.2 for details) has competitive results with the full network, while the license plate (LP) processing time is significantly lower (especially on CPU). When considering the processing speed of the full pipeline on GTX 1080 (LP detection and recognition) it can fit within 20 ms per Full-HD frame. We consider 5 ms for detection, 10 LPs per image for recognition (8 ms), and processing overhead.

Examples of correctly and incorrectly recognized LPs from ReId and HDR datasets are shown in Figure 8.4. The *top* row contains correctly recognized LPs and the *bottom* row contains incorrectly recognized LPs. Both rows are shown with the best scoring two results for each plate.

8.4 Holistic LPR – Summary

This paper presents a holistic license plate recognition method using the promising CNN technique. The last convolutional layer of used CNNs is connected to 8 branches of fully connected layers, each with 36 outputs, for predicting characters at the respective positions in the image. Each network is trained to localize and recognize characters in the image at different locations, which are learned automatically. Evaluation of the proposed method proves that the CNN can localize the position of characters automatically and it can learn such distinctive features that are robust to various illumination, rotation, occlusion, and image blur.

The results show that the proposed networks significantly outperform existing open-source and commercial solutions on numerous datasets, while the processing speed is comparable when used on CPUs and our solution is easy to use on GPUs.

Chapter 9

Geometric Alignment by Deep Learning for Recognition of Challenging License Plates

Abstract In this paper, we explore the problem of license plate recognition in-the-wild (in the meaning of capturing data in unconstrained conditions, taken from arbitrary viewpoints and distances). We propose a method for automatic license plate recognition in-the-wild based on a geometric alignment of license plates as a preceding step for holistic license plate recognition. The alignment is done by a Convolutional Neural Network that estimates control points for rectifying the image and the following rectification step is formulated so that the whole alignment and recognition process can be assembled into one computational graph of a contemporary neural network framework, such as Tensorflow. The experiments show that the use of the aligner helps the recognition considerably: the error rate dropped from 9.6 % to 2.1 % on real-life images of license plates. The experiments also show that the solution is fast – it is capable of real-time processing even on an embedded and low-power platform (Jetson TX2). We collected and annotated a dataset of license plates called *CamCar6k*, containing 6,064 images with annotated corner points and ground truth texts. We make this dataset publicly available.

9.1 License Plate Recognition in Unconstrained Environment

Automatic License Plate Recognition (ALPR) is the backbone for many applications in traffic surveillance and intelligent transportation systems (automatic parking systems, security surveillance systems, toll gates, etc.). In many such applications, the cameras are fixed and positioned so that the license plates share a common size (image resolution), orientation and they are not skewed. In such scenarios, the existing recognizers of license plates achieve almost perfect results. However, mobile monitoring platforms are used more frequently for parking enforcement and other applications, and also the availability and properties of PTZ (pan-tilt-zoom) cameras offer for much less restricted scenarios. Existing solutions of automatic license plate recognition (an overview is available in Section 2.8) are not designed for these unconstrained cases and tend to achieve poor results.

Our paper described in previous chapter proved that holistic (i.e. refraining from segmenting the characters) recognition outperforms other available recognition methods. However, our previous work does not deal with challenging license plates captured from different

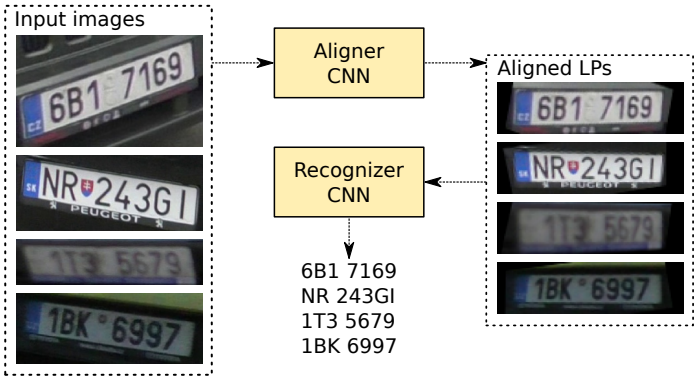


Figure 9.1: Overview of our approach. The input license plates of various resolutions, padding, rotation, and skew are analyzed by the aligner CNN, rectified and processed by a holistic license plate recognizer. The whole process can be incorporated into one computational graph and computed as a whole, for example in a GPU.

viewpoints. In this work, we propose a solution how to overcome that limitation. We designed a new convolutional neural network (CNN) whose purpose is to predict four corner points of the license plate in the unaligned image. These points define the transformation which rectifies the image for subsequent processing by the holistic license plate recognizer. Although the neural networks are trained separately, they can be assembled into one computational graph so that the solution works as a whole, its use is no more complicated than using only the recognizer. The proposed framework can be seen in Figure 9.1.

We carried out experimental evaluation of the proposed approach. The proposed aligner network decreases the recognition error on real-life license plates considerably. We evaluated the sensitivity of the newly proposed pipeline to various distortions of the input images: rotation, skew, blur, noise, etc. Again, the aligner network helps the stability notably. We also evaluated the speed of the proposed recognition pipeline. Thanks to the fact that it can be assembled with the recognizer into one computational graph, it can be efficiently (in real time) executed even on low-power and embedded devices. All these findings make the proposed approach interesting in applications of traffic enforcement and intelligent transportation systems.

The contributions of this paper are the following:

- A method for aligning license plates before recognition based on CNN.
- Novel **CamCar6k** dataset of in-the-wild license plates.
- Detailed evaluation of aligner/recognizer performance influenced by different distortions.

9.2 Aligner-CNN – Methodology

This section presents our processing pipeline, mostly the newly added aligner CNN and the way the whole system is assembled and learned. It should be noted that this work is not focused on license plate detection as we assume that the license plates are detected by an existing technique in software [2, 56, 86, 121, 178], hardware [259], or GPUs [81]. The license plate detector does not need to be very sophisticated and the system can

Figure 9.2: The whole license plate processing pipeline. The aligner outputs four heatmaps based on which the LP's corner points are found and the license plate is transformed/aligned. Then, the existing state-of-the-art recognizer processes it.

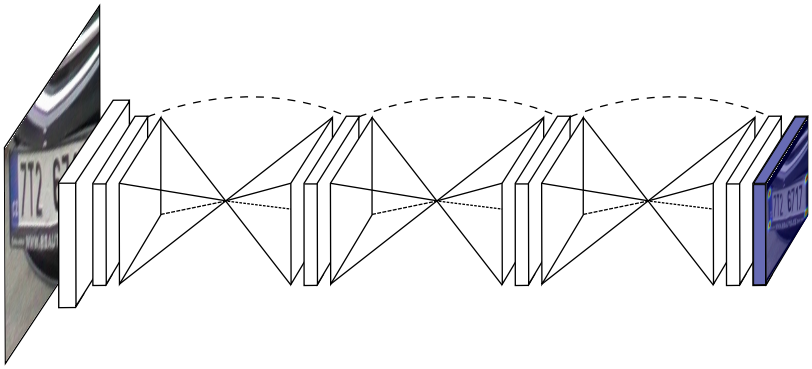


Figure 9.3: Hourglass architecture of the proposed Aligner CNN for estimation of the corner points. It consists of three stacked hourglass modules, allowing repetitive top-down, bottom-up inference with skip connections. The Aligner CNN estimates the probability map for each corner point independently, in the case of a missing corner points, the probability map is (close to) zero over all its surface.

tolerate its relatively high false positive rate, since the candidate locations are verified by the aligner/recognizer presented in this work.

9.2.1 Hourglass Network for Keypoint Detection

The proposed Aligner CNN is shown in Figure 9.3. It is based on the stacked hourglass neural network as designed by Newell *et al.* [162]. The Aligner CNN is a fully convolutional network which for each image point evaluates the probability that the point contains one of the four corner points. The network contains three hourglass modules which downsample and upsample the features in the spatial dimension. For better gradient flow, the network contains skip connections and an additional output layer with MSE or Binary Cross-Entropy losses after each hourglass module. The hourglass design allows the network to process and consolidate features across different scales. It has the capacity to capture all of these features together and create per-pixel predictions at the output. For further details about the model, we refer readers to the original paper [162].

The network is trained using randomly rotated images of license plates and ground truth probability maps for each corner point are used as the supervision. To reduce the computational complexity, we used only three hourglass modules and feature size 64.

Figure 9.4: Examples of the alignment process. From left to right: input image, merged predictions, detected keypoints, aligned license plates.

9.2.2 License Plate Processing Pipeline

The hourglass network estimates the locations of the inner corner points of the license plate. Its outputs are four probability maps for the four corner points in a specified order (top left, top right, bottom right, bottom left). If a keypoint is not present in the input image (occlusion, damage, image does not contain a license plate), the peak is missing in the appropriate heatmap. An example of the predicted heatmaps merged into one can be found in Figure 9.2 middle (*merged predictions*) and in Figure 9.4.

The maximum of each predicted heatmap is used as the detected keypoint, shown in Figure 9.2 (*detected keypoints*). The detected keypoints are then used to normalize the license plate using homography $\mathbf{H}$ between the estimated points and predefined axis-aligned corner points.

Finally, aligned license plates depicted in Figure 9.2 (*aligned license plate*) are passed to the recognizer network to provide the final recognition.

9.2.3 Recognizer Network

Recognition of the aligned license plates is based on a CNN proposed by Špaňhel *et al.* [208] (described in previous Chapter 8). The proposed network processes the whole image at once without character segmentation. During training, the CNN learns the presumed location for each output layer (each character of the LP) from the training data. The network is composed of three blocks sharing the same structure: three identical subblocks containing the convolutions and nonlinearity (3×3 convolution layer + ReLU + Batch normalization) followed by one 2×2 max pooling. At the end, the network predicts eight different characters independently.

During inference for *aligner+recognizer* network, both variants are evaluated (even the variant without the aligner) and the result with higher confidence is selected as the final output. It results in an increase of recognition accuracy; however, the processing speed decreases, which can be seen in Section 9.3, Table 9.2.

Table 9.1: Accuracy of license plate recognition with different versions of aligner (rows)/recognizer (columns).

	RECOGNIZER		
ALIGNER	Real	Real+Synth	OpenALPR
None	87.5	90.4	69.4
Real	96.0	97.9	73.6
Real+Synth	95.8	97.7	71.4

9.3 Experiments with Aligned License Plates

In our experiments, we evaluated different combinations of the aligner and the recognizer networks. We trained two variants of the **aligner** network, different in the datasets used for training. The **Real** variant is trained on the *training* split of the *CamCar6k* dataset only because this real-world dataset contains the annotations of the LP corners. The **Real+Synth** option expands the training set by adding the synthetic data in the training phase. The networks were trained with learning rate 2.5e-4, batch size 16, and input resolution 128×128. The output probability maps have shape 32×32×4 as we are estimating 4 corner points. The network was shown 5 millions randomly transformed samples during the training.

On the other hand, the **recognizer** network trained on the real-world data (denoted by **Real**) is trained using all data from *ReId* and *HDR* and the *training* part of the *CamCar6k* dataset because the corner point annotations are not necessary. The second variant **Real+Synth** expands the training set by synthetically generated images just like in the case of the aligner network. The networks were trained with learning rate 0.01 and Adam optimizer for 20 epochs. The input image shape is 128×35. During the training, the input license plates were randomly rotated in range $\pm 15°$, shifted, and resized. These different variants are denoted in the form of **ALIGNER / RECOGNIZER** throughout the whole paper.

For evaluating the recognition accuracy and the processing speed, only the *test* split of the *CamCar6k* dataset was used (depicted in Section 3.2 on page 36). The evaluation of each variant of the aligner/recognizer networks can be found in Table 9.1. The results show that using the aligner CNN considerably improves the recognition rate on the in-the-wild data: error rate reduced from 12.5 % to 4.0 %. Extending the real-world dataset by synthetic data did not improve the aligner's performance (it even slightly degraded). On the other hand, using the synthetic data when learning the recognizer network helped visibly: error rate reduced from 4.0 % to 2.1 %. Two publicly available license plate recognizers were also evaluated. OpenALPR[1] which is the most famous open-source solution for license plate recognition and license plate recognition Sighthound Cloud API[2]. Both solutions suffer from being based on character segmentation (Section 2.8). Interestingly, Sighthound was able to detect at least some characters only on 1 % of license plates in the dataset so we omitted it from Table 9.1.

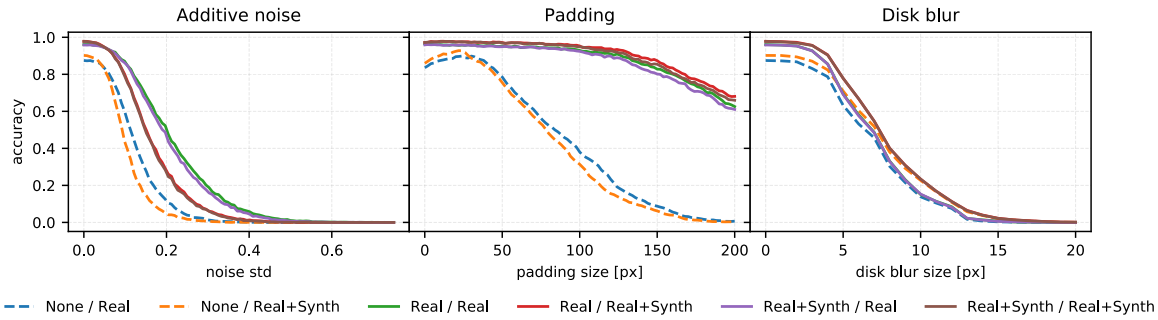


Figure 9.5: Three different distortions added to the real-world images (testing part of the *CamCar6k* dataset). In all cases, the aligner improves the recognition performance considerably. In the case of added padding (the LP does not cover the whole region of interest, but a margin is added around), the recognizer helps tremendously – allowing for a high tolerance for the detector of the LPs.

9.3.1 Robustness of the Proposed Method

Robustness of the method against various distortions and influences is crucial for license plate recognition in the wild, so we made experiments with different types of distortions.

All the distortions were applied on the original image data (testing part of the *CamCar6k* dataset). The detailed results of these tests are shown in Figure 9.5 from left to right, respectively.

Additive noise Noise added to the source image simulates varying quality of the used camera or its setting. Use of the aligner keeps the performance constantly better than without it and the drop with the increasing noise comes later. Interestingly, recognizer trained on purely real data does not drop its performance as quickly as the one trained also with synthetic data (though the second one performs slightly better when the noise is not added).

Padding As expected, the aligner detects the extra padding around the license plate and the recognizer then can focus on the license plate instead of on its surroundings. The performance with the aligner remains almost unchanged and it starts to fall only with such a large padding which was not covered in the training data.

Out of focus blur The recognition seems to be equally sensitive to disc blurring regardless of the use of the aligner.

The following distortions have two parameters, so they are visualized by using 2D graphs in Figure 9.6.

Rotation/Shear The purpose of the aligner is to compensate for random rotations and skewed license plates. The test verifies that the aligner is very successful in this task and broadens the tolerance to these distortions greatly.

[1]OpenALPR – https://www.openalpr.com
[2]Sighthound Cloud API – https://www.sighthound.com/products/cloud

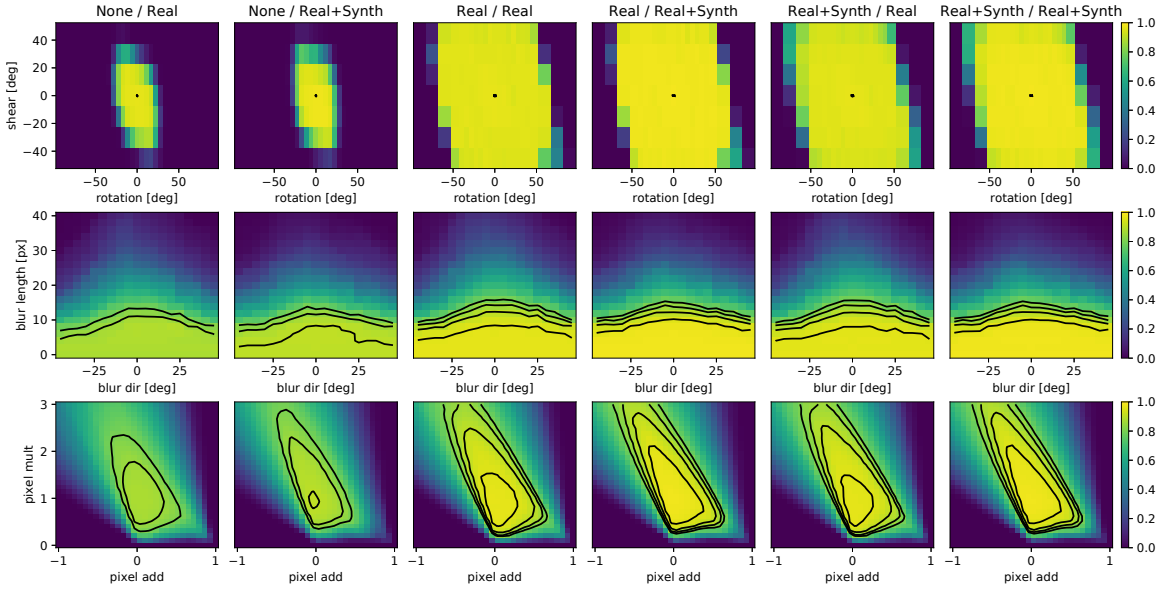


Figure 9.6: Three different distortions added to the real-world images (testing part of the *CamCar6k* dataset), top to bottom: rotation vs. shear, simulated motion blur (direction + length), brightness / contrast adjustment. The versions with the aligner at work (columns 4 to 6) are clearly superior in all cases. Also, it should be noted that the Aligner CNN was trained for rotations ±45°.

