VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

TRAFFIC MONITORING FROM AERIAL VIDEO DATA

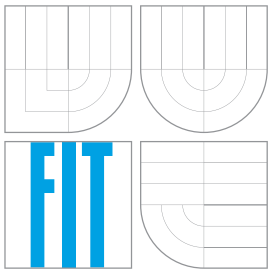
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. ADAM BABINEC

BRNO 2015

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

MONITOROVÁNÍ DOPRAVY Z LETECKÝCH VIDEÍ

TRAFFIC MONITORING FROM AERIAL VIDEO DATA

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. ADAM BABINEC

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. JAROSLAV ROZMAN, Ph.D.

BRNO 2015

Abstrakt

Tato diplomová práce se zabývá návrhem a implementací systému pro extrakci trajektorií vozidel z leteckých videí. Navržený systém analyzuje po sobě jdoucí letecké snímky dopravní křižovatky, které byly pořízeny kamerou z výšky přibližně 150 metrů pomocí autonomně létajícího stroje. Každý snímek je geo-registrovaný na základě vizuálních klíčových bodů ORB. Pro detekci vozidel v obraze byl využit kaskádový klasifikátor a příznaky MB-LBP. Oblast detekce vozidel byla omezena pomocí algoritmů pro detekci pohybu a s využitím informace o geometrii sledované křižovatky. Samotné sledování detekovaných vozidel je postaveno na částicovém filtru, který při evaluaci částic kombinuje výstup z detektoru vozidel a vizuální i pohybový model sledovaného vozidla. Systém byl otestovaný na třech ručně anotovaných video sekvencích, přičemž 92 % odhadnutých trajektorií koresponduje s realitou. Systém našel svoje uplatnění ve výzkumné činnosti dopravních analytiků, kde se využívá pro pokročilejší analýzy chování účastníků provozu v oblasti dopravních křižovatek a jejich vzájemné interakce.

Abstract

This thesis proposes a system for extraction of vehicle trajectories from aerial video data for traffic analysis. The system is designed to analyse video sequence of a single traffic scene captured by an action camera mounted on an arbitrary UAV flying at the altitudes of approximately 150 m. Each video frame is geo-registered using visual correspondence of extracted ORB features. For the detection of vehicles, MB-LBP classifier cascade is deployed, with additional step of pre-filtering of detection candidates based on movement and scene context. Multi-object tracking is achieved by Bayesian bootstrap filter with an aid of the detection algorithm. The performance of the system was evaluated on three extensively annotated datasets. The results show that on the average, 92 % of all extracted trajectories are corresponding to the reality. The system is already being used in the research to aid the process of design and analysis of road infrastructures.

Klíčová slova

detekce objektů, sledování mnoha objektů, lokální binární vzor, částicový filtr, letecké video, UAV, analýza dopravy

Keywords

object detection, multiple object tracking, Local Binary Pattern, aerial video, particle filter, unmanned aerial vehicle, traffic analysis

Citace

Adam Babinec: Traffic Monitoring from Aerial Video Data, diplomová práce, Brno, FIT VUT v Brně, 2015

Traffic Monitoring from Aerial Video Data

Prohlášení

Čestně prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením Ing. Jaroslava Rozmana, Ph.D. a odborného konzultanta Ing. Davida Hermana. Všechny literární prameny a publikace, ze kterých jsem čerpal, jsou uvedeny v bibliografii.

.....
Adam Babinec
26. května 2015

Poděkování

Rád bych poděkoval Ing. Jaroslavu Rozmanovi, Ph.D. za jeho vedení. Velké poděkování patří Ing. Davidu Hermanovi za odborné konzultace, neustálou morální podporu a pomoc při získávání potřebných informací a dat. Dále děkuji Ing. Jiřímu Apeltauerovi za uvedení do problematiky dopravní analýzy silničních křižovatek konvenčními metodami. Data využitá v této práci byla poskytnuta firmou RCE systems s.r.o., Česká Republika, a firmou Vertex Access, Spojené Království, za což jim děkuji.

Contents

Chapter 1

Introduction

The transportation systems are inevitable part of human society as they provide means to move goods, animals or people from one location to another. Thanks to the effective transportation systems the distances between producers of goods and their consumers became less important and therefore enabling more competitive trade environment — resulting in higher effectiveness of economy. As traffic itself does not produce anything valuable, the costs spent on transportation system are understood as actual overhead of human society. Therefore, the overall aim is to reduce this overhead by minimizing the transportation costs and improve its effectiveness.

In modern culture, the road transportation systems are being aided by automatic and intelligent solutions which form together one complex system which is usually described as intelligent transportation systems. Intelligent transportation systems represents group of technological solutions and advanced applications which are applied in transportation system to help maximize the system's efficiency. Their outcomes may improve the transportation system by altering traffic flow, for example by aided vehicle guidance, intelligent traffic lights systems etc., or they may provide the necessary data to further improve traffic system's infrastructure, e.g. by giving the information about road congestion to help decision process in new road planning.

The traffic monitoring, analysis and surveillance systems represents an essential part of intelligent transportation systems. The purpose of such systems is to gather and process traffic data which are further analysed in areas of traffic management, transportation planning, transportation infrastructure design and by public safety organizations (emergency services and law enforcement agencies). Generally, traffic data gathering technologies can be split according to the nature of gathered data in two categories: standard technologies which gather data at single point of the transporting system (i.e. vehicle counts and in some cases types and speed), and advanced technologies which can gather more detailed or broader scope of data across an area of transporting system (i.e. vehicle trajectories, route patterns, etc.).

In this chapter an introduction to traffic monitoring technologies and trends is presented and in that context the motivation and goal of this work is described. The structure of this paper is outlined at the end of this chapter.

1.1 Standard Traffic Monitoring Technologies

Vehicle counting technologies implement ways to collect data about number of passed vehicles through given node of road infrastructure. Additionally, they can detect vehicle weight and/or number a of axles, and in case of multiple collection points on a single road segment, they can be used to estimate the vehicle speed[1]. Such technologies are listed hereafter (cited from [2, 3]):

- **Intrusive Technologies** require the integration of sensing unit directly on/into the road surface or into the road bed, which increases the installation costs as well as it complicates the sensor maintenance. The length of lifetime of such sensors is also dependent of traffic density and weight of vehicles crossing the sensors as the repeated heavy pressure on sensor's installation can cause malfunction and/or damage to sensor.
  - **Pneumatic Road Tubes** are placed across the road lanes to detect vehicles from pressure changes that are produced when a vehicle axle passes over the tube. Generated pulse of the air pressure is recorded and processed by data processing unit located by the side of the road. The drawbacks of this technology is limited lane coverage and its efficiency and precision is subject to weather, temperature and traffic conditions.
  - **Piezoelectric Sensors** are incorporated into the road surface and work on principle of converting mechanical pressure into electric impulses on principle of deformation of piezoelectric material. The deformation modifies the surface charge density of the material, which results in electric potential difference between the electrodes installed into the sensor. The nature of generated signals is dependent on pressure applied to the sensor, and therefore the weight of the vehicle's axle can be measured as well.
  - **Inductive Loops** represent the most conventional technology used to collect long-term traffic data. They are placed into the road bed and generate magnetic field. A passing vehicle disrupts the magnetic field inducing Foucault currents in the wire loops. This disruption is detected as inductance decrease in the wiring and is recorded as passed vehicle. This sensor is however only capable to detect objects containing bigger masses of ferromagnetic material and therefore it may not detect passage of pedestrians as well as bicycles and scooters. This technology is robust to weather and traffic conditions but may as well as previous technologies be damaged by dense and heavy traffic.
- **Non-intrusive Technologies** incorporates remote sensing approaches to passing vehicle detection. These technologies do not require modification of road structure to incorporate sensing unit, instead they use remotely captured signals to collect traffic data. The current trends are to prefer non-intrusive technologies as they may lead to more cost efficient solutions in long term.
  - **Passive and Active Infra-red** technologies make use of energy radiated in near field or far field infra-red wavelength spectrum from detection area. The

[1]The estimation of vehicle speed in these cases (with exception of radar technology) is however implemented as indirect estimation method and it is a subject to error as it depends on vehicle re-identification technique applied between data collection points, which is not available for some of the technologies and/or may produce unreliable results [1].

information about presence, speed and type of passed vehicle can be extracted from captured data. This approach is robust to illumination conditions however it is sensitive to bad weather and have limited lane coverage.

- **Passive Magnetic** sensors are similar to inductive loops, but instead they are mounted over the top of the road - usually at road fare stations and similar road infrastructures. Their principle is based on recording of induced magnetic signatures of vehicles, which can be used to identify vehicle presence and type.
- **Microwave Radar** can detect moving vehicles and using Doppler effect it can detect vehicle speed. It can be used to record vehicle count data, their speed and simple vehicle classification. This technology is robust to weather and illumination conditions.
- **Acoustic sensors** can either be passive or active. The passive sensors are mounted beside the road or over the lane to detect sound signatures of vehicles to collect data of their counts and types. Active sensors are usually mounted over the traffic lane where they emit sound waves which are then bounced back from present vehicle or road surface. The temporal shift between emitted and received signal and the frequency change can be used to derive vehicle presence and speed information.
- **Manual Counts** require a trained human observer to be present in the traffic scene and manually record data about passed vehicles. This way the vehicle counts and types can be collected, and also some data that cannot be collected or distinguished by automatic systems — vehicle occupancy rate, pedestrians and more precise classification (e.g. taxi cabs, ambulance vehicles and so).

During the long history of transportation systems, various traffic monitoring approaches were applied to gather valuable traffic data. However, in terms of long-term and comprehensive data gathering, even though there was increasing pressure to develop more efficient traffic monitoring approaches, the cost of acceptable solutions is still deem to be too high to gather precise data continuously. Instead, the statistical approach is given to such problem, which consist of data sampling by collecting information about traffic on segments of the transportation system only during a short-term periods planned across the whole traffic analysis term and afterwards, the statistical methods are applied to produce interpolated/approximated estimates. The most deployed traffic counting technology is either based on inductive loops (which are usually used to trigger adaptive traffic lights systems and as side-product it collect the traffic counts) or manual counting by people (used to provide annual traffic estimates by producing data samples over short term data gathering periods). These approaches usually produce data bearing vehicle count and type information. Such produced data may deem to be sufficient for some of the applications in improving transportation system's efficiency, however others areas of traffic analysis may use more precise data to provide means for more efficient transportation systems. Especially in recent years, there is present growing need for precise and reliable vehicle trajectory data and telemetry of vehicles gathered in critical segments of transportation system. Due to these requirements, the current trends in traffic monitoring targets to gather more precise data ideally in long term, with as low capital and operational expenditures as possible. Currently, there are two approaches used widely which can provide comprehensive vehicle traffic data: visual based technologies and Floating Car Data techniques [2].

1.2 Floating Car Data

Technologies based on Floating Car Data (FCD) principle use an agent — either an active or passive unit — build into the vehicle or carried on-board of the vehicle to collect the necessary data or to identify the vehicle at traffic data collecting points of the transportation system. Such systems are often described as *vehicle telematics systems* and are used to track vehicles, trailers and containers in process of transport. Also, they are widely used for fleet management of large company's vehicle fleet.

The passive agents do not collect any vehicle position, dynamics or trajectory data, but instead they are used as detection and identification tags at the data collecting points. The most common technologies that use passive agents are based on RFID (Radio-Frequency IDentification) or Bluetooth re-identification methods. RFID microchips are issued by transportation system management agencies and installed into selected vehicles which take the roles of data samples — the passing of such vehicles along the traffic data collecting points is registered by a RFID detectors mounted beside the road and recorded into the database. If the network of the collecting points is dense enough, the data about usual vehicle routes, average speed on road segments and other useful information can be extracted.

The Bluetooth based approach works similarly — the vehicle is identified by physical address of present Bluetooth devices in the vehicle (hands-free sets, cell phones, infotainment systems, etc.). In comparison to approach based on identification of RFID chips, it does not require additional device to be present in the vehicle. On the other hand, this approach tend to produce uncertain results if the vehicle contains numerous Bluetooth devices and it completely misses all vehicles without any Bluetooth device turned on. In case of older and low-priced vehicles it may lead to statistical bias.

In contrast to passive agents, the active agents collect, record and optionally distribute vehicle position, dynamics and trajectory data captured by sensors installed in agents themselves or in the vehicle itself. Such agents can be part of vehicle's infotainment system, automotive navigation system or passengers' smart phones. The collected data are usually transferred to central traffic management database using wireless Internet connection on reasonable short intervals to provide the up-to-date data about traffic situation on the roads.

Such methods either uses cellular network data (CDMA, GSM, UMTS, GPRS) to estimate the vehicle position and route (this approach is known as Floating Cellular Data) or they make use of built-in GPS sensors, inertial unit and gyroscope to estimate precisely their local and global position and subsequently trajectory and velocity of the vehicle. Such data can be affected by a noise in grades of units or tens of meters of absolute position error, and therefore there is need for signal processing to filter out the noise. The great advantage of these methods is support by manufacturers of cell phones and developers of the cell phone software with motivation of providing real-time congestion reports that assist automotive navigation systems and statistical data for insurance companies [4].

1.3 Vision Based Technologies

Vision based technologies for traffic monitoring are based on analysis of video data captured by cameras in visible electromagnetic spectrum or near infra-red spectrum. Such captured image data contains information about vehicle shape, colour, type and position and thanks to temporal dimension of video sequences, the information about vehicle speed and trajectory can be extracted as well. Even though, all these data may be present in

the video sequences, their extraction is subject to field of computer vision analysis, as the data are usually affected heavily by noise and the visual representation of vehicles is diverse and usually variable during the movement of the vehicle across the captured scene in video sequence. Additionally, the problems of environment occlusion, intra-target occlusion and dependency of visual representation of vehicles on different lightning and weather conditions make the task of vehicle trajectory extraction very cumbersome. To overcome these difficulties, the algorithms of image–reality registration, object detection and multi-object tracking, as well as signal processing algorithms to smooth out the noise, need to be all deployed to solve the problem of traffic monitoring in visual data.

The most widely used approaches make use of stationary cameras mounted beside the roads or over the lanes at the road infrastructure buildings and poles. Such cameras can be used jointly with speed measuring radar systems to identify speeding subjects on highways, or installed alone to detect and track vehicles in video data, to produce informations about vehicle counts, their types and speed and in some instances they can identify vehicles before and after their course through the crossroad to produce origin-destination matrices. This approach can be affected by bad illumination conditions. To surpass this weakness, the cameras may operate in near infra-red electromagnetic spectrum to cope with the task of vehicle detection and tracking during the night or in cases of bad illuminated traffic scenes. The downside of such systems is the requirement of permanent installation beside the road and permanent access to electricity resulting in increased cost of traffic infrastructure. Additionally, the field of view of a single camera is very limited, so when the exhaustive traffic scene has to be monitored in its entirety, high number of such stationary cameras and data processing units has to be deployed resulting in higher costs and lower flexibility of such systems.

Another way is to extract vehicle trajectories from satellite imagery. Generally, such data are globally aligned and capture undeniably the widest spatial extent of all aforementioned methods. The main obstacles in this approach are the limited availability and high cost of such data, as well as poor spatial resolution or temporal resolution of publicly available satellite image data. Also, the quality of data is depending on weather and/or illumination conditions and other factors. However even still images capturing the wide areas in low resolution (0.5 to 2 metres per pixel) can still offer a clues about traffic density and congestion [5, 6].

Finally, the last way to gather a visual data suitable for traffic monitoring is by using an aerial vehicles, such as helicopters, airplanes and UAVs. At the current state, such data is used only as decision support for human observers who examine the visual data to produce a subjective conclusion about a traffic state and congestion at the time of the capture. The quality of aerial video data is subject to weather and illumination conditions and it may be limited in inhabited areas. Another obstacle in using aerial video data in automatic traffic monitoring is the problem of its geo-registration — mapping every single image of aerial video sequence into the real world coordinate system. Similarly to the satellite imagery, the information about a traffic density and congestion can be extracted even from still images of road segment.

1.4 This Work

According to [9], the current demographic and economic trends are changing transport demands and therefore the new view on traffic analysis and planning should be considered. The vehicle travel is peaking in most developed countries and current demographic

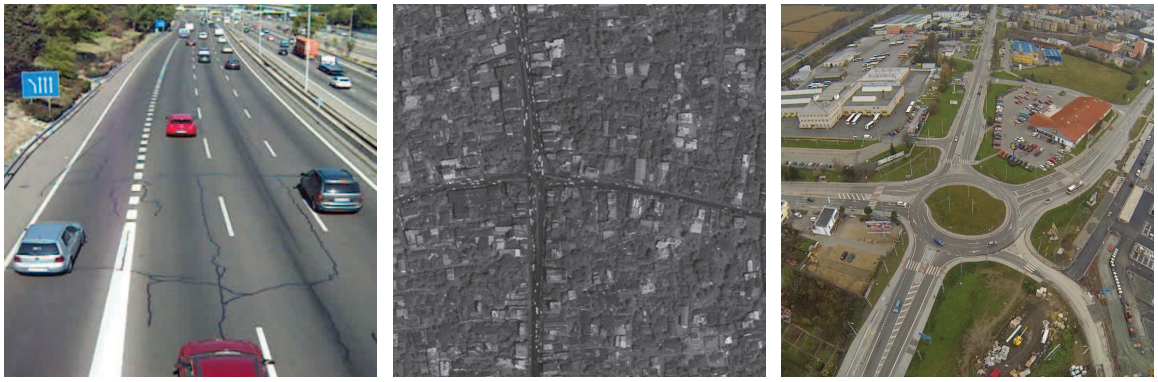


Figure 1.1: Example of imagery data that is used in vision based technologies. Left: video frame captured by a stationary camera. Source: [7]. Middle: panchromatic satellite imagery with resolution of 1 pixel per meter. Source: [8]. Right: video frame captured by a flying UAV.

and economic trends (ageing population, rising fuel prices, urbanization, increasing health and environmental concerns and changing consumer preferences) are increasing demand for walking, cycling and public transport. On this wave of paradigm shift, the transport planning seeks for solution of arisen problems and/or alternative solutions for existing problems. Therefore, there is an opportunity to gain momentum for new approaches across whole field of transportation system analysis and planning.

We believe that the new methodologies and technologies can change the way the people look at the traffic and transportation. As these changes are happening right now there is an opportunity to design and develop new techniques of traffic data extraction based on new technologies in different areas. One of such technologies is video capture using unmanned aerial video. In latest years, the development of low-cost multicopter aerial vehicles has opened the opportunities of their deployment in various applications from entertainment, through video capture of sports and cultural events, to package delivery and surveillance. This lead to considerable price drop of such vehicles as well as improvement in the length of their flight time, position stabilisation algorithms, load capacity and availability of supporting technologies. Therefore it is reasonable to consider a deployment of such vehicles in traffic analysis and monitoring.

This work proposes a system which will allow to extract comprehensive traffic data of single traffic scene from aerial video data captured by flying UAV. The system may form a precursor and a basis for a complex solution of traffic data gathering and analysis using unmanned aerial vehicles such as multi-copters or balloons. As aerial video data bears no personal information about drivers or identification of vehicles (more specific than their type and colour), the system necessary fulfils the requirement of data anonymity. In comparison to methods based on floating car data, this approach does not require an active node present in each vehicle and furthermore it does not need any additional data gathering points or database systems incorporated by vehicle producers. Also, in the future the system may be further improved in terms of computational speed and accuracy to provide real time data which no other alternative can do in current time — precise vehicle trajectory data with minimal absolute spatial error. Such data property may represent crucial requirements in some applications, for example — road structure design analysis, inter-vehicle interaction and behaviour analysis and so forth. We believe that it may eventually

lead to a technologically flexible and low-cost alternative to existing solutions and therefore make transportation systems more cheaper and efficient.

To show that proposed system may represent a feasible solution to traffic data gathering from aerial videos, as part of this work we implement the prototype and evaluate its vehicle detection and tracking abilities as well as reliability trajectory extraction. The evaluation is carried out on the hand annotated dataset of aerial video data provided by companies *RCE systems s.r.o.,* Brno, Czech Republic and *Vertex Access*, Rotherham, United Kingdom.

This paper is organised as follows: In second chapter, the problem statement and a overview of proposed system is given. In the third chapter, the sub-problem of geo-registration of aerial video data is discussed and proposed solution to this problem is presented. The sub-problems of vehicle detection and tracking are described and analysed with proposed solutions in chapters 4 and 5 respectively. Additionally, in the chapter 6, we describe the data post-processing phase of the system which deals with noise of estimated vehicle movement data and missing data in case of geo-registration failure. Also, the filtering of potentially erroneous vehicle trajectory data is presented there as well. The implementation of the proposed system is outlined in chapter 7. The evaluation methodologies, metrics and evaluation results of the detection and tracking alone as well as whole output of the proposed system, are presented in chapter 8. The summary of the whole work, discussion about the results of the work and description of future plans and possible improvements are given in the last chapter of the paper — chapter 9.

Chapter 2

Overview

This chapter aims to provide the overview of the proposed system in context of the task of vehicle trajectory extraction from aerial video data and existing solutions to this problem. We will firstly formulate the goal of trajectory extraction from aerial video data, and then we sketch out the design of the proposed system which we have developed with the intention to create the solution to the formulated problem. Also the general logic in problem break down is explained. The proposed solutions to arisen sub-problems in the architecture of the system are described later in the paper in their respective chapters.

2.1 Goal Formulation

The proposed system is solving the task of vehicle trajectory extraction from aerial video data. To accurately understand the problem, the detailed description of all its aspects is given hereby.

Aerial video data represents video sequences captured by an aerial vehicle for further scene understanding and object detection and tracking. In the context of this work, we understand aerial video data as single video sequence capturing traffic scene in its entirety. The traffic scene should be ideally captured from top view, but the application should be able to deal with scene captured from an angle, therefore it may be distorted by perspective transformation of the scene due to considerable view incident angle. The traffic scene may be occluded in long-term spells, but generally, it should be captured on the video most of the time. To provide sufficient visual data, we expect the aerial vehicle to be floating in height of 80 m to 200 m above the ground surface and the spatial resolution of captured data should be about 6 pixels per meter[1]. The temporal resolution of video should be reasonable to provide data for continuous object tracking — at the case of vehicles, we assume the maximum speed of 60 miles per hour (cca. 96 kilometres per hour) which leads to positional change of vehicle of approximately 26.67 metres per second. Therefore in this case, the temporal resolution of at least 10 frames per second (FPS) is appropriate, resulting in inter-frame positional change of 2.667 metre. This requirement can be easily met by reasonably priced action cameras, such as GoPro Hero3 or Sony AS20[2].

[1] However, even in the evaluation datasets used in this work, the spatial resolution varies over the scene area and it goes as low as 3 pixels per meter.

[2] For further technical specifications, please, see the products' web pages: GoPro Hero3 — http://shop.gopro.com/EMEA/cameras/hero3-white/CHDHE-302-EU.html, and Sony AS20 — http://store.sony.com/compact-pov-action-cam-zid27-HDRAS20/B/cat-27-catid-All-Action-Cam.

Traffic scene is a delimited part of the real world which contains roads, intersections, junctions and/or roundabouts, where the vehicle movement is taking place. In this work, it is assumed that all the traffic that is present in the scene is happening in manner, that vehicles at the first enter the scene, move on the road surface and leave the scene — ergo no vehicle just moves inside the scene without ever arriving to the scene or leaving the scene.

For the sake of better understanding of the traffic scene by computer program, we apply additional requirements: we define traffic scene as delimited part of the road surface present in the video. On this part of the road surface, the detection and tracking of the vehicles will be carried out. Also, to emphasise the important part of the traffic scene — the *region of interest* is defined.

Region of interest is a delimited subset of road surface in the traffic scene which represent a limited area for which the trajectories of the passing vehicles needs to be extracted. The area is defined by all possible entry points trough which the vehicle can enter the region of interest, and all possible exit points through which the vehicle can exit the region of the interest. This delimitation is modelled by virtual entry traffic gates and exit traffic gates, which are part of the annotation of traffic scene derived from geographic information system (GIS).

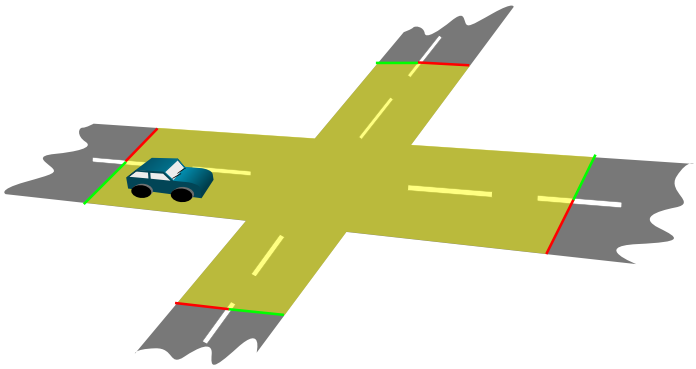


Figure 2.1: Illustration of a traffic scene. The green lines represent entry gates. The red lines represents exit gates. The highlighted part of the road surface is a region of interest.

Standard driver behaviour in traffic scene that the proposed system is expecting and accepting as only valid behaviour is following: vehicle enters the traffic scene, moving on the road surface. Subsequently, it crosses an entry gate and enters the region of interest. Afterwards, the vehicle moves through the region of interest until it exits the region by crossing one of the exit gates. In the end, the vehicle leaves the traffic scene. The vehicle must stay on the road surface during the whole movement through the region of interest.

Vehicle trajectory is a sequence of vehicle states, describing its movement across the region of interest in the traffic scene. It specify vehicle global (absolute) position at every given time moment of vehicle's presence in the region of interest. Additionally, information about the vehicle velocity and tangential and lateral accelerations can be derived from such data.

In the context of aforementioned description, we hereby formulate the goal of the proposed system:

Goal. *Given the annotation of a traffic scene and an aerial video sequence capturing the said traffic scene, the goal of the system is to extract vehicle trajectory data of all vehicles*

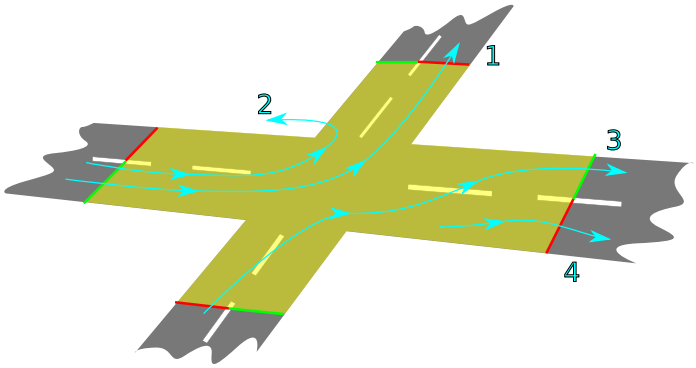


Figure 2.2: Illustration of selected trajectories of vehicles. Only the vehicle which produced the trajectory 1 conforms to the standard driver behaviour, as it enters region of interest through entry gate (green line), moves on the road surface, and leaves through exit gate (red line). Other trajectories does not conform the standard driver behaviour for the following reasons: The trajectory 2 does not leave through an exit gate. The trajectory 3 does not cross the entry and exit gates in the correct order. The trajectory 4 does not enter the region of interest trough an entry gate.

that follows standard driver behaviour during their movement through the said traffic scene which is in its entirety captured in said aerial video sequence.

2.2 Sub-problems Discussion

The task of vehicle trajectory extraction from aerial video data can be broken apart into 5 sub-problems:

- Traffic scene understanding
- Video frame pre-processing
- Vehicle detection
- Vehicle tracking
- Post-processing of extracted data

In the following subsections, we will discuss these problems and afterwards, we will present the design of the proposed system.

2.2.1 Traffic Scene Understanding

Traffic scene understanding provides a conceptual comprehension of traffic scene with relevant relationships between the entities of the traffic scene. The system can make use of the knowledge of the scene to improve its precision and robustness, e.g. by allowing only meaningful vehicle movements or by expecting some well known patterns of vehicle behaviour in given context. Additionally, such knowledge may also lead to better computation performance of the system and/or lowered requirements for the input data quality.

Traffic scene understanding should provide following knowledge to the system: definition of geometry of road surface, expected entry and exit points of the region of the interest,

geometric transformation of the scene into the real world coordinates, which can be all extracted from GIS database, and respective visual representation of the scene, which can be extracted either from high-resolution orto-photo maps or the analysed video sequence itself.

2.2.2 Video Frame Pre-processing

This sub-problem covers the task of video stabilisation and geo-registration, and also it targets to remove image distortion and noise present in the video frames.

Every input video frame may be affected by imperfection and properties of the camera lens. This leads to geometrical distortion of the captured frames, which is undesirable in computer vision applications, especially when high spatial precision of extracted data is needed. Another imperfection of input video frame may be represented by a noise in colour (or brightness) dimension of the image, caused by electromagnetic noise, video compression or used capture technology.

Furthermore, the video captured by a camera mounted on a UAV can suffer from shaking and therefore it needs to be reasonably stabilised. Also, the information about how is the captured traffic scene projected into the video frame (or the other way: how is the captured video frame positioned in the traffic scene) needs to be retrieved. Both of these problems are tasks for video stabilisation and geo-registration.

2.2.3 Vehicle Detection and Tracking

The main sub-problems of this work are vehicle detection and vehicle tracking in the video sequence. The vehicle detection sub-problem lies in the effective extraction of positions of all vehicles in the given input frame. The aim of the vehicle tracking sub-problem is to retrieve the sequences of states of all target vehicles during their movement through the region of interest in the traffic scene.

To solve these tasks, the appropriate algorithms must be deployed, which can cope with the nature of aerial video data — environmental occlusion, multi-target overlap, low resolution and feature salinity of tracked objects and their appearance changes during their movement through traffic scene.

2.2.4 Post-processing of Extracted Data

Extracted video data may be affected by spatial noise due to the imperfect video stabilisation and geo-registration of video frames and missteps or inaccuracy of vehicle detection and tracking. Such noise can have undesirable results on the vehicle trajectory analysis and may lead to incorrect estimations of vehicle velocity and acceleration. Therefore, it needs to be dealt with by filtering it out using signal analysis and processing techniques.

Additionally, due to occlusion or failure of video stabilisation and geo-registration, the extracted trajectory may contain missed, undefined segments. These portions of trajectories needs to be estimated by data interpolation or extrapolation algorithms.

2.3 Proposed Design

The proposed system was designed as a prototype to explore the possibilities of vehicle trajectory extraction from aerial video data by blending together the solutions to aforementioned sub-problems.