Table 9.2: Processing speed of recognizer on different platforms.

		Recognizer		**Aligner+Recognizer**	
Platform	**Model**	ms	FPS	ms	FPS
CPU	i5-6500	7.944	125.9	25.850	38.7
GPU	GTX 1080	0.686	1,457.3	1.877	532.7
SoC	Jetson TX2	4.456	224.4	16.466	60.7

Motion blur The motion blur (important factor in surveillance and monitoring of moving vehicles and/or from a moving vehicle) is simulated by applying blur of given length (in pixels) in a given direction (in degrees). Use of the aligner improves the results also in this case.

Brightness/Contrast changes The last test studies how the performance is influenced by brightness/contrast changes, simulated by adding and multiplying the individual pixel values.

9.3.2 Processing Speed

The processing speed of the recognizer and the combination of the aligner plus the recognizer was evaluated on different platforms: on CPU, GPU, and on an embedded device (SoC). The results are presented in Table 9.2. The results show that the solution is able to greatly benefit from using contemporary GPUs – the coupled aligner with recognizer can process over 500 LPs per second. Even an embedded system, represented by NVIDIA Jetson TX2

platform (Tegra X2 chip, Pascal architecture, 256 CUDA cores), is capable of processing at least 60 LPs per second, with power consumption of only 7.5 Watts. This performance is sufficient for processing data from a connected camera in real time and it is suitable for usage in real-world applications.

9.4 License Plate Alignment – Summary

In this paper, we are dealing with the problem of recognizing license plates in the wild – rotated, skewed, blurred, noised, etc. Recognition of such license plate images is crucial for many applications from the domain of traffic enforcement and intelligent transportation systems.

We propose to precede the LP recognizer by an aligner having the Hourglass CNN architecture. The experimental results show that harnessing the aligner helped considerably: the error rate dropped from 9.6 % without it to 2.1 % on a real-world dataset captured by cameras mounted on a vehicle. The aligner is designed so that it can be assembled with the recognizer into one computational graph used by contemporary neural network platforms. This greatly helps the efficiency and the whole solution is able to work in real time even on low-power and embedded architectures (represented by Jetson TX2 in the experiments). The speed of processing 60.7 LPs per second on an embedded device outperforms current solutions significantly.

Along with this research, we collected a dataset of 6,064 in-the-wild license plates and annotated their ground truth texts and four inner corner points. We made the dataset **CamCar6k** publicly available for non-commercial use.

Part V

Application of Presented Research in the Real World

The previous parts of this thesis described advances in various areas of traffic analysis, the contributions of different methods and published datasets, described in previous Chapters 5, 6, 7, 8, 9. This part briefly overviews previously described methods within the context of NVIDIA AI City Challenges[3] or contractual research.

General Traffic Analysis (Challenges) NVIDIA AI City Challenge started in the year 2017. As the challenge evolves, it attracts more researchers' attention and allows them to evaluate proposed solutions against top methods/solutions that are fine-tuned for each challenge track. In traffic analysis tracks, we have tried to avoid such fine-tuning as much as possible to test the ability of our proposed methods across multiple challenge tracks *(Speed Measurement, Vehicle Re-Identification, Multi-Target Multi-Camera Vehicle Tracking, Vehicle Trajectory Analysis)*. Chapter 10 contains our a brief overview of our submissions to this challenges [203, 205, 50]

Analysis of Vehicle Trajectories (Contractual research) This contractual research aimed to analyze the behavior of vehicles on third-grade roads with and without horizontal lane markings with slight curvature ($R \leq 200m$). The roads are not frequented by many vehicles, so a general short-term study would not provide enough data. We used recording devices for long-term (weeks) traffic recording and designed a system for analyzing the trajectories of the vehicles employing computer vision, based on *camera calibration, precise LP detection and tracking* We collected a dataset at 6 distinct locations, containing 1 010 hours of daytime video. In this dataset, we tracked over 12 000 cars and analyzed their trajectories. The results show that the selected approach is functional and provides information that would be hard to mine otherwise. After applying the horizontal markings, the drivers slowed down and shifted slightly toward the outer side of the curvature. This research study was conducted for Transport Research Center *(Cetrum Dopravního Výzkum*[4]*)* in Brno. Conclusions from this research were published in the following works [72, 206]. A summary of this research is provided in Chapter 11 on page 104.

The results show that after the lines were drawn, the drivers tended to slow down (by around 4 km/h) and to move slightly (around 20 cm) towards the outer side of the curvature. When the center line was drawn (contrary to the situation when the shoulder line markings were drawn), the cars were less spread across the driving lane, i.e., the drivers generally stayed closer to the ideal center line of the driving lane.

[3] https:\\www.aicitychallenge.org
[4] https:\\www.cdv.cz

Chapter 10

NVIDIA AI City Challenges

We have attended multiple NVIDIA AI City Challenges, organized as *Computer Vision and Pattern Recognition Workshop* since 2017. Over several years, we have submitted multiple submissions covering the various tracks of this challenge.

10.1 Speed Measurement and Vehicle Re-Identification from Video (AIC 2018)

In this submission, we address the tasks of vehicle speed measurement and re-identification (i.e. *Task1* and *Task3*). We used our approach to camera calibration (Section 6.2). Also, for the re-identification task, we use the approach proposed in our LFTD paper (Section 7.2). In the paper, we show that using "unpacked" versions of vehicles (Section 5.2) improves the re-identification performance.

The vehicles were detected by a CNN-based detector, tracked in time, and a 3D bounding box was constructed for every detected vehicle. These 3D bounding boxes are used for different purposes in the tasks. In the re-identification task, we use the 3D bounding box to produce an "unpacked" version of the vehicle and normalize the input image. In the speed measurement task, the 3D bounding box is used to compute the center of the vehicle's base, as it is a point in the road plane; therefore, it can be used for speed measurement.

10.1.1 Vehicle Detection and Data Preprocessing

As a first step of processing the video, it is necessary to **detect vehicles** in all frames. We used Faster-RCNN [182] with ResNet101 [76] backbone. The detector was trained on UA-DETRAC [150] and COD20K [100] datasets. The detections were merged to tracks using Kalman filter [101]. Examples of the detections can be found in Figure 10.1 (left). Furthermore, we detected **vanishing points** in every video using our recent algorithm [199]. Figure 10.2 (top left) shows a visualization of the detected vanishing points.

For each detected vehicle, we used general object contour detector [248] to estimate the **contours** [202] of the vehicles (Figure 10.1 – center left and center right) and then constructed **3D bounding boxes** of the vehicles [43] (Figure 10.1 right). The 3D bounding box is used for two main purposes. First, it is possible to use it to normalize the image for vehicle fine-grained recognition [198, 202] and re-identification [209]. Second, it is possible to estimate a point on the road plane which can be used for speed measurement of the vehicles [43, 199].

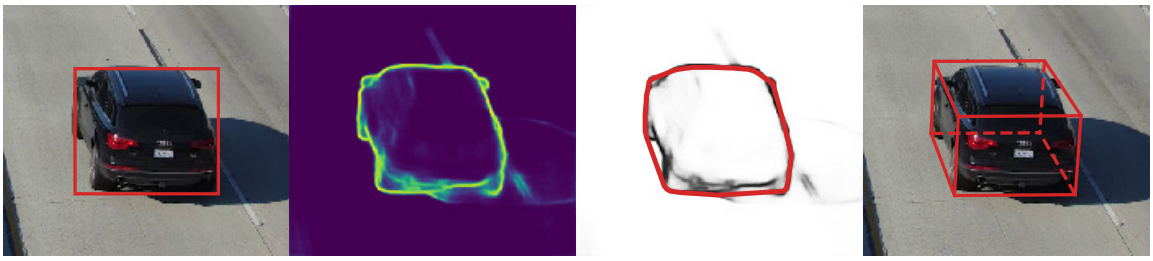


Figure 10.1: **Left:** detected vehicle using Faster-RCNN detector [182], **center left:** contour probability map estimated by general object contour detector [248], **center right:** estimated contour from the contour probability map [202], **right:** constructed 3D bounding box [43, 202].

10.1.2 Vehicle Speed Measurement

To **estimate the scale** for vanishing point-based calibration, we used Google Earth to measure the real-world distance of two points in the road plane (Figure 10.2 – top right). However, as we assume that the measurements will be imprecise, we used many of them ($\sim$ 40) to reduce the error. The set of measurements $\mathcal{M}$ was divided into two groups. The first group $\mathcal{M}_\mathbf{u}$ contains measurements in the direction to the first vanishing point $\mathbf{u}$ (represented by red arrows in Fig. 10.2, top left). The other group $\mathcal{M}_\mathbf{v}$ is computed as $\mathcal{M}_\mathbf{v} = \mathcal{M} \setminus \mathcal{M}_\mathbf{u}$.

As the second vanishing point $\mathbf{v}$ (green arrows in Fig. 10.2 – top left) is sometimes detected imprecisely (see horizon line in the same figure), we further refine its position. To achieve that, we optimize the following term

$$\mathbf{v}^* = \arg\min_{\mathbf{v}} \sum_{\mathbf{p}_1,\mathbf{p}_2,m\in\mathcal{M}_\mathbf{u}} \left| m - d_{\mathbf{u},\mathbf{v},\mathcal{M}_\mathbf{v}}(\mathbf{p}_1,\mathbf{p}_2) \right|, \tag{10.1}$$

where $d_{\mathbf{u},\mathbf{v},\mathcal{M}_\mathbf{v}}(\mathbf{p}_1,\mathbf{p}_2)$ represents the distance of two image points $\mathbf{p}_1$ and $\mathbf{p}_2$ in meters. It should be noted that the distance depends on the vanishing points and on the scene scale (which is computed using measurements $\mathcal{M}_\mathbf{v}$); therefore, we include all these variables in the lower index of the function. The final scene scale λ is computed using all measurements $\mathcal{M}$. For further details about the optimization and scene scale computation, see our paper [199]. **Final calibrations** represented by the regular orthogonal grid can be found in Figure 10.3. Using this calibration, measuring the distance of two image points in the road plane in meters is possible. Furthermore, with known video framerate, it is possible to compute the speed of the vehicles. To stabilize the speed measurement, we measure the speed between detections 10 video frames apart.

The point in the center of the base of the 3D bounding box (intersection of its diagonals) is used to measure speed in the road plane. Short tracks are represented by the median value for speed measurement noise elimination. Longer tracks are filtered to suppress high frequencies in the speed signal. Figure 10.4 (left) shows all measured speeds in all videos.

10.1.3 Approach to Vehicle Re-Identification

The InceptionResNet-V2 [216] CNN with "unpacked" vehicle images [202] with size 331×331 was fine-tuned to *identification* task in the first place on an early version of *CarsReId74k* dataset. The fine-tuning was done with Adam [106] optimizer and learning rate 1e-4.

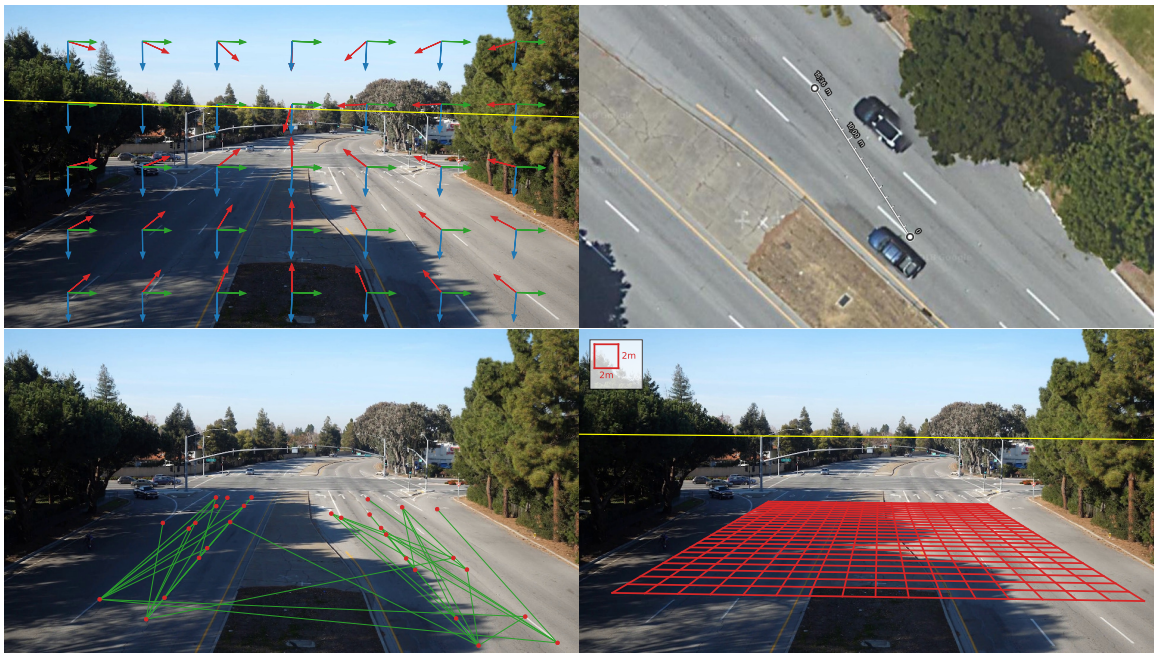


Figure 10.2: **Top left:** visualization of estimated 3 vanishing points (arrows with different color coding) and horizon (yellow line), **top right:** an example of used measurement on the road plane from Google Earth, **bottom left:** all measurements on the road plane, **bottom right:** final regular orthogonal grid with 2m sides.

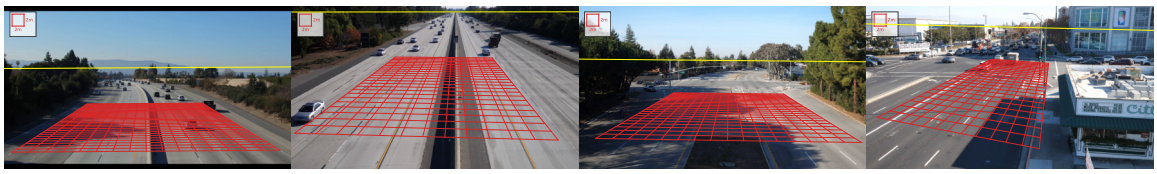


Figure 10.3: Examples of regular orthogonal grids and horizon lines for every location (from top left: Loc1 – Loc4). The size of the grid cells is 2×2 meters.

Afterward, we cached the ID features and trained **LFTD network** to aggregate the features in the temporal domain as there are multiple observations for the vehicle as they pass in front of the cameras. The LFTD network used had 1,024 output features and tanh non-linearity, and **Weighted Euclidean distance** (**WE**) was adopted.

We kept queries from part of our dataset for validation. The Hit@1 with ID features and average temporal pooling was 69 %. It increased to 78 % with the LFTD network and standard Euclidean distance. Finally, using the WE distance pushed the performance on validation data to 79 %.

These 1,024-dimensional features and learned WE distance was used for computation of **pairwise distance** between all detected and tracked vehicles at every location. We used these distances to construct quadruplets with one vehicle track from every location. These quadruplets are supposed to represent vehicle tracks with the same identity. As the vehicle needs to be observed at every location, we used the maximal distance of pairs within the quadruplet to measure the total similarity of the quadruplet. Therefore, we are interested only in the quadruplets with low maximal distance of pairs in the quadruplet.

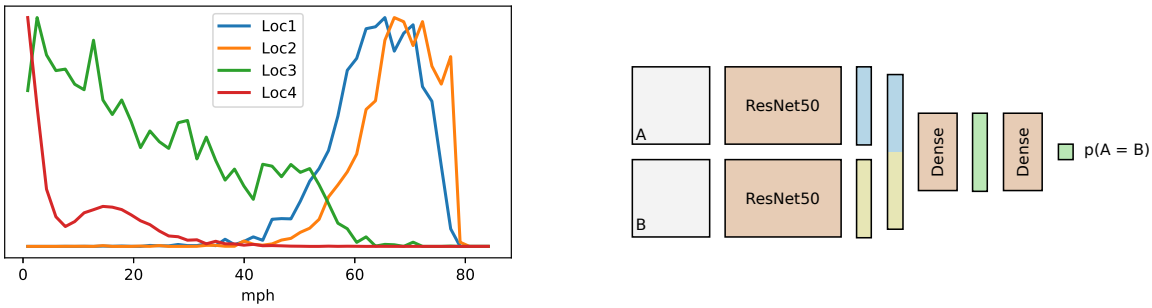


Figure 10.4: **Left:** Histogram of all measured speeds of observed vehicles.**Right:** Validation net used to refine the vehicle re-id results. The network takes two images of vehicles as inputs and produces the probability that the identity is the same for both of them.

To further improve the re-id accuracy, we trained a **validation neural network** which takes two images of vehicles and produces the probability that they have the same identity. The schematic design of the validation network can be found in Figure 10.4 (right). This validation net was applied with every pair in the quadruplets. Median and minimal probability values were used as a threshold for the quadruplets, and the ones with probabilities below the thresholds were removed.

Finally, we used these sorted and filtered quadruplets as the identities. We took only five best-scoring quadruplets to limit false positives. As the vehicles could be observed multiple times at every location, we merged these five quadruplets with any high-scoring quadruplet containing at least once the exact vehicle. This way, we acquire sets of vehicles with the assumed same identity as required by the task rules. For examples of such sets, see Figure 10.5.

10.2 Vehicle Re-ID and Multi-Camera Tracking in City-Scale Environment (AIC 2019)

In this submission, we address the tasks of vehicle multi-camera tracking and re-identification (i.e., *Track1* and *Track2*) on the CityFlow dataset [224].

Our approach to visual vehicle re-id is based on extracting feature vectors using a convolutional neural network and aggregating extracted feature vectors from observed vehicles in the temporal domain. We use standard CNNs [76, 217, 83] trained for the identification task, and we employ an LFTD network [209] for feature aggregation.

For the multi-camera tracking part, we propose a method for matching points from vehicle trajectories in real-world linear coordinate system space. This approach is based on a projection of 2D image points into the real-world linear space [236] and the matching of vehicles in this linear space with respect to time and space constraints. Furthermore, this approach can also be combined with the extraction of feature vectors for all observed and pre-matched tracks.

10.2.1 Vehicle Re-Identification

Approach to vehicle re-id task follows the previous year (Section 10.2.1 on this page). Additionally, ResNet-50 [76] was trained and evaluated. Also, we could not use our previously

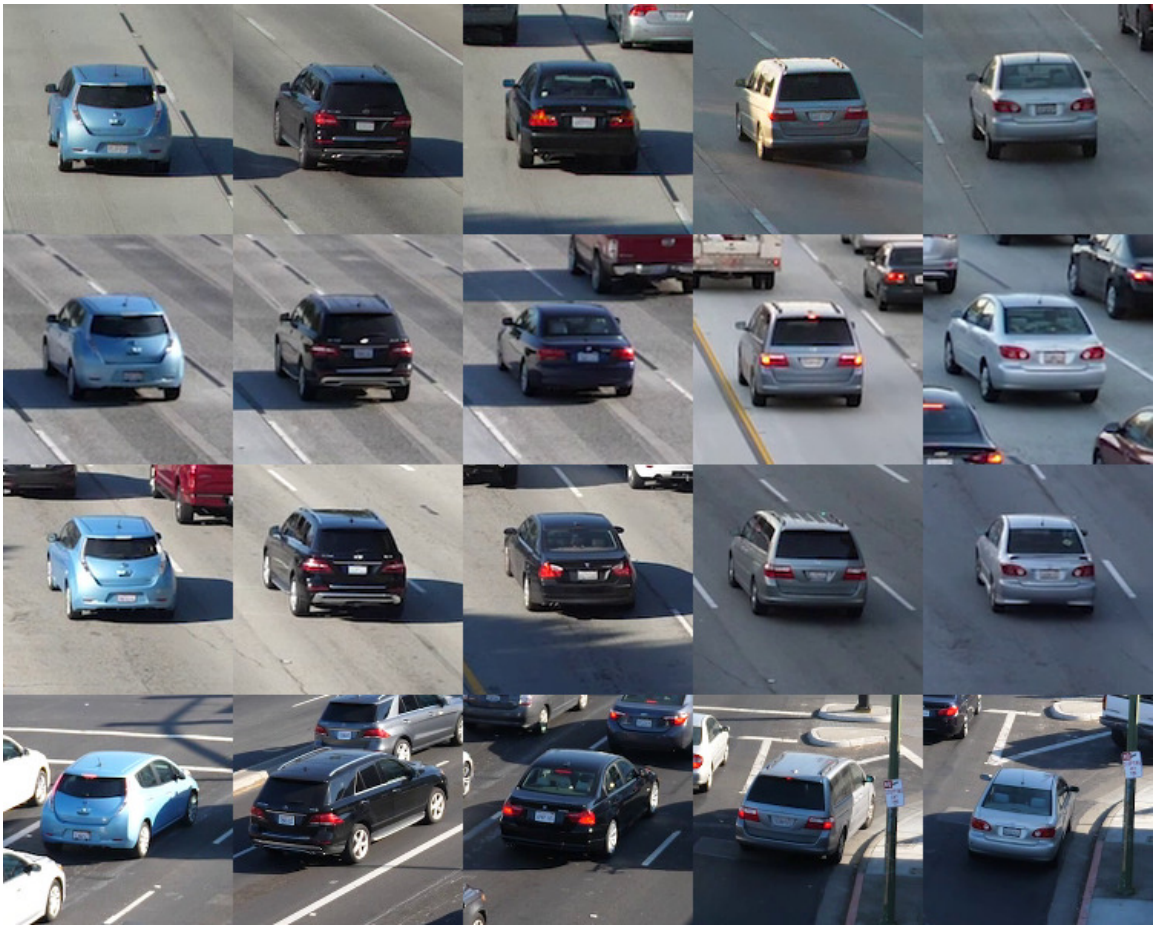


Figure 10.5: Each column represents a set of vehicles from all locations (from top to bottom: Loc1 to Loc4), which are considered by our re-identification method to share their identity. Only one track per location is shown.

proposed modification using the "unpacked" version of vehicle images due to viewpoints limitations and already cropped images in *Track2*.

We evaluated different variants of backbone networks together with the influence of using *image modifiers* [198, 202] and pre-training the networks on different datasets.

Vehicle Re-Id Design Changes

The results of our submissions from the evaluation server showed unbalanced values compared to our evaluation, which led to design changes in our methodology proposed above. Inspired by previous works [112, 224], we tried to replace our feature extractor with a much smaller CNN MobileNet [83] with feature vector dimensionality reduced to 128 dimensions. The second change was replacing the cross-entropy loss with triplet loss combined with **semi-hard batch sampling** [188]. The rest of our design remained the same. We tried multiple variants of *image modifiers* and pooling methods.

Figure 10.6: *left:* Detections' bottom points used for localization of camera's view area. *right:* Corresponding points transformed to the linear space (viewpoint selected as convex hull of these points).

10.2.2 Multi-Camera Tracking

Unlike the vehicle re-id task, the multi-camera tracking task does not have to be solved by visually comparing detected vehicles. The problem can be solved even using **positional matching** with knowledge of GPS coordinates of cameras, known distances and time synchronization between them, and their calibrations. In this case, matching is based on a projection of the 2D point of vehicle trajectory from the image space into the world coordinate space (linear system in our case). These projections from multiple cameras can be matched with each other for every time step in order to obtain matching between tracks across multiple cameras.

It should be noted that the approach described below assumes that for each camera within the session, an overlap exists in the camera's view area with at least one other camera in the same session. This condition is satisfied for almost all test-session cameras, as seen in Figure 10.9. In other cases, vehicles from the camera without overlap cannot be matched with the rest of the cameras, and the matching procedure had to be modified. However, this modification is straightforward with knowledge of time synchronization and distances between the cameras.

Vehicle Trajectory Estimation

Positional matching counts on the estimation of trajectory points of each observed vehicle. The selection of a point from vehicle detection may influence the precision of point localization in the world space. One solution is to construct the 3D bounding box [198, 202] around the vehicle and select the middle point of the vehicle base lying on the ground plane. However, this 3D bounding box construction is computationally expensive as it relies on the vehicle's silhouette. We use the middle point of 2D detections' bottom line provided instead, as this point performs the best from available data.

Transformation from Image to World Coordinate System

The transformation process assumes that the calibration parameters for each camera are known. We used camera calibration provided with the dataset to compute a homography matrix describing the transformation from the image plane to GPS coordinates in DD (*Decimal degrees*) format. The transformation between coordinate systems is a straightfor-

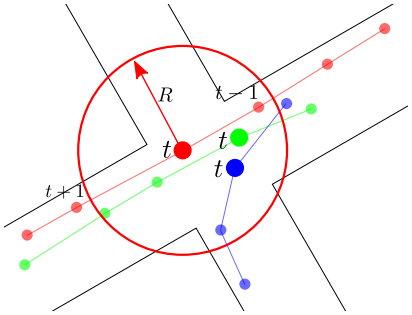


Figure 10.7: Visualization of the positional matching method. **Red** and **green** tracks corresponds to same vehicle observed by multiple cameras. **Blue** track represents another track of another vehicle. Positional matching iteratively determines if points from one trajectory correspond to points from another trajectory by constructing a circle with radius R in each time-step t, and matches are accumulated to the *matching matrix*. This matrix contains a score for each possible combination of camera-track pairs.

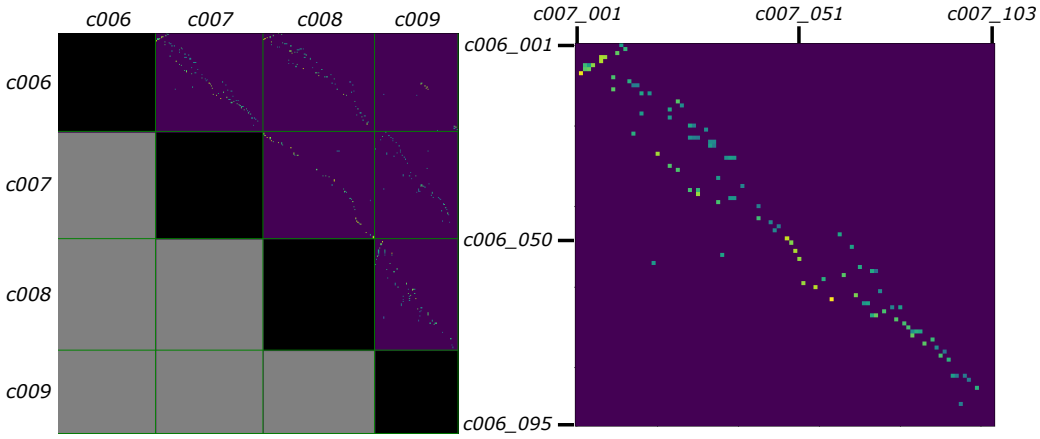


Figure 10.8: **Left:** Matching matrix for *S02*. Each block size differs based on the count of tracks detected in single cameras. **Right:** Matrix corresponding to sub-block *c006-c007*. Each cell contains the count of matches between the track in each row and each column.

ward operation made only by matrix multiplication — homography matrix **H** multiplied by GPS coordinates in homogeneous format to transform from GPS to the image plane (*i.e.*, inverted homography matrix multiplied by image point in homogeneous format to transform image plane point to GPS coordinates). Two cameras in the dataset (*c005* and *c035*) are fisheye, and thus compensation for the distortion of the point is necessary before transformation to GPS coordinate systems.

Projection of GPS to Linear System Since GPS coordinates are known in the DD format and not as positions on the flat plane, the distances, and positions do not correspond precisely to the real world because of the curvature of the Earth. Although distances in the DD format can be computed by *Haversine formula*, they can potentially suffer from some inaccuracies, and thus transformation to the linear space was done. We used transformation from *EPSG:4326* to *EPSG:26975* (corresponds to *North Iowa* where the dataset was collected).

Figure 10.9: Localized viewpoints for part of all cameras in *S04* and *S05*.

Camera View Area Estimation Our solution is automatically detecting the camera view area (polygon covering part of the real world, where a specific camera can see objects). The two bottom corners of the vehicle's bounding box are transformed into the linear space for each observed vehicle. The Convex hull of the points is used as a polygon covering the camera's view area. An example of the points used during the detection of a camera's viewpoint together with the corresponding linear space is depicted in Figure 10.6. Examples of localized viewpoints for a part of all cameras in a session can be seen in Figure 10.9. Our experiments show that it is convenient not to use all detections but to limit them in some way — detections must be larger than 1000 pixels (in area), and all detections should be no further than 300 meters from the camera in the linear space.

Multi-Camera Tracks Positional Matching

Positional matching between vehicle tracks observed by multiple cameras at one session is based on comparing mutual positions of individual trajectory points from multiple cameras in the real-world linear coordinate system in each time step. Trajectory points from each camera in a session are sorted by their observation time (*time steps*). For each time step and each trajectory point observed by one camera, we construct a circle in the linear space with radius R, and we are looking for trajectory points from other cameras in the session, which are contained inside the constructed circle (for better understanding, please see Figure 10.7). These pairwise camera–track *matches* are accumulated in a *matching matrix* M. This matrix contains all possible matches from each track in one camera to all tracks in the other cameras. Figure 10.8 shows an example of the matching matrix.

The matching matrix is split into pairwise camera blocks. In each row of these blocks, we look for maximal accumulated values in other camera blocks separately using *Linear Sum Assingment* solver. These maximal values correspond to the best matching tracks between all cameras in the session. Best matches are further processed and joined into bigger groups if some element of *pairs, triplets, quadruplets,...* is missing in the other set, which has at least one shared element.

Even visual features can be employed in the proposed method for solving multi-camera tracking problem. We can extract features (by using the same convolutional neural network with pooling as described in Section 10.2.1) from vehicle tracks given by camera-track

Figure 10.10: Example result of our positional matching method with a projection of trajectories' points into a linear coordinate system. Arrows depict transformed points from image to real-world space in a specific time-step (displayed camera frame).

indices of the matching matrix and construct a pairwise distance matrix with the same shape as the matching matrix. This distance matrix is then used to weigh elements in the matching matrix.

10.3 Determining Vehicle Turn Counts (AIC 2020)

In this submission, we address the task of vehicle counting by their class at multiple intersections (i.e., *Track1*). This track is focused on counting four-wheel vehicles and freight trucks that follow pre-defined movements (travel directions) (see Fig. 10.11) on different camera scenes, which should help DOTs' traffic engineers in traffic analysis and planning. In this challenge's track, the emphasis is placed on effectiveness (precision of counting) and, along with that, also on efficiency.

Our solution is based on *counting by tracking* approach. It benefits from using convolutional neural networks in the detection and tracking steps. Trajectories of vehicles obtained for the CNN feature-based tracker are further processed and analyzed to determine each vehicle's entry and exit point for correct counting of trajectories.

10.3.1 Determining Entry and Exit Areas

Assigning a vehicle trajectory to a specific travel direction requires knowledge of the vehicle entrance/exit area on the road. This task is crucial to obtain correct counts for every possible travel direction. In our case, the entrance and exit areas were manually annotated using polygonal representation from 3 to 5 points for each polygon. We also have a user-defined trajectory for each travel direction defined by pair of an entrance/exit area. Examples of areas defined for each camera together with region-of-interest can be found in Figure 10.12.

Figure 10.11: The main idea of this task — counting vehicles for every pre-defined movement of interest (travel direction).

10.3.2 Vehicle Detection and Tracking

The proposed solution is based on vehicle detection and tracking. For every input video frame, we first detect vehicles using a CNN-based detector. In order to achieve real-time detection or even quicker processing but still have a high detection accuracy, we decided to use the YOLOv3-416 detector [181]. For the tracking task, we decided to use *Simple Online and Real-time Tracking with Deep Association Metric* (DeepSORT) [238].