As input, it accepts the aerial video sequence, camera intrinsic parameters and annotated reference image containing the whole examined traffic scene. The camera intrinsic parameters are required for image distortion removal as well as estimation of position of the camera. The annotated reference image is utilized for video stabilisation and its further geo-registration — transformation of the video sequence into real world coordinate system. The annotations of the reference image are also used to describe important aspects of traffic scene, define the road surface and region of interest for vehicle trajectory extraction, forming together the traffic scene understanding data.

The proposed system is targeted to analyse single-level intersections and it is therefore limited to road segments and intersections without any under-the-surface or over-the-surface lane interchanges. We also assume that whole area of analysed traffic scene can be visible from a UAV and captured on the video sequence — i.e. the traffic scene does not contain any tunnels or permanent large-scale structures occluding the crucial part of the traffic scene. For the sake of simplicity of implemented supporting algorithms, the area of traffic scene is considered to be planar and is therefore approximated by two-dimensional plane.

The output of the system is a database of estimated vehicle trajectories in real world coordinate system, suitable for further analysis of traffic congestion, driver's behaviour, dynamic properties of vehicles, etc. The data can be used to calculate velocity and acceleration profiles, critical aspects of intersections such as places with high lateral acceleration of vehicles or frequent corner cuttings.

The system works in following 2 phases:

- Phase 1: **Detection and Tracking** — the system detects vehicles in the video sequence and tracks them during their movement through the traffic scene.
- Phase 2: **Data Post-processing** — prepares generated data for vehicle behaviour analysis. Namely, the interpolation of missing data, smoothing of generated trajectories and re-filtration of trajectory database is carried out.

During phase 1 — *Detection and Tracking*, every frame of input video sequence is processed according to illustration depicted in figure 2.3. It shows that the processing of the single frame is divided into following three stages:

- **Input pre-processing stage** — solves the tasks of video frame pre-processing.
- **Detection stage** — solves the task of vehicle detection.
- **Tracking stage** — solves the task of vehicle tracking from previous frame to current frame of the video sequence.

During the input pre-processing stage, the image is loaded from input video as a single video frame. Using the information about camera intrinsic parameters, the frame's geometrical distortion is removed. Undistorted frame is stabilised and geo-registered by finding a geometrical transformation between already geo-registered reference frame and undistorted video frame, based on visual similarity. In detection phase, the geo-registered frame is used to update background model and compute foreground mask. Background model represents the system's knowledge about how the static scene without moving objects looks like. It is used to compare the geo-registered video frame with this knowledge to generate foreground mask — a two-dimensional array which carries an information about the areas of the input video frame which are considered to be a foreground — moving objects. The foreground

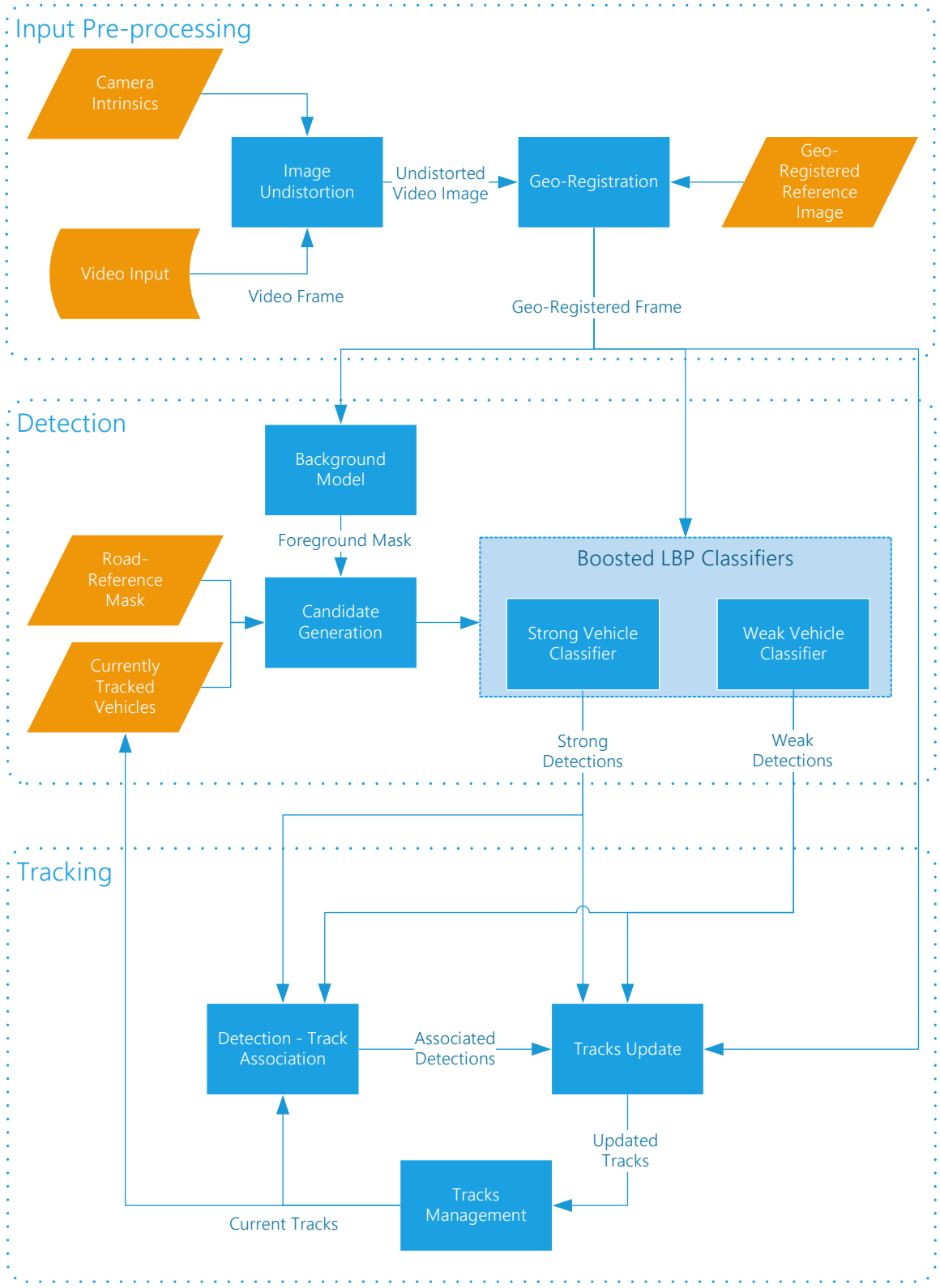


Figure 2.3: The data flow for analysis of a single frame of the input video sequence during vehicle detection and tracking. The analysis consist of three stages: input pre-processing stage, detection stage and tracking stage. All these stages will be further described in following chapters of the paper.

model is fused together with road-reference mask and database of currently tracked vehicles to generate candidates for detection. The detection is carried out using two vehicle classifiers — strong vehicle classifier accepting only salient candidates, and more benevolent weak vehicle classifier with low target miss ratio but higher false positive ratio. In tracking stage, sets of weak and strong detections are applied to aid vehicle tracking in step of detection–track association and during tracks update. In the track management step the tracks are, according to their states, added, rejected or saved for further analysis or prepared for output as successful tracks.

In the following chapters (3–5) we will describe all stages of single video frame analysis used in *Detection and Tracking* phase, namely: input pre-processing stage, tracking stage and detection stage respectively. Additionally, in chapter 6, we will describe *Data Post-processing* phase, concluding the description of the algorithms employed in the proposed system.

Chapter 3

Input Pre-processing

In this chapter we will specify the necessary inputs of the proposed system and provide the description of selected approaches to input pre-processing tasks. The pre-processing stage is responsible for stabilisation and projection of input video frame into the real world coordinate system, resulting in geo-registered video frame. The process consist of three steps:

1. Distortion removal from input video frame.
2. Calculate correspondence between input video frame and reference image.
3. Geo-register input frame by computing the transformation estimation of input video frame into the real world coordinates using calculated correspondence and traffic scene annotations.

Additionally, at the end of this chapter, we will discuss the problem of plane approximation of the scene tied together with detection and tracking of three-dimensional objects (vehicles) from geometric point of view. We will propose the algorithm to estimate corrected position of such detected/tracked object, called *Offset Fix*.

3.1 Traffic Scene Annotation

As we already stated in subsection 2.2.1, traffic scene understanding is an important part of the vehicle trajectory extraction from aerial video data. To provide necessary data for scene understanding to the proposed system, the annotated reference frame is provided as one of its inputs. It consists of a reference image which captures the entire analysed traffic scene and traffic scene annotations. Traffic scene annotation is collection of geometric objects defined in reference image space, which contains additional relevant data. These objects describe important aspects of traffic scene, such as correspondence between reference image and real world coordinate system, road surface, region of interest and unstable regions.

Annotations are divided into 4 groups:

- *Corresponding Point* — describes geometrical correspondence with real world coordinate system. Each point is defined for certain point in reference image, for which real world coordinates are known. Each corresponding point therefore consist of coordinates in reference image and corresponding coordinates in real world. There must be defined at least 4 corresponding points, while none triplet of such points may lay on

line, to capture the geometric correspondence and therefore transformation between real world coordinate system and reference image.

- *Traffic Lane* — describes part of road surface, and it is represented by polygon. For proper operation of proposed system, provided traffic lanes should cover all road surface of traffic scene where vehicles may move, since road-reference mask used in latter stages of presented algorithm is generated from traffic lane information contained in the traffic scene annotation.

- *Traffic Gate* — describes the traffic analysis gate, and it is represented by a polyline. For each traffic gate an information about vehicle crossing across the defining polyline is stored, including time of crossing, vehicle identification and velocity. There are three types of traffic gates:

  - *Neutral Gate* — used solely for recording of vehicle crossing events.
  - *Entry Gate* — represents entry border of region of interest for traffic analysis: place across which vehicles enter the region of interest.
  - *Exit Gate* — represents exit border of region of interest for traffic analysis: place across which vehicles leave the region of interest.

  Each traffic gate can be either single-directional or bi-directional. Bi-directional traffic gate is sensitive to vehicle crossing events in both directions of vehicle movement across the gate. A single-directional gate is sensitive only to vehicle crossing event that happen in predetermined direction of vehicle movement across the gate. This direction of the traffic gate is defined by the user. Properly annotated traffic scene should have defined all possible entry and exit gates leading into/from region of interest.

- *Unstable Region* represents an area of the reference image that is experiencing permanent movement or is formed by scene foreground — objects that does not stay in the same place for the whole sequence of video data. The definition of unstable regions is optional, however it may have helpful effect on video stabilisation and geo-registration.

We expect that in the future, generation of traffic scene annotation can be automatized using GIS databases, as there is permanent development in the area of GIS systems. However, during the development and evaluation of the system, we did not have an access to such data and therefore in this work, we have decided to use traffic scene annotation data defined by a human, although extracted from GIS. To ease the generation of such data, we have implemented the application which generate traffic scene annotation with human assistance — see section 7.4.

3.1.1 Real World Coordinate System

To provide the definition of corresponding points and geo-registration of video sequence, a convenient real world coordinate system had to be chosen. The most common geographic coordinate system — World Geodetic System — is based on projection of the Earth surface on a standard spheroidal reference surface (also known as reference ellipsoid) [10]. The coordinates in this system are defined by standard coordinate system — longitude and latitude angles. The value of longitude coordinate represents an angle between IERS reference Meridian half-plane, and a line crossing the centre of mass of the Earth and given point on

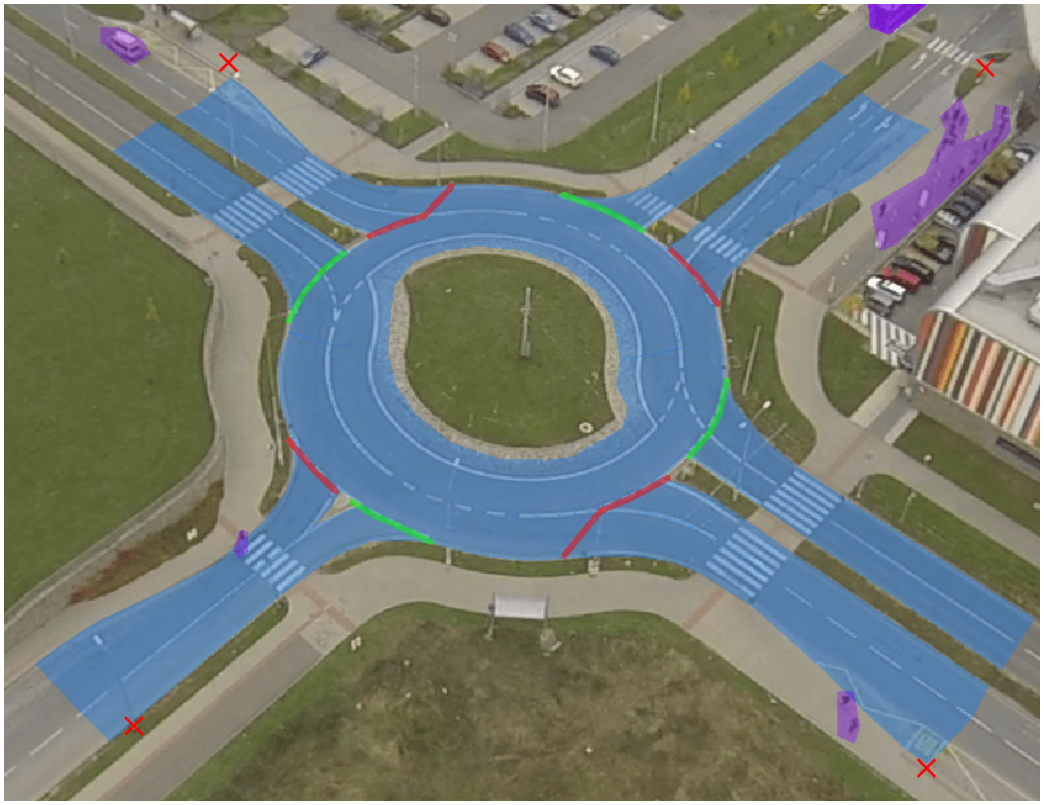


Figure 3.1: Example of an annotated traffic scene. Little red crosses represent corresponding points. Blue area represents traffic lanes. Green and red polylines represent entry and exit gates, respectively. Region of interest is bounded by these gates, and is therefore the roundabout itself. Purple areas represent unstable regions.

the the surface of the Earth. The value of latitude coordinate represents an angle between equatorial plane and a line crossing the centre of mass of the Earth and given point on the surface of the Earth. The current revision — WGS84, was established in 1984 and latest revision was published in 2004, known as EPSG:4326. This geographic coordinate system is used widely almost in every application that deals with geographic coordinates. However, it represents the points on the surface of Earth by polar angular coordinates, thus it complicates the transformation estimations and conversions between image space, which uses Cartesian coordinates.

For that sake, we have decided to use geographic coordinate system which projects points from the surface of the Earth onto a planes — into two-dimensional Cartesian coordinate system. The transformation between image space and such coordinate system can be expressed by single matrix, and the conversion is done by simple multiplication of transformation matrix and homogeneous vector representing the given point. The most prevalent such geographic coordinate system is Universal Transverse Mercator (UTM) developed at by US army in 1940s. It does not project all the points of the Earth surface on single map projection plane, but instead it divides the Earth surface into 60 zones [10]. As projection technique, it uses secant transverse Mercator projection using WGS84 ellipsoid. Due to this fact, there is conflict at the boundaries of UTM zones — as the each zone uses different reference point for its coordinate system, for the calculation of distances in such areas it is

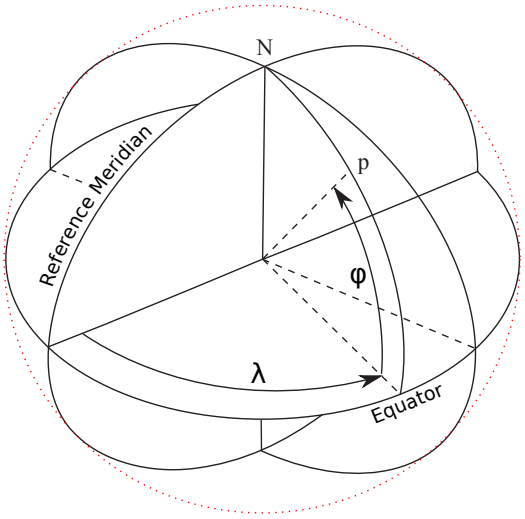


Figure 3.2: Illustration of World Geodetic System. The position of the point p is represented by an anglular coordinates — longitute λ and latitude ϕ.

convenient to make all calculations in coordinate system of one selected zone. Another disadvantage is planar projection of otherwise ellipsoid surface, which leads to scale distortion near the zone boundaries. This distortion is however very small (less than 0.1%) and it is usually ignored.

The presented system has built in support for Universal Transverse Mercator geographic coordinate system. However, to support local geographic coordinate systems (usually published by local governments or cartographic bureaus, such as Křovák's projection used in Czech Republic and Slovakia) which can provide better local position accuracy, the system is able to accept any geographic coordinate system based on projection to two-dimensional Cartesian space.

3.2 Image Undistortion

Due to the usage of wide field-of-view lenses and imperfection of conventional cameras, every input frame in the video sequence is affected by the lens distortion and sensor chip misalignment. This leads to the image geometrical distortion that may have undesirable effects on quality of outputs of image registration process and estimation of position in real world coordinate system.

In the presented system, we assume translational, radial and tangential distortion of image. This assumption is considered acceptable according to Bradsky and Kaehler [12] for use in most computer vision applications. The solutions for removal of such distortion were addressed in literature several times [13, 14]. To recover such distortion, the intrinsics parameters of camera — camera projection matrix and lens distortion coefficient vector, must be known. The required camera information can be obtained by camera calibration, using specialized calibration patterns. Such calibration methods are implemented in widespread used vision library OpenCV [15] and scientific computational suite MATLAB [16].

In our system, we have employed the image distortion removal algorithm implemented in OpenCV library [15]. After the distortion removal procedure, the straight lines in real world are depicted as straight lines in resulting undistorted image, making the undistorted

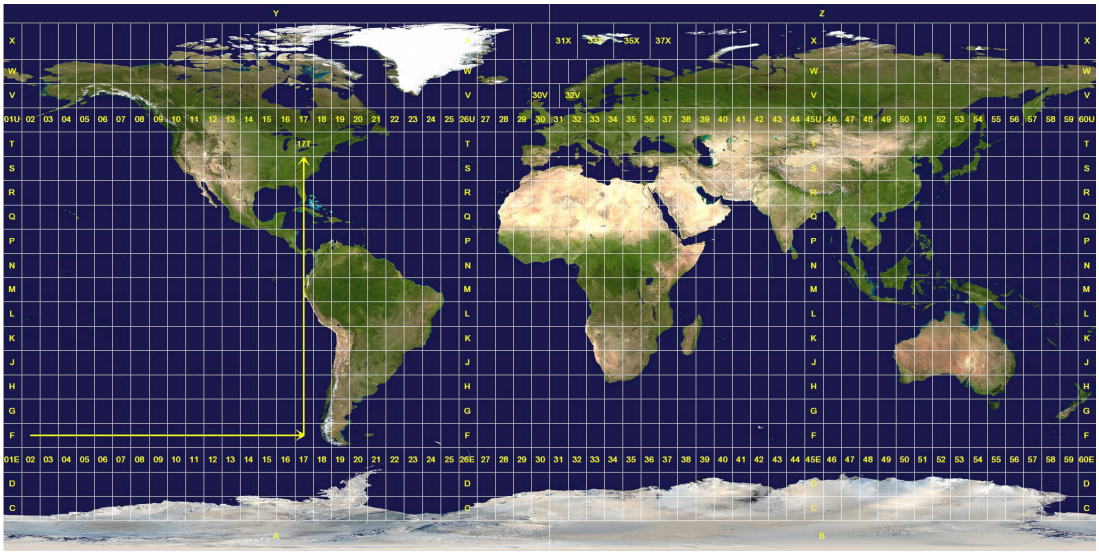


Figure 3.3: The subdivision of the Earth surface into the UTM zones. Each zone is projected by a secant transverse Mercator projection on a segement of a cylindrical surface. Note that the grid is uniform with exception of the six zones in the Northern Europe. Source: [11]

image a perspective transformation projection of reality onto the plane. Such undistorted image can be further analysed to estimate its transformations to various coordinate systems.

3.3 Video Stabilisation and Geo-registration

Video stabilisation and geo-registration of the input video sequence transforms the video sequence to the real world coordinate system. Such transformed data are more suitable for vehicle detection and tracking for following reason:

- The size of the vehicles in real world coordinate system is consistent during its movement through the scene. This is generally not true for the images which are not geo-registered and therefore affected by perspective projection. In such images, the

(a) Distorted Video Frame (b) Undistorted Video Frame

Figure 3.4: The comparison of distorted and undistorted video frame capturing the traffic scene. Note that in the undistorted frame, straight lines are preserved.

range of possible sizes of the vehicles varies over the area of the image which may lead to increased computational complexity.

- The dynamics model of the vehicle can be implemented directly in the coordinate system of geo-registered image, as the transformation of such image to real world coordinate system is only composed of rotation, translation and uniform scale. This leads to simplified and faster implementation of tracking algorithm, as no transformations are needed for conversions between dynamics model of vehicle and its tracked position in the video frame.

- The transformation to real world coordinate system is necessary for the usability of extracted data.

The geo-registration process of the video frame consist of two transformation estimations: transformation estimation between video frame and reference frame and transformation estimation between reference frame and real world coordinate system.

3.3.1 Image Correspondence

To estimate transformation between input video frame and reference image, we deploy algorithm based on sparse local image feature correspondence. The used image feature detector and descriptor is ORB (Oriented FAST and Rotated BRIEF) [17]. The image feature correspondence set is generated according to the descriptor distances of detected features and is cross-validated as follows:

Let the V_r be the set of extracted image features found in reference image and V_i the set of extracted image features found in undistorted video frame. The feature correspondence set contains every pair $u\left(v^r, v^i\right)$, where $v^r \in V_r$ and $v^i \in V_i$, such that the following conditions are met:

$$\begin{aligned} \forall v_x \in V_r \setminus \{v^r\} : \quad & \mathrm{dist_d}\left(v^i, v^r\right) + \mathit{diff}_{min} \leq \mathrm{dist_d}\left(v^i, v_x\right) \\ \forall v_y \in V_i \setminus \{v^i\} : \quad & \mathrm{dist_d}\left(v^i, v^r\right) + \mathit{diff}_{min} \leq \mathrm{dist_d}\left(v_y, v^r\right) \quad , \end{aligned} \tag{3.1}$$

where operator $\mathrm{dist_d}\left(a, b\right)$ represents Hamming distance of image features a and b according to their descriptors and diff_{min} is some arbitrary constant with empirically estimated value.

The feature correspondence set is further filtered so it does not contain points (such as moving objects) which may affect the quality of calculated image transformation between the reference image and undistorted video frame. This may be done by rejecting all correspondences that are situated in areas that are deem to be scene foreground (see subsection 4.2.1). This approach however have following obstacles: the scene foreground for current video frame can be calculated only for already geo-registered frame which is not available at this step of the algorithm, and possible corrupted scene foreground may lead to rejection of too many correspondences which may further result to even more corrupted background model. Therefore, we decided to reject only correspondences that are situated in areas which are estimated to be currently occupied by already detected and tracked vehicles, or in the areas defined by unstable regions (see section 3.1). The estimation of areas occupied by already tracked vehicles is based upon dynamics model of vehicles described in chapter 5.

Figure 3.5: Geo-registration of the input video frame. Left: before the transformation into real world coordinate system. Right: after the transformation into real world coordinate system. Note that the vehicles at the top of the untransformed frame (left) are considerably smaller — this size distortion is caused by perspective effect: farther objects appear to be smaller. In the geo-registered frame (right), such effect is not present.

3.3.2 Transformation Estimation

We assume that the geometric transformation between reference image and undistorted video frame is perspective transformation. The calculation of transformation is based on minimisation of square error of back-projection [15]. To achieve the robustness of transformation estimation, the RANSAC (RANdom SAmple Consensus) procedure is applied to the set of feature correspondences [18]. Target inliers ratio is set to 0.8, number of iterations is limited to 50 and final transformation is calculated from the whole set of inliers of the best sample correspondence set.

The geometric transformation of the the input video frame to the two-dimensional real world coordinate system can be achieved by following multiplication of transformation matrices:

$$\mathbf{M}_{im} = \mathbf{M}_{rm}\mathbf{M}_{ir} \quad , \tag{3.2}$$

where $\mathbf{M}_{im}$ is calculated geometric transformation matrix from input video frame to two-dimensional real world coordinate system, $\mathbf{M}_{rm}$ is geometric transformation matrix representing transformation of reference image to two-dimensional real world coordinate system and $\mathbf{M}_{ir}$ is geometric transformation matrix representing transformation from input video frame to reference image. $\mathbf{M}_{ir}$ is result of previously mentioned RANSAC procedure and $\mathbf{M}_{rm}$ can be estimated from traffic scene annotation defined in reference image. Traffic scene annotation contains the definition of at least 4 corresponding point pairs between reference image and real world coordinate system. Therefore, the transformation can be estimated the same way as transformation between input video frame and reference image, with dropping of RANSAC procedure — assuming that the annotation does not contain outliers. Example of input video frame transformed into real world coordinate system is shown at figure 3.5.

This explained approach naturally incorporates video stabilisation process, because it registers every input video frame independently to the same coordinate system.

3.4 Coordinate Systems

Internally, the system however does not store the transformed geo-registered frame in scale 1:1. This approach would reduce the contained visual data significantly, as each squared meter of traffic scene surface in real world coordinate system would be represented only by one pixel of the stored frame. Also, due to arbitrary enormous values of real world coordinates, the representing two-dimensional array would need to be either sparse or impractically huge. Therefore, the system stores the geo-registered frame in so called *working space*.

Working space is coordinate system which is based on scaled up, rotated and translated version of real world coordinate system. It is constructed by scaling up the real world coordinate system by factor 6 in both axes x and y. This system is thereafter rotated so that the projection of up-pointing vector (vector with magnitude of 1 and pointing negative direction of y axis) from the middle of the reference image to work space points the same way in working space. Subsequently, the x axis is flipped to preserve left/right orientation in the space. Finally, the space is translated to the nearest possible point to its origin (point [0, 0]), while keeping all the road surface, defined in reference frame and transformed to work space, in $(+, +)$ quadrant. The flip of the x axis is necessary due to different axis orientations of real world coordinate system and image coordinate system — while real world coordinate system's x axis points to right and y axis points upward, in image coordinate system the y axis points left but the x axis points downwards. This working space preserves consistent scale across the area of traffic scene, provides sufficient spatial resolution (6 pixels per meter) and allows for fast access to image data (in constant time) and efficient memory representation.

To summarize, the system uses in total 4 coordinate systems:

- Raw space — coordinate system based on image coordinates of current video frame
- Reference space — coordinate system based on image coordinates of reference image.
- Real world coordinate system
- Working space coordinate system

When using valid traffic scene annotation and after successful geo-registration of the current video frame, all transformations between these systems are known or can be calculated by matrix multiplication and/or inversion.

3.5 Offset Fix

Due to the planar approximation of the traffic scene, tracked three-dimensional objects are projected onto the plane, resulting in loss of information of their third dimension, and are therefore represented as two-dimensional blobs. We also cannot assume that the scene is captured as orthogonal projection — tracked objects are not projected onto the image directly from above, instead they are captured obliquely. This results in skewed representation of the vehicle and is responsible for difference between the real position of the vehicle on the road and position of visual centre of vehicle (see figure 3.6. In case of bigger vehicles, such as lorries, this difference may lead to significant error in resulting estimated trajectories. We therefore propose following algorithm to fix the said positional difference, called *Offset Fix*.

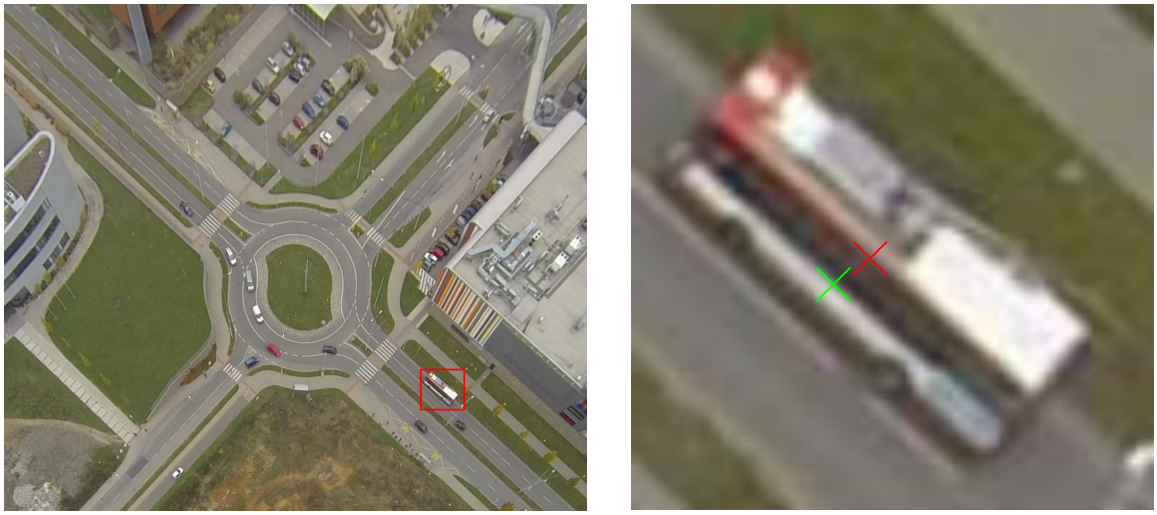


Figure 3.6: Illustration of positional difference between visual centre of selected vehicle (red cross) and actual position of the vehicle at the road surface (green cross).

Firstly, we assume, that the visual centre of vehicle in coordinate system of input video frame is projection of three-dimensional geometric centre of vehicle in real world coordinate system into the two-dimensional coordinate system of input video frame. We also assume that actual position of vehicle on the road surface is an orthogonal projection of three-dimensional geometric centre of vehicle onto the road surface and that the three-dimensional geometric centre of vehicle is at height h above the road surface.

Now, let $\mathbf{x}_{ih}$ be two-dimensional homogeneous vector representing the position of visual centre of detected vehicle in input video frame, $\mathbf{x}_{m3Dh}$ be three-dimensional homogeneous vector representing geometric centre of vehicle in real world three-dimensional coordinate system and $\mathbf{x}_{m3D0}$ its position on the road surface — also three-dimensional homogeneous vector. Let $\mathbf{x}_{m2D0}$ be representation of the position of the vehicle on the road surface in two-dimensional coordinate system of real world (such as UTM map coordinates). Therefore it must be that:

$$\mathbf{x}_{m3Dh} = \begin{bmatrix} \mathbf{x}_{m2D0}\,(0) \\ \mathbf{x}_{m2D0}\,(1) \\ h \\ 1 \end{bmatrix} \tag{3.3}$$

and

$$\mathbf{x}_{m3D0} = \begin{bmatrix} \mathbf{x}_{m2D0}\,(0) \\ \mathbf{x}_{m2D0}\,(1) \\ 0 \\ 1 \end{bmatrix}. \tag{3.4}$$

We need to find projection matrix $\mathbf{P}_h$, for known h, which can be estimated from size of the detected object, so that the following condition is met:

$$\mathbf{x}_{ih} = \mathbf{P}_h \mathbf{x}_{m2D0} \quad . \tag{3.5}$$

To calculate matrix $\mathbf{P}_h$, we need to calculate projection matrix $\mathbf{P}$, which projects real world three-dimensional points onto the plane of input video frame. This matrix is dependent on

position of camera and its intrinsic parameters in form of camera projection matrix.

The pose of the camera can be reconstructed using algorithm of homography decomposition presented by Malis and Vargas in [19], from matrix $\mathbf{H}_{2D}$ which is defined as:

$$\mathbf{H}_{2D} = \mathbf{M}_{mi} C^{-1} \quad , \tag{3.6}$$

where $\mathbf{M}_{mi}$ is inverse of matrix $\mathbf{M}_{mi}$ defined in equation (3.2). The decomposition however have multiple solutions. Due to geometrical and contextual constraints, the incoherent solutions are eliminated and the correct solution is formed by 3×3 rotation matrix R and three-dimensional translation vector t. Together, they can be expressed by 3×4 rigid transformation matrix H:

$$\mathbf{H} = [R|t] = \begin{bmatrix} R_{11} & R_{12} & R_{13} & t_1 \\ R_{21} & R_{22} & R_{23} & t_2 \\ R_{31} & R_{32} & R_{33} & t_3 \end{bmatrix} . \tag{3.7}$$

Therefore, we are able to construct the projection matrix $\mathbf{P}$:

$$\mathbf{P} = \mathbf{CH} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \quad . \tag{3.8}$$

The matrix $\mathbf{P}$ projects three-dimensional real world coordinate point $[X, Y, h]^T$, where h is elevation above the road surface, onto input video frame, forming point $[x, y]^T$:

$$\begin{bmatrix} x/z \\ y/z \\ 1/z \end{bmatrix} = P \begin{bmatrix} X \\ Y \\ h \\ 1 \end{bmatrix} \quad , \tag{3.9}$$

where z is homogeneous coefficient. After multiplication with z it leads to:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \begin{bmatrix} Xz \\ Yz \\ hz \\ z \end{bmatrix} \quad , \tag{3.10}$$

which further leads to:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13}h & p_{14} \\ p_{21} & p_{22} & p_{23}h & p_{24} \\ p_{31} & p_{32} & p_{33}h & p_{34} \end{bmatrix} \begin{bmatrix} Xz \\ Yz \\ z \\ z \end{bmatrix} \quad , \tag{3.11}$$

and further to:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13}h + p_{14} \\ p_{21} & p_{22} & p_{23}h + p_{24} \\ p_{31} & p_{32} & p_{33}h + p_{34} \end{bmatrix} \begin{bmatrix} Xz \\ Yz \\ z \end{bmatrix} \quad , \tag{3.12}$$

where matrix

$$\mathbf{P}_h = \begin{bmatrix} p_{11} & p_{12} & p_{13}h + p_{14} \\ p_{21} & p_{22} & p_{23}h + p_{24} \\ p_{31} & p_{32} & p_{33}h + p_{34} \end{bmatrix} \tag{3.13}$$

is square and generally invertible. This matrix maps two-dimensional real world coordinates with known elevation h above the road surface onto input video frame.

If $\mathbf{x}_{ih}$ and h are known, it is possible to compute $\mathbf{P}_h$ from $\mathbf{P}$ as already shown. Then:

$$\mathbf{x}_{m2D0} = \mathbf{P}_h^{-1}\mathbf{x}_{ih} \quad , \tag{3.14}$$

which represent actual position of the vehicle on the road surface expressed in two-dimensional real world coordinate system. The three-dimensional vectors $\mathbf{x}_{m3Dh}$ and $\mathbf{x}_{m3D0}$ can be expressed from equations (3.3) and (3.4) respectively.

3.6 Summary

This chapter was aimed at the outline of *input pre-processing* stage of the proposed algorithm. It presented a description of traffic scene annotation. Then the steps of video frame undistortion, and its subsequent stabilisation and geo-registration were presented. At the end of the chapter, *Offset Fix* algorithm which deals with geometric distortion of traffic scene due to projective transformation was outlined.

Chapter 4

Detection

This chapter is aimed at the problem of vehicle detection in traffic scene captured in aerial imagery and video sequences. Firstly, the problem of vehicle detection in image data will be introduced. Then we discuss the existing approaches in the task of vehicle detection and aspects of aerial video data that may affect the performance of detection algorithms. Finally, we will describe our suggested approach that is implemented in the detection stage of the proposed system and the motivation behind the decision of its design.

Generally speaking, the purpose of object detection is to detect (i.e. find) instances of certain object class in data. In terms of computer vision, the object detection is usually performed directly in image data. In this part of the work we aim to develop an object detection algorithm which will for given video frame from aerial video sequence find all instances of vehicles present in the traffic scene and extract their positions and bounding rectangles.

4.1 Existing Approaches

The design of efficient and robust vehicle detection methods in aerial images has been addressed several times in the past. In general, these methods can be classified into two categories depending on whether an explicit or implicit model is utilized [20].

4.1.1 Explicit Models

Detection using explicit model approach requires a user provided description of detected object — a generic two-dimensional or three-dimensional model of vehicle. Therefore, the detection predominately relies on geometric features of vehicle, such as edges and surfaces of vehicle body, or cast shadows.

Such explicit models were applied in work of Moon et al. [21], with generic model of car containing its rectangular body boundaries, front and rear windscreen. The detection was carried out in feature space generated from response of Canny edge detector. To speed up the computation, the orientation of the vehicle was assumed to be the same, as the orientation of the road. Very similar approach is used in work of Zhao and Nevatia [22], with addition of possible vehicle shadow presence to its model representation and incorporation of Bayesian network in final decision-making step of detection algorithm. Kozempel and Reulke in [23] provided a very fast solution which is based on four edge kernel filters representing four edges of vehicles. Their responses are fused together to generate detection candidates which are subsequently filtered to reject possible double detections.

This algorithm, however, requires input information about road orientation to the street database.

Another promising usage of explicit model for vehicle detection was proposed by Kim and Malik [24], which used probabilistic three-dimensional model of line features oriented the same direction. Detection of vehicles based on such model was computationally very light-weight, nonetheless, it required detailed definition of possible vehicle shape with probabilistic distributions of its dimensions.

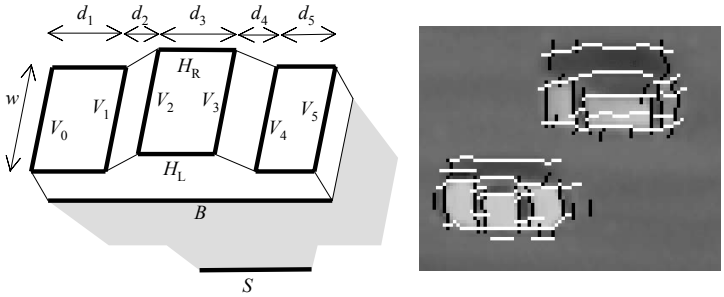


Figure 4.1: Example of an explicit model, as used in the work by Kim and Malik [24]. Left: Vehicle model with annotated dimensions of the vehicle. For each dimension, a probabilistic model of its possible values must be provided. Right: Example of line features that are to be fitted to the model. Source: [24].

4.1.2 Implicit Models

In case of implicit approaches, the internal representation is derived through collecting statistics over the extracted features, such as Histogram of Oriented Gradients (HoG), Local Binary Patterns (LBP), specialised pixel-wise features and so forth. The detection for candidate image regions is performed by computing the feature vectors and classifying them against the internal representation. Among the main generic disadvantages of these approaches are the need for a huge amount of annotated training data, a lot of miss detections of rotated vehicles, and computational expensiveness during the training phase, as the features are usually passed to a cascade classifier training algorithm, such as AdaBoost [25] or more complex dynamic Bayesian networks [26].

To achieve real-time performance, Gleason et al. [27] employed a two-stage classifier. The first stage performs a fast filtration based on the density of corners detected by Harris corner detector, color-spatial profiles of detection candidates. The second stage is more computationally complex: it computes Histogram of Oriented Gradients and Histogram of Gabor Coefficients as features for binary classifier. Similar pre-processing phases are often used in order to make the detection faster and more reliable, as seen in work by Moranduzzo and Melgani [28]. They coupled SIFT feature saliency cues and Support Vector Machine classifier together to generate candidates, which are subsequently grouped together. Their approach is however suitable only for high-resolution aerial images, due to the application of SIFT feature detector. Another strategy for elimination of false positive detections is restricting the areas for vehicle detection by application of street extraction techniques, using region-growing applied to Gaussian smoothed image based on Euclidean colour distance between neighbouring pixels, as seen in work of Truemer et al. in [29], or using 2.5D data, as seen in work of Pacher et al. [30].

In contrast to the above mentioned approaches, which take information from a single image only, Truemer et al. in [31] additionally tried to enhance the performance by incorporating temporal information from motion analysis and background subtraction into the detection process. Another similar approach is seen in [32], where the authors employed a motion analysis by a three-frame subtraction scheme; moreover, they proposed a method for track associations by graph matching and vehicle behaviour modelling.

Another outstanding approach to vehicle detection was presented by Cheng et al. in [33]. They dropped the idea of region-based classification and detection, and instead introduced pixel-wise classification. However features extracted at each pixel (Harris corner detector with Tsai's moment-preserving thresholding [34] method applied to Canny edge detector) are based on local neighbourhood. This approach uses histogram based background colour removal and classification by Support Vector Machine for colour classification and dynamic Bayesian networks for final detection decision.

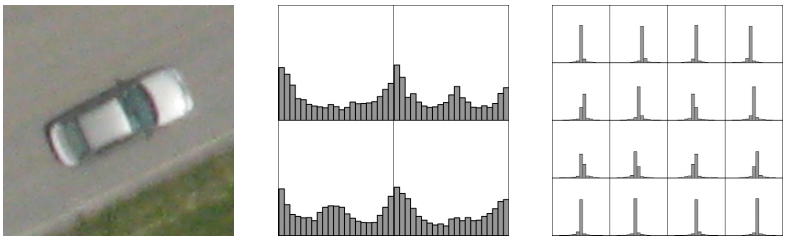


Figure 4.2: Example of features used to construct an implicit model of vehicle. Left: Aerial image of the vehicle. Middle: Histograms of Oriented Gradients (HoG) produced for the given image. Note that the image is divided into four blocks and for each block, its corresponding HoG is computed, dividing orientation space of gradients into 21 bins. Right: Histograms of Gabor coefficients produced by applying 16 filters constructed by combination of four rotations and four phase offsets of Gabor kernel. Source: [27].

4.2 Our Approach

During the design of detection algorithm used in the proposed system we had to take the specific nature of aerial video data into account. As seen on figure 4.3, the visual representation of objects to be detected in video sequences captured by a flying UAV are characteristic by low resolution, blur, low feature salinity and appearance diversity and variability.

To address these characteristics we decided to use an approach that would be able to cope with such data. Due to the low feature salinity and occasionally blurred visual representation of vehicles in video sequences, the approaches based on detection and clustering of more complex visual features (such as SIFT features used in work of Moranduzzo and Melgani [28]) and sharp corners or edges were not considered as feasible. Therefore, we expected that only algorithms that perform object detection by classification of candidates against a model that describes both macro- and micro-structure are suitable for such task.

As a solution, we decided to deploy a detection algorithm based on Local Binary Patterns (LBP) features. LBP features are used widely in face recognition tasks and show promising results even for smaller training sets (common for face recognition systems) [35]. Also, wast number of modifications of LBP feature based detectors have been developed for various

(a) Blur, Low Resolution and insufficient feature salinity.

(b) Appearance diversity of the objects.

(c) Appearance variability of a single object.

Figure 4.3: Characteristic problems of vehicle representation in aerial video data. All images are extracted from evaluation datasets (see section 8.1).

purposes [36] and are incorporated to of publicly available computing libraries [15, 37, 38], which leads to simplicity of implementation. For these reasons, we decided to use cascade classifier based on Multi-scale Block Local Binary Patterns (MB-LBP) features [39].

Also, similarly to presented approaches by Gleason et al. [27] and Truemer et al. in [31], our algorithm performs a fast filtration of all possible detection candidates to reduce false positive ratio and improve the computational speed of classification. This step is called candidate generation and it fuses together the information about traffic scene, temporal motion and already tracked vehicles to provide a set of all reasonable candidates for vehicle classification step.

In the following subsections we firstly describe generation of detection candidates, and afterwards the construction and principle of used MB-LBP classifier is described.

4.2.1 Generation of Detection Candidates

To generate candidates for vehicle detection, input video frame is transformed into working space coordinate system, removing perspective projection effect. This leads to orthographic representation of the scene, reducing the range of possible candidate sizes.

We limit the generation of candidates for vehicle detection to road surface area, which is retrieved from traffic scene annotation. Furthermore, we limit the candidates generation to square candidates which fulfil at least one of the following condition:

- Center of candidate area exhibits the signs of motion. We assume that the initial detection of vehicle may be limited to moving vehicles, as there is no use to track vehicles which are static for the movement of the whole video sequence that is analysed.
- Candidate area is overlapped significantly by currently tracked vehicle. This condition supports generation of detections for currently tracked vehicles. Such generated detections will aid the tracking algorithm, which we will describe in chapter 5.

To detect the motion, we employed background subtraction method based on Gaussian Mixture Model as presented in [40]. This method constructs the background model — the model describing scene without any moving object, using multiple Gaussian nodes (i.e. Gaussian Models) per each pixel. To construct the internal background model, the

Figure 4.4: Illustration of the background subtraction algorithm. Left: input video frame. Middle: illustration of background model — pixels are set to colour value with highest probability. Right: extracted foreground mask. Note that the bus on the right upper corner is present in the background model and it is not signaled in the foreground mask. This is because the bus was stationary for long time interval.

algorithm accepts a sequence of images of the static scene. These images however may contain moving objects — as long as the moving objects do not predominate at the same place for the significant portion of the video sequence, the algorithm can cope with their presence.

By that virtue, in the presented system, the background model is initialised by reference frame transformed into working space coordinate system. Further more, during the video sequence analysis, the model is updated using the video frames from aerial video sequence itself. The video frame is firstly compared to internal background model to detect present motion — the comparison is carried out also per each pixel, by evaluation of Mahalanobis distance of the video frame pixel colour value to the corresponding Gaussian Mixture nodes of given pixel in the background model. Further the distance is compared against a empirically estimated threshold to produce foreground mask. The foreground mask is represented as two-dimensional matrix with element values set to 1 where motion is detected, and to 0, where there is no motion detected. Subsequently, the video frame is used to update the internal background model.

During the development we have encountered the failure of foreground extraction in some video frames, as seen on figure 4.5. This is caused by imperfect transformation estimation in step of geo-registration of input video frame (see section 3.3), which is in turn caused by image deformation due to violent camera shaking. The action cameras are usually equipped with cheaper image capture chips which use rolling shutter. Rolling shutter is a method of image capture, which does not capture the whole image at once (in single moment), but instead by scanning either by rows or columns of the image capture chip. Therefore, each line of the image is captured in time moment with a little temporal offset against the rest of the lines of the image. In presence of fast movement or camera shaking, this results into a undesired distortion of the image. Such distortion cannot be trivially removed and can cause imprecision or failure in process of transformation estimation.

To address this issue the resulting foreground mask is generated by accumulating blurred foreground masks[1] for the current video frame and two previous frames of the video sequence and the subsequent thresholding of the result.

[1]The applied blur is based on convolution of Gaussian kernel with size 7 pixels × 7 pixels and standard deviation of 1.4 pixel.

Figure 4.5: Failure of background subtraction algorithm. Left: input video frame. Middle: illustration of background model. Right: foreground mask. Red arrow points at the area that is mistakenly considered as a foreground.

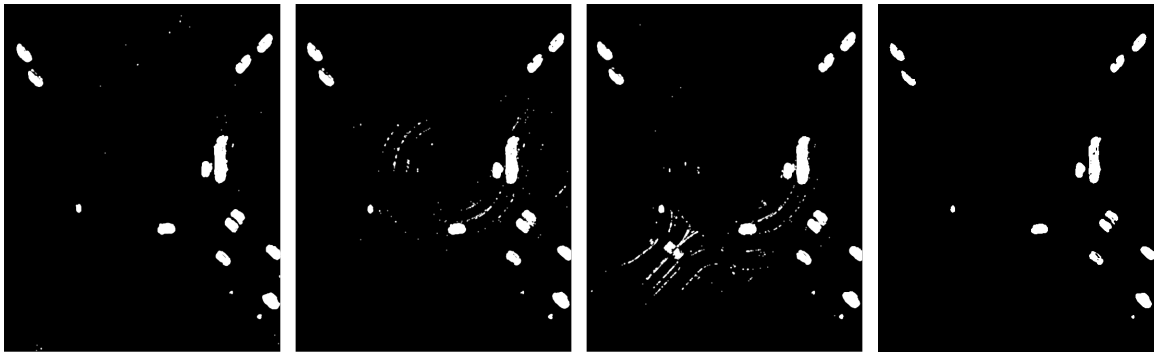


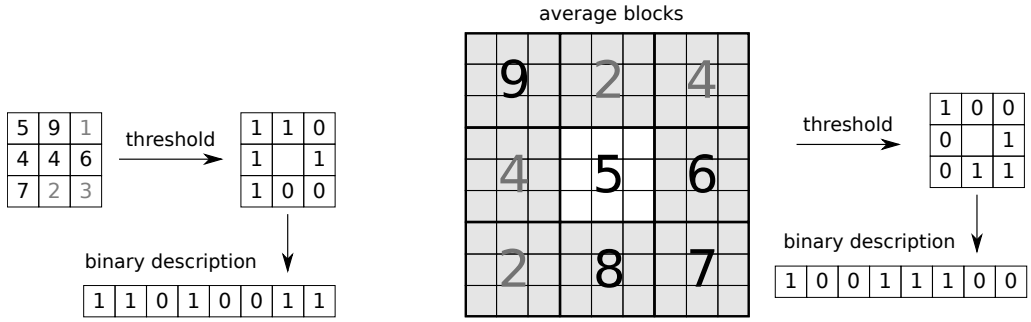
Figure 4.6: Example of forground mask accumulation. The first three images represent foreground masks from current and previous two frames of the video sequence. The resulting fused foreground mask is shown at the fourth image.

To detect the overlap of detection candidate with any of currently tracked vehicles, the position of vehicle must be predicted. To predict the position, the motion model of vehicle as described in chapter 5 is applied. As the prediction may not be perfect, small area around the centre of the tracked vehicle is assumed as vehicle area. Any detection candidate which centre is coincident with this area is considered meeting the condition of overlapping with tracked vehicle.

The resulting set of candidates is represented by two-dimensional array called *detection mask* O_{dm} which is constructed by element-wise fusion of road surface mask O_{rs}, mask of already tracked vehicles O_{tv} and foreground mask O_{fm}:

$$O_{dm} = O_{rs} \,\&\, (O_{tv} \,|\, O_{fm}) \quad , \tag{4.1}$$

where & is binary operator of element-wise logical AND and | is binary operator of element-wise logical OR. Detection mask, with addition of range of all possible vehicle sizes (3 m to 6 m or 18 pixels to 42 pixels in working space coordinate system), forms a compact representation of all reasonable vehicle detection candidates — objects present on the road surface that are moving or being tracked.



(a) LBP features.

(b) MB-LBP features.

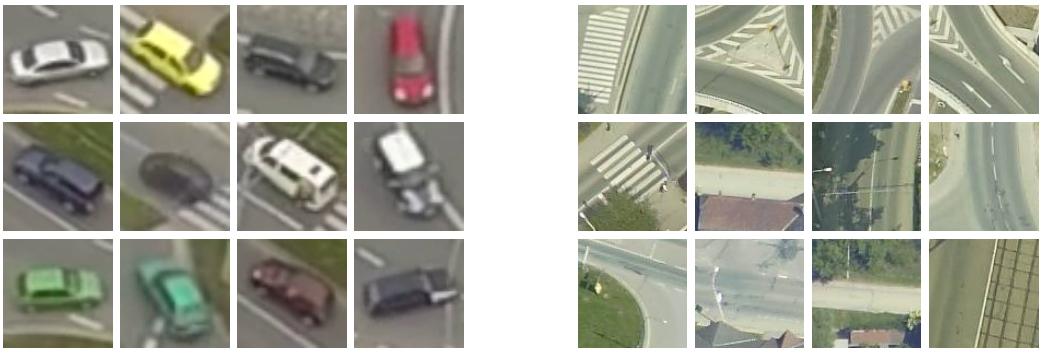
Figure 4.7: Comparison of extraction of simple LBP features (left), and MB-LBP features (right). LBP features use only pixel values of very neighbouring pixels, which are thresholded according to value of central pixel. The resulting values form binary descriptor. MB-LBP features are computed by calulating average values in neigbouring blocks of pixels. Average values are then tresholded according to average value of central block. The rest of the algorithm is unchanged. Note, that size of blocks may be ordinary.

4.2.2 Vehicle Detection

For automatic selection of appropriate features and construction of a robust detector, we have utilized Viola and Jones's Adaboost algorithm [41]. In their original paper, the authors used HAAR features for object representation; however, HAAR features have poor robustness to illumination changes and may lead to high false alarm rate. Also, learning and detection phase of such detector is computationally expensive. To alleviate these weaknesses, its modification using MB-LBP features, has been employed. Comparing to the original LBP calculated in 3×3 neighbourhood, MB-LBP is computed from average values of block subregions, and is therefore more robust, since it encodes both micro-structures and macro-structures providing more complex image representation.

According to [42], MB-LBP features have significantly smaller false alarm rate than HAAR features, while keeping comparable hit rate. Another motivation to choose Adaboost cascade coupled together with MB-LBP feature classifier is ease of implementation and possible future detector specialisation or restrictions of search domain, as we are planning to develop the proposed system beyond the scope of this master thesis.

To train the classifier, a hand annotated dataset of positive and negative samples of vehicles and their surroundings was generated. The aerial image data for this purpose was provided by *RCE systems s.r.o.* It contained video sequences captured by flying UAVs and high-resolution aerial images captured by plane. These video sequences and images were hand-annotated to produce set of learning data for detection algorithm. The dataset contained more than 20 000 positive samples and 20 000 negative samples in total. The positive sample set contained square images of car vehicles of different types, colours and orientations. Trucks, buses and other heavy vehicles were excluded from the dataset due to their insufficient quantity in the provided imagery data. This however limits the ability of the presented system to detect (and subsequently track) only car vehicles. Negative samples contained images of road surface, surroundings of intersections, road structures and pathways. The special emphasis was put on gathering sufficient number of samples containing road markings (horizontal signalisation), as those were more likely to be falsely



(a) Positive samples.

(b) Negative samples.

Figure 4.8: Selected examples of samples used to train the vehicle classifier. Note that the samples are scaled for a better illustration.

detected by trained detectors. All samples were collected in raw image data without geo-registration or other image transformations. For the purpose of classifier training, the positive samples were resized to size of 32×32 pixels. The selected examples of training data are shown in figure 4.8.

The resulting trained classifier consisted of 18 stages of decision cascade of simple MB-LBP classifiers. For the further improvement of detection assistance for tracking algorithm, the trained detector was split to two classifiers:

- *strong vehicle classifier* — consisting of all 18 stages of trained classifier, having small false alarm rate. The detections signalled by this classifier, further referenced as *strong detections*, are considered as very significant indication of vehicle presence in detection candidate, and are treated as such further in tracking stage of the algorithm, especially as basis for new tracking targets.
- *weak vehicle classifier* — consisting of first 11 stages of trained classifier, having higher false alarm rate. This classifier is more benevolent than *strong vehicle classifier* accepting much more detection candidates as vehicles and its output is used to aid vehicle tracking, acting as tracking attractors. These detections, further referenced as *weak detections*, form basis for *heat map*.

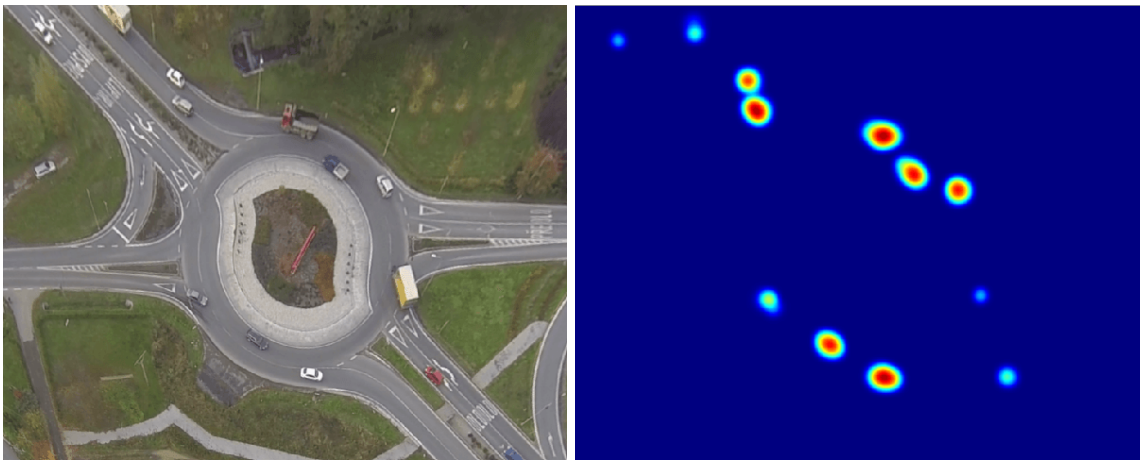
4.2.3 Heat Map

To speed up the operations with *weak detections*, we proposed following function which for given point in two-dimensional real world coordinate system returns the significance of this point according to the set of *weak detections* :

$$h\left(\mathbf{x}\right) = 1 + \sum_{d \in \mathcal{D}_{weak}} f\left(\mathbf{x}, \mathbf{x}_d, \sigma_d\right) \quad , \tag{4.2}$$

where $\mathcal{D}_{weak}$ is a set of current *weak detections*, $\mathbf{x}_d$ is a position of the detection d in real world coordinate system, σ_d is size of detection d in real world coordinate system, $f\left(\mathbf{x}, \boldsymbol{\mu}, \sigma\right)$ represents the value of a multivariate normal distribution $\mathcal{N}\left(\boldsymbol{\mu}, \boldsymbol{\Sigma}\right)$ expressed at point $\mathbf{x}$. Matrix Σ is constructed as follows:

$$\Sigma = \begin{bmatrix} (0.16\sigma)^2 & 0 \\ 0 & (0.16\sigma)^2 \end{bmatrix} \quad . \tag{4.3}$$



(a) Geo-registered video frame (b) Corresponding *heat map*

Figure 4.9: Example of *heat map* constructed from a set of weak detections.

Heat map is two-dimensional array consisting of values of heat function $h(\mathbf{x})$ across whole area of traffic analysis scene.

4.3 Summary

This chapter provided deployed solution to the problem of vehicle detection in aerial imagery. Firstly, an overview of existing approaches in this field was given. In that context, a detection algorithm was proposed, based on pre-filtration of detection candidates using background model and subsequent classification of candidates with two cascade classifiers. At the end of the chapter, *heat map* structure was described.

Chapter 5

Tracking

This chapter explains the tracking algorithm used in the tracking stage of the proposed system. Firstly, the definition of tracking problem is presented, then the existing approaches of vehicle tracking in aerial imagery are analysed. The rest of the chapter is devoted to description of tracking algorithm that is deployed for task of vehicle tracking in the proposed system.

The aim of vehicle tracking is to retrieve the sequences of states of all target vehicles during their movement through the region of interest in the traffic scene, given the geo-registered aerial video sequence and the results of the preceding stage of the system — vehicle detections. This is the most important part of the proposed system as the generated sequences of states are a raw representation of vehicle trajectories. Therefore, the ability of tracking algorithm to track vehicles continuously and without error is a crucial aspect affecting the quality of generated output of the whole system.

5.1 Existing Approaches

Video-based moving object tracking is one of the most popular research problems in computer vision. However, it is still a challenging task due to the presence of noise, occlusion, dynamic and cluttered backgrounds, and changes in the appearance of the tracked object, which are all very common in aerial images. Numerous tracking approaches have been presented in recent years; a detailed survey can be found in [43]. Typically, object tracking algorithms employed in traffic analysis may be divided in two groups: algorithms using Bayesian filters and off-line data association algorithms.

5.1.1 Bayesian Filtering

Bayesian filters present a family of estimation algorithms that use series of noisy (uncertain) measurements observed over time and produces estimates of unknown variables that tend to be more precise than those based on single measurement data [44].

The application of Kalman filter for object tracking was proposed as early as in 1970 [45], and since then it has been used widely in the field. For purpose of multi-target vehicle tracking, a more mature approach incorporating Kalman filter algorithm was proposed already in 1993 in work of Koller et al. [46], citing its robustness, simplicity and speed. Their approach consisted of detection and segmentation of motion based upon background modelling. Tracking targets were represented by moving blobs bounded by closed cubic spline contours. They decomposed the estimation problem into two Kalman filters — one

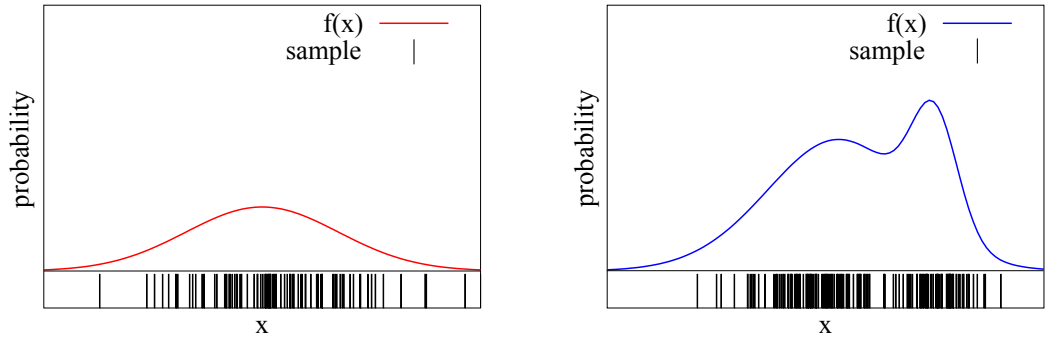


Figure 5.1: Two cases of probability function and its respective approximation by particles. Left (red): The probability function $f(x)$ can be approximated by single-modal Gaussian distribution. Righ (blue): The probabiltity function $f(x)$ can not be approximated by single-modal Gaussian distribution accuratelly. However, the particle distribution can model probability accuratelly, depending on the number of particles. Figure based upon: [52].

for the motion estimation, and the second one for the shape estimation. Additionally, to deal with multi-target occlusions, they proposed occlusion reasoning based on geometric properties of the scene and camera position.