For each vehicle, we save its trajectory points. The centroid of the detected vehicle's bounding box is used as a point representing the vehicle in the trajectory. When the vehicle is no longer detected but still appears in a frame of the video (its trajectory has not left the ROI yet), we predict the next position of the vehicle based on its last trajectory points. We compute the average angle between the last 5 points of the vehicle trajectory and the Euclidean distance between the last 2 points. Based on the computed angle and distance, we predict the next trajectory point of the vehicle.

Merging Broken Trajectories

Before the tracker prediction step, we deal with long-term occlusion and detection inaccuracies by a trajectory merging step. When the vehicle is not visible for a certain period and then reappears in a subsequent frame, it starts its new trajectory even though the old trajectory of this vehicle is still predicted.

To deal with this situation, when one vehicle is being tracked twice, the proposed method tries to merge broken trajectories. For each trajectory predicted by the tracker, we try to find a newly detected vehicle trajectory formed by 2 or 3 points. We compare the latest predicted points of the tracker-predicted trajectory with the location and direction of this new trajectory.

If the trajectories are similar in their location and speed, these two trajectories are merged. This merging prevents counting two vehicles instead of a single correct one. This approach may lead to identity switching, but this is fine if vehicles are traveling in the same direction – which they are. Otherwise, they would not be merged by the algorithm.

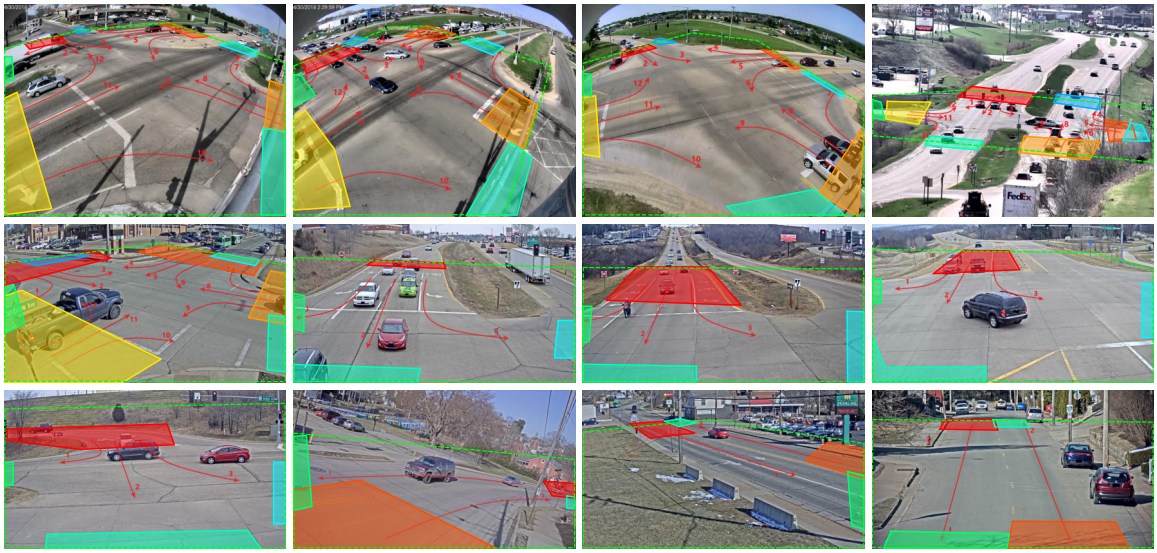


Figure 10.12: Examples of images from a few cameras in the dataset with annotated entrance/exit areas and pre-defined movements of interests and ROIs. Images were updated to fit the view better. The ordering of images does not fully correspond to the sequence of camera IDs.

10.3.3 Travel Direction Assignment

The final step is determining the correct travel direction (movement of interest) for each ended trajectory outside pre-defined ROIs. Pairs of the entrance/exit areas define the travel directions.

For each detected trajectory, the entrance area is defined by its intersection with the beginning of this trajectory or by the distance to the closest one. The same principle is applied even for the exit area of each trajectory. This pair then defines a travel direction.

Unfinished trajectories In some situations, the detector can lose the vehicle in the middle of a pre-defined ROI. This situation may occur, for example, when the vehicle is turning from one road to another. In such a case, the prediction-based trajectory of the vehicle will continue in the determined direction (e.g., straight, even if the vehicle is turning). In contrast, the detection-based trajectory ends **before** it exceeds the border of the ROI. If new detections no longer update the prediction-based trajectory, it may cause the vehicle will be counted in the wrong direction.

For these cases, our annotations for every camera contain the average trajectory for each travel direction. When every point of the detection-based trajectory can be fit to this average trajectory (point from detection-based trajectory is inside of polygon, which defines average trajectory), the prediction-based trajectory will follow this direction, even if the detection is lost.

Chapter 11

Analysis of Vehicle Trajectories for Determining Cross-Sectional Load Density Based on Computer Vision

The principal assumption of this research is that the driver controls their vehicle according to previous stimuli coming from the roadway. The driver adjusts the vehicle's path and/or speed based on these. It is known that the driver approaching a narrow part of the roadway decelerates [96]. The speed, together with the reaction times of the driver, further influence the trajectory of the vehicle, having a direct impact on conflict situations and traffic accidents.

Dangerous situations often occur in directional bends with smaller radii, where the roadway is narrowed, and the drivers are acting recklessly (they do not decrease the speed). Widening the road or changing its location would be too expensive and often unfeasible. A low-cost solution that could help to some extent would be horizontal traffic marking on the road, either as a shoulderline or as the centerline between the lanes. Which of these (or their combination) is more suitable/influential, and what effect can be expected from their application?

In this work, we used long-term visual recording and computer vision analysis of the recorded videos to get answers to these questions related to transportation safety and improve the situation with meager costs. The information could be tough to obtain, mainly when the locations of interest were on third-grade roads with meager traffic intensity, where human observation would be inefficient or impossible.

Our measurements were related to traffic density across the lateral cross-section of the road in the directional arc as dependent on the horizontal lane markings. The distance of the vehicle from the side of the road and its speed can provide evidence of the influence that the presence of the central/side lane markings might have on the vehicle's trajectory. The assumption is that the drivers keep their distance from these markings and try to adjust their driving style.

We took long video recordings on several analyzed locations before and after horizontal road markings were applied and processed the videos. First, we calibrated the cameras to enable measurements on the road plane. We detected and tracked passing vehicles and measured their position on three road cross-sections at each location. Finally, we analyzed how the distribution of car speeds and positions changes with the presence of road markings.

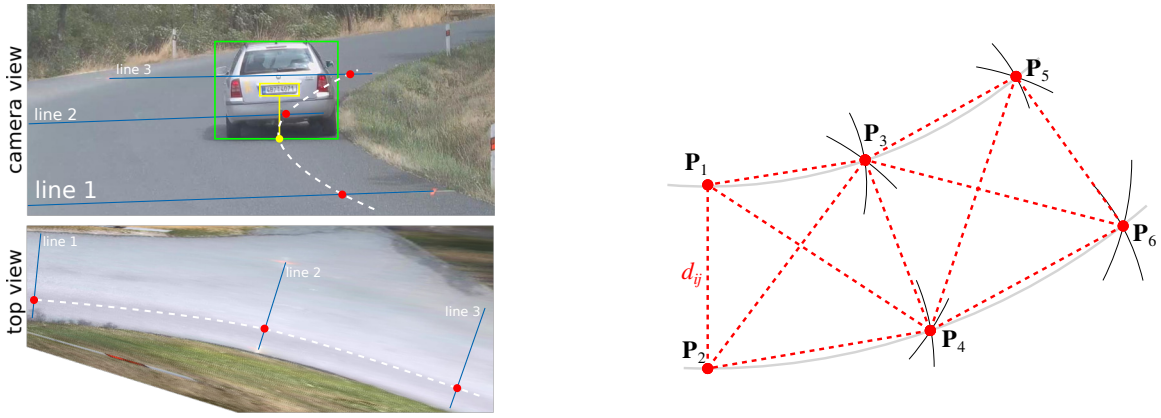


Figure 11.1: **Left:** We detect and track cars in videos and record lateral position of cars on three road cross sections. We analyze how driver behavior changes when horizontal line markings are drawn. **Right:** Construction of road plane coordinates $\mathbf{P}_i$ based on the measured distances d_{ij}.

11.1 Methodology

We installed a static camera at each location and visually captured several days of traffic before and after road markings were drawn. Our goal was to analyze lower-class roads with low traffic intensity, and multiple days are required to capture a sufficient number of trajectories.

In order to measure distances in the captured image space, the camera needs to be calibrated (including the scale). After camera installation, the operators marked several clearly visible points on the road surface and measured their distances.We define three *virtual cross-sections* in which the positions of the passing cars are being measured and evaluated. Figure 11.2 shows a camera view from one of analyzed locations with virtual lines and measured distances. After camera installation, the speed of several cars was measured by a handheld radar in order to get reference speeds for validating the visual vehicle tracker.

11.1.1 Camera calibration

From the marker locations $\mathbf{p}_i$ in the camera image (annotated manually) and their distances d_{ij}, we recovered locations of markers on the road plane $\mathbf{P}_i$ (see Figure 11.1 – Right). To estimate the homogeneous positions of the markers in the road plane $\mathbf{P}_i$, we first set $\mathbf{P}_1 = (0, 0, 1)^\intercal$ and $\mathbf{P}_2 = (0, d_{12}, 1)^\intercal$. Then, it is possible to construct other points $\mathbf{P}_i$ always as an intersection of two circles. Formally, point $\mathbf{P}_3$ is computed as:

$$\mathbf{P}_3 = \arg\min_{\mathbf{P}} \sum_{j \in \{1,2\}} \Big| d_{j3} - \|\mathbf{P}_j - \mathbf{P}\| \Big|. \tag{11.1}$$

As the two circles have two intersections, the ones with higher x coordinates are used. Following this procedure, the rest of the points are constructed. Finally, as distances d_{34} and d_{56} are not used for constructing the points, we used them as validation measurements, and the relative error was approximately 1 %.

To construct the mapping between the image space and the road plane, 2D homography $\mathbf{H}$ is computed between annotated positions in the image space $\mathbf{p}_i$ and the corresponding

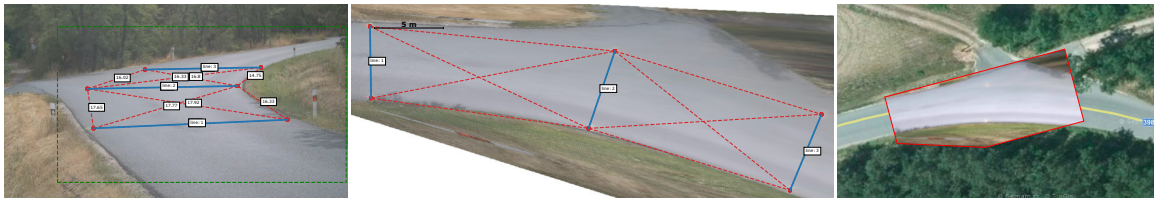


Figure 11.2: An example of the calibration markings and measurements. A minimal of 6 points, 3 on each side of the road are marked on the side of the road and the mutual distances are measured in real world. **Left:** camera image where positions of markers were defined. **Middle:** birds-eye view of the scene rendered using homography recovered form point distances. **Right:** our reconstruction of the road overlayed on orthographic map shows that the reconstruction is correct.

points on the road plane $\mathbf{P}_i$. Since there are 6-point correspondences, the linear system is over-determined, and linear least squares is used to estimate the homography.

In this step, the scene is assumed to be approximately planar – this condition must be locally satisfied, at least in the measured area. After this calibration, any image point can be mapped to its position on the road plane, and the real-world distances between image points can thus be measured.

11.1.2 Video processing

Tens of hours of Full HD video recording have to be processed. Videos were pre-processed by custom-trained detector [38] for license plate (LP) detection, followed by Faster R-CNN detector [182] for precise vehicle detection within pre-filtered video segments (nighttime videos were discarded). A Kalman filter tracked car detections, and LP detections were associated with tracks. To measure the position of a car on the road plane, we simply project the center of its license plate to the bottom edge of the car's bounding box and transform this point using homography $\mathbf{H}$.

Formally, car track T of length N_T is a sequence of image points $\mathbf{x}_i$ transformed with the calibration homography $\mathbf{H}$ to the road plane, associated with a timestamp t.

$$T = \{(\mathbf{H}\mathbf{x}_i, t_i) \mid i = 1 \dots N_T\} \tag{11.2}$$

To remove noise from the tracking, positions $\mathbf{H}\mathbf{x}_i$ are filtered by a third-order polynomial, and the interpolated track S is obtained.

For a trajectory S, crossing points l_j with the road cross sections $j \in \{1, 2, 3\}$ are obtained. The average speed v between the first and the last cross-section is determined by measuring the traveled distance on the road plane and the time difference. Finally, for purposes of statistics, a trajectory is represented by its speed and the three positions on the cross sections $\mathcal{T}_k = (l_j, v), j \in \{1, 2, 3\}$.

11.2 Results

We analyzed 6 locations on lower-class roads. On each location, a static camera was installed several days before line markings were drawn and recorded the traffic before and after the line marking application. The study includes 1,010 hours of traffic video recording.

Although the cameras were meant to be static, the long recording shows a slight drift of the camera image due to wind and instability of camera mounting. Therefore we used

Figure 11.3: Visualization of the measured localities. **top:** camera view before lane makings applied, **middle:** view after lane markings applied, **bottom:** camera view projected on the ground plane computed from the calibration measurements **blue** color marks the envelope of trajectories prior to road marking, **orange** marks envelopes after road markings were drawn.

Table 11.1: Details of the measurements at different localities. Average speed is in km/h.

	Location 1				Location 2				Location 3			
Curve	Inner		Outer		Inner		Outer		Inner		Outer	
Marking	✘	✔	✘	✔	✘	✔	✘	✔	✘	✔	✘	✔
# cars	131	41	844	367	777	709	673	621	255	307	520	668
Avg. speed	66.1	71.7	66.3	69.1	70.3	63.7	66.7	64.9	61.3	61.6	61.5	60.0

	Location 4				Location 5				Location 6			
Curve	Inner		Outer		Inner		Outer		Inner		Outer	
Marking	✘	✔	✘	✔	✘	✔	✘	✔	✘	✔	✘	✔
# cars	352	222	51	61	308	201	382	274	1201	979	1690	880
Avg. speed	48.5	47.9	48.6	47.4	56.9	55.0	56.7	55.1	62.5	64.1	61.7	62.0

different sets of annotated marker locations for video before and after the road marking application. On Location 2, the recording was affected by severe wind, which caused camera vibrations and subsequent camera image shaking. This resulted in a significant error in the measurements of car positions (where the assumption is a fixed camera image). We did not find any reliable method to stabilize the image to fix the position of markers automatically. We include the results from this location; however, they are not to be considered significant.

The individual scenes included in the study are visualized by Figure 11.3. The illustration shows an illustrative frame from the video before and after applying the horizontal line markings. It also shows the camera view re-projected to the horizontal plane based on the calibration points. In these views, the three cross-section lines are shown, a visualization of the distribution of cars going in each direction before (blue) and after (orange) the application of the horizontal line markings.

More detailed histograms of the vehicles' lateral positions within section 2 (the middle one) and the vehicles' speeds are shown in Figure 11.4. The lateral shifts in the vehicles' position are more significant than the vehicles' speed change. Table 11.1 shows detailed location information, number of observed vehicles, and average speed for the inner and outer driving lanes. Finally, Figure 11.5 visualizes the change in speed and lateral position within the cross-sections for all the locations (separately reporting the inner and the outer driving lane).