To deal with the non-linearity of the target behaviour model, the extended Kalman filter using first-order approximation of the non-linear equation of the dynamics model was proposed. The application of the extended Kalman filter for moving vehicle tracking was suggested by Obolensky in [47]. However, Kalman filter, representing linear quadratic estimation algorithm, is suited only for tasks modelled by single-modal Gaussian probability density [48], which is a limiting factor, especially in tasks of vehicle tracking in aerial imagery.

Therefore, most recently, Bayesian bootstrap filter algorithm, also called particle filter, is being employed for object tracking in cluttered background or multi-target tracking. This is due to the probabilistic sampling nature of the algorithm: Bayesian bootstrap filter samples the probability domain of object state by a set of particles, which are at the end of each iteration resampled according to their importance weight based upon their consent with observation. If the probability domain is sampled with sufficiently numerous set of particles, the probability density function can be arbitrary — not limited to normal distribution [48]. This is a very important property of Bayesian bootstrap filter algorithm, especially when used in object tracking in cluttered background and with low feature salinity of tracked objects, where the state probability function may happen to be multi-modal on many occasions.

For the task of multiple object tracking, Bayesian bootstrap filter algorithm was proposed in work of Karlsson and Gustafsson [49]. The application of particle filter for unmanned aerial vehicle surveillance is thoroughly described in work of Samuelson [50]. An outstanding approach was presented by Hess and Fern in [51]. For the purposes of tracking the players during a football match, they developed an algorithm which uses pseudo-independent particle filters parametrized by log-linear models with error-driven discriminative filter training. During the training of the filter, the algorithm attempts to minimize the squared residual mean of the filter output by modifying the weights of various features used in the observation step of the filter.

5.1.2 Offline Data Association

As the computational abilities of computers have improved significantly over the past few years, the most current approaches in vehicle tracking from aerial or satellite imagery aim at off-line optimization of data association, usually using more complex graph based algorithms or probabilistic correspondence mapping.

In [53], Reilly et al. presented tracking by data association based upon bipartite graph matching in graph of possible tracking pairs with edges weighted according to spatial proximity and velocity orientation components calculated from consecutive video frames. Similar approach was used by Xiao et al. in [54], but instead, the edges of joint probability graphs are weighted by values of kinematic measure, shape matching and appearance matching of the target pairs.

Hierarchical approach in vehicle tracking is presented in work of Prokaj et al. [55], which is based on hierarchical connecting of shorter estimations of parts of trajectories — tracklets, into longer ones, eventually leading to forming whole trajectories.

In work done by Saleemi and Shah in [56], they maintain multiple possible candidate tracks per object using a context-aware association (vehicle leading model, avoidance of track intersection) and applying a weighted hypothetical measurement derived from the observed measurement distribution to revise temporal tracking correspondence.

5.2 Our Approach

To track vehicles in the proposed system, a sequential Monte Carlo method, also known as Bayesian bootstrap filter or particle filter, was adapted. It suits the task of tracking multiple objects through a complex scene due to its simplicity and ability to cope with non-linear and non-Gaussian tasks. Instead of estimating a complete state of the traffic scene (i.e. the states of all present vehicles), we deploy one Bayesian bootstrap filter per each target.

For an easier understanding of the following sections, we introduce the term *track*. *Track* is defined as an incomplete sequence of vehicle states estimated by algorithm of vehicle tracking for the target. Each estimated track corresponds to exactly one tracking target and vice versa.

For each processed video frame of the sequence, the algorithm executes the following three steps:

- *Detection–Track Association* — associates weak and strong detections to currently tracked vehicles.
- *Tracks Update* — deploys Bayesian bootstrap filter to perform tracking of vehicles from previous to current video frame according to their previous state, a set of weak detections, the associated detections and the current geo-registered video frame.
- *Tracks Management* — adds or removes tracking targets according to their states.

In the following subsections we will provide an insight into the whole tracking algorithm.

5.2.1 Detection–Track Association

For each detection from both sets of weak and strong detections, the best fitting currently tracked target is found and vice versa, while strong detections being favoured. Formally, let

$\mathcal{D}_{weak}$ be a set of weak detections, $\mathcal{D}_{strong}$ be a set of strong detections, $\mathcal{D} = \mathcal{D}_{weak} \cup \mathcal{D}_{strong}$ be a set of all detections, $\mathcal{T}$ be a set of all currently tracked targets and $\mathcal{A}$ is set of constructed associations. Then for every constructed association $(d, t) \in \mathcal{A}$, where $d \in \mathcal{D}$ and $t \in \mathcal{T}$, it must be that:

$$\begin{aligned} \forall d' \in \mathcal{D} \setminus \{d\} : \operatorname{diff}(d', t) \geq \operatorname{diff}(d, t) \vee \\ d \in \mathcal{D}_{strong} \wedge d' \in \mathcal{D}_{weak} \end{aligned} \tag{5.1}$$

and

$$\begin{aligned} \forall t' \in \mathcal{T} \setminus \{t\} : \operatorname{diff}(d, t') \geq \operatorname{diff}(d, t) \wedge \\ \operatorname{diff}(d, t) \leq \mathit{diff}_{max} \quad , \end{aligned} \tag{5.2}$$

where diff_{max} is a parameter of the algorithm and $\operatorname{diff}(d, t)$ is a function, which returns the difference of rectangles representing detection d and target t. It is defined as follows:

$$\operatorname{diff}(d, t) = \frac{|r_d| + |r_t| - 2\,|r_d \,\&\, r_d|}{|r_d| + |r_t|} \quad , \tag{5.3}$$

where r_d is a rectangle representing detection d, r_t is a rectangle representing target t, function $|r|$ returns the area of rectangle r and binary operator & represents an intersection of given operands.

Due to the said conditions, associations may not completely cover the entire sets of currently tracked targets or detections. The strong detections which are not associated with any target will form basis for new tracking targets in Tracks Management step (refer to subsection 5.2.4 for more information).

5.2.2 Target Representation

For the algorithm to be able to track its target, the description of tracked object – target representation – is necessary. In our case, due to low resolution of tracked objects, we employed simple target representation: rectangular descriptor template derived from detection area of the target. It consist of 3 colour channels (red, green and blue) and 3 edge map channels, which are constructed as a sum of absolute responses of Scharr operator in both x and y directions in the image, separately for all three colour channels. This template is extracted from input video frame transformed into work space, bounded by area defined by initial detection of the target and resized to rectangle of 32×32 pixels. This type of representation is able to carry both spatial and colour information and its computation is very fast.

To achieve the plasticity of target representation, the template $\mathbf{T}_t$ of target t is updated over the period of tracking if at least one of the following events happens:

- There is a currently associated strong detection to the target t during the *Detection-Track Association* step (see subsection 5.2.1).
- There is a currently associated weak detection to the target t during the *Detection-Track Association* step (see subsection 5.2.1), and the value of *heat* function evaluated at the estimated position of the target t is greater than threshold T_{heat}. This value is calculated from *heat map* according to equation (4.2).

Figure 5.2: Examples of selected target descriptor templates. Each column represents the same target. Upper row: RGB channels of descriptors. Lower row: Edge channels of descriptors visualised in RGB space. Histograms of all descriptors are equalised for a better illustration.

In the case of template update, the values of the template are altered by weighted average of the former template and the new template extracted from the currently processed video frame, where the former template has weight of $(1 - \alpha)$ and the new template has weight of α. Value α represents the update rate of template and is given as a parameter of the algorithm. Additionally, to prevent undesirable swaps between the targets, the template update is disabled if multiple targets overlap.

5.2.3 Bayesian Bootstrap Filter

To illustrate the principle of the Bayesian bootstrap filter in greater detail, we first explain basic sequential Monte Carlo approximation. Generally, Monte Carlo techniques are used to estimate the unknown parameters of some entity (or system) by numerous random sampling. Now, let us assume that we do not know the state of the entity, but we are able to extract an observation — a value that is somehow correlated to the state of the entity. Also, from the state of the entity, we can calculate the observation, but we are not able to do it the other way around — directly extract the state of the entity from the observation. The sequential Monte Carlo approximation methods can be seen as a simulation of randomly generated models of the entity — particles — and comparing the observations of the particles to the observations of the entity. Each particle is assigned a weight — a number that represents how closely its observations are similar to the observations of the entity. As time evolves, in each step the weights of all particles are updated according to the most current observation, until the end of the simulation is reached. At the end of the simulation, the weight of every particle is proportional to the probability of its state to be the true state of the analysed entity. This approach in fact models the probability function of entity state by random sampling across the subspace of all possible states [48].

The case when internal state of the entity changes as time evolves is handled by incorporating transition step. The transition step alters the states of the particles according to dynamics, which models expected behaviour of entity in such state, and random noise. In such case, for the reasonable estimation of the true entity state, a huge number of particles needs to be generated. The drawback of such approach is its computational complexity. Also, it was discovered that many of the particles tend to reach weights close to zero (or zero) very quickly. This phenomenon is called weight degeneracy. To address this problem, a reasampling technique was introduced to the algorithm in work of Gordon et al. [57], resulting in the so called Bayesian bootstrap filter. The Bayesian bootstrap filter deploys

sampling-importance resampling (SIR) scheme in order to reduce the computational costs. SIR is applied in each iteration of Bayesian bootstrap filter algorithm as follows [49, 48]:

1. **Particle Evaluation:** Every particle is assigned a weight according to the similarity of its current observation to the current observation of modelled entity.
2. **Weight Normalisation:** Weight of every particle is divided by the sum of weights of all particles. Therefore, the sum of normalized weights of all particles is 1.
3. **Resampling:** generates a new set of particles by resampling the particles from the original set with probability proportional to the weights of the particles.
4. **Transition:** alters the states of the particles according to their dynamics model and random noise.

In the presented tracking algorithm, we deployed a set of Bayesian bootstrap filters — one for each target. Therefore, each Bayesian bootstrap filter estimates the sequence of states of a single vehicle. The analysed entity is represented by the vehicle. The internal state of the entity is represented by the position of the vehicle, velocity vector and size of its bounding rectangle. The observation of the entity is based on the descriptor template of target representation (see subsection 5.2.2). The observation of particle is based on descriptor template derived from its estimated state. In the following paragraphs, we will provide a further insight into the implementation of this approach.

Dynamics model and noise

In our system, the dynamics model of the tracked entity, and therefore a particle from the particle filter, is defined by following parameters:

- $\mathbf{x}$ — two-dimensional vector representing position of the centre of the vehicle in real world coordinate system.
- $\mathbf{v}$ — two-dimensional vector representing vehicle velocity in real world coordinate system normalised by video frame rate. Therefore, if $\mathbf{v}_{real}$ is real world velocity of the vehicle, then $\mathbf{v} = \mathbf{v}_{real}/f_{video}$, where f_{video} is video frame rate.
- s — size of the bounding rectangle of the tracked vehicle in real world coordinate system. The size represents the lengths of all sides of the rectangle, limiting the shape of the bounding rectangle to a square.

All these parameters forms five-dimensional particle state vector $\mathbf{p}$ which is used in calculations:

$$\mathbf{p} = \begin{bmatrix} \mathbf{x}\,(0) \\ \mathbf{x}\,(1) \\ \mathbf{v}\,(0) \\ \mathbf{v}\,(1) \\ s \end{bmatrix} \quad . \tag{5.4}$$

As the transition model, we consider target position to be an integration of target velocity. Therefore, the transition between two consecutive frames can be represented by following

matrix:

$$\mathbf{D} = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad , \tag{5.5}$$

This model changes the particle position $\mathbf{x}$ according to its velocity. All other parameters are assumed unchanged during the transition. The transition from particle $\mathbf{p}^{(i)}$ defined in time moment of video frame i into particle $\mathbf{p}^{(i+1)}$ defined in time moment of video frame $i+1$ is therefore:

$$\mathbf{p}^{(i+1)} = \mathbf{D}\left(\mathbf{p}^{(i)} + \mathbf{n}\right) \quad , \tag{5.6}$$

where $\mathbf{n}$ is noise vector:

$$\mathbf{n} = \begin{bmatrix} n_{x0} \\ n_{x1} \\ n_{v0} \\ n_{v1} \\ n_s \end{bmatrix} \quad . \tag{5.7}$$

Therefore, the equation (5.6) has the same meaning as the following:

$$\mathbf{x}^{(i+1)} = \mathbf{x}^{(i)} + \mathbf{v}^{(i+1)} + \begin{bmatrix} n_{x0} \\ n_{x1} \end{bmatrix} \quad , \quad \mathbf{v}^{(i+1)} = \mathbf{v}^{(i)} + \begin{bmatrix} n_{v0} \\ n_{v1} \end{bmatrix} \quad \text{and} \quad s^{(i+1)} = s^{(i)} + n_s \quad . \tag{5.8}$$

The elements of noise vector are generated randomly according to normal distribution $\mathcal{N}(\mu, \sigma^2)$, where $\mu = 0$ and value of σ are user defined parameters for each element of noise vector. In case of velocity noise and size noise, the parameter σ is constant during the entire course of tracking, whilst for position noise, the value of parameter σ evolves. Let j be a sequential index of video frame from analysed video sequence when the tracking of target t modelled by particle with state vector $\mathbf{p}$ was commenced, and i be sequential index of the current video frame. Also let $\sigma_{\mathbf{x}}^{(i)}$ be covariance of normal distribution generating values of random noise n_{x0} and n_{x1}. Then its value is:

$$\sigma_{\mathbf{x}}^{(i)} = \sigma_{\mathbf{x}}\left(1 + f^i m\right) \quad , \tag{5.9}$$

where $\sigma_{\mathbf{x}}$ is value of $\sigma_{\mathbf{x}}^{(i)}$ for $i \to \infty$, f is falloff rate and m is initial multiplier of covariance. All three values are input parameters of the tracking algorithm. Figure 5.3 illustrates the dependence of $\sigma_{\mathbf{x}}^{(i)}$ on parameters $\sigma_{\mathbf{x}}$, f and m.

This approach is intended to help the initialisation of the filter, especially the initialisation of velocity parameters of particles. The application of this approach causes the position of particle to be affected more by random noise than by its velocity during the early stage of tracking, and afterwards slowly elevating the effect of velocity. This way, the particle's velocity may slowly adapt while elevating its effect on particle behaviour.

Evaluation and Resampling

The evaluation of particles is based on two aspects: appearance similarity and attraction factor. The appearance similarity models how much is the given particle visually similar to the visual representation of the target — its descriptor template. It is represented by

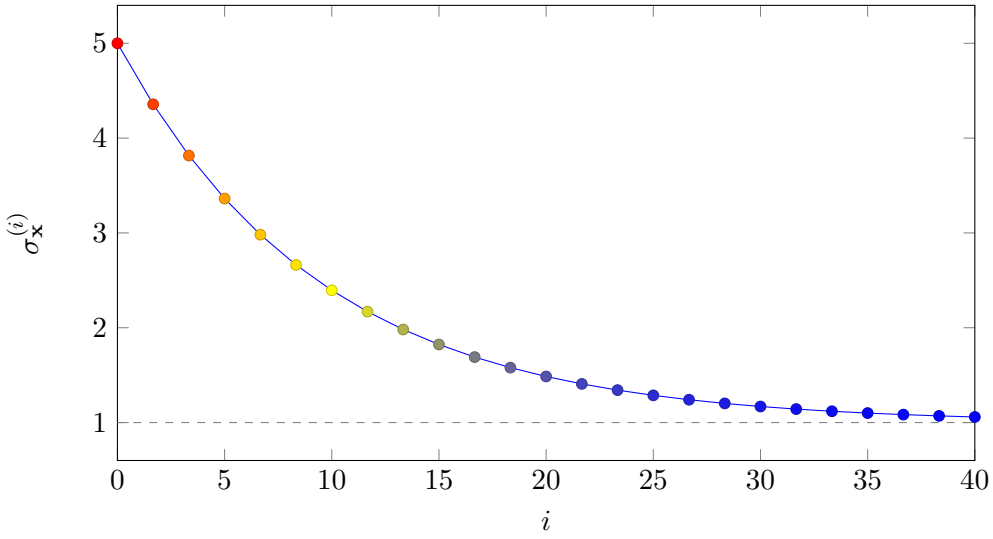


Figure 5.3: Graph of function $\sigma_{\mathbf{x}}^{(i)}$ defined in equation (5.9) according to values of i for parameters $\sigma_{\mathbf{x}} = 1$, $f = 0.9$ and $m = 4$.

appearance similarity function $App\,(p,t)$ of particle p to target t, which is evaluated as follows:

$$App\,(p,t) = \frac{1}{1 + \mathrm{SAD_C}\,(T_t, T_p)} \quad , \tag{5.10}$$

where T_t is descriptor template of target t, T_p is descriptor template of fictitious target based on particle p, and $\mathrm{SAD_C}\,(T_1, T_2)$ is sum of absolute differences of templates T_1 and T_2 across all their channels. The differences are, however, spatially weighted by circular mask around the centre of the templates at point $c = (16, 16)\,\mathrm{px}$, with radius of $16\,\mathrm{px}$, to emphasise the central areas of the descriptor templates, where vehicles are to be present.

The attraction factor represents the aid of the detection stage of the system. Its purpose is to keep the particles of the filter at the region with higher probability of vehicle presence. For each particle p, it is represented by attraction factor function $Att\,(p)$, which is defined as:

$$Att\,(p) = h\,(\mathbf{x}_p) \quad , \tag{5.11}$$

where $h\,(\mathbf{x})$ is *heat function* as defined in equation (4.2) and $\mathbf{x}_p$ is the position of particle p in real world coordinate system.

At the particle evaluation step, each particle is assigned its importance weight, which is later used to estimate state of the target and resample the set of particles. The importance weight $\mathcal{W}\,(p,t)$ of particle p for tracked target t is defined as:

$$\mathcal{W}\,(p,t) = e^{App(p,t)^2 \cdot Att(p)} \quad . \tag{5.12}$$

The estimated state $\mathcal{E}\,(t)$ of the target t is represented by the highest-weighted particle (*maximum a posteriori*), i.e.:

$$\mathcal{E}\,(t) = \underset{p \in \mathcal{X}_{(t)}}{\arg\max}\,(\mathcal{W}\,(p,t)) \quad , \tag{5.13}$$

where $\mathcal{X}_{(t)}$ is the set of all particles of the filter modelling tracked target t.

Resampling step of the filter for particle set $\mathcal{X}_{(t)}$ is carried out using weight proportionate random selection, also known as roulette wheel principle, according to values of particle's importance weight. The new (resampled) set $\mathcal{X'}_{(t)}$ is created as an empty set and is computed from set $\mathcal{X}_{(t)}$ by repeatedly inserting the copy of randomly selected particle p_i from $\mathcal{X}_{(t)}$ until the size of the set $\mathcal{X'}_{(t)}$ reaches the size of set $\mathcal{X}_{(t)}$. The probability $q\left(p_i\right)$ of selection of particle $p_i \in \mathcal{X}_{(t)}$ for resampling is following:

$$q\left(p_i\right) = \frac{\mathcal{W}\left(p_i, t\right)}{\sum_{p \in \mathcal{X}_{(t)}} \mathcal{W}\left(p, t\right)}. \tag{5.14}$$

5.2.4 Tracks Management

During this step of tracking stage, the new targets for tracking are being introduced into the system, the tracks of targets are updated, and tracking termination of selected targets is performed.

The introduction of new target is a result of existence of a strong detection that has not been associated to any currently tracked objects during Detection-Track Association step (see subsection 5.2.1). Let $\mathcal{D}'_{strong}$ be a set of current unassociated strong detections, e.g. for which the following is true:

$$\forall d' \in \mathcal{D}'_{strong} : \nexists \left(d', t\right) \in \mathcal{A} \quad , \tag{5.15}$$

where $\mathcal{A}$ is a set of current associations. For every unassociated detection $d' \in \mathcal{D}'_{strong}$, a new tracking target t^* is added to the set of active targets and is initialized as follows:

- The bounding rectangle r_{t^*} of the target t^* is copy of bounding rectangle of the detection d'.
- The description template model T_{t^*} of the target t^* is extracted from the patch of current video frame transformed into work space, bounded by rectangle r_{t^*}.
- New particle filter $\mathcal{F}_{t^*}$ is created with dynamics model described in subsection 5.2.3, and with particle set $\mathcal{X}_{(t^*)}$, for which each particle $p \in \mathcal{X}_{(t^*)}$ is initialised with parameter vector $\mathbf{p}$:

$$\mathbf{p} = \begin{bmatrix} \mathbf{x}\left(0\right) \\ \mathbf{x}\left(1\right) \\ \mathbf{v}\left(0\right) \\ \mathbf{v}\left(1\right) \\ s \end{bmatrix} \quad , \tag{5.16}$$

  where position $\mathbf{x}$ is initialised as the centre of rectangle r_{t^*}, velocity $\mathbf{v}$ is initialized to zero, and size s is initialized as the length of the side of rectangle r_{t^*}. Rectangle r_{t^*} is actually a square, as every yielded detection from both weak and strong vehicle classifiera is defined as square-shaped. This is due to the condition that every detection candidate is considered as square area — see subsection 4.2.1 for more details.

For each target t in the currently tracked object list, its generated track is updated. Let $\mathcal{E}(t)$ be estimated state of target t calculated according to equation (5.13). As the tracking algorithm tracks the centre of visual representation of the vehicle, its position on the road surface is distorted and has to be adjusted to match the actual position of the tracked vehicle on the road surface in real world coordinate system. To do so, we employ formerly

proposed algorithm *Offset Fix* as described in section 3.5, to transform estimated state $\mathcal{E}(t)$ to adjusted estimated state $\mathcal{E}'(t)$. Finally, the track generated for target t is appended by the state $\mathcal{E}'(t)$.

Tracking of target $t^\dagger$ is terminated when one of the following conditions are met:

- Target $t^\dagger$ leaves the area defined by annotated road surface.
- Target model of target $t^\dagger$ has not been updated for a certain amount of time steps.
- Target t is overlapping with another target t' for a certain amount of time steps and the following condition is met:

$$App\left(p^*_{t'}, t'\right) > App\left(p^*_{t^\dagger}, t^\dagger\right) \quad , \tag{5.17}$$

  where $App\left(p, t\right)$ is appearance similarity function as described in equation (5.10), $p^*_{t'}$ is the highest weighted particle of particle set $\mathcal{X}_{(t')}$ tied with target t' and $p^*_{t^\dagger}$ is the highest weighted particle of particle set $\mathcal{X}_{(t^\dagger)}$ tied with target $t^\dagger$.

In case of tracking termination of target $t^\dagger$, its generated track is analysed. If it is found that the target $t^\dagger$ has fully entered the region of interest of the traffic scene through an entry gate, passed through it and left trough an exit gate, in that order, it is considered as successful tracking of target $t^\dagger$ and the generated track is saved for *Data Post-processing* phase. Otherwise, the tracking is considered as unsuccessful and is therefore rejected.

5.3 Summary

This chapter provided an insight into the tracking algorithm deployed in the proposed system. First, it introduced the existing approaches to the problem of vehicle tracking in aerial video sequences. In that context, we described our approach based on Bayesian bootstrap filter with the aid of detection clues generated in the detection stage of the system.

Chapter 6

Data Post-processing

Data Post-processing phase is the last phase of the proposed system. Its purpose is to prepare trajectory data for vehicle behaviour analysis. For that purpose, it is necessary that generated trajectories are defined for every time moment of the respective vehicle's presence in the region of interest, and the spatio-temporal data are suitable for estimation of vehicle velocity and acceleration. Namely, interpolation of missing parts of tracks, their smoothing and re-filtration of trajectory database is carried out. This way the database of trajectories is generated from the set of successful tracks. The following sections provide an insight into the algorithms applied in this phase.

6.1 Interpolation of Skipped Frames

During the analysis of aerial video sequence, some of the input video frames may not be mapped onto the reference image successfully, due to the failure of video geo-registration algorithm (see section 3.3). This may happen for various reasons — a temporal occlusion of a major part of the scene by a building or a tree, a sudden change of the illumination caused by direct sunlight or a change of camera point of view due to strong wind. In these and many other cases, the image correspondence between affected video frame and reference frame may not be established properly, and therefore the transformation between the video frame and reference image is unknown.

In *Vehicle Detection and Tracking* phase, the proposed system deals with such situation by skipping the analysis of the affected video frame. To compensate for increased time difference between the preceding and following analysed frames, the dynamics model of vehicle (see subsection 5.2.3) is applied on the particles of the particle filter. This may still lead to inaccurate estimation of the target's state in the affected frame; therefore the information about skipped frame is stored to allow more accurate interpolation of the missing target's state in *Data Post-processing* phase.

The estimation of missing data in *Data Post-processing* phase is carried out by interpolation from the neighbouring states. For every event of skipped frame i, the affected targets trajectories are analysed. For every affected target t, its missing state $\mathcal{E}\left(t\right)^{(i)}$ in the frame i is estimated by linear interpolation based on two nearest known target states in time domain.

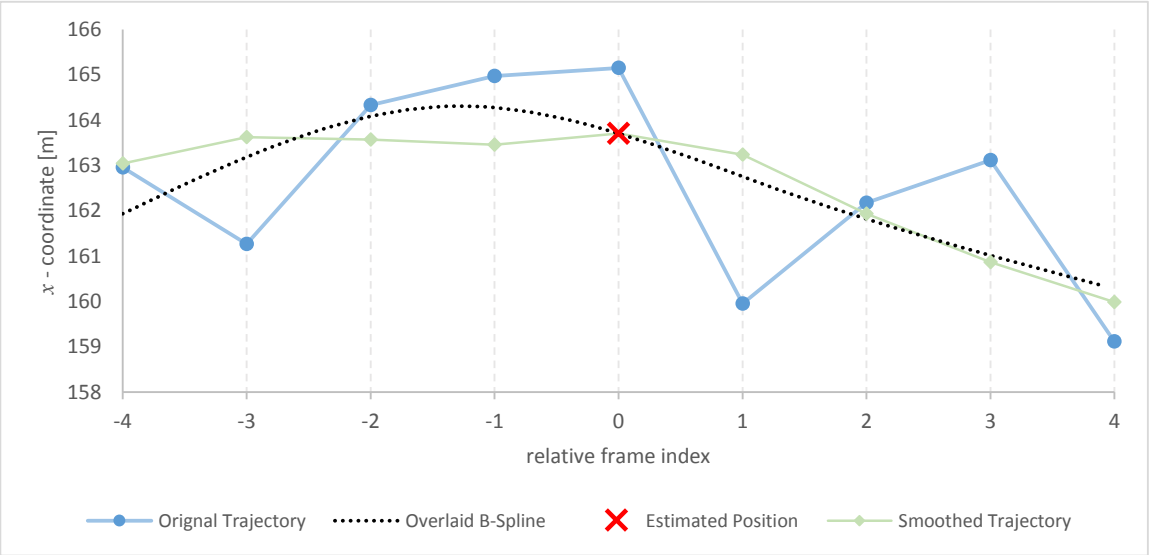


Figure 6.1: Illustration of smoothing window with width of $d_w = 9$ in the neighbourhood of estimated state at relative frame index 0. For better illustration only x-coordinates are visualised. The state of the trajectory at relative frame index 0 is approximated using neighbouring states in smoothing window (with relative frame indices -4 to 4). These states are overlaid by approximating B-spline curve. The value of the curve at relative frame index 0 represents smoothed value of the original state, visualised as the resulting state. This way, each point of the trajectory is smoothed out. The segment of the resulting respective trajectory is illustrated in ligth green.

6.2 Trajectory Smoothing

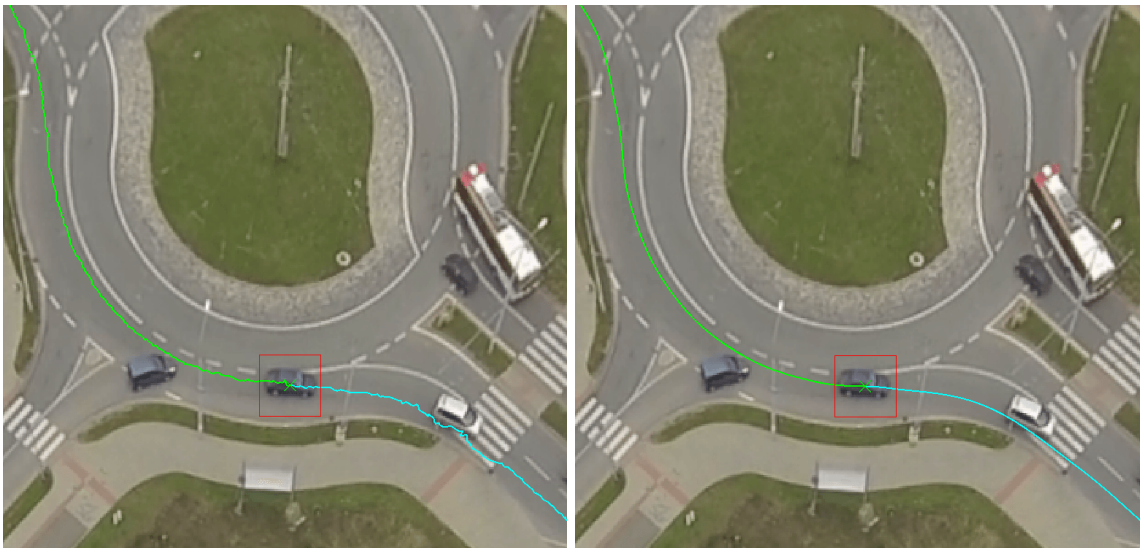
Generated trajectories suffer from spatial noise due to selected representation of estimated state of target t, when only best particle of particle filter is selected (see equation (5.13)). Therefore, it is necessary to perform smoothing of generated data in order to provide a set of trajectories that are suitable for estimation of vehicle speed an acceleration (which are first and second derivatives of the position). Firstly, the trajectory is smoothed out by applying rectangular smoothing window in time domain. However, for the given smoothing window, the resulting state is not represented as average of the states in the window; instead, the states in the smoothing window are overlaid by approximating B-Spline curve [58] in time domain. The resulting state is represented by a point of the generated curve defined for the given time moment.

Subsequently, to remove greater deviations, the trajectory is sampled in time domain to produce a sparse set of control points, which are overlaid by interpolating cubic spline [58]. The resulting curve defines the shape of the trajectory. Due to the non-linearity of cubic spline parameters in time domain, the position of the vehicle cannot be derived directly from cubic equations, while preserving continuity of vehicle velocity. To achieve this, for each control point of the curve, travelled distance of respective vehicle to that point is calculated.

The acquired series of travelled distances in time domain are interpolated using Monotone Piecewise Cubic Interpolation algorithm [59]. The result of the interpolation is monotone and continuous in its first derivation. Monotonicity of the interpolation guarantees

that the travelled distance of vehicle can only increase as time proceeds. The continuity of the first derivation of the travelled distance guarantees that the velocity of the vehicle is continuous, therefore conforming to real world: velocity of an object must be continuous in time domain, otherwise its acceleration would be undefined or infinite at the point of velocity discontinuity, both of which being impossible.

Let $d_t^{(i)}$ be the value of travelled distance of the vehicle at time moment i, calculated using the said interpolation. The position of the vehicle at the time moment i is defined as point of the the interpolating cubic spline, which distance from the beginning of the trajectory along the curve is equal to $d_t^{(i)}$. The trajectory composed of such produced series of points is smooth in spatial domain, and continuous in its first derivation — object velocity.



(a) Before Smoothing (b) After Smoothing

Figure 6.2: Example of smoothing applied to trajectory data.

6.3 Refiltration

After the interpolation of skipped frames and smoothing of generated trajectories has been applied, the shapes of vehicle trajectories changed, and therefore the vehicle crossings of entry gates and exit gates may have changed as well. Thus, it is necessary to reapply the required conditions defining a successfully tracked vehicle trajectory (see subsection 5.2.4) by refiltration. For each trajectory in updated trajectory database, we analyse its shape to obtain information whether its corresponding object t crossed both entry gate and exit gate in that order. If so, then the trajectory is considered as representing a successfully tracked object and is accepted. In all other cases, the trajectory is considered as unsuccessful and is therefore rejected.

Additionally, the whole database is also once again analysed for target overlaps. Each overlap sequence of two targets is inspected: if the overlap sequence last for at least a certain amount of time steps, the trajectory of one of the overlapping targets is rejected according to the equation (5.17) in subsection 5.2.4.

6.4 Summary

In this chapter, we described Data Post-processing phase of the proposed sytem. The motivation for and approach to interpolation of skipped frames, trajectory smoothing and trajectory database refiltration were presented.

Chapter 7

Implementation

In this chapter we will provide an overview of how the presented system was implemented. We will first specify the tools and libraries used for the implementation of the system. Then the outline of the system's architecture is presented together with the description of important classes. At the end of the chapter, a short description of all applications developed for this work is provided.

7.1 Implementation Tools

Due to the broad nature of the system's design (see section 2.3), we decided to use an object oriented approach to implement the system. C++ was selected as the most suitable programming language for such task. The advantage of C++ lies in its multi-purpose and multi-paradigm properties enabling development of extensive systems while still offering tools to implement high performance algorithms. Additionally, in the field of computer vision, most of the publicly available implementations of algorithms offer native C++ API.

In the implementation of the system the following third party libraries were used:

- **Qt**[1] — general data manipulation, implementation of basic GUI, input and output to files, XML parsing using Document Object Model, multi-threading.
- **OpenCV**[2] — image processing, object detection, geometry calculations, video sequence management.
- **OpenCVX**[3] — basic implementation of Bayesian bootstrap filter.
- **Eigen**[4] — geometry calculations, vector mathematics, CPU optimisations.
- **TCLAP**[5] — definition, parsing and accessing command line arguments.
- **MurMurHash**[6] — hashing video frames for accurate seeking in video sequences.
- **BSpline**[7] — approximating B-spline curve

[1] https://www.qt.io/
[2] http://opencv.org/
[3] https://github.com/sonots/opencvx
[4] http://eigen.tuxfamily.org/
[5] http://tclap.sourceforge.net/
[6] https://code.google.com/p/smhasher/
[7] http://www.eol.ucar.edu/homes/granger/bspline/doc/

The application implements only CPU optimisation and parallelism, using SSE instruction sets and Microsoft Concurrency Runtime[8] library.

The source files are managed under numerous Qt sub-projects for better modularity and easier management inside QtCreator IDE. The application was developed in a Windows 7 system environment for Microsoft Visual C++ compiler. However, all tools and libraries are multi-platform, with the exception of Microsoft Concurrency Runtime library, which can be switched off. The application is implemented as a command line utility. However, it provides a very basic visualisation of a current tracking state.

7.2 System Architecture

In total, the system consists of 107 classes, so only a brief description of the most important modules and classes is provided here. The application was designed to have hierarchical architecture with multiple data and algorithm encapsulations. This hierarchy is illustrated on graph shown in figure 7.1.

Generally, we can divide all classes into two groups: data classes and algorithm classes. A simplified overview of objects representing algorithms and data inside the application, and their communication is shown in figure 7.1.

7.2.1 Data Classes

Data classes are used to encapsulate data for storage and provide methods for simplified manipulation of encapsulated data. The following data classes have the most important role in the application:

- **RCrossroadMarkups** — represents traffic scene annotation data (see section 3.1). It provides methods for optimised traffic gate crossing event detection, querying of road surface presence at a given point and saving/loading data to/from XML structures.
- **RTrajectory** — represents estimated trajectory of a single vehicle. It contains a series of vehicle states in time domain. It provides methods for fast trajectory manipulation, estimation of speed and velocity.
- **RTrackingHistoryBase** — encapsulates information about vehicle movement in the traffic scene. It contains its trajectory, stored as *RTrajectory* object, its unique identification, and information about passed entry and exit gates.
- **RTrackingHistory** — derived from *RTrackingHistoryBase*, it encapsulated information about tracking of a single vehicle. With addition to *RTrackingHistoryBase*, it contains information about initial detection of vehicle and history of its appearance function (see subsection 5.2.3).
- **RTrackingLog** — encapsulates output of the whole system: generated tracking histories of vehicles in form of database of *RTrackingHistory* objects, selected input parameters of the system and database of all transformations between thr coordinate systems used in application (see section 3.4).

[8] https://msdn.microsoft.com/en-us/library/vstudio/dd504870.aspx

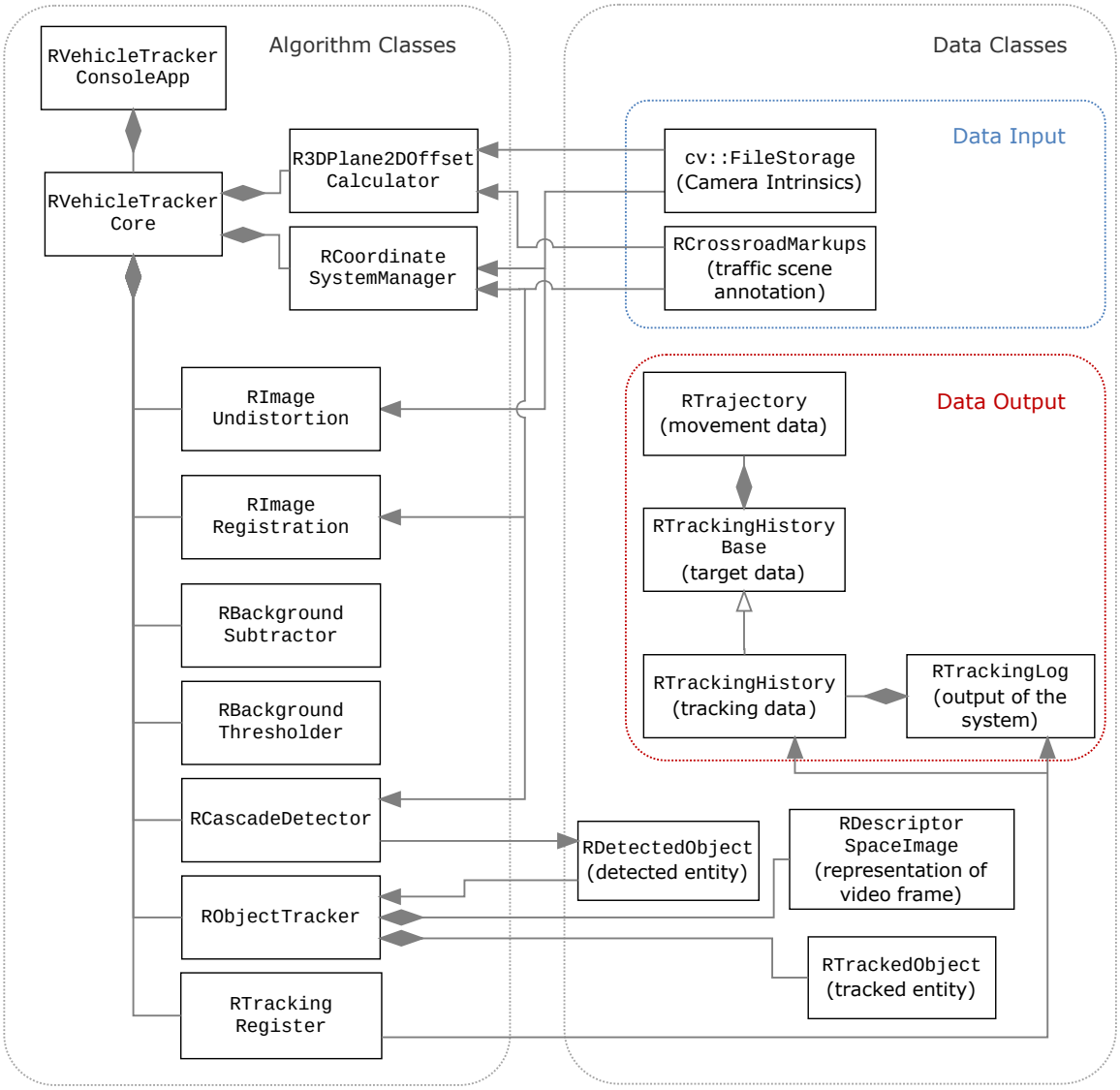


Figure 7.1: Hybrid graph capturing class composition and selected input and output data. Composition relationship is depicted by arrows with filled diamond heads. Inheritance relationship is depicted by the arrow with an empty triangular head. Data flow is represented by arrows with filled triangular heads. For the sake of clarity, only initialisation and output data flow is shown, with the addition of detected entities.

- **RDetectedObject** — encapsulates a single detection of vehicle detection algorithm. It contains information about the position of the detected vehicle in the geo-registered video frame.
- **RTrackedObject** — encapsulates information about tracked object in the traffic scene: its unique identification, descriptor template, information about current estimated state and particles of the particle filter. It provides methods to analyse its estimation and methods to execute steps of particle filter.
- **RDescriptorSpaceImage** — represents a geo-registered video frame in the descrip-

tor colour space, containing red, green, blue and edge map channels (see subsection 5.2.2). Edge map channels are calculated as the sum of absolute responses of Scharr operator in both x and y directions in the video frame. Its purpose is to speed up extraction of the descriptor templates for the targets and particles of Bayesian bootstrap filter.

7.2.2 Algorithm Classes

Algorithm classes encapsulate algorithms and their parameters. Each algorithm class provides at least a method for loading its parameters from theconfiguration file and execution of the algorithm. The following algorithm classes play an important role in the application:

- **RCoordinateSystemManager** — provides methods to manage various coordinate systems in the application (see section 3.4). It contains an algorithm to calculate working space coordinate system according to the traffic scene annotations and the set of corresponding points between reference frame and real world coordinate system.
- **R3DPlane2DOffsetCalculator** — encapsulates an implementation of *Offset Fix* algorithm as explained in section 3.5.
- **RImageUndistortion** — encapsulates algorithm for the image undistortion (see section 3.2).
- **RImageRegistration** — provides a method for finding correspondence between reference image and video frame, allowing its subsequent geo-registration (see section 3.3).
- **RBackgroundSubtractor** and **RBackgroundThresholder** — implement the algorithms of background subtraction and subsequent accumulation as described in subsection 4.2.1.
- **RCascadeDetector** — encapsulates the algorithm for vehicle detection. It is represent by two objects, one for weak vehicle classifier and other for strong vehicle classifier (see subsection 4.2.2). Upon each detection, it produces a list of *RDetectedObject*s.
- **RObjectTracker** — manages tracking of targets by maintaining a database of *RTrackedObject*s. It provides methods for estimation of targets' states in the video frame and calculation of the mask of currently tracked objects. It also implements the algorithm of detection-track association and tracks management as explained in chapter 5. To speed up the computation, it employs *RDescriptorSpaceImage* object constructed from the video frame.
- **RTrackingRegister** — manages generated tracks per each target tracked in the traffic scene. It contains a database of *RTrackingHistory* objects, and provides methods for additional filtering used in *Data Post-processing* phase (see chapter 6).
- **RVehicleTrackerCore** — this class is the most important class of the whole application. It manages every algorithm and computation that happens during the application execution and it is responsible for data communication between contained algorithm objects. It also manages data input and output of the results.

- **RVehicleTrackerConsoleApp** — provides console wrapper for *RVehicleTrackerCore* for console application. It parses the application arguments and manages lifetime of *RVehicleTrackerCore* object.

7.3 Usage

The application is designed to be run from command line using the following parameters:

- `-s <path>` — defines path to settings file. Settings file contains parameters of deployed algorithms neatly structured in configuration INI file. Parameters of each algorithm are stored in separate groups inside the file.
- `-r <path>` — defines path to a reference image of the traffic scene.
- `-a <path>` — defines path to traffic scene annotation file. The file is stored in XML structure and is produced by supplementing application *TrafficSceneAnnotator* (see section 7.4). On the startup of the application, the traffic scene annotation file is open and loaded using *RCrossroadMarkups* object.
- `-c <path>` — defines path to file containing camera annotation intrinsic parameters: camera matrix and camera distortion coefficients vector. The file has XML structure, and is parsed using *cv::FileStorage* object from OpenCV library.
- `-v <path>` — path to input aerial video sequence that is to be analysed. The supported containers are *AVI* and *MP4*. The supported codecs depend on computer system, ffmpeg module and the version of OpenCV library used for application compilation.
- `-b <number>` — index of the first frame to be analysed in the video sequence.
- `-e <number>` — index of the last frame to be analysed in the video sequence.
- `-o <path>` — path to an output file which the output will be written to. The output will be written as XML DOM representation of *RTrackingLog* object.
- `-p <path>` — path to a file, where the output video sequence will be written to. The output video sequence is represented as geo-registered sequence of input video frames with visualisation of tracked vehicles. This parameter is optional.

During the application execution, a visualisation of current progress is shown to the user. The processing can be paused by pressing key *'P'*. The analysis can be aborted by pressing key *'Q'*.

7.4 Supplementing Applications

During the work on this thesis, several additional applications had to be developed, for the purposes of assistance in creating data input and analysis of data output:

- **PositiveSampleCreator** and **NegativeSampleCreator** — both applications were used to produce positive and negative samples for training of the vehicle classifiers (see subsection 4.2.2). The applications have very simple graphic user interface, which provides means of extracting samples from video or imagery data.

- **TrafficSceneAnnotator** — an GUI application that enables a simple way of producing traffic scene annotation using reference image, camera intrinsic parameters and GIS data.
- **DetectionEvaluator** — a testbed application for evaluation of deployed detection algorithm. For more information, see section 8.2.
- **TrackingEvaluator** — a testbed application for evaluation of deployed tracking algorithm. For more information, see section 8.3.
- **SystemEvaluator** — application that analyses the output produced by the presented system and compares it with a hand annotations of the video sequences to evaluate the performance of the system as a whole.

7.5 Summary

In this chapter, we provided an overview of the implementation of the presented system. The implementation tools were described in the leading section of the chapter, followed by the description of system architecture and a brief description of usage of the resulting application. In the last section, an overview of supplementing applications was provided.

Chapter 8

Evaluation

This chapter describes evaluation of the presented system. According to the subject of evaluation, the evaluation of the system is divided into following three parts:

- **Evaluation of detection algorithm** — aims at evaluation of detection stage of the proposed system. The performance of detection using both weak and strong vehicle classifiers across various coordinate systems and with or without pre-filtering by background model is evaluated and compared.
- **Evaluation of tracking algorithm** — aims at evaluation of tracking stage per single tracking target. The accuracy and performance of tracking with and without aid of the detection stage is evaluated, as well as the tracking performance across different update rates of target descriptor template is analysed.
- **System evaluation** — evaluates the performance and accuracy of the proposed system as whole. For that purpose, the specialised metrics suited for evaluation of multi-target tracking are applied.

In the following sections of this chapter, firstly, the description of evaluation datasets is given. Then, for each of the aforementioned parts of the evaluation, its methodology, results of the evaluation and discussion of the results are presented.

8.1 Evaluation Datasets

The evaluation was carried out using hand annotated dataset derived from aerial video data provided by private companies *RCE systems s.r.o.,* Brno, Czech Republic and *Vertex Access*, Rotherham, United Kingdom. The dataset contained three sequences of aerial video data captured by an UAV:

1: Netroufalky — this sequence captures the traffic at the roundabout near Netroufalky construction site in Bohunice, Brno, Czech Republic. The UTM coordinates of the scene are following: E = 614 231.35, N = 5 448 430.28, zone 33U. The video sequence was recorded on 18th October 2013.

2: Olomouc — this sequence captures traffic at the roundabout junction of Hamerská road and Lipenská road near Olomouc, Czech Republic. The UTM coordinates of the scene are following: E = 667 204.60, N = 5 495 514.72, zone 33U. The video sequence was recorded on 24th October 2013.

3: **Sheffield** — this sequence captures traffic on the Sheffield road near Magna Science Adventure Center, Rotherham, United Kingdom. The UTM coordinates of the scene are following: E = 607 352.37 m, N = 5 920 000.12 m, zone 30U. The video sequence was recorded on 20th November 2014.

The video sequences Netroufalky and Olomouc were captured by an unmanned aerial vehicle — hexacopter MiroKopter MK Hexa XL. As a video capture device, the action camera GoPro Hero3 Black Edition was used. The camera was equipped with ultra-wide optical lenses which provide 133.6° diagonal field of view. The video was captured at resolution of 1920 px × 980 px at 29.97 Hz.

The video sequence Sheffield was captured by an unmanned aerial vehicle — octacopter AscTec Falcon 8. As a video capture device, the action camera GoPro Hero3+ Black Edition was used. The camera was equipped with medium-wide optical lenses which provide 107.1° diagonal field of view. The video was captured at resolution of 2704 px × 1524 px at 29.97 Hz.

During the course of data acquisition of all four sequences, the UAV was steadily flying at altitudes of approximately 150 m above the road surface beside the analysed traffic scene — therefore the scene is captured obliquely causing a perspective distortion of the scene affecting the quality of image data across the area of the traffic scene depending on view inclination to given point of the traffic scene. Selected video frames from evaluation sequences are shown in figure 8.1.

Each sequence is supplemented by a hand annotated database of car trajectories that correspond to the traffic captured at the scene. For each car that has passed through the scene during the sequence, its position and bounding rectangle is known for each moment of its presence in the region of interest of the scene. In total, all annotations contain trajectories of 446 vehicles with total travelled distance of 73.493 km and duration of 3 hours 21 minutes and 10 seconds. In the sequences Netroufalky and Olomouc, the Křovák projection was used as real world coordinate system. In the sequence Sheffield, Universal Transverse Mercator (UTM) projection was used as real world coordinate system.

For evaluation process, the ground truth trajectories are created by truncating hand annotated trajectories from the database, so they all lie inside the region of interest of the traffic scene. To address the inaccuracy of hand annotated data, the following metric is deployed to evaluate spatial error between the true (annotated) position of vehicle represented by vector $\mathbf{t}$ and the estimated position of the vehicle represented by vector $\mathbf{d}$, both in real world coordinate system:

$$Err\left(\mathbf{d},\mathbf{t}\right) = \max\left(\|\mathbf{d}-\mathbf{t}\| - 0.5\,\mathrm{m}, 0\,\mathrm{m}\right) \quad , \tag{8.1}$$

where $\max\left(a,b\right)$ returns the maximum of values a and b, and $\|c\|$ returns euclidean norm (i.e. distance from origin) of vector c. The insensitivity of spatial error at distances from 0.0 m to 0.5 m is incorporated because of subtle inaccuracies in provided annotations.

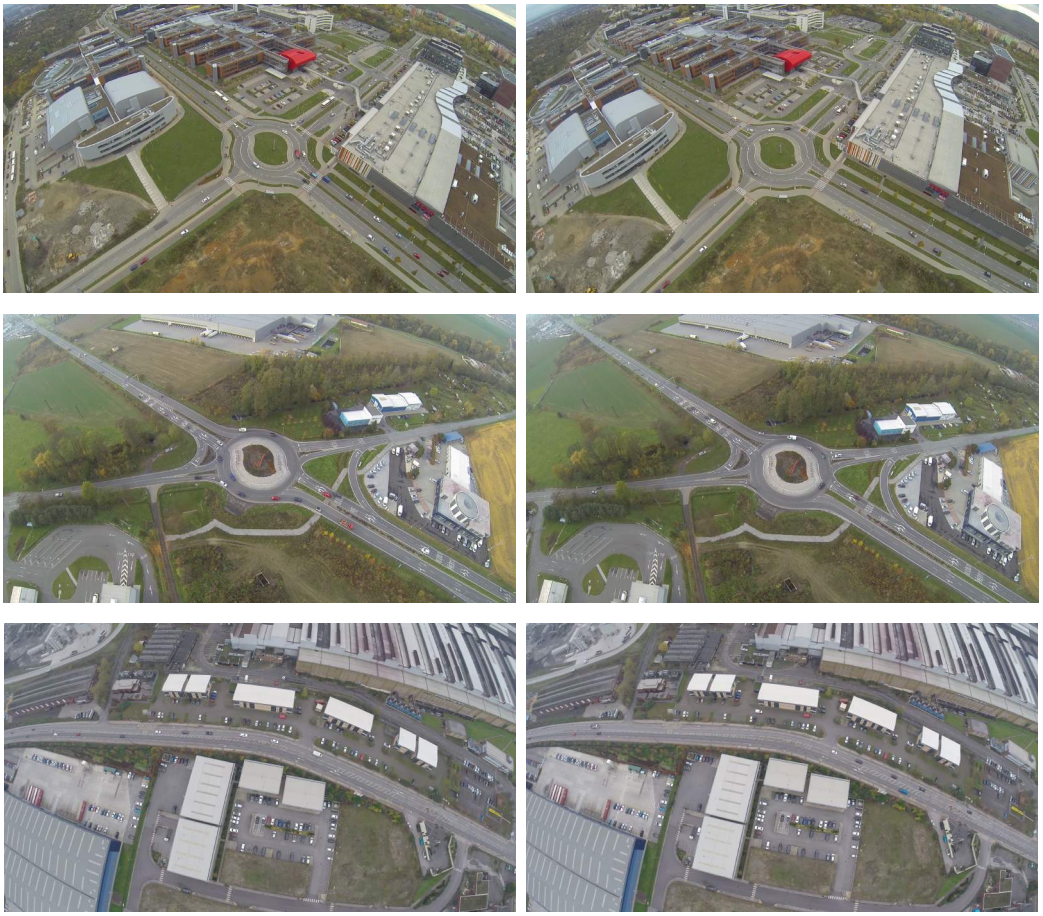


Figure 8.1: Selected frames from evaluation sequences. First row: Netroufalky video sequence. Second row: Olomouc video sequence. Third row: Sheffield video sequence.

8.2 Evaluation of Detection Algorithm

The section describes the evaluation of used vehicle detection algorithm. We aimed to compare performance of detector with both weak and strong vehicle classifiers. Also, an evaluation of the performance of the detectors with and without pre-filtering of detection candidates was carried out. To inspect, whether the transformations between different coordinate systems have effect on the performance of detectors, the evaluation was carried out in all three image coordinate systems: *raw space*, *reference space* and *working space coordinate system* (see section 3.4). Note that the training data for the classifiers used in the evaluated detection algorithm were extracted from different aerial image data.

8.2.1 Methodology

Test data was generated as follows: For each video sequence, 1000 video frames were selected uniformly across the whole sequence. In total, there were 19 547 true targets present in the selected video frames — 5687 in Netroufalky video sequence, 10 210 in Olomouc video sequence and 3651 in Sheffield video sequence. For each selected video frame, the detection of vehicles in the region of interest was carried out. If the detector produced multiple overlaying rectangles, they were fused together using partition algorithm as implemented

in [15], to form single rectangle. The results of detection were compared with the ground truth extracted from the hand annotated database of vehicle trajectories. The association between the set of generated detections and the set of true vehicle positions was done by finding the nearest true vehicle position for each generated detection and vice versa. Spatial error between the true vehicle position t and its corresponding detection position d is expressed as in equation (8.1), where vector $\mathbf{d}$ is given as centre of bounding box of detection d projected on road surface (see algorithm *Offset Fix* in section 3.5) and vector $\mathbf{t}$ represents the hand annotated position of centre of the vehicle on the road surface. If the value of $Err\,(d,t)$ exceeds threshold $dist_{max} = 3\,\mathrm{m}$, it is considered as miss and such pair is rejected.

For every evaluated configuration of detection algorithm the following data is recorded independently per each frame, and also accumulated for the whole sequence:

- *RMSE* — root of mean squared spatial errors of associated detection and ground truth pairs (see equation (8.1)).
- Number of valid detections VD. Valid detection is a detection that is associated with some ground truth.
- Number of false detections FD. False detection is a detection that is not associated with any ground truth.
- Number of missed truths MT. Missed truth is a ground truth that is not associated with any detection.
- Number of all ground truths TT, respective to the frame or set of all selected frames.
- Computational cost T_c. It is represented as a length of the elapsed time that passed during the execution of detection algorithm.

From the collected evaluation data, the following metrics were calculated for each evaluated configuration:

- P — precision of the detector. It represents the ratio of correctly detected objects to all detected objects. It is computed as follows:

$$P = \frac{VD}{VD + FD}. \tag{8.2}$$

- R – recall of the detector. It represents the ratio of correctly detected objects to all ground truths. It is computed as follows:

$$R = \frac{VD}{TT}. \tag{8.3}$$

For a perfect detection system with no missed truths and false detections, both precision and recall equal to 1. The presented system was developed to produce reliable data with higher emphasis on precision than recall.

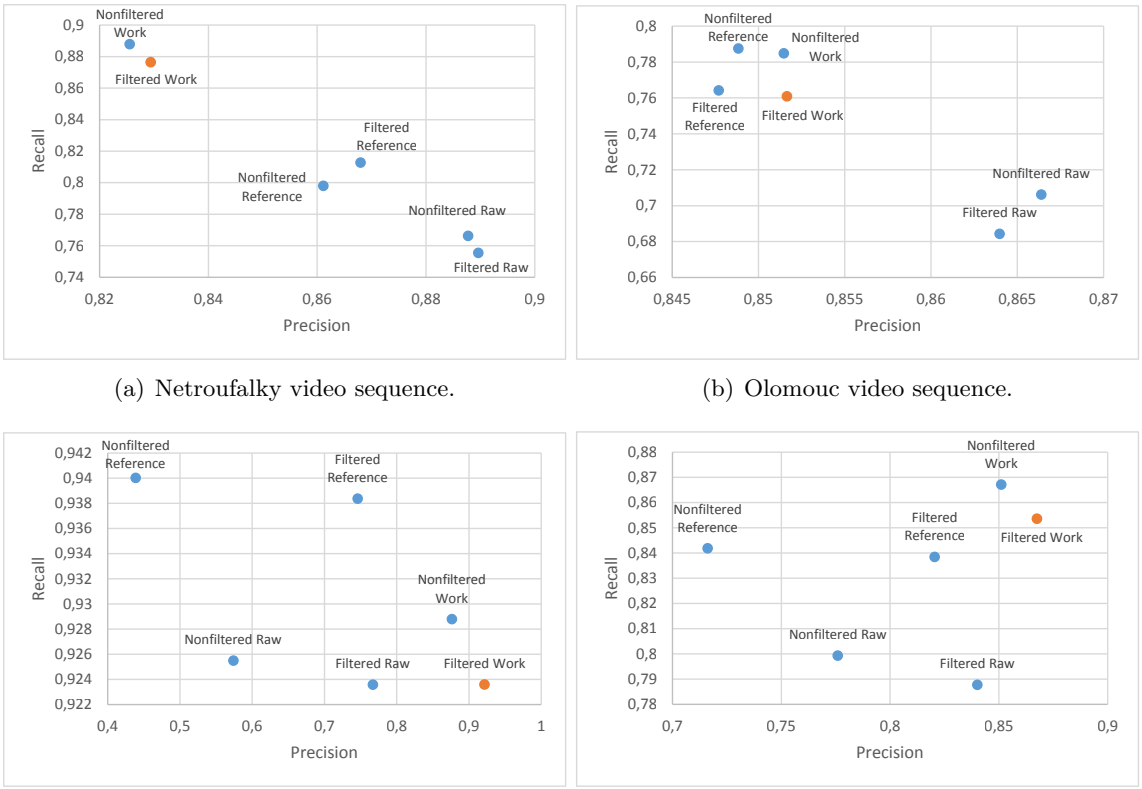
8.2.2 Results

In total, 36 configurations of the evaluated system were analysed, 12 per each evaluation sequence. To see the overview of the results for all evaluated configurations, please, refer to the Appendix, section B.1. In this section further, we will discuss the resulting values of precision P and recall R.

Charts illustrating precision and recall of vehicle detection using strong vehicle classifier for all three datasets are shown in figure 8.2. The configuration deployed in the proposed system is highlighted by orange colour. The results vary a lot between different video sequences. For video sequence Netroufalky (figure 8.2(a)), it is evident that precision and recall of the detection is dependent on image transformation. When a video frame is transformed into work space, detection yields better recall values and worse precision. On the other hand, detection in raw space yields better precision and worse recall. Similar performance is observed for video sequence Olomouc (figure 8.2(b)). In that case, the detection algorithm exhibits high precision and low recall for images in raw space. Contrary to results for Netroufalky video sequence, detection in reference space yields worse precision than detection in work space. Another interesting observation is, that detection algorithm with candidate filtering which is supposed to filter out false detection candidates and therefore reduce false detection rate (see subsection 4.2.1), has worse precision in both reference space and raw space. Precision in work space is slightly better only for configuration with candidate filtering. The greatest performance differences between the configurations of detection algorithm were observed in Sheffield video sequence (figure 8.2(c)) — the best precision with value of $P = 0.9216$ is achieved by deploying detection algorithm with candidate filtering on video frames in work space. The second best precision $P = 0.8762$ for this video sequence is observed for the same image space but without candidate filtering. This configuration yields better recall value, however the difference of recall values for these two configuration is very small — only 0.0052. It is notable, that in the Sheffield video sequence, candidate filtration has vast impact on precision values, and its impact on recall is comparatively small.