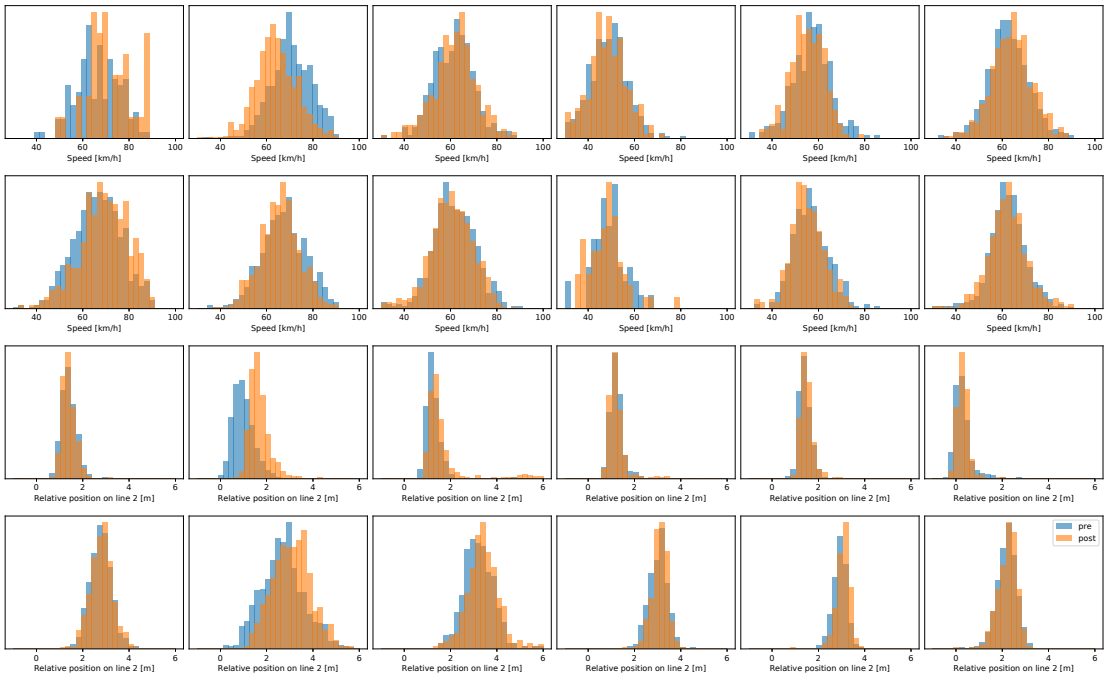


Figure 11.4: Distributions of the changes of speeds (rows 1 and 2 for *inner* and *outer* curve) and positions of cars (rows 3 and 4 for *inner* and *outer* curve) for all locations (in columns). **blue:** distribution before application of lines, **orange:** distribution after application of lines.

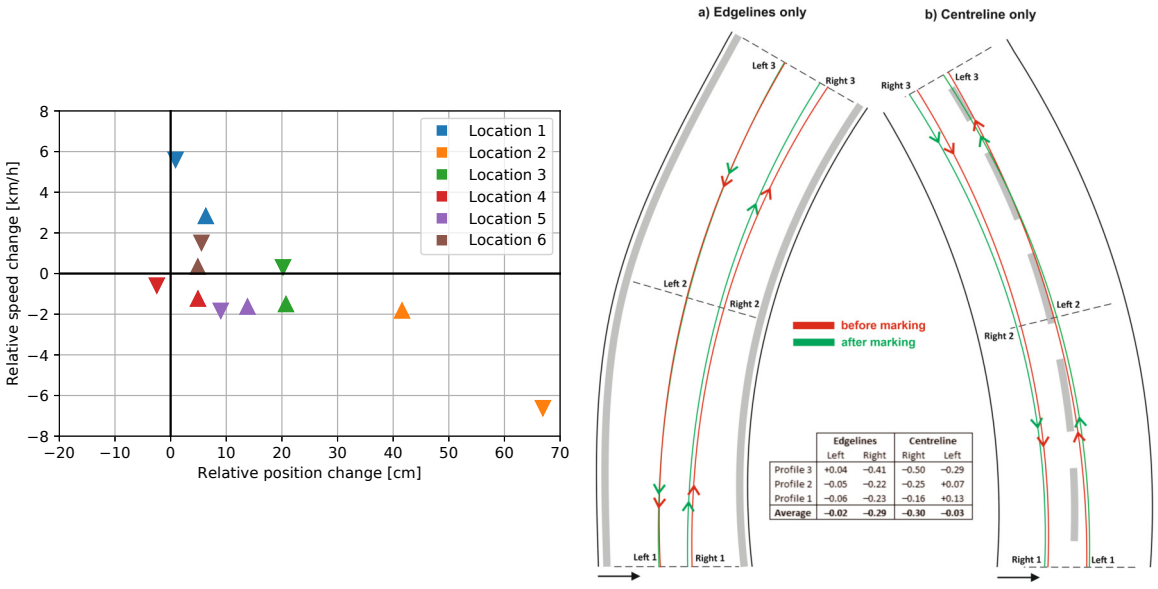


Figure 11.5: **Left:** Change of mean car position (horizontal axis) and mean car speed (vertical axis) after the markings were drawn. The car position is measured on the middle cross section. Locations are color coded. On most locations, cars tend to slow down with the markings present. Triangle pointing down: inner lane in the curve; up: outer lane. **Right:** Visual representation of approximate driving trajectories, based on obtained differences in lateral positions in three profiles.

Chapter 12

Conclusion

This thesis presents a contribution to the traffic analysis community in four critical aspects of Intelligent Transportation Systems (ITS), namely Vehicle Fine-Grained Recognition, Vehicle Re-Identification, License Plate Recognition, and Monocular Vehicle Speed Measurement. Throughout this research, significant strides were made by advancing the robustness of Computer Vision methods for traffic analysis, an essential component of ITS.

This includes novel method and augmentation techniques for vehicle fine-grained recognition Chapter 5, a novel method for aggregation visual features for vehicle re-identification Chapter 7 and innovative approach to license plate recognition using alignment of the license plate and holistic recognition Part IV.

Another part of this contribution is made by providing rigorously curated datasets in the areas of license plate recognition, vehicle re-identification, vehicle fine-grained recognition, and monocular vehicle speed measurement. The benchmarking of algorithms, performance evaluation, and advancement of research in traffic analysis applications are only a few advantages these datasets provide. The availability of these datasets enables researchers to push the boundaries of traffic analysis, ultimately leading to safer, more efficient transportation systems.

The following articles represent the culmination of this research, each contributing innovative approaches and solutions to its respective field. Their acceptance in the prestigious conferences and journals in the ITS field has validated the research efforts and confirmed their alignment with the current research trends and priorities in ITS.

- Špaňhel *et al.*, Learning feature aggregation in temporal domain for re-identification, CVIU 2020, (Q1, IF 2023: 4.886).
- Špaňhel *et al.*, Holistic recognition of low quality license plates by CNN using track annotated data, IEEE AVSS 2017 (CORE Rank: B).
- Špaňhel *et al.*, Geometric alignment by deep learning for recognition of challenging license plates, IEEE ITSC 2018.
- Sochor *et al.*, BoxCars: Improving fine-grained recognition of vehicles using 3-d bounding boxes in traffic surveillance, IEEE T-ITS 2018 (Q1, IF 2023: 9.551).
- Sochor *et al.*, Comprehensive data set for automatic single camera visual speed measurement, IEEE T-ITS 2018 (Q1, IF 2023: 9.551).

Despite the advances made, it is acknowledged that the field of ITS, especially concerning the application of Computer Vision methods, is continuously evolving. Therefore,

there are several potential avenues for future research. For example, achieving very accurate detection of license plate corners to sub-pixel accuracy could be a step forward for a monocular speed measurement system certified by the relevant authorities.

Moreover, as ITS become more pervasive, it would be crucial to focus on their scalability and interoperability while considering their societal, economic, and environmental impacts. The current state-of-the-art methods in ITS often rely on the availability of large amounts of computational resources, and many problems are solved by enlarging models or combinations of models. However, these steps are incompatible with real-world deployments of these approaches, where methods often need to run on hardware with limited performance, such as traffic cameras. In this case, methods based on knowledge distillation could be the leaders of the field in the following years.

In summary, this dissertation represents a step in the ongoing journey of understanding and improving Computer Vision methods for traffic analysis and their robustness. Hopefully, this research's findings, insights, and contributions will inspire and inform future endeavors in this exciting and transformative field.

Bibliography

[1] Agarwal, S., Awan, A., and Roth, D. Learning to detect objects in images via a sparse, part-based representation. *IEEE Trans. on Pattern Analysis and Machine Intelligence (T-PAMI)*. 2004, vol. 26, no. 11.

[2] Anagnostopoulos, C. N. E., Anagnostopoulos, I. E., Loumos, V. and Kayafas, E. A license plate-recognition algorithm for intelligent transportation system applications. *IEEE Trans. on Intelligent Transportation Systems (T-ITS)*. 2006, vol. 7, no. 3.

[3] Arth, C., Leistner, C. and Bischof, H. Object reacquisition and tracking in large-scale smart camera networks. In: IEEE. *ACM/IEEE International Conference on Distributed Smart Cameras*. 2007.

[4] Asselin Miller, N., Biedka, M., Gibson, G., Kirsch, F., Hill, N. et al. Study on the deployment of C-ITS in Europe: Final Report. *Report for DG MOVE MOVE/C*. 2016, vol. 3.

[5] Atienza, R. Vision transformer for fast and efficient scene text recognition. In: Springer. *International Conference on Document Analysis and Recognition (ICDAR)*. 2021.

[6] Baek, J., Kim, G., Lee, J., Park, S., Han, D. et al. What is wrong with scene text recognition model comparisons? dataset and model analysis. In: *IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019.

[7] Baran, R., Glowacz, A. and Matiolanski, A. The efficient real- and non-real-time make and model recognition of cars. *Multimedia Tools and Applications*. 2015, vol. 74, no. 12. ISSN 1380-7501.

[8] Bartl, V. and Herout, A. Optinopt: Dual optimization for automatic camera calibration by multi-target observations. In: IEEE. *IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*. 2019.

[9] Bartl, V., Juranek, R., Špaňhel, J. and Herout, A. Planecalib: Automatic camera calibration by multiple observations of rigid objects on plane. In: IEEE. *Digital Image Computing: Techniques and Applications (DICTA)*. 2020.

[10] Bartl, V., Špaňhel, J., Dobeš, P., Juranek, R. and Herout, A. Automatic camera calibration by landmarks on rigid objects. *Machine Vision and Applications (MVAP)*. 2021, vol. 32.

[11] Bay, H., Tuytelaars, T. and Van Gool, L. Surf: Speeded up robust features. In: Springer. *IEEE European Conference on Computer Vision (ECCV)*. 2006.

[12] Bell, S., Lawrence Zitnick, C., Bala, K. and Girshick, R. Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2016.

[13] Bernsen, J. Dynamic thresholding of grey-level images. In: *International Conference on Pattern Recognition (ICPR)*. 1986, vol. 2.

[14] Boonsim, N. and Prakoonwit, S. Car make and model recognition under limited lighting conditions at night. *Pattern Analysis and Applications*. 2016. ISSN 1433-755X.

[15] Broomhead, D. S. and Lowe, D. *Radial basis functions, multi-variable functional interpolation and adaptive networks*. 1988.

[16] Buch, N., Orwell, J. and Velastin, S. A. 3D Extended Histogram of Oriented Gradients (3DHOG) for Classification of Road Users in Urban Scenes. 2009. doi:10.5244/C.23.15.

[17] Canny, J. A Computational Approach to Edge Detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence (T-PAMI)*. 1986, PAMI-8, no. 6. ISSN 0162-8828.

[18] Caraffi, C., Vojir, T., Trefny, J., Sochman, J. and Matas, J. A System for Real-time Detection and Tracking of Vehicles from a Single Car-mounted Camera. In: *IEEE Intelligent Transportation Systems Conference (ITSC)*. 2012.

[19] Cathey, F. and Dailey, D. A novel technique to dynamically measure vehicle speed using uncalibrated roadway cameras. In: *Intelligent Vehicles Symposium*. 2005.

[20] Chai, Y., Rahtu, E., Lempitsky, V., Van Gool, L. and Zisserman, A. TriCoS: A Tri-level Class-Discriminative Co-Segmentation Method for Image Classification. In: *IEEE European Conference on Computer Vision (ECCV)*. 2012.

[21] Chang, S.-L., Chen, L.-S., Chung, Y.-C. and Chen, S.-W. Automatic license plate recognition. *IEEE Trans. on Intelligent Transportation Systems (T-ITS)*. 2004, vol. 5, no. 1.

[22] Chatfield, K., Simonyan, K., Vedaldi, A. and Zisserman, A. Return of the Devil in the Details: Delving Deep into Convolutional Nets. In: *British Machine Vision Conference (BMVC)*. 2014.

[23] Cheang, T. K., Chong, Y. S. and Tay, Y. H. Segmentation-free Vehicle License Plate Recognition using ConvNet-RNN. *ArXiv preprint arXiv:1701.06439*. 2017.

[24] Chen, J., Wang, Y., Qin, J., Liu, L. and Shao, L. Fast Person Re-Identification via Cross-Camera Semantic Binary Transformation. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2017.

[25] Chen, L., Yang, H., Zhu, J., Zhou, Q., Wu, S. et al. Deep Spatial-Temporal Fusion Network for Video-Based Person Re-Identification. In: *IEEE Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2017.

[26] Chen, W., Chen, X., Zhang, J. and Huang, K. Beyond Triplet Loss: A Deep Quadruplet Network for Person Re-Identification. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2017.

[27] Cheng, D., Gong, Y., Zhou, S., Wang, J. and Zheng, N. Person Re-Identification by Multi-Channel Parts-Based CNN With Improved Triplet Loss Function. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2016.

[28] Cho, Y.-J. and Yoon, K.-J. Improving Person Re-Identification via Pose-Aware Multi-Shot Matching. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2016.

[29] Chopra, S., Hadsell, R. and LeCun, Y. Learning a similarity metric discriminatively, with application to face verification. In: IEEE. *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2005, vol. 1.

[30] Clady, X., Negri, P., Milgram, M. and Poulenard, R. Multi-class Vehicle Type Recognition System. In: *IAPR Workshop on Artificial Neural Networks in Pattern Recognition*. Berlin, Heidelberg: [b.n.], 2008. ANNPR '08. ISBN 978-3-540-69938-5.

[31] Cucchiara, R., Piccardi, M. and Mello, P. Image analysis and rule-based reasoning for a traffic monitoring system. *IEEE Trans. on Intelligent Transportation Systems (T-ITS)*. 2000, vol. 1, no. 2.

[32] Dai, J., He, K. and Sun, J. Instance-aware semantic segmentation via multi-task network cascades. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2016.

[33] Dai, J., Li, Y., He, K. and Sun, J. R-FCN: Object detection via region-based fully convolutional networks. In: *Advances in Neural Information Processing Systems (NIPS)*. 2016.

[34] Dailey, D., Cathey, F. and Pumrin, S. An algorithm to estimate mean traffic speed using uncalibrated cameras. *IEEE Trans. on Intelligent Transportation Systems (T-ITS)*. 2000, vol. 1, no. 2. ISSN 1524-9050.

[35] Dalal, N. and Triggs, B. Histograms of oriented gradients for human detection. In: IEEE. *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2005, vol. 1.

[36] Dlagnekov, L. and Belongie, S. *Recognizing Cars*. 2005.

[37] Do, V.-H., Nghiem, L.-H., Thi, N. P. and Ngoc, N. P. A simple camera calibration method for vehicle velocity estimation. In: *Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), 2015 12th International Conference on*. 2015.

[38] Dollár, P., Appel, R., Belongie, S. and Perona, P. Fast Feature Pyramids for Object Detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence (T-PAMI)*. 2014, vol. 36, no. 8. ISSN 0162-8828.

[39] Du, S., Ibrahim, M., Shehata, M. and Badawy, W. Automatic license plate recognition (ALPR): A state-of-the-art review. *IEEE Trans. on Circuits and Systems for Video Technology*. 2013, vol. 23, no. 2.

[40] Duan, K., Parikh, D., Crandall, D. and Grauman, K. Discovering Localized Attributes for Fine-grained Recognition. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2012.

[41] Dubská, M., Herout, A., Juránek, R. and Sochor, J. Fully Automatic Roadside Camera Calibration for Traffic Surveillance. *IEEE Trans. on Intelligent Transportation Systems (T-ITS)*. 2015, vol. 16, no. 3. ISSN 1524-9050.

[42] Dubská, M. and Herout, A. Real Projective Plane Mapping for Detection of Orthogonal Vanishing Points. In: *British Machine Vision Conference (BMVC)*. Bristol, GB: [b.n.], 2013.

[43] Dubská, M., Sochor, J. and Herout, A. Automatic Camera Calibration for Traffic Understanding. In: *British Machine Vision Conference (BMVC)*. 2014.

[44] Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J. and Zisserman, A. The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision*. 2010, vol. 88, no. 2.

[45] Fan, X. and Zhao, W. Improving robustness of license plates automatic recognition in natural scenes. *IEEE Trans. on Intelligent Transportation Systems (T-ITS)*. 2022, vol. 23, no. 10.