To summarise the results of detection algorithm with strong vehicle classifier, we provide also a chart of overall precision and recall aggregated across all three sequences by arithmetic mean. It is shown in figure 8.2(d). We can conclude, that however candidate filtering may worsen detection performance (evidently seen in results for video sequence Olomouc), it may also rapidly reduce false detections (seen in results for video sequence Sheffield). Interesting is also the fact, that even though the training dataset for vehicle classifiers was extracted from untransformed imagery, the detection does not perform best in untransformed video frames (in raw space).

The comparison of detection algorithm performance using strong vehicle classifier and weak vehicle classifier is shown in figure 8.3. The recall metric for weak classifier has higher values — therefore the number of missed truths is very low for such detectors. It is also noticeable that filtering of detection candidates greatly improves values of precision of weak classifier, while having only slight overall negative effect on recall.



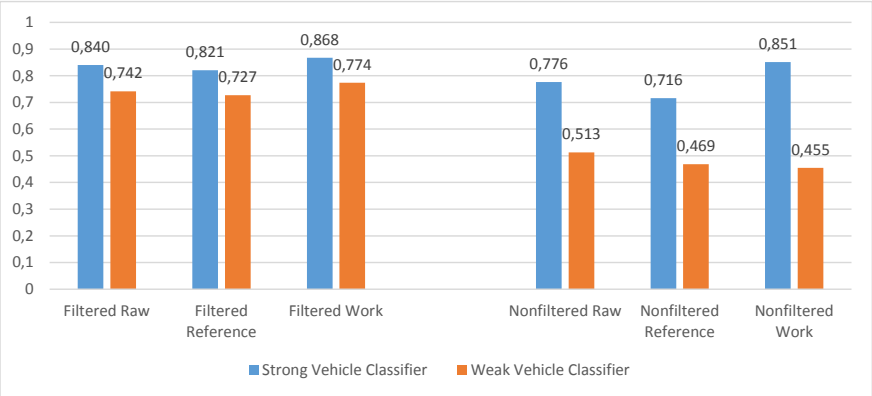
(a) Netroufalky video sequence.

(b) Olomouc video sequence.

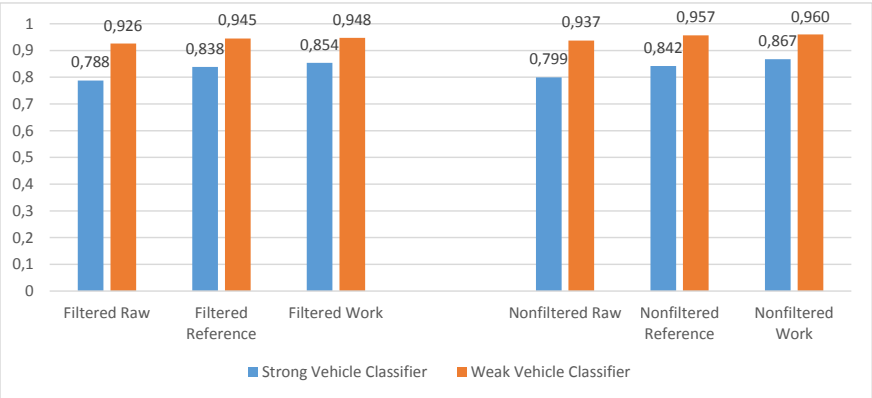
(c) Sheffield video sequence.

(d) Overall detection performance.

Figure 8.2: Performance of various configurations of detection algorithm in evaluation sequences. Each data point represents a performance result for a single configuration of detection algorithm. Position on horizontal axis represents precision of the detection. Position on vertical axis represents recall of the detection. Data labels identify algorithm configuration: *'Filtered'* means that filtration of detection candidates has been performed. *'Nonfiltered'* means that the filtration has not been performed. To indicate the image space in which the detection has been carried out, tags *'Raw'*,*'Reference'* and *'Work'* identify raw space, reference space and work space, respectively (see section 3.4). The configuration deployed in the proposed system is highlighted by orange colour. Note that axes of each chart are scaled independently.



(a) Aggregated precision.



(b) Aggregated recall.

Figure 8.3: Comparision of precision and recall of detection algorithm for various configurations using weak and strong vehicle classifires, aggregated across video sequences. Data labels identify algorithm configuration: *'Filtered'* means that filtration of detection candidates has been performed. *'Nonfiltered'* means that the filtration has not been performed. To indicate the image space in which the detection has been carried out, tags *'Raw'*,*'Reference'* and *'Work'* identify raw space, reference space and work space, respectively (see section 3.4).

8.3 Evaluation of Tracking Algorithm

Tracking evaluation aims to analyse performance of the proposed tracking algorithm deployed for a single target depending on update rate of the descriptor template (see subsection 5.2.2)). The update rates of 0.0 to 0.12 are evaluated with a step of 0.02. Also, the effect of tracking aid using detections yielded by the detection stage is analysed as well: The performance of tracking algorithm is evaluated with an aid of produced weak and strong detections, as described in chapter 5, i.e. the attraction factor for particles in Bayesian bootstrap filter is calculated as in equation (5.11) and target description template is updated only in case when the requirements stated in subsection 5.2.2 are met. For the comparison, the tracking algorithm is evaluated without the aid of produced weak and strong detections, i.e. the target descriptor template is updated at each tracking step and attraction factor function is $Att\,(p) = 1$.

For each video sequence, the remaining parameters of the tracking algorithm were fixed: the Bayesian bootstrap filter contained 512 particles per target, inter-frame random noise is set to $\sigma_{\mathbf{x}} = 0.016\,\mathrm{m}$ for particle position, $\sigma_{\mathbf{v}} = 0.013$ for particle velocity and $\sigma_s = 0.016$ for particle size. Initial multiplier of covariance is set to $m = 5$ and falloff rate is set to $f = 0.95$. For information on these parameters, see subsection 5.2.3.

8.3.1 Methodology

The evaluation of each configuration of tracking algorithm was executed on whole sequences by processing all video frames in order (in direction of time). Each frame is geo-registered and the tracking algorithm is deployed in *working space* (see section 3.4). The new tracking target is spawned at the beginning of each true trajectory that is longer than 100 frames. Initial position of the target and bounding rectangle is based on position and bounding rectangle of the true trajectory. For each target, independent tracking is deployed and evaluated. The tracking of the target is terminated when at least one of the following conditions is met:

- the corresponding true trajectory terminates at the given video frame.
- the track diverges from the corresponding true trajectory too much and it was tracked for more than 400 video frames in the sequence.

To assume that track diverged from its true trajectory, its spatial error as defined in equation (8.1) must be greater than 3.0 m for at least 30 video frames in sequence.

For every evaluated track the following values were recorded:

- Error vector $\mathbf{E}_{err}$. It is a vector of spatial errors between track and corresponding true trajectory as defined in equation (8.1), expressed for whole track. Its i-th element contains value of $Err\,(d, t)$ for the i-th frame from beginning of the track.
- Time point of first divergence τ_d. If the track has diverged from the respective true trajectory, its first point of divergence is recorded. The point of divergence is defined as the beginning of the first time period when its spatial error was greater than 3.0 metres for at least 30 frames of the video sequence. If the track has not diverged from its true trajectory, the time point of its first divergence is not recorded.

To provide the overall results for each analysed configuration of the tracking algorithm, the following data was gathered:

- Number of all divergent tracks N_d.
- Average error vector $\mathbf{E}_{AVG}$. It is a vector which is calculated as element-wise arithmetic mean of all error vectors $\mathbf{E}_{err}$ defined for given evaluation configuration. It represents a trend of average spatial error in time domain from the beginning of the track.
- Divergence vector DIV. It represents the progress of divergence of generated tracks in time domain. Value of each element of vector DIV at index i represents the number of tracks that have diverged at the i-th frame or before i-th video frame from the beginning of tracks.

8.3.2 Results

In total, 42 configurations of tracking algorithm were evaluated, 14 per each evaluated sequence. Number of targets used in evaluation is 433, of which 144 targets are present in sequence Netroufalky, 165 in sequence Olomouc and 124 in sequence Sheffield. The comprehensive overview of all results is included in Appendix, section B.2. To analyse the performance and caveats of the tracking algorithm further, we will present in this section only selected cases that we deem to be important and characteristic.

The most important aspect of the tracking algorithm is that it does not drift away from its target. To express, how well the different configurations of tracking algorithm can meet this goal, the numbers for all diverged targets for each evaluated configuration are shown in charts in figure 8.5.

It is shown, that the lowest number of diverged tracks for all three evaluated video sequences is produced by configuration of tracking algorithm with aid from detection stage and zero update rate of descriptor template. To further analyse the results of the evaluation, we only select the following configurations:

- Aided tracking with zero update rate — the best performing aided configuration.
- Aided tracking with update rate $\alpha = 0.08$ — selected for comparison as average performing aided tracking configuration with non-zero update rate.
- Non-aided tracking with update rate $\alpha = 0.04$ — the best performing non-aided configuration in video sequences Netroufalky and Olomouc.
- Non-aided tracking with update rate $\alpha = 0.12$ — the best performing non-aided configuration in Sheffield video sequence.

The graphs expressing values of divergence vectors DIV for selected configurations of tracking algorithm are shown in figure 8.6. It is evident that the configuration of aided tracking with zero update rate diverges the least. Only in Sheffield video sequence, non-aided tracking with update rate $\alpha = 0.12$ is on par. However, this configuration leads to considerably worse results in the other two sequences. In sequence Sheffield, a relatively high number of tracks diverged at the very beginning of tracks. We assume that this is due to the high speed of vehicles captured in the sequence, which makes the particles of the particle filter deployed in the tracking algorithm unable to adapt they velocities quickly enough at the beginning of tracking. Also, generally, configurations of aided tracking with non-zero update rate of descriptor template show poor performance in almost every aspect. We expect that this is caused by blurring of descriptor template at the update steps due to

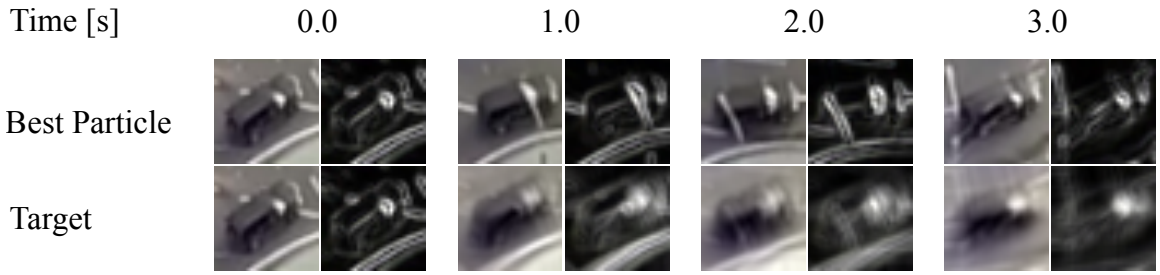
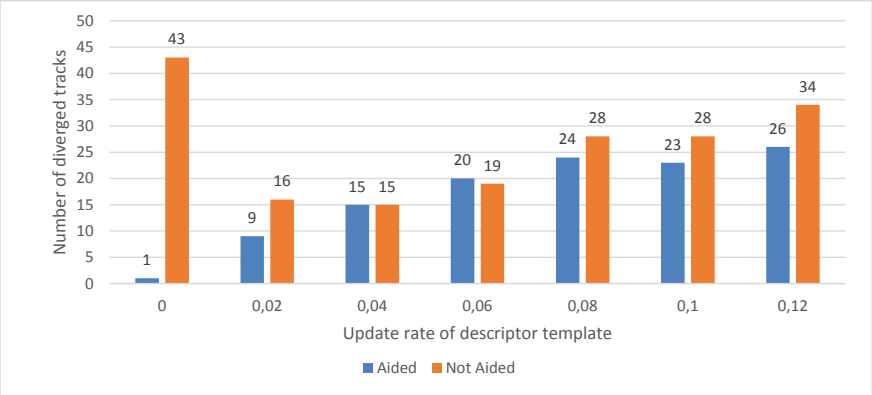


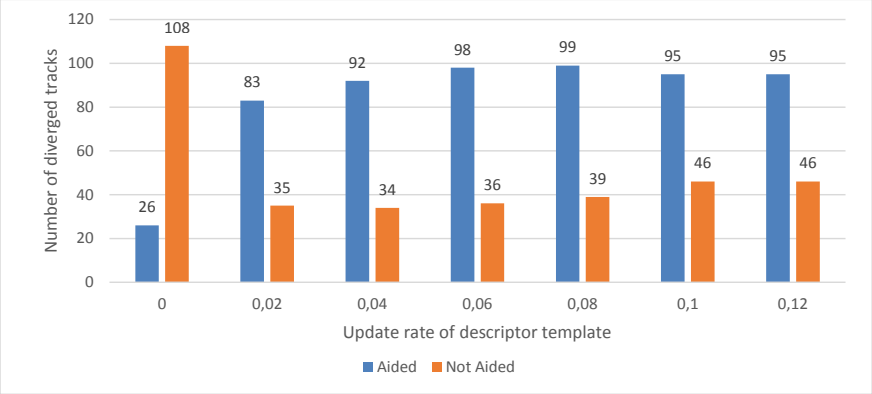
Figure 8.4: Illustration of descriptor template degradation in configuration of aided tracking algorithm with update rate $\alpha = 0.08$. This figure visualises descriptor templates (RGB and Edge channels) of a target in selected moments from beginning of the tracking. *Best Particle* row provides visualisation of descriptor template extracted from corresponding video frame according to state of the best weighted particle (see subsection 5.2.3). *Target* row provides visualisation of descriptor template for target representation, which was gradualy updated. It is clearly noticeable that details in the target representation are more degraded and distorted in later phase of the tracking.

non-perfect align of update descriptors (i.e. the detection does not contain vehicle exactly in its centre), which causes degradation of details in the resulting descriptor templates (see figure 8.4). Due to this, the appearance similarity function (see equation (5.10)) may yield wrong results.

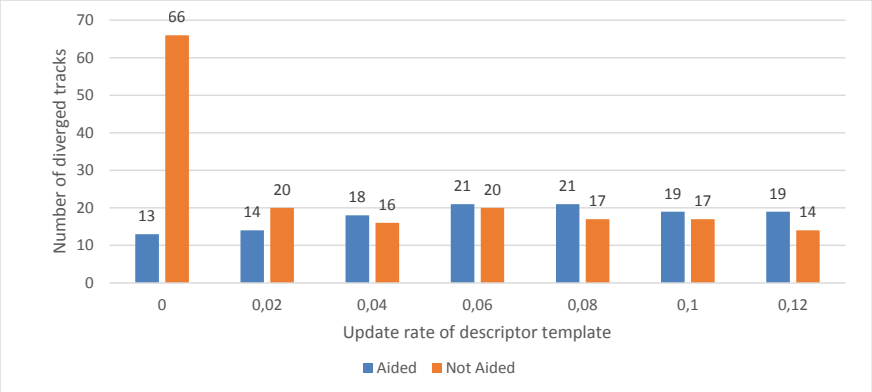
The graphs shown in figure 8.7 express the progress of average spatial error for selected configurations of tracking algorithm for first 400 frames of tracks. For sequences Netroufalky and Olomouc, the configuration of aided tracking algorithm with zero update rate yields the best results. For sequence Sheffield this is not true. We assume that this is caused by the high false detection rate of weak vehicle classifier. These false detections may induce deceptive tracking attractors and in turn lead to diverged tracking. In this case, the configuration of non-aided tracking algorithm with update rate of $\alpha = 0.12$ leads to overall better results. However, the very same configuration is also observed to produce much worse results for video sequences Netroufalky and Olomouc. It may be caused by usual movement motifs of vehicles in the scenes: while in both Netroufalky and Olomouc sequences, tracked vehicles execute series of turns and changes of speed due to roundabout, in Sheffield sequence the usual vehicle trajectory is mostly straight with one slight turn and without major changes of speed. Also, the density of traffic differs much among the sequences — in Sheffield, the density is considerably lower, and therefore, the probability of target swap is lower as well. Additionally, Olomouc traffic scene contains multi-lane roads where vehicles move beside each other. In this case, the probability of target swap and misleading by deceptive tracking attractor may be higher.



(a) Netroufalky, using 144 true targets.

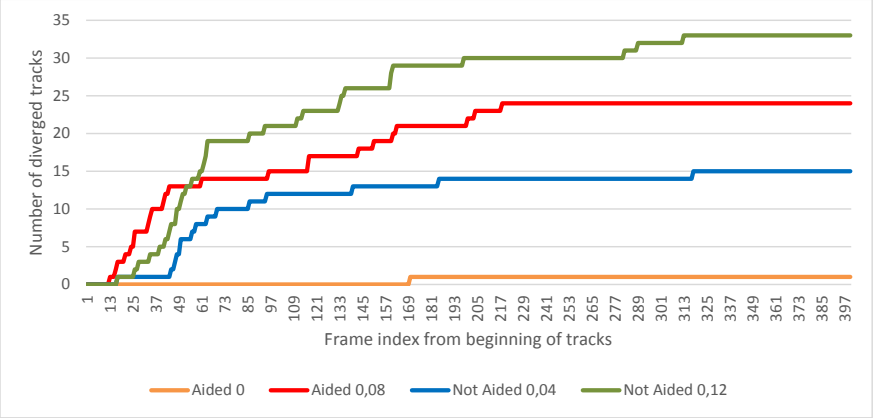


(b) Olomouc, using 165 targets.

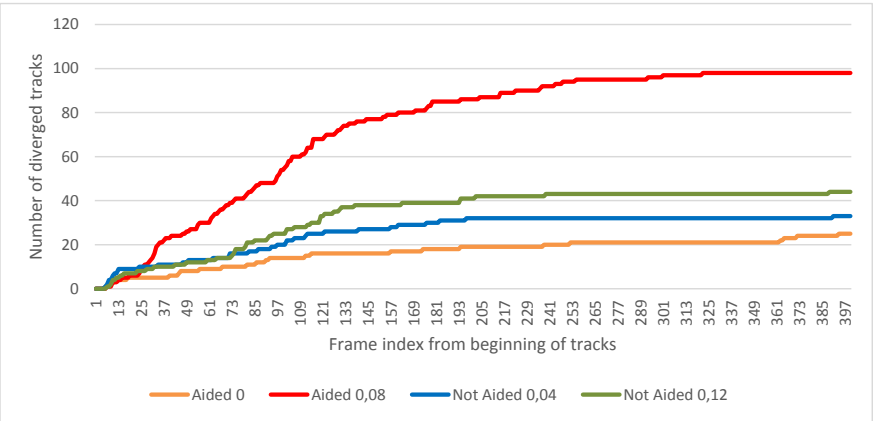


(c) Sheffield, using 124 targets.

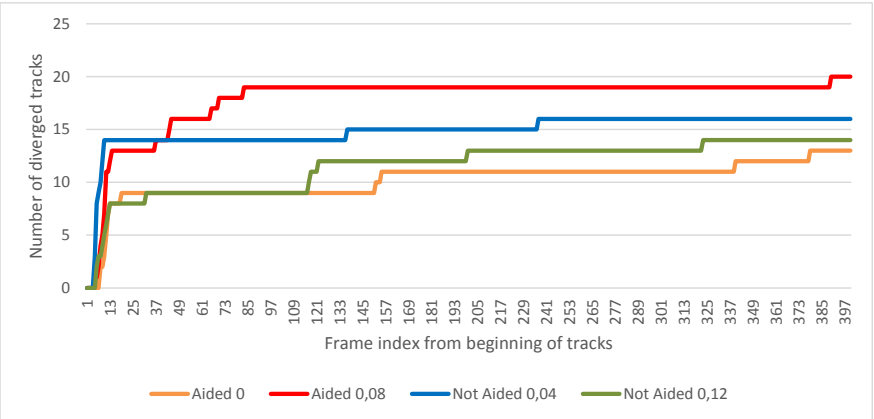
Figure 8.5: Number of all diverged targets per each evaluated configuration of tracking algorithm, depending on update rate of descriptor template and incorporation of aid from detection stage. Note that configuration with aid from detection stage and zero update rate yields the best results in all three evaluated video sequences.



(a) Netroufalky, using 144 true targets.

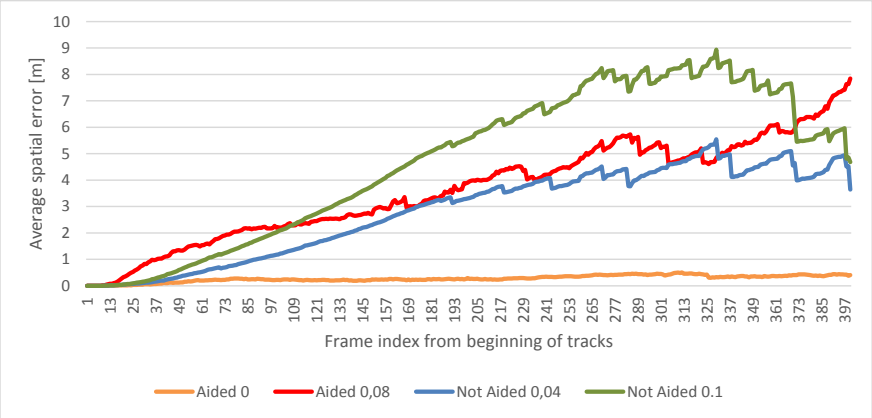


(b) Olomouc, using 165 targets.

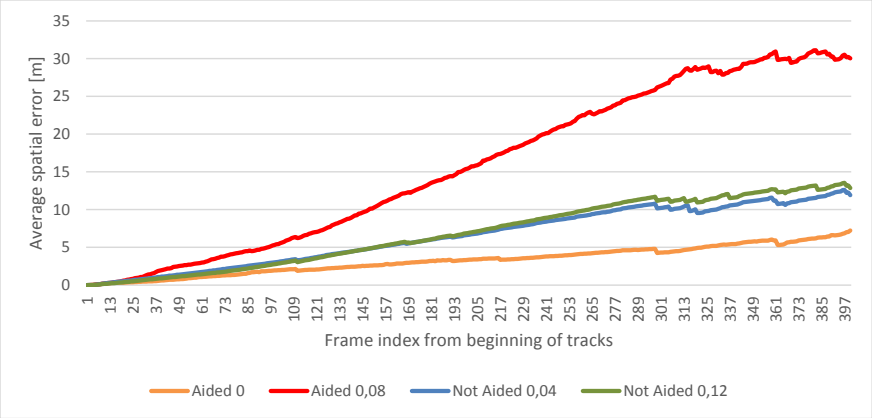


(c) Sheffield, using 124 targets.

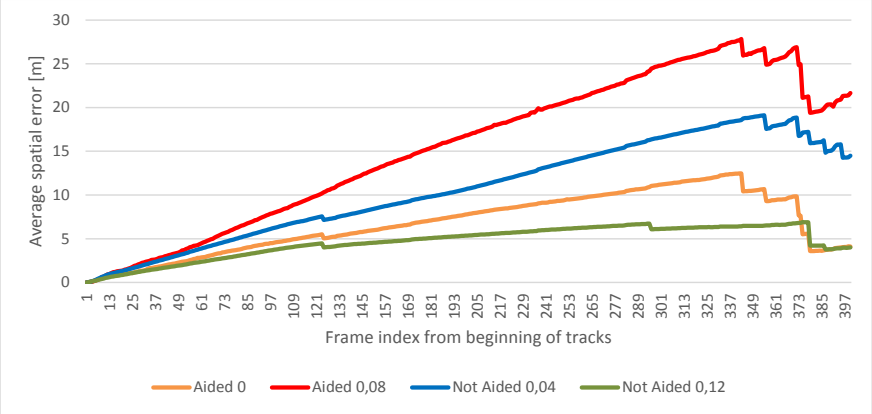
Figure 8.6: Graphs expressing values of divergence vectors per each evaluated sequence. The value of each element of divergend vector represent number of tracks that have diverged at or before the given frame relative to the beginning of tracks. Note that vertical axis is scaled independently for each graph. Also note that graphs show only the first 400 frames of the tracks, and therefore their last value may differ from values at charts in figure 8.5.



(a) Netroufalky video sequence.



(b) Olomouc video sequence.



(c) Sheffield video sequence.

Figure 8.7: Progress of average spatial error for selected configurations of tracking algorithm. The number after labels *Aided*/*Not Aided* in the legend of each graph represents update rate of target descriptor template. The sudden declines in average spatial error happen due to termination of tracking for targets which true trajectory ends at the given frame. It is noticeable in cases when the terminated tracking diverged too much from its true trajectory.

8.4 System Evaluation

The system evaluation aims to analyse quality of the generated output by the whole presented system. As shown in the previous sections, both detection and tracking algorithms which are deployed in the presented system produce results that are imperfect. The vehicle detection algorithm using strong vehicle classifier has average precision of approx. 0.87 and recall of approx. 0.85, which on average means that for each 20 produced detections, around 3 of them are false and 3 true targets are missed. The vehicle tracking algorithm is shown to have difficulties with fast moving targets, complex scenes and vehicle movements.

To evaluate the impact of the said shortcomings of both detection and tracking algorithms, we run the presented system on all video sequences from the evaluation dataset (see section 8.1). These sequences contained in total 423 ground truth trajectories that conformed the standard driver behaviour (see section 2.1). The detection of vehicles was carried out in working space with filtering of detection candidates, as explained in chapter 4. For the task of vehicle tracking, the configuration of tracking algorithm with zero update rate of descriptor template ($\alpha = 0.0$, see subsection 5.2.2) and aid by detection algorithm was selected, as it performs well on all evaluated sequences in tracking evaluation (see section 8.3).

8.4.1 Methodology

The evaluation process was the same for every video sequence. The reference image was extracted from the video sequence, with an aim to select the frame which contains minimum of moving objects. It was further used to produce the traffic scene annotation (see section 3.1). Then, the background model of the scene (see section 4.2.1) was initialised by the reference frame and further refined using video frames from the first 30 seconds of the video sequence. Subsequently, the system analysed the whole video sequence as described in this thesis. The output of the system was compared against the provided database of ground truth trajectories.

To provide evaluation results that are comparable with other systems, we used CLEAR Multi-Object Tracking metrics which were presented by Bernardin and Stiefelhagen [60]. They propose specialised metrics especially suitable for multi-object tracking evaluation. The main principle is based on finding continuous sequences of hypothesis–object correspondences (hypothesis is an estimated trajectory generated by the evaluated system, and an object represents the true trajectory of objects extracted from ground truth annotations). The set of hypothesis–object correspondences is calculated at each frame of the evaluation sequence, based on set of correspondences from previous step and spatial error. If the spatial error between the corresponding hypothesis and object is greater that given threshold $d_{max} = 3.0\,\mathrm{m}$, the correspondence is deemed invalid. To address the inaccuracy of hand annotated data, we calculate the spatial error between true object and estimated hypothesis as in equation (8.1), which results in ignoring any spatial differences under 0.5 m. Additionally, only the parts of the trajectories that lie inside the region of interest defined in traffic scene annotation are used for evaluation. The values of the following metrics from [60] are calculated:

- *MOTP* — multiple object tracking precision:

$$MOTP = \frac{\sum_{i,t} d_t^i}{\sum_t c_t} \quad , \tag{8.4}$$

where d_t^i is spatial error of i-th correspondence defined at video frame t, and c_t is number of all correspondences at the frame t. It is the total spatial error of estimated positions for all correspondences averaged by total number of correspondences. It represents the ability of system to estimate precise object positions, independent from its ability of object detection and keeping consistent trajectories.

- $\overline{m}$ — ratio of misses:

$$\overline{m} = \frac{\sum_t m_t}{\sum_t g_t} \quad , \tag{8.5}$$

where m_t is number of missed true targets in frame t and g_t is number of all true targets in frame t.

- $\overline{fp}$ — ratio of false positives:

$$\overline{fp} = \frac{\sum_t fp_t}{\sum_t g_t} \quad , \tag{8.6}$$

where fp_t is number of false positives at frame t.

- $\overline{mme}$ — ratio of mismatches (or also target swaps):

$$\overline{mme} = \frac{\sum_t mme_t}{\sum_t g_t} \quad , \tag{8.7}$$

where mme_t is number of detected mismatches (inconsistencies of hypothesis–object correspondences in temporal domain, also understand as target swaps) at frame t.

- $MOTA$ — multiple object tracking accuracy:

$$MOTA = 1 - \left(\overline{m} + \overline{fp} + \overline{mme}\right). \tag{8.8}$$

For metrics $MOTP$, $\overline{m}$, $\overline{fp}$ and $\overline{mme}$, the lower values are better (zero for perfect solution). For metric $MOTA$ the higher value is better ($MOTA = 1.0$ for perfect solution).

Additionally, to analyse performance of the presented system in aspect of its ability to produce reliable output — consistent trajectories of all vehicles present in the scene — we calculated following data per each evaluated sequence:

- VT — valid trajectories. It represents the number of estimated trajectories that have been consistently assigned to the same true trajectory during the whole movement of the respective vehicle trough the region of interest.

- IT — invalid trajectories. It represents the number of all trajectories that have not been consistently assigned to the same true trajectory during the whole movement of the respective vehicle through the region of interest.

- MT — missed truths. It represents number of true trajectories that has not been consistently mapped to the same estimated trajectory during the whole movement of the respective vehicle trough the region of interest.

- P — precision of the evaluated system. It is calculated as follows:

$$P = \frac{VT}{VT + IT}. \tag{8.9}$$

- R — recall of the evaluated system. It is calculated as follows:

$$R = \frac{VD}{TT} \quad , \tag{8.10}$$

  where TT is number of all true trajectories.

8.4.2 Results

The evaluation was carried out for each sequence separately. The results of the evaluation are shown in table 8.1, and will be discussed hereafter. The aggregate results of respective metrics were calculated by arithmetic mean of the metrics.

Table 8.1: Results of system evaluation.

Sequence	**Netroufalky**	**Olomouc**	**Sheffield**	**Aggregate**
$MOTP$	0.368	0.378	0.359	0.368
$\overline{m}$	0.083	0.227	0.047	0.119
$\overline{fp}$ ($\times 10^{-4}$)	2.223	0.518	4.418	2.386
$\overline{mme}$ ($\times 10^{-5}$)	2.167	6.647	4.370	4.395
$MOTA$	0.894	0.768	0.909	0.857
Precision P	0.931	0.921	0.917	0.923
Recall R	0.899	0.707	0.909	0.839
True trajectories TT	149	164	110	–
Valid trajectories VT	134	116	100	–
Invalid trajectories IT	10	10	9	–
Missed truths MT	15	48	10	–

Firstly, we will discuss CLEAR MOT metrics, that are aimed to evaluate tracking performance of the system. The values of multiple object tracking precision $MOTP$, which indicates the ability of the evaluated system to estimate correct target position, are very similar for all three evaluation sequences, with an average of $MOTP = 0.368$. This means, that on average, the estimated spatial error between true and estimated position of vehicle was around 0.368 m. We need to point out that the spatial error is calculated according to equation (8.1), which ignores differences under 0.5 m. Therefore, the actual average spatial error of the output of the system is somewhere in the interval $[0.368\,\text{m}, 0.868\,\text{m}]$.