[46] Fang, J., Zhou, Y., Yu, Y. and Du, S. Fine-Grained Vehicle Model Recognition Using A Coarse-to-Fine Convolutional Neural Network Architecture. *IEEE Trans. on Intelligent Transportation Systems (T-ITS)*. 2016, PP, no. 99. ISSN 1524-9050.

[47] Felzenszwalb, P. F., Girshick, R. B. and McAllester, D. Cascade object detection with deformable part models. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2010. ISSN 1063-6919.

[48] Feris, R. S., Siddiquie, B., Petterson, J., Zhai, Y., Datta, A. et al. Large-scale vehicle detection, indexing, and search in urban surveillance videos. *IEEE Trans. on Multimedia*. 2012, vol. 14, no. 1.

[49] Filipiak, P., Golenko, B. and Dolega, C. Applications of Evolutionary Computation. In: Squillero, G. and Burelli, P., ed. Cham: [b.n.], 2016, chap. NSGA-II Based Auto-Calibration of Automatic Number Plate Recognition Camera for Vehicle Speed Measurement. ISBN 978-3-319-31204-0.

[50] Folenta, J., Spanhel, J., Bartl, V. and Herout, A. Determining vehicle turn counts at multiple intersections by separated vehicle classes using CNNs. In: *IEEE/CVF Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2020.

[51] Gao, C., Wang, J., Liu, L., Yu, J. G. and Sang, N. Temporally aligned pooling representation for video-based person re-identification. In: *IEEE International Conference on Image Processing (ICIP)*. 2016.

[52] Gavves, E., Fernando, B., Snoek, C., Smeulders, A. and Tuytelaars, T. Local Alignments for Fine-Grained Categorization. *International Journal of Computer Vision*. 2015, vol. 111, no. 2. ISSN 0920-5691.

[53] Ge, Y., Chen, D. and Li, H. Mutual mean-teaching: Pseudo label refinery for unsupervised domain adaptation on person re-identification. *ArXiv preprint arXiv:2001.01526*. 2020.

[54] Gebru, T., Krause, J., Wang, Y., Chen, D., Deng, J. et al. *Using Deep Learning and Google Street View to Estimate the Demographic Makeup of the US*. 2017.

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

[56] Giannoukos, I., Anagnostopoulos, C.-N., Loumos, V. and Kayafas, E. Operator context scanning to support high segmentation rates for real time license plate recognition. *International Conference on Pattern Recognition (ICPR)*. 2010, vol. 43, no. 11.

[57] Girshick, R. Fast r-cnn. In: *IEEE International Conference on Computer Vision (ICCV)*. 2015.

[58] Girshick, R., Donahue, J., Darrell, T. and Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2014.

[59] Glasner, D., Galun, M., Alpert, S., Basri, R. and Shakhnarovich, G. Viewpoint-Aware Object Detection and Continuous Pose Estimation. *Image&Vision Comp.* 2012.

[60] Gonçalves, G. R., Diniz, M. A., Laroca, R., Menotti, D. and Schwartz, W. R. Real-time automatic license plate recognition through deep multi-task networks. In: IEEE. *Conference on Graphics, Patterns and Images (SIBGRAPI)*. 2018.

[61] Gonçalves, G. R., Silva, S. P. G. da, Menotti, D. and Schwartz, W. R. Benchmark for license plate character segmentation. *Journal of Electronic Imaging*. 2016, vol. 25, no. 5.

[62] Goodfellow, I. J., Bulatov, Y., Ibarz, J., Arnoud, S. and Shet, V. Multi-digit Number Recognition from Street View Imagery using Deep Convolutional Neural Networks. *ArXiv preprint arXiv:1312.6082*. 2013.

[63] Grammatikopoulos, L., Karras, G. and Petsa, E. Automatic estimation of vehicle speed from uncalibrated video sequences. In: *International Symposium on Modern Technologies, Educationa nd Profeesional Practice in Geodesy and Related Fields*. 2005.

[64] Graves, A., Liwicki, M., Fernández, S., Bertolami, R., Bunke, H. et al. A novel connectionist system for unconstrained handwriting recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence (T-PAMI)*. 2009, vol. 31, no. 5.

[65] Gu, H.-Z. and Lee, S.-Y. Car model recognition by utilizing symmetric property to overcome severe pose variation. *Machine Vision and Applications (MVAP)*. 2013, vol. 24, no. 2. ISSN 0932-8092.

[66] Guo, H., Zhu, K., Tang, M. and Wang, J. Two-level attention network with multi-grain ranking loss for vehicle re-identification. *IEEE Trans. on Image Processing*. 2019, vol. 28, no. 9.

[67] Guo, J.-M. and Liu, Y.-F. License plate localization and character segmentation with feedback self-learning and hybrid binarization techniques. *IEEE Trans. on Vehicular Technology*. 2008, vol. 57, no. 3.

[68] Gupte, S., Masoud, O., Martin, R. F. and Papanikolopoulos, N. P. Detection and classification of vehicles. *IEEE Trans. on Intelligent Transportation Systems (T-ITS)*. 2002, vol. 3, no. 1.

[69] Hadsell, R., Chopra, S. and LeCun, Y. Dimensionality Reduction by Learning an Invariant Mapping. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2006, vol. 2. ISSN 1063-6919.

[70] Harris, C. and Stephens, M. A combined corner and edge detector. In: Citeseer. *Alvey vision conference*. 1988, vol. 15.

[71] Hartley, R. I. and Zisserman, A. *Multiple View Geometry in Computer Vision*. Secondth ed. 2004.

[72] Havránek, P., Zøuvala, R., Špaňhel, J., Herout, A., Valentová, V. et al. How does road marking in horizontal curves influence driving behaviour? *European transport research review*. 2020, vol. 12.

[73] He, B., Li, J., Zhao, Y. and Tian, Y. Part-regularized near-duplicate vehicle re-identification. In: *IEEE/CVF Computer Vision and Pattern Recognition (CVPR)*. 2019.

[74] He, H., Shao, Z. and Tan, J. Recognition of Car Makes and Models From a Single Traffic-Camera Image. *IEEE Trans. on Intelligent Transportation Systems (T-ITS)*. 2015, PP, no. 99. ISSN 1524-9050.

[75] He, J., Chen, J.-N., Liu, S., Kortylewski, A., Yang, C. et al. Transfg: A transformer architecture for fine-grained recognition. In: *AAAI Conference on Artificial Intelligence*. 2022, vol. 36, no. 1, p. 852–860.

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

[77] He, S., Luo, H., Wang, P., Wang, F., Li, H. et al. Transreid: Transformer-based object re-identification. In: *IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021.

[78] He, X. C. and Yung, N. H. C. A Novel Algorithm for Estimating Vehicle Speed from Two Consecutive Images. In: *IEEE Workshop on Applications of Computer Vision (WACV)*. 2007. ISSN 1550-5790.

[79] He, X. C. and Yung, N. H. C. New method for overcoming ill-conditioning in vanishing-point-based camera calibration. *Optical Engineering*. 2007, vol. 46, no. 3.

[80] Hermans, A., Beyer, L. and Leibe, B. In Defense of the Triplet Loss for Person Re-Identification. *ArXiv preprint arXiv:1703.07737* [arXiv:1703.07737]. 2017.

[81] Herout, A., Jošth, R., Juránek, R., Havel, J., Hradiš, M. et al. Real-time object detection on CUDA. *Journal of Real-Time Image Processing.* 2011, vol. 6, no. 3. ISSN 1861-8219.

[82] Hirzer, M., Beleznai, C., Roth, P. M. and Bischof, H. Person Re-Identification by Descriptive and Discriminative Classification. In: *Proc. Scandinavian Conference on Image Analysis (SCIA).* 2011.

[83] Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W. et al. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *ArXiv preprint arXiv:1704.04861.* 2017.

[84] Hsiao, E., Sinha, S., Ramnath, K., Baker, S., Zitnick, L. et al. Car make and model recognition using 3D curve alignment. In: *IEEE Workshop on Applications of Computer Vision (WACV).* 2014.

[85] Hsieh, J.-W., Chen, L.-C. and Chen, D.-Y. Symmetrical SURF and Its Applications to Vehicle Detection and Vehicle Make and Model Recognition. *IEEE Trans. on Intelligent Transportation Systems (T-ITS).* 2014, vol. 15, no. 1. ISSN 1524-9050.

[86] Hsu, G. S., Chen, J. C. and Chung, Y. Z. Application-Oriented License Plate Recognition. *IEEE Trans. on Vehicular Technology.* 2013, vol. 62, no. 2. ISSN 0018-9545.

[87] Hu, A., Sun, Z., Li, Q., Xu, Y., Zhu, Y. et al. Fine-grained traffic video vehicle recognition based orientation estimation and temporal information. *Multimedia Tools and Applications.* 2023, vol. 82, no. 9, p. 13745–13763.

[88] Hu, C., Bai, X., Qi, L., Wang, X., Xue, G. et al. Learning Discriminative Pattern for Real-Time Car Brand Recognition. *IEEE Trans. on Intelligent Transportation Systems (T-ITS).* 2015, vol. 16, no. 6. ISSN 1524-9050.

[89] Hu, Q., Wang, H., Li, T. and Shen, C. Deep CNN|s With Spatially Weighted Pooling for Fine-Grained Car Recognition. *IEEE Trans. on Intelligent Transportation Systems (T-ITS).* 2017, PP, no. 99. ISSN 1524-9050.

[90] Hu, Y., Jin, X., Zhang, Y., Hong, H., Zhang, J. et al. Rams-trans: Recurrent attention multi-scale transformer for fine-grained image recognition. In: *ACM International Conference on Multimedia.* 2021, p. 4239–4248.

[91] Huang, Q., Cai, Z. and Lan, T. A single neural network for mixed style license plate detection and recognition. *IEEE Access.* 2021, vol. 9, p. 21777–21785.

[92] Huang, S., Xu, Z., Tao, D. and Zhang, Y. Part-Stacked CNN for Fine-Grained Visual Categorization. In: *IEEE Computer Vision and Pattern Recognition (CVPR).* 2016.

[93] Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *ArXiv preprint arXiv:1502.03167.* 2015.

[94] Jaderberg, M., Simonyan, K., Vedaldi, A. and Zisserman, A. Reading Text in the Wild with Convolutional Neural Networks. *International Journal of Computer Vision*. 2016, vol. 116, no. 1. ISSN 1573-1405.

[95] Jain, V., Sasindran, Z., Rajagopal, A., Biswas, S., Bharadwaj, H. S. et al. Deep Automatic License Plate Recognition System. In: *Indian Conference on Computer Vision, Graphics and Image Processing*. New York, NY, USA: [b.n.], 2016. ICVGIP '16. ISBN 978-1-4503-4753-2.

[96] Jamieson, N. *Clear Zones, Barriers and Driving Lines – Mitigating the Effects of Crashes on Corners (Horizontal Curves)*. 12-529B33. Lower Hutt, New Zealand, 2012.

[97] Jiang, M., Zhang, X., Yu, Y., Bai, Z., Zheng, Z. et al. Robust vehicle re-identification via rigid structure prior. In: *IEEE/CVF Computer Vision and Pattern Recognition (CVPR)*. 2021.

[98] Jiang, Y., Jiang, F., Luo, H., Lin, H., Yao, J. et al. An Efficient and Unified Recognition Method for Multiple License Plates in Unconstrained Scenarios. *IEEE Trans. on Intelligent Transportation Systems (T-ITS)*. 2023.

[99] Johari, M., Keyvan Ekbatani, M., Leclercq, L., Ngoduy, D. and Mahmassani, H. S. Macroscopic network-level traffic models: Bridging fifty years of development toward the next era. *Transportation Research Part C: Emerging Technologies*. 2021, vol. 131.

[100] Juránek, R., Herout, A., Dubská, M. and Zemčík, P. Real-Time Pose Estimation Piggybacked on Object Detection. In: *IEEE International Conference on Computer Vision (ICCV)*. 2015.

[101] Kalman, R. E. A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME–Journal of Basic Engineering*. 1960, vol. 82, Series D.

[102] Kanaci, A., Zhu, X. and Gong, S. Vehicle re-identification in context. In: Springer. *International Conference on Pattern Recognition (ICPR)*. 2019.

[103] Kanal, L. N. Problem-solving models and search strategies for pattern recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence (T-PAMI)*. 1979, no. 2.

[104] Ke, X., Zeng, G. and Guo, W. An Ultra-Fast Automatic License Plate Recognition Approach for Unconstrained Scenarios. *IEEE Trans. on Intelligent Transportation Systems (T-ITS)*. 2023.

[105] Khan, F. M. and Brèmond, F. Multi-shot Person Re-Identification Using Part Appearance Mixture. In: *IEEE Winter Conference on Applications of Computer Vision (WACV)*. 2017.

[106] Kinga, D. and Adam, J. B. A method for stochastic optimization. In: *International Conference on Learning Representations (ICLR)*. 2015.

[107] Kluwak, K., Segen, J., Kulbacki, M., Drabik, A. and Wojciechowski, K. ALPR - Extension to Traditional Plate Recognition Methods. In: *Intelligent Information and Database Systems*. Berlin, Heidelberg: [b.n.], 2016. ISBN 978-3-662-49390-8.

[108] Krause, J., Gebru, T., Deng, J., Li, L. J. and Fei Fei, L. Learning Features and Parts for Fine-Grained Recognition. In: *International Conference on Pattern Recognition (ICPR)*. 2014. ISSN 1051-4651.

[109] Krause, J., Jin, H., Yang, J. and Fei Fei, L. Fine-Grained Recognition without Part Annotations. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2015.

[110] Krause, J., Stark, M., Deng, J. and Fei Fei, L. 3D Object Representations for Fine-Grained Categorization. In: *IEEE International Conference on Computer Vision Workshops (ICCVW)*. 2013.

[111] Krizhevsky, A., Sutskever, I. and Hinton, G. E. ImageNet Classification with Deep Convolutional Neural Networks. In: Pereira, F., Burges, C., Bottou, L. and Weinberger, K., ed. *Advances in Neural Information Processing Systems (NIPS)*. 2012.

[112] Kumar, R., Weill, E., Aghdasi, F. and Sriram, P. Vehicle re-identification: an efficient baseline using triplet embedding. *ArXiv preprint arXiv:1901.01015*. 2019.

[113] Köstinger, M., Hirzer, M., Wohlhart, P., Roth, P. M. and Bischof, H. Large scale metric learning from equivalence constraints. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2012. ISSN 1063-6919.

[114] Lan, J., Li, J., Hu, G., Ran, B. and Wang, L. Vehicle speed measurement based on gray constraint optical flow algorithm. *International Journal for Light and Electron Optics*. 2014, vol. 125, no. 1. ISSN 0030-4026.

[115] Laroca, R., Estevam, V., Britto, A. S., Minetto, R. and Menotti, D. Do we train on test data? The impact of near-duplicates on license plate recognition. In: IEEE. *International Joint Conference on Neural Networks (IJCNN)*. 2023.

[116] Laroca, R., Severo, E., Zanlorensi, L. A., Oliveira, L. S., Gonçalves, G. R. et al. A Robust Real-Time Automatic License Plate Recognition based on the YOLO Detector. *CoRR*. 2018. Available at: http://arxiv.org/abs/1802.09567.

[117] Lee, S., Gwak, J. and Jeon, M. Vehicle model recognition in video. *International Journal of Signal Processing, Image Processing and Pattern Recognition*. 2013, vol. 6, no. 2.

[118] Leibe, B., Cornelis, N., Cornelis, K. and Van Gool, L. Dynamic 3D Scene Analysis from a Moving Vehicle. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2007. ISSN 1063-6919.

[119] Leibe, B. and Schiele, B. Analyzing appearance and contour based methods for object categorization. In: IEEE. *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2003, vol. 2.

[120] Li, D., Chen, X., Zhang, Z. and Huang, K. Learning Deep Context-Aware Features Over Body and Latent Parts for Person Re-Identification. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2017.

[121] Li, H. and Shen, C. Reading Car License Plates Using Deep Convolutional Neural Networks and LSTMs. *ArXiv preprint arXiv:1601.05610*. 2016.

[122] Li, H., Wang, P. and Shen, C. Towards End-to-End Car License Plates Detection and Recognition with Deep Neural Networks. *ArXiv preprint arXiv:1709.08828*. 2017.

[123] Li, L., Guo, Y., Xie, L., Kong, X. and Tian, Q. Fine-Grained Visual Categorization with Fine-Tuned Segmentation. *IEEE International Conference on Image Processing (ICIP)*. 2015.

[124] Li, S., Bak, S., Carr, P. and Wang, X. Diversity regularized spatiotemporal attention for video-based person re-identification. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2018.

[125] Liao, S., Hu, Y., Zhu, X. and Li, S. Z. Person Re-Identification by Local Maximal Occurrence Representation and Metric Learning. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2015.

[126] Lin, D., Shen, X., Lu, C. and Jia, J. Deep LAC: Deep Localization, Alignment and Classification for Fine-Grained Recognition. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2015.

[127] Lin, J., Ren, L., Lu, J., Feng, J. and Zhou, J. Consistent-Aware Deep Learning for Person Re-Identification in a Camera Network. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2017.

[128] Lin, T.-Y., RoyChowdhury, A. and Maji, S. Bilinear CNN Models for Fine-grained Visual Recognition. In: *IEEE International Conference on Computer Vision (ICCV)*. 2015.

[129] Lin, Y.-L., Morariu, V. I., Hsu, W. and Davis, L. S. Jointly Optimizing 3D Model Fitting and Fine-Grained Classification. In: *IEEE European Conference on Computer Vision (ECCV)*. 2014.

[130] Lin, Y., Zheng, L., Zheng, Z., Wu, Y., Hu, Z. et al. Improving person re-identification by attribute and identity learning. *International Conference on Pattern Recognition (ICPR)*. 2019, vol. 95.

[131] Liu, C., Zhang, Y., Luo, H., Tang, J., Chen, W. et al. City-scale multi-camera vehicle tracking guided by crossroad zones. In: *IEEE/CVF Computer Vision and Pattern Recognition (CVPR)*. 2021.

[132] Liu, H., Tian, Y., Yang, Y., Pang, L. and Huang, T. Deep Relative Distance Learning: Tell the Difference Between Similar Vehicles. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2016.

[133] Liu, K., Ma, B., Zhang, W. and Huang, R. A Spatio-Temporal Appearance Representation for Viceo-Based Pedestrian Re-Identification. In: *IEEE International Conference on Computer Vision (ICCV)*. 2015.

[134] Liu, W., Anguelov, D., Erhan, D., Szegedy, C. and Reed, S. SSD: Single Shot MultiBox Detector. *ArXiv preprint arXiv:1512.02325*. 2015.

[135] Liu, W., Chen, C., Wong, K.-Y. K., Su, Z. and Han, J. Star-net: a spatial attention residue network for scene text recognition. In: *British Machine Vision Conference (BMVC)*. 2016, vol. 2.

[136] Liu, X., Zhang, S., Huang, Q. and Gao, W. Ram: a region-aware deep model for vehicle re-identification. In: IEEE. *IEEE International Conference on Multimedia and Expo (ICME)*. 2018.

[137] Liu, X., Liu, W., Ma, H. and Fu, H. Large-scale vehicle re-identification in urban surveillance videos. In: IEEE. *IEEE International Conference on Multimedia and Expo (ICME)*. 2016.

[138] Liu, X., Liu, W., Mei, T. and Ma, H. A Deep Learning-Based Approach to Progressive Vehicle Re-identification for Urban Surveillance. In: Springer. *IEEE European Conference on Computer Vision (ECCV)*. 2016.

[139] Liu, Y., Zhang, D., Zhang, Q. and Han, J. Part-object relational visual saliency. *IEEE Trans. on Pattern Analysis and Machine Intelligence (T-PAMI)*. 2021, vol. 44, no. 7.

[140] Liu, Y., Yan, J. and Ouyang, W. Quality aware network for set to set recognition. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2017.

[141] Llorca, D. F., Colás, D., Daza, I. G., Parra, I. and Sotelo, M. A. Vehicle model recognition using geometry and appearance of car emblems from rear view images. In: *IEEE Intelligent Transportation Systems Conference (ITSC)*. 2014. ISSN 2153-0009.

[142] Lou, Y., Bai, Y., Liu, J., Wang, S. and Duan, L.-Y. Embedding adversarial learning for vehicle re-identification. *IEEE Trans. on Image Processing*. 2019, vol. 28, no. 8.

[143] Lowe, D. G. Object recognition from local scale-invariant features. In: Ieee. *IEEE International Conference on Computer Vision (ICCV)*. 1999, vol. 2.

[144] Lu, L., Cai, Y., Huang, H. and Wang, P. An efficient fine-grained vehicle recognition method based on part-level feature optimization. *Neurocomputing*. 2023, vol. 536, p. 40–49.

[145] Lu, L. and Huang, H. Component-based feature extraction and representation schemes for vehicle make and model recognition. *Neurocomputing*. 2020, vol. 372, p. 92–99.

[146] Lu, L., Wang, P. and Cao, Y. A novel part-level feature extraction method for fine-grained vehicle recognition. *International Conference on Pattern Recognition (ICPR)*. 2022, vol. 131.

[147] Luo, H., Gu, Y., Liao, X., Lai, S. and Jiang, W. Bag of tricks and a strong baseline for deep person re-identification. In: *IEEE/CVF Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2019.

[148] Luo, H., Jiang, W., Gu, Y., Liu, F., Liao, X. et al. A strong baseline and batch normalization neck for deep person re-identification. *IEEE Trans. on Multimedia.* 2019, vol. 22, no. 10.

[149] Luvizon, D., Nassu, B. and Minetto, R. Vehicle speed estimation by license plate detection and tracking. In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* 2014.

[150] Lyu, S., Chang, M.-C., Du, D., Wen, L., Qi, H. et al. UA-DETRAC 2017: Report of AVSS2017 & IWT4S Challenge on Advanced Traffic Monitoring. In: IEEE. *IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS).* 2017.

[151] Ma, X. and Grimson, W. E. L. Edge-based rich representation for vehicle classification. In: IEEE. *IEEE International Conference on Computer Vision (ICCV).* 2005, vol. 2.

[152] Maduro, C., Batista, K., Peixoto, P. and Batista, J. Estimation of vehicle velocity and traffic intensity using rectified images. In: *IEEE International Conference on Image Processing (ICIP).* 2008. ISSN 1522-4880.

[153] Mainka, A. *Smart world cities in the 21st century.* 2018.

[154] Marr, D. and Hildreth, E. Theory of edge detection. *Royal Society of London B: Biological Sciences.* 1980, vol. 207, no. 1167, p. 187–217.

[155] Masood, S. Z., Shu, G., Dehghan, A. and Ortiz, E. G. License Plate Detection and Recognition Using Deeply Learned Convolutional Neural Networks. *ArXiv preprint arXiv:1703.07330.* 2017.

[156] Matsukawa, T., Okabe, T., Suzuki, E. and Sato, Y. Hierarchical Gaussian Descriptor for Person Re-Identification. In: *IEEE Computer Vision and Pattern Recognition (CVPR).* 2016.

[157] Matzen, K. and Snavely, N. NYC3DCars: A Dataset of 3D Vehicles in Geographic Context. In: *IEEE International Conference on Computer Vision (ICCV).* 2013.

[158] McLaughlin, N., Rincon, J. Martinez del and Miller, P. Recurrent Convolutional Network for Video-Based Person Re-Identification. In: *IEEE Computer Vision and Pattern Recognition (CVPR).* 2016.

[159] Mikolajczyk, K. and Schmid, C. A performance evaluation of local descriptors. *IEEE Trans. on Pattern Analysis and Machine Intelligence (T-PAMI).* 2005, vol. 27, no. 10.

[160] Moravec, H. P. *Obstacle avoidance and navigation in the real world by a seeing robot rover.* 1980.

[161] Murthy, S. K. Automatic construction of decision trees from data: A multi-disciplinary survey. *Data Mining and Knowledge Discovery.* 1998, vol. 2, no. 4.

[162] Newell, A., Yang, K. and Deng, J. Stacked hourglass networks for human pose estimation. In: Springer. *IEEE European Conference on Computer Vision (ECCV)*. 2016.

[163] Nomura, S., Yamanaka, K., Katai, O., Kawakami, H. and Shiose, T. A novel adaptive morphological approach for degraded character image segmentation. *International Conference on Pattern Recognition (ICPR)*. 2005, vol. 38, no. 11.

[164] Nurhadiyatna, A., Hardjono, B., Wibisono, A., Sina, I., Jatmiko, W. et al. Improved vehicle speed estimation using Gaussian mixture model and hole filling algorithm. In: *International Conference on Advanced Computer Science and Information Systems (ICACSIS)*. 2013.

[165] Oliveira, I. O. de and Sousa Pio, J. L. de. Object reidentification in multiple cameras system. In: IEEE. *International Conference on Embedded and Multimedia Computing*. 2009.

[166] Opelt, A., Fussenegger, M., Pinz, A. and Auer, P. Generic object recognition with boosting. *IEEE Trans. on Pattern Analysis and Machine Intelligence (T-PAMI)*. 2004, TR-EMT-2004-01.

[167] Ottlik, A. and Nagel, H.-H. Initialization of model-based vehicle tracking in video sequences of inner-city intersections. *International Journal of Computer Vision*. 2008, vol. 80, no. 2, p. 211–225.

[168] Özuysal, M., Lepetit, V. and Fua, P. Pose estimation for category specific multiview object localization. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2009. ISSN 1063-6919.

[169] Papageorgiou, C. and Poggio, T. *A Trainable Object Detection System: Car Detection in Static Images*. 1673. 1999. (CBCL Memo 180).

[170] Pearce, G. and Pears, N. Automatic make and model recognition from frontal images of cars. In: *IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*. 2011.

[171] Petrovic, V. and Cootes, T. F. Analysis of Features for Rigid Structure Vehicle Type Recognition. In: *British Machine Vision Conference (BMVC)*. 2004.

[172] Pirsiavash, H., Ramanan, D. and Fowlkes, C. C. Bilinear classifiers for visual recognition. In: Bengio, Y., Schuurmans, D., Lafferty, J., Williams, C. and Culotta, A., ed. *Advances in Neural Information Processing Systems (NIPS)*. 2009.

[173] Prokaj, J. and Medioni, G. 3-D model based vehicle recognition. In: *IEEE Workshop on Applications of Computer Vision (WACV)*. 2009. ISSN 1550-5790.

[174] Psyllos, A., Anagnostopoulos, C. and Kayafas, E. Vehicle model recognition from frontal view image measurements. *Computer Standards & Interfaces*. 2011, vol. 33, no. 2. ISSN 0920-5489.

[175] Qian, J., Jiang, W., Luo, H. and Yu, H. Stripe-based and attribute-aware network: A two-branch deep model for vehicle re-identification. *Measurement Science and Technology*. 2020, vol. 31, no. 9, p. 095401.

[176] Qiao, S., Zhu, Y., Li, X., Liu, T. and Zhang, B. Research of improving the accuracy of license plate character segmentation. In: IEEE. *Frontier of Computer Science and Technology (FCST)*. 2010.

[177] Qin, S. and Liu, S. Towards end-to-end car license plate location and recognition in unconstrained scenarios. *Neural Computing and Applications*. 2022, vol. 34, no. 24, p. 21551–21566.

[178] Rasheed, S., Naeem, A. and Ishaq, O. Automated number plate recognition using Hough lines and template matching. In: *World Congress on Engineering and Computer Science*. 2012, vol. 1.

[179] Redmon, J., Divvala, S., Girshick, R. and Farhadi, A. You only look once: Unified, real-time object detection. *ArXiv preprint arXiv:1506.02640*. 2015.

[180] Redmon, J., Divvala, S. K., Girshick, R. B. and Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. *CoRR*. 2015, abs/1506.02640. Available at: http://arxiv.org/abs/1506.02640.

[181] Redmon, J. and Farhadi, A. YOLOv3: An Incremental Improvement. *ArXiv*. 2018.

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

[183] Rothe, R., Timofte, R. and Van Gool, L. Deep Expectation of Real and Apparent Age from a Single Image Without Facial Landmarks. *International Journal of Computer Vision*. 2016. ISSN 1573-1405. Available at: http://dx.doi.org/10.1007/s11263-016-0940-3.

[184] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S. et al. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*. 2015.

[185] Sanakoyeu, A., Tschernezki, V., Buchler, U. and Ommer, B. Divide and conquer the embedding space for metric learning. In: *IEEE/CVF Computer Vision and Pattern Recognition (CVPR)*. 2019.

[186] Savarese, S. and Fei Fei, L. 3D generic object categorization, localization and pose estimation. In: IEEE. *IEEE International Conference on Computer Vision (ICCV)*. 2007.

[187] Schoepflin, T. and Dailey, D. Dynamic camera calibration of roadside traffic management cameras for vehicle speed estimation. *IEEE Trans. on Intelligent Transportation Systems (T-ITS)*. 2003, vol. 4, no. 2. ISSN 1524-9050.

[188] Schroff, F., Kalenichenko, D. and Philbin, J. Facenet: A unified embedding for face recognition and clustering. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2015.

[189] Schwab, K. *The fourth industrial revolution*. 2017.

[190] Shen, Y., Xiao, T., Li, H., Yi, S. and Wang, X. Learning Deep Neural Networks for Vehicle Re-ID With Visual-Spatio-Temporal Path Proposals. In: *IEEE International Conference on Computer Vision (ICCV)*. 2017.

[191] Shi, H., Yang, Y., Zhu, X., Liao, S., Lei, Z. et al. Embedding Deep Metric for Person Re-identification: A Study Against Large Variations. In: Leibe, B., Matas, J., Sebe, N. and Welling, M., ed. *IEEE European Conference on Computer Vision (ECCV)*. Cham: [b.n.], 2016. ISBN 978-3-319-46448-0.

[192] Shrivastava, A., Gupta, A. and Girshick, R. Training region-based object detectors with online hard example mining. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2016.

[193] Simon, M. and Rodner, E. Neural Activation Constellations: Unsupervised Part Model Discovery with Convolutional Networks. In: *IEEE International Conference on Computer Vision (ICCV)*. 2015.

[194] Simonyan, K. and Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *CoRR*. 2014, abs/1409.1556.

[195] Sina, I., Wibisono, A., Nurhadiyatna, A., Hardjono, B., Jatmiko, W. et al. Vehicle counting and speed measurement using headlight detection. In: *International Conference on Advanced Computer Science and Information Systems (ICACSIS)*. 2013.

[196] Sivic, J. and Zisserman, A. Video Google: A text retrieval approach to object matching in videos. In: IEEE. *IEEE International Conference on Computer Vision (ICCV)*. 2003.

[197] Sobel, I. and Feldman, G. A 3x3 isotropic gradient operator for image processing. *Talk at the Stanford Artificial Intelligence Project (SAIL)*. 1968.

[198] Sochor, J., Herout, A. and Havel, J. BoxCars: 3D Boxes as CNN Input for Improved Fine-Grained Vehicle Recognition. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2016.

[199] Sochor, J., Juránek, R. and Herout, A. Traffic surveillance camera calibration by 3D model bounding box alignment for accurate vehicle speed measurement. *Computer Vision and Image Understanding*. 2017, vol. 161. ISSN 1077-3142.

[200] Sochor, J., Juránek, R., Špaňhel, J., Maršík, L., Širokỳ, A. et al. Comprehensive data set for automatic single camera visual speed measurement. *IEEE Trans. on Intelligent Transportation Systems (T-ITS)*. 2018, vol. 20, no. 5.

[201] Sochor, J., Juránek, R., Špaňhel, J., Maršík, L., Široký, A. et al. *BrnoCompSpeed: Review of Traffic Camera Calibration and A Comprehensive Dataset for Monocular Speed Measurement* [arXiv:1702.06441]. 2017.

[202] Sochor, J., Špaňhel, J. and Herout, A. BoxCars: Improving Fine-Grained Recognition of Vehicles Using 3-D Bounding Boxes in Traffic Surveillance. *IEEE Trans. on Intelligent Transportation Systems (T-ITS)*. 2018, PP, no. 99. ISSN 1524-9050.

[203] Sochor, J., Špaňhel, J., Juránek, R., Dobes, P. and Herout, A. Graph@ fit submission to the nvidia ai city challenge 2018. In: *IEEE Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2018.

[204] Song, L., Wang, C., Zhang, L., Du, B., Zhang, Q. et al. Unsupervised domain adaptive re-identification: Theory and practice. *International Conference on Pattern Recognition (ICPR)*. 2020, vol. 102.

[205] Špaňhel, J., Bartl, V., Juránek, R. and Herout, A. Vehicle re-identification and multi-camera tracking in challenging city-scale environment. In: *IEEE Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2019, vol. 2.

[206] Špaňhel, J., Juránek, R., Herout, A., Novák, J. and Havránek, P. Analysis of vehicle trajectories for determining cross-sectional load density based on computer vision. In: IEEE. *IEEE Intelligent Transportation Systems Conference (ITSC)*. 2019.