The ratio of misses $\overline{m}$ is very close to zero in sequences Netroufalky and Sheffield. However, this is not true for Olomouc sequence, where $\overline{m} = 0.227$, which means that more than 22 % of true trajectory states were not estimated correctly at all. This may be addressed to high complexity of traffic scene and high density of traffic in the said sequence. Therefore, the vehicle movement is more complex and targets may overlap, which is more difficult to grasp by the deployed tracking algorithm. This may lead to undesirable target swaps, tracking terminations or generation of trajectories that does not conform the expected standard driver behaviour (see section 2.1).

On the other hand, the ratio of false positives $\overline{fp}$ is lowest for Olomouc sequence. Nonetheless, we need to point out that both ratios of false positives $\overline{fp}$ and mismatches $\overline{mme}$ are very low for all evaluated sequences. Therefore, the system is less prone to production of false trajectories, or trajectories that are invalid due to target swap. This is a very important observation, as one of the requirements on the presented system was that it will not produce invalid trajectories that may badly affect the subsequent traffic analysis.

The final CLEAR MOT metrics, multiple object accuracy *MOTA*, which summarizes metrics $\overline{m}$, $\overline{fp}$ and $\overline{mme}$, is lowest (the worst) for Olomouc sequence. This is due to the high ratio of misses $\overline{m}$ which we discussed in previous paragraphs.

In terms of output precision and recall, the system performed well on the sequences Netroufalky and Sheffield, achieving both precision P and recall R very close to or above 0.9 mark. This roughly means that the system was able to produce only one false trajectory and missed only one truth per each ten estimated trajectories. The system output for sequence Olomouc does have similar precision, $P = 0.921$, however the recall is much lower, at value $R = 0.707$. Therefore, the system missed on average 3 out of 10 true trajectories in Olomouc video sequence. This is caused by high ratio of misses $\overline{m}$ in the said sequence. To put things in perspective: ratio of misses $\overline{m} = 0.227$ caused that 29.3 % of all true trajectories were not estimated correctly — mostly missed.

On average, it was observed that the system produces output with precision $P = 0.923$ and recall $R = 0.839$. We can say that system has relatively good ability to produce only well formed trajectories which correspond to reality. This is however for price of reduced ability to extract trajectories of all cars in the scene. Nevertheless, in our case the precision is more important, as stated in the beginning of the section.

8.5 Summary

This chapter was aimed at the evaluation of the presented system. Firstly, we have provided description of the evaluation dataset — video sequences and their annotations. Then the separate evaluations of detection and tracking algorithms were described and their respective results were discussed. Finally, the output of the whole proposed system was evaluated and the results were discussed as well.

Chapter 9

Conclusion

This thesis presented the design, architecture and evaluation of a prototype system for traffic monitoring from aerial video data, aiming to solve the problem of extraction of vehicle trajectories from video sequences captured by unmanned aerial vehicles.

Firstly, we provided a brief description of available monitoring technologies of road transportation systems. In that juxtaposition we outlined the motivation and possible feasibility of traffic monitoring using UAVs, as well as possible shortcomings and sub-problems of such approach. Subsequently, we proposed a design of a system, that would provide a solutions for the said sub-problems and form a compact application for vehicle trajectory extraction from aerial video data.

To show that the proposed system may offer a reasonable output data quality for subsequent traffic analysis, we implemented a prototype of the system and executed a set of evaluations. The evaluation was executed on three exhaustively hand annotated video sequences containing trajectories of 446 vehicles with total travelled distance of more than 73 km with duration of 3 hours 21 minutes and 10 seconds.

The evaluation was divided into three parts. Firstly, we evaluated the system's ability to detect vehicles in the aerial imagery. The second evaluation aimed at analysis of deployed tracking algorithm and its ability to track single object in aerial video sequences. The third, final, evaluation analysed the output of the proposed system as a whole.

The results of the evaluation indicate that even despite the inaccurate results of vehicle detection and tracking algorithms, the system as whole is robust and is not prone to produce invalid estimations of trajectories. Overall results of evaluation show that the system's output contains on average 92.3 % valid trajectories, which mean that less than 1 invalid trajectory per each 10 estimated trajectories is produced by the system. This comes, however, at the price of slightly lower recall of the system — on average the system correctly estimates trajectories of only 83.9 % of all present vehicles in the scene.

These results, even though indicating a weakness of the proposed solutions, can nonetheless be still considered as very competitive, as there are very few alternative solutions which can offer such comprehensive traffic data. The viability of the proposed system is emphasised by fact that the prototype is already being used at the *Institute of Road Structures* under *Faculty of Civil Engineering, Brno University of Technology* to aid the process of design and analysis of new class of traffic intersections and roundabouts. Additionally, the system is also deployed in research of transportation systems at *Department of Civil, Environmental, Aerospace, Materials Engineering (DICAM), University of Palermo.*

A part of this work was published in a paper, which was produced as joint research with *Institute of Road Structures, Faculty of Civil Engineering, Brno University of Technology.*

The paper was peer-reviewed and accepted for the joint ISPRS conference PIA15/HRIGI15, that was held in Munich, 2015. The presented system was also described in a paper published at a student conference Excel@FIT 2015, where it was awarded first prize in the category of *Market Potential* and second prize in the category of *Public Benefit*. Both papers are provided in Appendix, section C and D respectively.

9.1 Further Research

Further work on the presented system is planned. Both *Institute of Road Structures* under *Faculty of Civil Engineering, Brno University of Technology* and *DICAM* under *University of Palermo* agreed to cooperate on research and development of the system with the goal of improving its performance and quality of generated output to enable its use across the whole spectrum of various applications in transporting system analysis and management.

For that purpose, the performance and accuracy of both vehicle detection and vehicle tracking algorithms have to be improved. In the case of detection algorithm, the relatively novel approach of deploying deep learning algorithms [61, 62] to the task of object detection is being considered. In the case of tracking algorithms, the approaches based on hierarchical data association optimisation (as explained in subsection 5.1.2) are being considered.

Additionally, further development of the presented system needs to be conduct to unlock its ability to function in heavy cluttered environment, complex non-planar traffic scenes with multi-level intersections and deeper traffic scene understanding. Also, the algorithm for automatic fusion of video data from multiple sources might be considered for cases when the traffic scene is captured by multiple flying UAVs. Optimisation of algorithm for real-time performance is also one of the goals for the next few years.

Subsequently, the algorithms for processing of generated data need to be developed as well — from estimation of traffic congestion, through detection of non-standard and hazardous situations, to navigation assistance. According to the responses that we already received from the researchers in the area of transportation systems, we expect that the collection of vehicle trajectory data using low-cost unmanned aerial vehicles can have positive impact on many aspects of our lives and therefore there are plentiful directions in which the proposed system can be improved.

Bibliography

[1] Cambridge Systematics, Inc.: *Travel Time Data Collection.* January 2012. White Paper.

[2] Leduc, G.: *Road traffic data: Collection methods and applications.* Luxembourg: Institute for Prospective Technological Studies, 2008. 55 p. Technical Note. Working Papers on Energy, Transport and Climate Change. JRC 47967.

[3] National Cooperative Highway Research Program: *Cost-effective performance measures for travel time delay, variation, and reliability.* Washington: Transportation Research Board, 2008. Report. ISSN 0077-5614. Available at: <http://www.camsys.com/pubs/WhitePaper_OD_TTData_Collection.pdf>. ISBN 978-0-309-11741-8.

[4] Bessler, S. a Paulin, T.: *Literature study on the state of the art of probe data systems in Europe.* Vienna: FTW Telecommunications Research Center, 2013. Available at: <http://www.fot-net.eu/download/fcd-report_final.pdf>.

[5] Chuvieco, E. a Huete, A.: *Fundamentals of satellite remote sensing.* Boca Raton, FL: CRC Press Inc., December 2009. ISBN 978-0-41531-084-0.

[6] Larsen, S. Ø., Koren, H. a Solberg, R.: Traffic monitoring using very high resolution satellite imagery. *Photogrammetric Engineering & Remote Sensing.* 2009, Vol. 75, Num. 7. p. 859–869.

[7] Guerrero Gomez Olmedo, R., Lopez Sastre, R. J., Maldonado Bascon, S. et al.: Vehicle Tracking by Simultaneous Detection and Viewpoint Estimation. In *International Work-conference on the Interplay between Natural and Artificial Computation 2013.* 2013. p. 306–316. Part II, LNCS 7931.

[8] Geoeye (2005): OrbView-3 scene. In *OrbView_U_0009016_00444260_2, Bellanwila-Attidiya Sanctuary, OrbView BASIC Enhanced.* Dulles, Virginia: GeoEye, May 2005.

[9] Litman, T. A.: *Smart Congestion Relief: Comprehensive Analysis Of Traffic Congestion Costs and Congestion Reduction Benefits.* Victoria, Canada: Victoria Transport Policy Institute, January 2011. Available at: <http://www.vtpi.org/cong_relief.pdf>.

[10] Grafarend, E. W. a Krumm, F. W.: *Lecture Notes Map Projections.* 2011. Institute of Geodesy, Stuttgart University. Rev 2.51. Available at: <http://www.uni-stuttgart.de/gi/geoengine/mappro/>.

[11] Wikimedia Commons: *Universal Transverse Mercator coordinate system: The UTM grid.* [online]. 2007. File: `Utm-zones.jpg`. Accessed: May 3rd 2015. Available at: <http://en.wikipedia.org/wiki/File:Utm-zones.jpg>.

[12] Bradski, G. a Kaehler, A.: *Learning OpenCV: computer vision with the OpenCV library.* Sebastopol, CA: O'Reilly Media, Incorporation, September 2008. 370-404 p. O'Reilly Series. ISBN 9780596516130.

[13] Beauchemin, S. S. a Bajcsy, R.: Modelling and Removing Radial and Tangential Distortions in Spherical Lenses. In *Multi-Image Analysis.* Berlin: Springer Berlin Heidelberg, 2001. p. 1–21. Lecture Notes in Computer Science, vol. 2032. Available at: <http://dx.doi.org/10.1007/3-540-45134-X_1>. ISBN 978-3-540-42122-1.

[14] Hongzhi, W., Meijing, L. a Liwei, Z.: The distortion correction of large view wide-angle lens for image mosaic based on OpenCV. In *International Conference on Mechatronic Science, Electric Engineering and Computer.* August 2011. p. 1074–1077.

[15] Bradski, G.: The OpenCV Library. *Dr. Dobb's Journal of Software Tools.* November 2000.

[16] MATLAB: *Version 7.10.0 (R2010a).* Natick, Massachusetts: The MathWorks Inc., 2010.

[17] Rublee, E., Rabaud, V., Konolige, K. et al.: ORB: An Efficient Alternative to SIFT or SURF. In *International Conference on Computer Vision.* Barcelona: Willow Garage, 2011.

[18] Fischler, M. A. a Bolles, R. C.: Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communication ACM.* June 1981, Vol. 24, Num. 6. p. 381–395. Available at: <http://doi.acm.org/10.1145/358669.358692>. ISSN 0001-0782.

[19] Malis, E. a Vargas, M.: *Deeper understanding of the homography decomposition for vision-based control.* 2007. 90 p. Research Report, RR-6303. Available at: <https://hal.inria.fr/inria-00174036v3>.

[20] Nguyen, T. T., Grabner, H., Gruber, B. et al.: On-line boosting for car detection from aerial images. In *IEEE International Conference on Reasearch, Innovation and Vision for the Future.* 2007. p. 87–95.

[21] Moon, H., Chellapa, R. a Rosenfeld, A.: Performance analysis of a simple vehicle detection algorithm. *Image and Vision Computing.* January 2002, Vol. 20. p. 1 – 13. Available at: <http://www.sciencedirect.com/science/article/pii/S0262885601000592>.

[22] Zhao, T. a Nevatia, R.: Car detection in low resolution aerial images. *Image and Vision Computing.* 2003, Vol. 21, Num. 8. p. 693–703.

[23] Kozempel, K. a Reulke, R.: Fast Vehicle Detection and Tracking in Aerial Image Bursts. *International Society for Photogrammetry and Remote Sensing.* September 2009, Vol. 38, Num. 3. p. 175 – 180. ISSN 1682-1750.

[24] Kim, Z. a Malik, J.: Fast vehicle detection with probabilistic feature grouping and its application to vehicle tracking. In *Proceedings on Ninth IEEE International Conference on Computer Vision*. October 2003. p. 524–531.

[25] Freund, Y. a Schapire, R. E.: A Short Introduction to Boosting. In *Journal of Japanese Society for Artificial Intelligence*. 1999. p. 771–780.

[26] Murphy, K. P.: *Dynamic bayesian networks: representation, inference and learning*. University of California, Berkeley, 2002. PhD. Thesis.

[27] Gleason, J., Nefian, A., Bouyssounousse, X. et al.: Vehicle detection from aerial imagery. In *2011 IEEE International Conference on Robotics and Automation (ICRA)*. May 2011. p. 2065–2070. ISSN 1050-4729.

[28] Moranduzzo, T. a Melgani, F.: Automatic Car Counting Method for Unmanned Aerial Vehicle Images. *IEEE Transactions on Geoscience and Remote Sensing*. March 2014, Vol. 52, Num. 3. p. 1635–1647. ISSN 0196-2892.

[29] Tuermer, S., Leitloff, J., Reinartz, P. et al.: Automatic Vehicle Detection in Aerial Image Sequences of Urban Areas using 3d HoG Features. *PCV 2010 - Photogrammetric Computer Vision and Image Analysis*. 2010, Vol. 28. p. 50–54.

[30] Pacher, G., Kluckner, S. a Bischof, H.: An Improved Car Detection using Street Layer Extraction. In *Computer Vision Winter Workshop, Ljubljana*. 2008.

[31] Tuermer, S., Leitloff, J., Reinartz, P. et al.: Motion Component Supported Boosted Classifier for Car Detection in Aerial Imagery. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Science*. 2011. ISSN 1682-1777.

[32] Xiao, J., Cheng, H., Sawhney, H. et al.: Vehicle detection and tracking in wide field-of-view aerial video. In *2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2010. p. 679–684. ISSN 1063-6919.

[33] Cheng, H.-Y., Weng, C.-C. a Chen, Y.-Y.: Vehicle Detection in Aerial Surveillance Using Dynamic Bayesian Networks. *Image Processing, IEEE Transactions on*. April 2012, Vol. 21, Num. 4. p. 2152–2159. ISSN 1057-7149.

[34] Tsai, W.-H.: Moment-preserving thresolding: A new approach. *Computer Vision, Graphics, and Image Processing*. 1985, Vol. 29, Num. 3. p. 377 – 393. Available at: <http://www.sciencedirect.com/science/article/pii/0734189X85901331>. ISSN 0734-189X.

[35] Chan, C. H.: *Multi-scale Local Binary Pattern Histogram for Face Recognition*. University of Surrey, 2008. PhD. Thesis. Available at: <http://www.ee.surrey.ac.uk/CVSSP/Publications/papers/Chan-PHD-2008.pdf>.

[36] Pietikäinen, M., Hadid, A., Zhao, G. et al.: *Local Binary Patterns for Still Images*. London: Springer London, 2011. 13-47 p. Computational Imaging and Vision, vol. 40. Available at: <http://dx.doi.org/10.1007/978-0-85729-748-8_2>. ISBN 978-0-85729-747-1.

[37] Coelho, L. P.: Mahotas: Open source software for scriptable computer vision. *Journal of Open Research Software*. 2013, Vol. 1, Num. 1.

[38] Vedaldi, A. a Fulkerson, B.: *VLFeat: An Open and Portable Library of Computer Vision Algorithms* [online]. 2008. Available at: <http://www.vlfeat.org/>.

[39] Liao, S., Zhu, X., Lei, Z. et al.: Learning Multi-scale Block Local Binary Patterns for Face Recognition. In Lee, S.-W. a Li, S. (ed.): *Advances in Biometrics*. Berlin: Springer Berlin Heidelberg, 2007. p. 828–837. Lecture Notes in Computer Science, vol. 4642. Available at: <http://dx.doi.org/10.1007/978-3-540-74549-5_87>. ISBN 978-3-540-74548-8.

[40] Zivkovic, Z.: Improved adaptive Gaussian mixture model for background subtraction. In *Proceedings of the 17th International Conference on Pattern Recognition*. August 2004. p. 28–31 Vol.2. ISSN 1051-4651.

[41] Viola, P. a Jones, M. J.: Robust real-time face detection. *International journal of computer vision*. 2004, Vol. 57, Num. 2. p. 137–154.

[42] Ju, Y., Zhang, H. a Xue, Y.: Research of Feature Selection and Comparison in AdaBoost based Object Detection System. In *Journal of Computational Information Systems*. 2013. ISSN 15539105.

[43] Yilmaz, A., Javed, O. a Shah, M.: Object Tracking: A Survey. *ACM Computing Surveys*. December 2006, Vol. 38, Num. 4. Available at: <http://doi.acm.org/10.1145/1177352.1177355>. ISSN 0360-0300.

[44] Särkkä, S.: *Bayesian Filtering and Smoothing*. New York: Cambridge University Press, October 2003. ISBN 9781107439627.

[45] Singer, R.: Estimating Optimal Tracking Filter Performance for Manned Maneuvering Targets. *IEEE Transactions on Aerospace and Electronic Systems*. July 1970, AES-6, Num. 4. p. 473–483. ISSN 0018-9251.

[46] Koller, D., Weber, J. a Malik, J.: *Robust Multiple Car Tracking with Occlusion Reasoning*. Berkeley: EECS Department, University of California, Berkeley, November 1993. UCB/CSD-93-780. Available at: <http://www.eecs.berkeley.edu/Pubs/TechRpts/1993/6308.html>.

[47] Obolensky, N.: *Kalman Filtering Methods for Moving Vehicle Tracking*. University of Florida, 2002. Master Thesis. Master Thesis.

[48] Chen, Z.: *Bayesian filtering: From Kalman filters to particle filters, and beyond* [online]. Hamilton: McMaster University, 2003. Manuscript. Accessed: May 2nd 2015]. Available at: <http://www.dsi.unifi.it/users/chisci/idfric/Nonlinear_filtering_Chen.pdf>.

[49] Karlsson, R. a Gustafsson, F.: Monte Carlo data association for multiple target tracking. In *Target Tracking: Algorithms and Applications*. October 2001. p. 13/1–13/5 vol.1.

[50] Samuelsson, O.: *Video Tracking Algorithm for Unmanned Aerial Vehicle Surveillance*. KTH Royal Institute of Technology, School of Electrical Engineering, 2012. Master Thesis.

[51] Hess, R. a Fern, A.: Discriminatively trained particle filters for complex multi-object tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*. June 2009. p. 240–247. ISSN 1063-6919.

[52] Stachniss, C.: *Exploration and Mapping with Mobile Robots*. University of Freiburg, Department of Computer Science, 2006. PhD. Thesis. Available at: <http://www.informatik.uni-freiburg.de/~stachnis/pdf/stachniss06phd.pdf>.

[53] Reilly, I. H. a Shah, M.: Detection and Tracking of Large Number of Targets in Wide Area Surveillance. In *11th European Conference on Computer Vision*. Berlin: Springer-Verlag Berlin Heidelberg, 2010. p. 186–199. ISBN 978-3-642-15558-1.

[54] Xiao, J., Cheng, H., Sawhney, H. et al.: Vehicle detection and tracking in wide field-of-view aerial video. In *2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2010. p. 679–684. ISSN 1063-6919.

[55] Prokaj, J., Zhao, X. a Medioni, G.: Tracking many vehicles in wide area aerial surveillance. In *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. June 2012. p. 37–43. ISSN 2160-7508.

[56] Saleemi, I. a Shah, M.: Multiframe Many-Many Point Correspondence for Vehicle Tracking in High Density Wide Area Aerial Videos. *International Journal of Computer Vision*. 2013, Vol. 104, Num. 2. p. 198–219. Available at: <http://dx.doi.org/10.1007/s11263-013-0624-1>. ISSN 0920-5691.

[57] Gordon, N. J., Salmond, D. J. a Smith, A. F.: Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In IET: *IEE Proceedings-F (Radar and Signal Processing)*. 1993. p. 107–113.

[58] Baker, K. A.: *Mathematics of Computer Graphics: Cubic Spline Curves*. 2002. Department of Mathematics, UCLA. Course Handouts. Available at: <http://www.math.ucla.edu/~baker/149.1.02w/handouts/dd_splines.pdf>.

[59] Fritsch, F. a Carlson, R.: Monotone Piecewise Cubic Interpolation. *SIAM Journal on Numerical Analysis*. 1980, Vol. 17, Num. 2. p. 238–246. Available at: <http://dx.doi.org/10.1137/0717021>.

[60] Bernardin, K. a Stiefelhagen, R.: Evaluating multiple object tracking performance: the CLEAR MOT metrics. *EURASIP Journal on Image and Video Processing*. 2008, Vol. 2008.

[61] Szegedy, C., Toshev, A. a Erhan, D.: Deep neural networks for object detection. In *Advances in Neural Information Processing Systems*. 2013. p. 2553–2561.

[62] Erhan, D., Szegedy, C., Toshev, A. et al.: Scalable object detection using deep neural networks. In IEEE: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. 2014. p. 2155–2162.

Appendix A

CD Contents

The physical version of this thesis contains CD medium with following content:

- `/thesis/pdf/` — this thesis in Portable Document Format file.
- `/thesis/tex/` — TeX source files of this thesis.
- `/video/` — introduction video of the presented system.
- `/poster/` — poster of the presented system that was published at Excel@FIT 2015 conference in A1 format.
- `/src/` — source code of the presented system and the supplementing applications.
- `/doc/` – generated documentation of the source code.
- `/bin/` — binary application of the presented system with a running example. The application is 32-bit Windows 7 executable binary file.
- `/evaluation/` — output of the evaluation in XML data format.
- `README.txt` — text file with description of the contents of the CD media, and guide how to build the application from the source and run it.

Appendix B

Detailed Evaluation Results

B.1 Evaluation of Detection Algorithm

This section contains detailed results of the evaluation of detection algorithm. The evaluation principle, methodology and overall results with discussion are presented in section 8.2.

B.1.1 Netroufalky

Number of all ground truths in process of evaluation of detection algorithm in Netroufalky video sequence is 5868.

Table B.1: Detection algorithm using strong vehicle classifier.

Coordinate System	Raw		Reference		Work	
Filtering	No	Yes	No	Yes	No	Yes
Valid detections VD	4357	4295	4682	4621	5048	4983
False detections FD	551	533	755	703	1067	1025
Missed truths MT	1329	1391	1004	1065	638	703
Comput. cost T_c (ms)	126.175	104.011	127.961	105.735	60.914	31.822
$RMSE$ (m)	0.309	0.311	0.302	0.306	0.472	0.479
Precision P	0.888	0.890	0.861	0.868	0.826	0.829
Recall R	0.766	0.755	0.798	0.813	0.888	0.876

Table B.2: Detection algorithm using weak vehicle classifier.

Coordinate System	Raw		Reference		Work	
Filtering	No	Yes	No	Yes	No	Yes
Valid detections VD	5374	5309	5484	5415	5578	5519
False detections FD	2212	1332	2656	1415	3953	1988
Missed truths MT	312	337	202	271	108	167
Comput. cost T_c (ms)	126.498	101.600	128.414	102.960	59.940	29.004
$RMSE$ (m)	0.311	0.315	0.319	0.322	0.451	0.456
Precision P	0.708	0.799	0.674	0.793	0.585	0.735
Recall R	0.945	0.934	0.964	0.952	0.981	0.971

B.1.2 Olomouc

Number of all ground truths in process of evaluation of detection algorithm in Olomouc video sequence is 10 210.

Table B.3: Detection algorithm using strong vehicle classifier.

Coordinate System	Raw		Reference		Work	
Filtering	No	Yes	No	Yes	No	Yes
Valid detections *VD*	7211	6987	8041	7803	8014	7768
False detections *FD*	1112	1100	1432	1402	1398	1353
Missed truths *MT*	2999	3223	2169	2407	2196	2442
Comput. cost T_c (ms)	158.759	106.228	166.007	110.390	94.272	44.892
$RMSE$ (m)	0.381	0.382	0.397	0.400	0.422	0.424
Precision P	0.866	0.864	0.849	0.848	0.851	0.852
Recall R	0.706	0.684	0.788	0.764	0.785	0.761

Table B.4: Detection algorithm using weak vehicle classifier.

Coordinate System	Raw		Reference		Work	
Filtering	No	Yes	No	Yes	No	Yes
Valid detections *VD*	9111	8891	9476	9253	9473	9225
False detections *FD*	3656	2660	5193	3107	7963	3149
Missed truths *MT*	1099	1319	734	957	737	985
Comput. cost T_c (ms)	160.775	111.236	168.491	116.042	95.933	45.787
$RMSE$ (m)	0.401	0.403	0.400	0.402	0.455	0.451
Precision P	0.714	0.770	0.646	0.749	0.543	0.746
Recall R	0.892	0.871	0.928	0.906	0.928	0.904

B.1.3 Sheffield

Number of all ground truths in process of evaluation of detection algorithm in Sheffield video sequence is 3651.

Table B.5: Detection algorithm using strong vehicle classifier.

Coordinate System	Raw		Reference		Work	
Filtering	No	Yes	No	Yes	No	Yes
Valid detections *VD*	3379	3372	3432	3426	3391	3372
False detections *FD*	2605	1025	4393	1167	479	287
Missed truths *MT*	272	279	219	225	260	279
Comput. cost T_c (ms)	222.170	103.740	234.478	106.526	112.841	44.084
$RMSE$ (m)	0.591	0.595	0.586	0.591	0.417	0.425
Precision P	0.574	0.767	0.439	0.746	0.876	0.922
Recall R	0.925	0.924	0.940	0.938	0.929	0.924

Table B.6: Detection algorithm using weak vehicle classifier.

Coordinate System	Raw		Reference		Work	
Filtering	No	Yes	No	Yes	No	Yes
Valid detections VD	3557	3551	3569	3564	3543	3537
False detections FD	26917	1867	38013	2008	11408	671
Missed truths MT	94	100	82	87	108	114
Comput. cost T_c (ms)	225.331	99.284	236.483	100.791	113.159	40.102
$RMSE$ (m)	0.529	0.534	0.530	0.539	0.414	0.421
Precision P	0.117	0.655	0.086	0.640	0.237	0.841
Recall R	0.974	0.973	0.978	0.976	0.970	0.969

B.2 Evaluation of Tracking Algorithm

This section contains detailed results of the evaluation of tracking algorithm — graphs of divergence vectors and progress of spatial error for all evaluated configurations of tracking algorithm. The evaluation principle, methodology and overall results with discussion are presented in section 8.3.

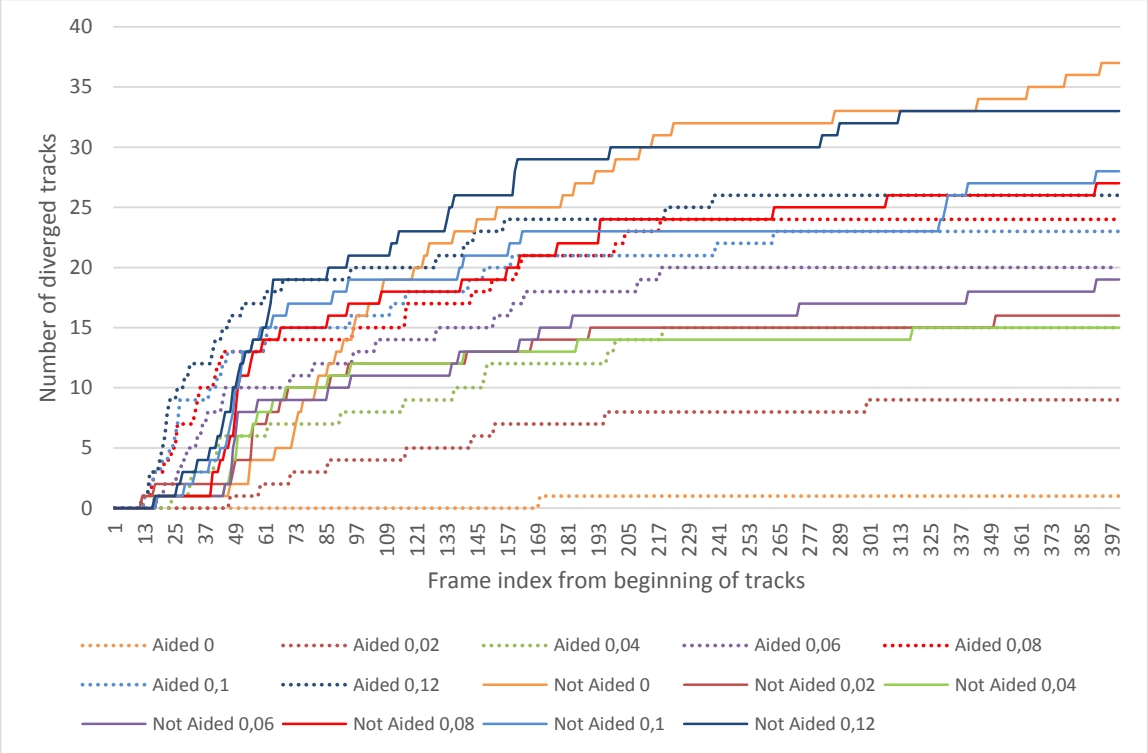


Figure B.1: Values of divergence vectors for evaluation sequence Netroufalky.