[207] Špaňhel, J., Sochor, J., Juránek, R. and Herout, A. Geometric Alignment by Deep Learning for Recognition of Challenging License Plates. In: *IEEE Intelligent Transportation Systems Conference (ITSC)*. 2018.

[208] Špaňhel, J., Sochor, J., Juránek, R., Herout, A., Maršík, L. et al. Holistic Recognition of Low Quality License Plates by CNN using Track Annotated Data. In: IEEE. *IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*. Lecce, IT: [b.n.], 2017. ISBN 978-1-5386-2939-0.

[209] Špaňhel, J., Sochor, J., Juránek, R., Dobeš, P., Bartl, V. et al. Learning Feature Aggregation in Temporal Domain for Re-Identificationn. *Computer Vision and Image Understanding*. 2020, vol. 192. ISSN 1077-3142.

[210] Stark, M., Krause, J., Pepik, B., Meger, D., Little, J. et al. Fine-Grained Categorization for 3D Scene Understanding. In: *British Machine Vision Conference (BMVC)*. 2012.

[211] Stauffer, C. and Grimson, W. E. L. Adaptive background mixture models for real-time tracking. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 1999, vol. 2.

[212] Su, C., Li, J., Zhang, S., Xing, J., Gao, W. et al. Pose-driven deep convolutional model for person re-identification. In: *IEEE/CVF International Conference on Computer Vision (ICCV)*. 2017.

[213] Su, C., Zhang, S., Xing, J., Gao, W. and Tian, Q. Deep Attributes Driven Multi-camera Person Re-identification. In: Leibe, B., Matas, J., Sebe, N. and Welling, M., ed. *IEEE European Conference on Computer Vision (ECCV)*. Cham: [b.n.], 2016. ISBN 978-3-319-46475-6.

[214] Sun, Y., Zheng, L., Deng, W. and Wang, S. SVDNet for Pedestrian Retrieval. *IEEE International Conference on Computer Vision (ICCV)*. 2017.

[215] Svoboda, P., Hradiš, M., Maršík, L. and Zemčík, P. CNN for license plate motion deblurring. In: *IEEE International Conference on Image Processing (ICIP)*. 2016.

[216] Szegedy, C., Ioffe, S., Vanhoucke, V. and Alemi, A. A. Inception-v4, inception-resnet and the impact of residual connections on learning. In: *AAAI*. 2017, vol. 4.

[217] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S. et al. Going Deeper with Convolutions. *CoRR*. 2014, abs/1409.4842. Available at: http://arxiv.org/abs/1409.4842.

[218] Szegedy, C., Reed, S., Erhan, D., Anguelov, D. and Ioffe, S. Scalable, high-quality object detection. *ArXiv preprint arXiv:1412.1441*. 2014.

[219] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J. and Wojna, Z. Rethinking the inception architecture for computer vision. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2016.

[220] Taigman, Y., Yang, M., Ranzato, M. and Wolf, L. DeepFace: Closing the Gap to Human-Level Performance in Face Verification. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2014.

[221] Tan, X., Wang, Z., Jiang, M., Yang, X., Wang, J. et al. Multi-camera vehicle tracking and re-identification based on visual and spatial-temporal features. In: *IEEE Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2019.

[222] Tang, X., Wang, W., Song, H. and Zhao, C. CenterLoc3D: monocular 3D vehicle localization network for roadside surveillance cameras. *Complex & Intelligent Systems*. 2023, p. 1–20.

[223] Tang, Y., Wu, D., Jin, Z., Zou, W. and Li, X. Multi-modal metric learning for vehicle re-identification in traffic surveillance environment. In: IEEE. *IEEE International Conference on Image Processing (ICIP)*. 2017.

[224] Tang, Z., Naphade, M., Liu, M.-Y., Yang, X., Birchfield, S. et al. CityFlow: A City-Scale Benchmark for Multi-Target Multi-Camera Vehicle Tracking and Re-Identification. *ArXiv preprint arXiv:1903.09254*. 2019.

[225] Teng, S., Liu, X., Zhang, S. and Huang, Q. Scan: Spatial and channel attention network for vehicle re-identification. In: Springer. *Advances in Multimedia Information Processing*. 2018.

[226] Tsai, L.-W., Hsieh, J.-W. and Fan, K.-C. Vehicle detection using normalized color and edge map. *IEEE Trans. on Image Processing*. 2007, vol. 16, no. 3, p. 850–864.

[227] Van De Weijer, J., Schmid, C., Verbeek, J. and Larlus, D. Learning color names for real-world applications. *IEEE Trans. on Image Processing*. 2009, vol. 18, no. 7.

[228] Viola, P. and Jones, M. J. Robust Real-Time Face Detection. *International Journal of Computer Vision*. Hingham, MA, USA: [b.n.]. 2004, vol. 57, no. 2. ISSN 0920-5691.

[229] Wang, F., Zuo, W., Lin, L., Zhang, D. and Zhang, L. Joint Learning of Single-Image and Cross-Image Representations for Person Re-Identification. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2016.

[230] Wang, F., Jiang, M., Qian, C., Yang, S., Li, C. et al. Residual attention network for image classification. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2017.

[231] Wang, H., Mazari, M., Pourhomayoun, M., Smith, J., Owens, H. et al. An end-to-end traffic vision and counting system using computer vision and machine learning: the challenges in real-time processing. *SIGNAL 2018 Editors*. 2018.

[232] Wang, J., Yang, J., Yu, K., Lv, F., Huang, T. et al. Locality-constrained Linear Coding for image classification. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2010. ISSN 1063-6919.

[233] Wang, P., Jiao, B., Yang, L., Yang, Y., Zhang, S. et al. Vehicle re-identification in aerial imagery: Dataset and approach. In: *IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019.

[234] Wang, T., Gong, S., Zhu, X. and Wang, S. Person Re-identification by Video Ranking. In: *IEEE European Conference on Computer Vision (ECCV)*. 2014. ISBN 978-3-319-10593-2.

[235] Wang, Z., Tang, L., Liu, X., Yao, Z., Yi, S. et al. Orientation Invariant Feature Embedding and Spatial Temporal Regularization for Vehicle Re-Identification. In: *IEEE International Conference on Computer Vision (ICCV)*. 2017.

[236] Wei, Y., Song, N., Ke, L., Chang, M.-C. and Lyu, S. Street object detection / tracking for AI city traffic analysis. *IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*. 2017.

[237] Wen, Y., Lu, Y., Yan, J., Zhou, Z., Deneen, K. M. von et al. An algorithm for license plate recognition applied to intelligent transportation system. *IEEE Trans. on Intelligent Transportation Systems (T-ITS)*. 2011, vol. 12, no. 3.

[238] Wojke, N., Bewley, A. and Paulus, D. Simple online and realtime tracking with a deep association metric. *IEEE International Conference on Image Processing (ICIP)*. 2017.

[239] Wu, S., Chen, Y. C., Li, X., Wu, A. C., You, J. J. et al. An enhanced deep feature representation for person re-identification. In: *IEEE Winter Conference on Applications of Computer Vision (WACV)*. 2016.

[240] Xiang, Y. and Savarese, S. Estimating the aspect layout of object categories. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2012.

[241] Xiao, T., Li, H., Ouyang, W. and Wang, X. Learning Deep Feature Representations With Domain Guided Dropout for Person Re-Identification. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2016.

[242] Xie, S., Yang, T., Wang, X. and Lin, Y. Hyper-Class Augmented and Regularized Deep Learning for Fine-Grained Image Classification. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2015.

[243] Xu, S., Cheng, Y., Gu, K., Yang, Y., Chang, S. et al. Jointly Attentive Spatial-Temporal Pooling Networks for Video-based Person Re-Identification. In: *IEEE International Conference on Computer Vision (ICCV)*. 2017.

[244] Yan, K., Tian, Y., Wang, Y., Zeng, W. and Huang, T. Exploiting Multi-Grain Ranking Constraints for Precisely Searching Visually-Similar Vehicles. In: *IEEE International Conference on Computer Vision (ICCV)*. 2017.

[245] Yan, Y., Ni, B., Song, Z., Ma, C., Yan, Y. et al. Person Re-identification via Recurrent Feature Aggregation. In: Leibe, B., Matas, J., Sebe, N. and Welling, M., ed. *IEEE European Conference on Computer Vision (ECCV)*. Cham: [b.n.], 2016. ISBN 978-3-319-46466-4.

[246] Yang, F., Li, K., Zhong, Z., Luo, Z., Sun, X. et al. Asymmetric co-teaching for unsupervised cross-domain person re-identification. In: *AAAI Conference on Artificial Intelligence*. 2020, vol. 34, no. 07, p. 12597–12604.

[247] Yang, J., Ren, P., Zhang, D., Chen, D., Wen, F. et al. Neural Aggregation Network for Video Face Recognition. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2017.

[248] Yang, J., Price, B., Cohen, S., Lee, H. and Yang, M.-H. Object contour detection with a fully convolutional encoder-decoder network. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2016.

[249] Yang, L., Luo, P., Change Loy, C. and Tang, X. A Large-Scale Car Dataset for Fine-Grained Categorization and Verification. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2015.

[250] Yang, S., Bo, L., Wang, J. and Shapiro, L. G. Unsupervised Template Learning for Fine-Grained Object Recognition. In: Pereira, F., Burges, C., Bottou, L. and Weinberger, K., ed. *Advances in Neural Information Processing Systems (NIPS)*. 2012.

[251] Yao, B. A Codebook-free and Annotation-free Approach for Fine-grained Image Categorization. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. Washington, DC, USA: [b.n.], 2012. CVPR '12. ISBN 978-1-4673-1226-4.

[252] Yao, Y., Zheng, L., Yang, X., Naphade, M. and Gedeon, T. Simulating content consistent vehicle datasets with attribute descent. In: Springer. *IEEE European Conference on Computer Vision (ECCV)*. 2020.

[253] Ye, J., Yang, X., Kang, S., He, Y., Zhang, W. et al. A robust mtmc tracking system for ai-city challenge 2021. In: *IEEE/CVF Computer Vision and Pattern Recognition (CVPR)*. 2021.

[254] You, J., Wu, A., Li, X. and Zheng, W.-S. Top-Push Video-Based Person Re-Identification. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2016.

[255] Yuan, Y., Yang, K. and Zhang, C. Hard-aware deeply cascaded embedding. In: *IEEE/CVF International Conference on Computer Vision (ICCV)*. 2017.

[256] Zagoruyko, S., Lerer, A., Lin, T.-Y., Pinheiro, P. O., Gross, S. et al. A multipath network for object detection. *ArXiv preprint arXiv:1604.02135*. 2016.

[257] Zapletal, D. and Herout, A. Vehicle Re-Identification for Automatic Video Traffic Surveillance. In: *IEEE Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2016.

[258] Zeiler, M. D. and Fergus, R. Visualizing and understanding convolutional networks. In: Springer. *IEEE European Conference on Computer Vision (ECCV)*. 2014.

[259] Zemcik, P., Juranek, R., Musil, P., Musil, M. and Hradis, M. High performance architecture for object detection in streamed videos. In: *International Conference on Field programmable Logic and Applications*. 2013. ISSN 1946-147X.

[260] Zhai, Y., Ye, Q., Lu, S., Jia, M., Ji, R. et al. Multiple expert brainstorming for domain adaptive person re-identification. In: Springer. *IEEE European Conference on Computer Vision (ECCV)*. 2020.

[261] Zhang, B. Reliable Classification of Vehicle Types Based on Cascade Classifier Ensembles. *IEEE Trans. on Intelligent Transportation Systems (T-ITS)*. 2013, vol. 14, no. 1. ISSN 1524-9050.

[262] Zhang, B. Classification and identification of vehicle type and make by cortex-like image descriptor HMAX. *International Journal of Computational Vision and Robotics*. 2014, vol. 4. ISSN 1752-9131.

[263] Zhang, C., Liu, W., Ma, H. and Fu, H. Siamese neural network based gait recognition for human identification. In: IEEE. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2016.

[264] Zhang, L., Yang, Y., Wang, M., Hong, R., Nie, L. et al. Detecting Densely Distributed Graph Patterns for Fine-Grained Image Categorization. *IEEE Trans. on Image Processing*. 2016, vol. 25, no. 2. ISSN 1057-7149.

[265] Zhang, N., Farrell, R. and Darrell, T. Pose pooling kernels for sub-category recognition. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2012. ISSN 1063-6919.

[266] Zhang, W., Yu, X. and He, X. Learning Bidirectional Temporal Cues for Video-based Person Re-Identification. *IEEE Trans. on Circuits and Systems for Video Technology*. 2017, PP, no. 99. ISSN 1051-8215.

[267] Zhang, W., Hu, S. and Liu, K. Learning compact appearance representation for video-based person re-identification. *ArXiv preprint arXiv:1702.06294*. 2017.

[268] Zhang, W., Song, H., Liu, L., Li, C., Mu, B. et al. Vehicle localisation and deep model for automatic calibration of monocular camera in expressway scenes. *IET Intelligent Transport Systems*. 2022, vol. 16, no. 4.

[269] Zhang, X., Xiong, H., Zhou, W., Lin, W. and Tian, Q. Picking Deep Filter Responses for Fine-Grained Image Recognition. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2016.

[270] Zhang, X., Cao, J., Shen, C. and You, M. Self-training with progressive augmentation for unsupervised cross-domain person re-identification. In: *IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019.

[271] Zhang, X., Zhang, R., Cao, J., Gong, D., You, M. et al. Part-guided attention learning for vehicle re-identification. *ArXiv preprint arXiv:1909.06023*. 2019, vol. 2, no. 8.

[272] Zhang, Y., Liu, D. and Zha, Z.-J. Improving triplet-wise training of convolutional neural network for vehicle re-identification. In: IEEE. *IEEE International Conference on Multimedia and Expo (ICME)*. 2017.

[273] Zhao, H., Tian, M., Sun, S., Shao, J., Yan, J. et al. Spindle Net: Person Re-Identification With Human Body Region Guided Feature Decomposition and Fusion. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2017.

[274] Zheng, K., Lan, C., Zeng, W., Zhang, Z. and Zha, Z.-J. Exploiting sample uncertainty for domain adaptive person re-identification. In: *AAAI Conference on Artificial Intelligence*. 2021, vol. 35, no. 4, p. 3538–3546.

[275] Zheng, L., Bie, Z., Sun, Y., Wang, J., Su, C. et al. MARS: A video benchmark for large-scale person re-identification. In: Springer. *IEEE International Conference on Computer Vision (ICCV)*. 2016.

[276] Zheng, L., He, X., Samali, B. and Yang, L. T. Accuracy enhancement for license plate recognition. In: IEEE. *Int. Conference on Computer and Information Technology (CIT)*. 2010.

[277] Zheng, Z., Jiang, M., Wang, Z., Wang, J., Bai, Z. et al. Going beyond real data: A robust visual representation for vehicle re-identification. In: *IEEE/CVF Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2020.

[278] Zheng, Z., Zheng, L. and Yang, Y. Pedestrian alignment network for large-scale person re-identification. *IEEE Trans. on Circuits and Systems for Video Technology*. 2018, vol. 29, no. 10.

[279] Zhong, Z., Zheng, L., Cao, D. and Li, S. Re-Ranking Person Re-Identification With k-Reciprocal Encoding. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2017.

[280] Zhou, F. and Lin, Y. Fine-Grained Image Classification by Exploring Bipartite-Graph Labels. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2016.

[281] Zhou, S., Wang, J., Wang, J., Gong, Y. and Zheng, N. Point to Set Similarity Based Deep Feature Learning for Person Re-Identification. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2017.

[282] Zhou, Y. and Shao, L. Cross-view GAN based vehicle generation for re-identification. In: *British Machine Vision Conference (BMVC)*. 2017, vol. 1.

[283] Zhou, Y. and Shao, L. Aware attentive multi-view inference for vehicle re-identification. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2018.

[284] Zhou, Z., Huang, Y., Wang, W., Wang, L. and Tan, T. See the Forest for the Trees: Joint Spatial and Temporal Recurrent Neural Networks for Video-Based Person Re-Identification. In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2017.

[285] Zhu, J., Zeng, H., Huang, J., Liao, S., Lei, Z. et al. Vehicle re-identification using quadruple directional deep learning features. *IEEE Trans. on Intelligent Transportation Systems (T-ITS)*. 2019, vol. 21, no. 1.

[286] Zivkovic, Z. Improved Adaptive Gaussian Mixture Model for Background Subtraction. In: *International Conference on Pattern Recognition (ICPR)*. 2004. ISSN 1051-4651.