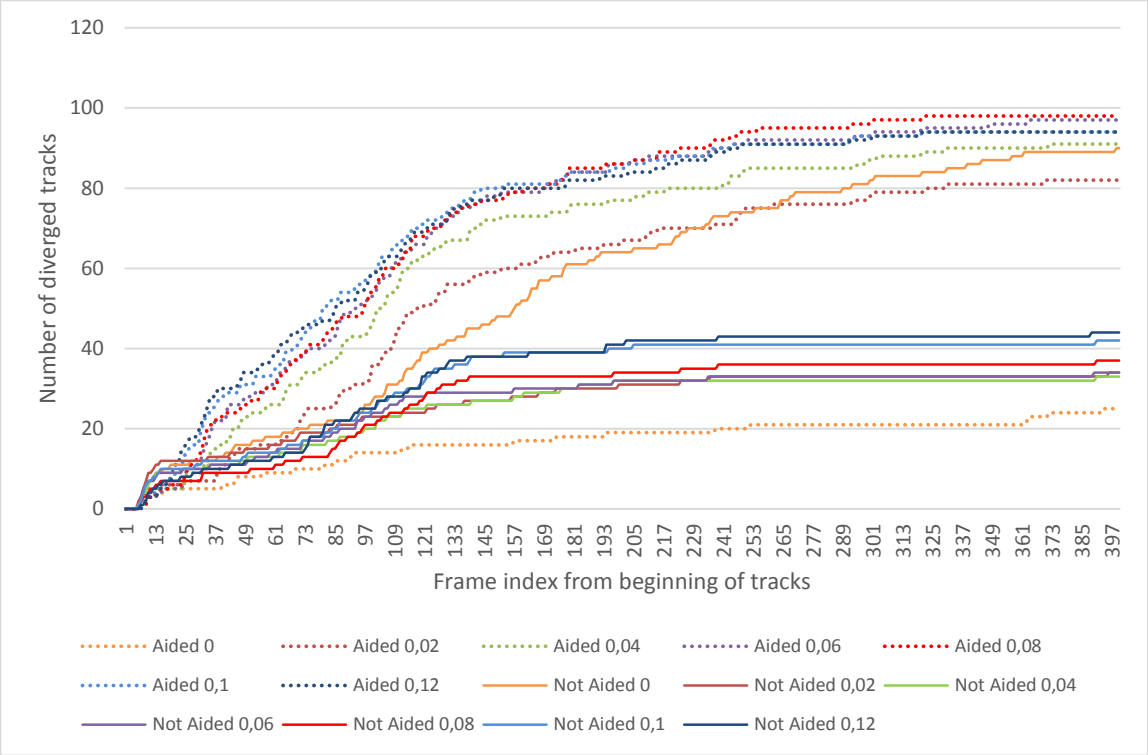


Figure B.2: Values of divergence vectors for evaluation sequence Olomouc.

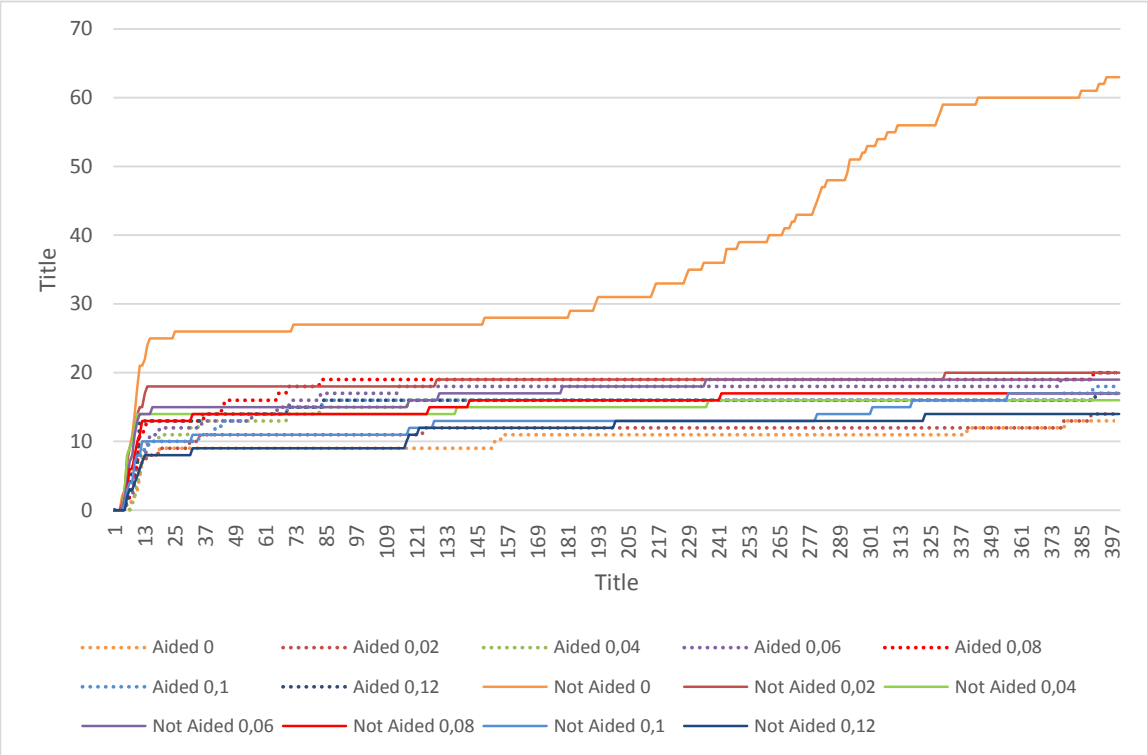


Figure B.3: Values of divergence vectors for evaluation sequence Sheffield.

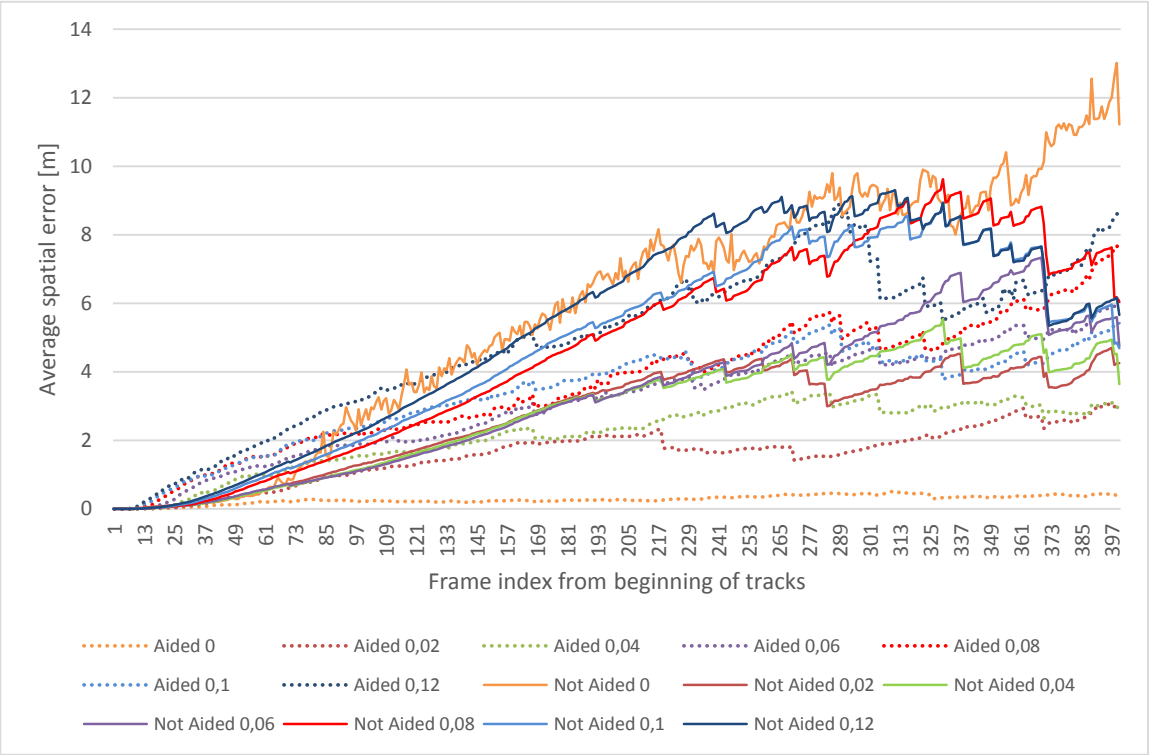


Figure B.4: Progress of average spatial error of tracking algorithm for sequence Netroufalky.

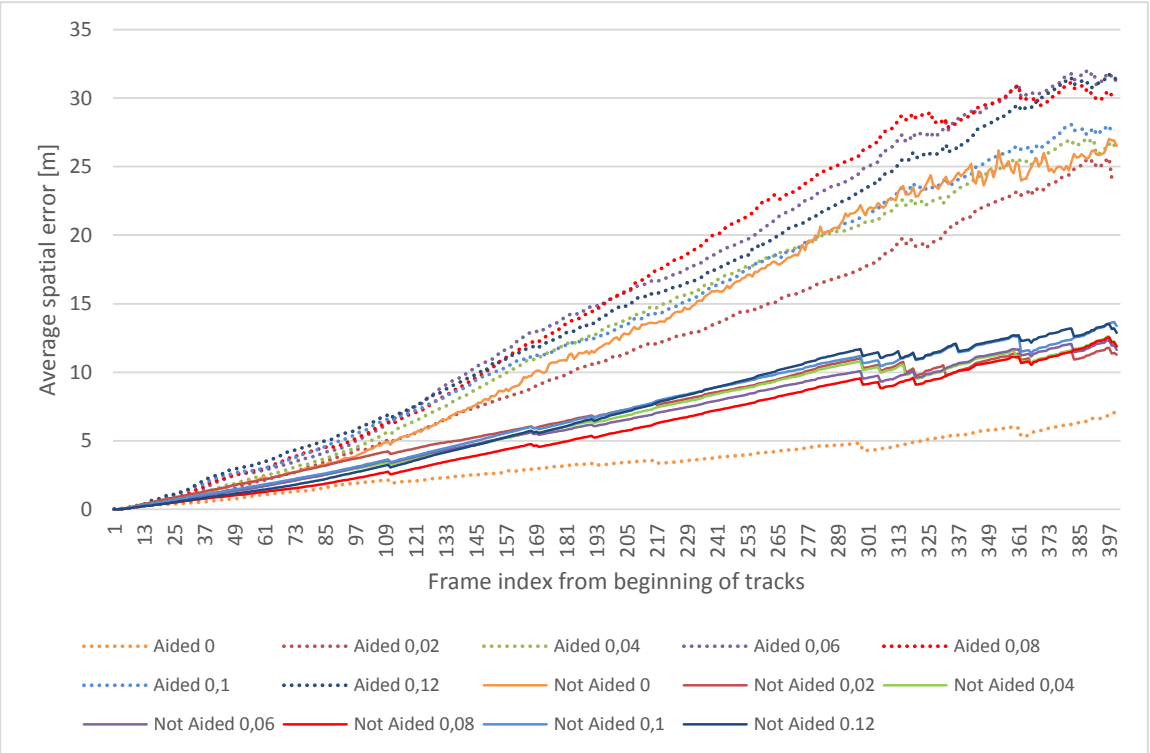


Figure B.5: Progress of average spatial error of tracking algorithm for sequence Olomouc.

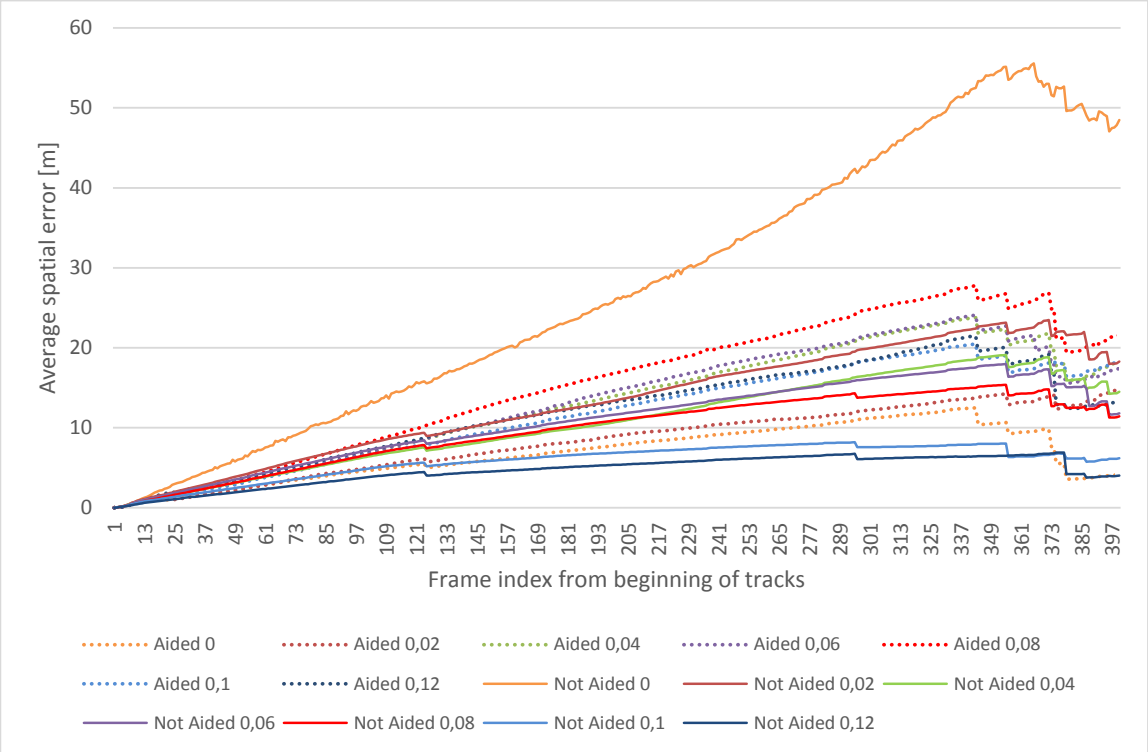


Figure B.6: Progress of average spatial error of tracking algorithm for sequence Sheffield.

Appendix C

Paper: PIA15/HRIGI15

AUTOMATIC VEHICLE TRAJECTORY EXTRACTION FOR TRAFFIC ANALYSIS FROM AERIAL VIDEO DATA

Jiří Apeltauer[a], Adam Babinec[b], David Herman[b], Tomáš Apeltauer[a]

[a] Faculty of Civil Engineering, Brno University of Technology, Czech Republic - (apeltauer.j, apeltauer.t)@fce.vutbr.cz
[b] RCE systems s.r.o., Svatopluka Čecha 1d, 612 00 Brno, Czech Republic - (adam.babinec, david.herman)@rcesystems.cz

KEY WORDS: traffic monitoring, vehicle detection, vehicle tracking, aerial imagery, UAV, particle filter

ABSTRACT:

This paper presents a new approach to simultaneous detection and tracking of vehicles moving through an intersection in aerial images acquired by an unmanned aerial vehicle (UAV). Detailed analysis of spatial and temporal utilization of an intersection is an important step for its design evaluation and further traffic inspection. Traffic flow at intersections is typically very dynamic and requires continuous and accurate monitoring systems. Conventional traffic surveillance relies on a set of fixed cameras or other detectors, requiring a high density of the said devices in order to monitor the intersection in its entirety and to provide data in sufficient quality. Alternatively, a UAV can be converted to a very agile and responsive mobile sensing platform for data collection from such large scenes. However, manual vehicle annotation in aerial images would involve tremendous effort. In this paper, the proposed combination of vehicle detection and tracking aims to tackle the problem of automatic traffic analysis at an intersection from visual data. The presented method has been evaluated in several real-life scenarios.

1. INTRODUCTION

A detailed analysis and evaluation of traffic flow is essential for a precise design and development of transport infrastructure. The current primary sources of traffic statistics are measurement stations based on induction loops and ultrasonic sensors, which count vehicles that pass a given point on the road. These conventional solutions typically provide data only in the form of basic frequency statistics, i.e. Annual Average Day Traffic (AADT). However, these systems have obvious shortcomings due to fixed installation, a very limited field of view and the type of sensors utilized.

Intersections are the most limiting factor to the capacity of the whole transport network, and therefore it is necessary to pay close attention to their design. Complex road junctions are of various types and can take up a large area, which is difficult to monitor in its entirety. For that reason, the standard monitoring systems are often placed only at the entrances and exits. However, this can be very restrictive in situations where the information about the behaviour of traffic participants during the passage through the intersection is crucial (e.g. the evaluation of the intersection design, a detailed comparison with simulated data, etc.). Furthermore, deploying fixed cameras or other on-ground counters over such wide range typically requires massive investment (which is far from realistic) and so alternatives are needed.

Aerial video surveillance using a wide field-of-view sensor has provided new opportunities in traffic monitoring over such extensive areas. In fact, unmanned aircraft systems equipped with automatic position stabilization units and high resolution cameras could be the most effective choice for data acquisition in sufficient quality. UAVs, unlike satellites or airplanes, are able to collect visual data from low altitudes, and therefore provide images with adequate spatial resolution for further traffic inspection, i.e. vehicle detection and tracking.

In this paper, we are concerned with tracking multiple vehicles in order to obtain detailed and accurate information about the vehicles' trajectories in the course of their passage through the intersection. The visual data for the analysis are captured by a camera mounted on an UAV. The operating time of a UAV is about twenty minutes due to its rapid consumption of battery power. The data is not analysed on board but recorded on a memory card and subsequently post-processed on a high-performance computer.

The remaining part of the present paper is organized as follows: Section 2 provides an overview of the related research conducted in the field of traffic surveillance and aerial image processing. Section 3 outlines the architecture of the proposed vehicle detection and tracking system. In section 4 our vehicle detection algorithm is described. A utilized vehicle tracking framework is presented in section 5. Section 6 provides insight into evaluation experiments and their results. Section 7 discusses the performance and applicability of the proposed system and possible future improvements.

2. RELATED WORK

Over the past few years, aerial image processing has become a popular research topic because of increased data availability. Aerial images can cover a large area in a single frame, which makes them attractive for monitoring and mapping tasks. Therefore, the utilization of UAVs operating in low-altitude for traffic inspection has been a major research interest in the past decade; an introduction to the current trends can be found in this brief survey paper (Lee and Kwak, 2014). Generally speaking, the task can be divided into two essential parts: vehicle detection and vehicle tracking.

The design of efficient and robust vehicle detection methods in aerial images has been addressed several times in the past. In general, these methods can be classified into two categories depending on whether explicit or implicit model is utilized (Nguyen et al., 2007). The explicit model approach uses a generic 2D or 3D model of vehicle and the detection predominately relies on geometric features like edges, lines and surfaces (Zhao and Nevatia, 2001, Moon et al., 2002). Kozempel and Reulke (Kozempel and Reulke, 2009) provided a very fast solution which is based on four special shaped edge filters aimed to represent an average vehicle. These filters, however, have to be pointed in a cor-

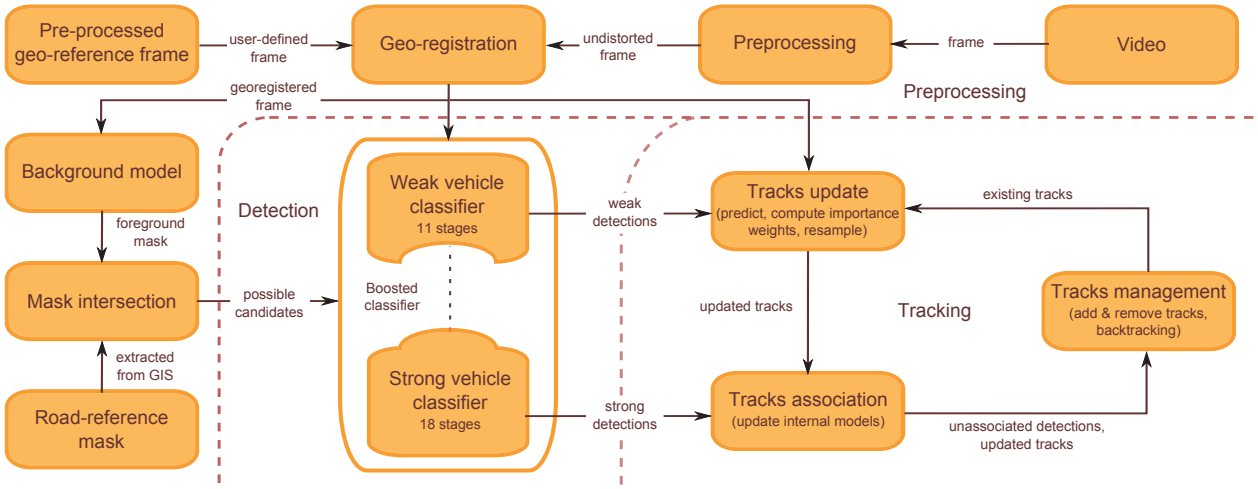


Figure 1. System architecture – the Boosted classifier is divided into two parts, the weak and the strong classifier. The detections taken from the strong classifier are used for initialization of new targets and/or for the update of the appearance models of existing targets when an association occurs. The detections returned by the weak classifier are used as clues in the tracking procedure.

rect direction according to the street database. In case of implicit approaches, the internal representation is derived through collecting statistics over the extracted features like histogram of oriented gradients (HoG), local binary patterns (LBP), etc. The detection for candidate image regions is performed by computing the feature vectors and classifying them against the internal representation (Nguyen et al., 2007, Sindoori et al., 2013, Lin et al., 2009, Tuermer et al., 2010). Among the main generic disadvantages of these approaches are the need for a huge amount of annotated training data, a lot of miss detections of rotated vehicles, and computational expensiveness during the training phase (the features are usually passed to a cascade classifier training algorithm).

To achieve real-time performance, Gleason and Nefian (Gleason et al., 2011) employed a two-stage classifier. The first stage performs a fast filtration based on the density of corners, colour profiles, and clustering. The second stage is more complex: it computes HoG and histogram of Gabor coefficients as features for binary classifier. Similar preprocessing phase is often used in order to make the detection faster and more reliable (Moranduzzo and Melgani, 2014). Another strategy for elimination of false positive detections is restricting the areas for vehicle detection by application of street extraction techniques (Pacher et al., 2008, Tuermer et al., 2010). In contrast to the above mentioned approaches, which take information from a single image only, Truemer et al. (Tuermer et al., 2011) tried to enhance the performance by incorporating temporal information from motion analysis into the detection process. In (Xiao et al., 2010), the authors also employed a motion analysis by a three-frame subtraction scheme; moreover, they proposed a method for track associations by graph matching and vehicle behaviour modelling. Next to the region or sliding window based method in (Cheng et al., 2012), they also designed a pixel-wise detector of vehicles which employs dynamic Bayesian network in the classification step. (Selvakumar and Kalaivani, 2013) presents a brief comparative study of detection techniques.

Video-based moving object tracking is one of the most popular research problems in computer vision. However, it is still a challenging task due to the presence of noise, occlusion, dynamic and cluttered backgrounds, and changes in the appearance of the tracked object, which are all very common in aerial images. Numerous tracking approaches have been presented in recent years; a detailed survey can be found in (Yilmaz et al., 2006). Our goal is to obtain the trajectories of targets over time and to maintain a correct, unique identification of each target throughout. Continuous tracking of multiple similar targets becomes tricky when the targets pass close to one another, which is very common at intersections. One of the early attempts to deal with the occlusions for traffic surveillance was proposed by Koller et al. (Koller et al., 1993), employing an explicit occlusion reasoning coupled with Kalman filters. However, to speed up the process of tracking and to accommodate the non-Gaussianness nature of the problem, a group of sequential Monte Carlo methods, also known as Particle Filters, is utilized (Rothrock and Drummond, 2000, Danescu et al., 2009, Hue et al., 2002). Particle filters can be discriminatively trained for specific environment and different objects-to-be-tracked tasks, as demonstrated by Hess and Fern in (Hess and Fern, 2009). Current approaches in vehicle tracking from aerial or satellite imagery aim at off-line optimization of data association, e.g. by deploying bipartite graph matching (Xiao et al., 2010, Reilly et al., 2010) or by revising temporal tracking correspondence as done by Saleemi and Shah in (Saleemi and Shah, 2013) by maintaining multiple possible candidate tracks per object using a context-aware association (vehicle leading model, avoidance of track intersection) and applying a weighted hypothetical measurement derived from the observed measurement distribution.

3. SYSTEM OVERVIEW

This paper proposes a method for detection and tracking of vehicles passing through an intersection for a detailed traffic analysis. The results are used for evaluation of the design of intersection and its contribution in the traffic network. The output from the analysis needs to be in the orthogonal coordinate system of the analysed intersection; therefore the transformation between the reference image and the intersection's coordinate system is known. For simplicity's sake the interchange types of intersections are not addressed and the analysed area is approximated by a plane. Figure 1 depicts the overall design of the system, which can be divided into three main parts: preprocessing, vehicle detection, and tracking.

In the preprocessing step, the acquired image is undistorted and geo-registered against a user-selected reference frame. The methods for image undistortion have been addressed in literature several times (Mallon and Whelan, 2004, Wang et al., 2011, Beauchemin and Bajcsy, 2001). In our case, radial and tangential

distortions are employed. The perspective transformation model is used in the geo-registration process. First, local ORB features (Rublee et al., 2011) are extracted both from the acquired undistorted frame and the reference frame. The features are then matched based on their descriptor distances and cross-validated, forming pairs of points which are used for estimation of geometrical transformation. Robustness of the algorithm is achieved utilizing RANSAC procedure (Fischler and Bolles, 1981).

Due to its outstanding detection performance, the boosting technique introduced by Viola and Jones was adopted for the vehicle detection (Viola and Jones, 2001, Lienhart and Maydt, 2002). In order to increase robustness to changes in illumination and to accelerate the training phase, Multi-scale Block Local Binary Patterns (MB-LBP) was employed. The searching space is restricted to the intersection of the motion and street masks with aim to considerably decrease false positive rate and computational demands. The detections which are not associated with existing tracks are added to the tracker as new targets.

For tracking, a sequential particle filter has been adopted. However, due to exponential complexity in the number of tracked targets, the system utilizes a set of fully independent Bootstrap particle filters (Isard and Blake, 1998), one filter per vehicle, rather than the joint particle filter (Khan et al., 2003). A target is represented by a gradually updated rectangular template. To further improve the tracker's robustness to cluttered background, a weak vehicle classifier with high positive detection rate is introduced to generate possible candidates for each frame. In fact, the weak classifier is obtained as an earlier stage of the robust one. Therefore, the acquired detections naturally contain a lot of false alarms; however, the probability of true positives is also not inconsiderable. Thanks to the high frame rate of input video, it seems to be beneficial to assume that the true positive detections and the predicted target states are very close to each other. The application of this assumption can effectively eliminate false alarms and can help avoid the tracking failures. The following sections provide detailed explanations of the most important parts of the whole system.

4. DETECTION

To detect vehicles, Viola and Jones's AdaBoost algorithm was utilized for the selection of appropriate features and construction of a robust detector (i.e. binary classifier). In their original paper, the authors used HAAR features for object representations; however, HAAR features have poor robustness to illumination changes and lead to a high false alarm rate (Ju et al., 2013). To alleviate these challenges and accelerate the learning phase, MB-LBP has been employed (Liao et al., 2007). Comparing with the original LBP calculated in a 3x3 pixel neighbourhood, MB-LBP is computed from average values of block sub-regions; therefore, MB-LBP is more robust since it encodes not only microstructures but also macrostructures, and provides more complete image representation. According to (Ju et al., 2013), MB-LBP features have a comparable hit rate to HAAR features, but a significantly smaller false alarm rate, which is crucial in the proposed system. Classification is performed in multi-scale and the obtained detections are grouped together with respect to their spatial similarity. The number of neighbours is used as a measure of confidence for determination of further addition of an unassociated vehicle as a new target to the tracker.

In order to eliminate false positives and accelerate detection, we restricted the searching area only to road surface. In our case, this information can be easily extracted from GIS because of the implicit requirement of geo-registration process. Moreover, the foreground mask is generated by background subtraction method based on Gaussian Mixture model (Zivkovic, 2004), and it is subsequently intersected with the acquired street mask. Afterwards, several morphological operations (erosion and dilation) are performed on the result of the intersection to reduce noise. This strategy reduces the false positive rate of the detector significantly.

The detector was trained on a hand annotated training dataset with 20,000 positive and 20,000 negative samples taken from aerial videos. The size of the samples is 32×32 pixels. The positive samples contain cars of different types, colours, and orientations. The negative samples were created from the surroundings of the analysed intersection, as well as from the road surface with the emphasis on horizontal traffic signs. Examples of both positive and negative samples are shown in Figure 2.

(a) Positive samples (scaled for illustration)

(b) Negative samples

Figure 2. Examples of samples used to train vehicle detector.

5. TRACKING

Over the last few years, particle filters, also known as sequential Monte Carlo methods, have proved to be a powerful technique for tracking purposes owing to their simplicity, flexibility, and the ability to cope with non-linear and non-Gaussian tasks (Rothrock and Drummond, 2000). Yet, particle filters may perform poorly when the posterior density is multi-modal. To overcome this difficulty, each target can be tracked by an individual particle filter as used in (Schulz et al., 2001, Danescu et al., 2009). In contrast to the approaches based on the extension of the state-space to include all targets, this approach considerably reduces the number of particles which are necessary for reliable tracking. Next, these individual particle filters can easily interact through the computation of the particle weights to handle the tracking scenario constraints (e.g. two vehicles cannot occupy the same place).

For individual trackers, a subtype of sequential importance sampling filter, Bayesian bootstrap particle filter, has been employed. Bootstrap filter uses transition density as a proposal density and performs resampling step in each iteration. In what follows, let $MPF = (C, \{\mathcal{X}_i\}_{i \in C}, \{M_i\}_{i \in C}, \mathcal{W}, t, \mathcal{E})$ denote a simplified particle representation of the $|C|$ tracked vehicles, where C represents the index set, $\mathcal{X}_i$ represents the set of particles and M_i is set of internal representations which belong to the i-th target. $\mathcal{W}$ stands for a function which computes the importance weight, t represents a transition function, and $\mathcal{E}$ returns the estimated state of the target as a single measurement.

5.1 Target representation and update

Target is represented by a rectangular descriptor template consisting of 4 channels: the sum of absolute response of Scharr operator and 3 colour channels (red, green, blue); all extracted from the processed image. This type of representation is able to carry both spatial and colour information, and its computation is very fast. To further emphasise the central part of the template, where the vehicle is to be present, we deploy a circular weighted mask.

The template is updated over the period of tracking only if one or both of the following events happen:

- The strong classifier yields a new detection in place where a significant overlap with the template occurs.
- The *heat* condition c_h of weak classifier is fulfilled, which is as follows:

$$c_h(\mathbf{x}) = true \Leftrightarrow \sum_{d \in \mathcal{D}_{weak}} f(\mathbf{x}, \mathbf{x}_d, 1.5) > T_{heat}, \quad (1)$$

where $\mathcal{D}_{weak}$ is a set of current detections of the weak classifier, $\mathbf{x}_d$ is a position of the detection d in the geo-registered image, $f(\mathbf{x}, \boldsymbol{\mu}, \sigma)$ represents the value of a multivariate normal distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ expressed at center $\mathbf{x}$ of the tracked object, $\boldsymbol{\Sigma}$ is diagonal 2×2 matrix with values of σ^2 at its diagonal, and $T_{heat} = 4$ is the predefined threshold.

Additionally, to prevent undesirable swaps between the targets, the template update is disabled if multiple targets overlap. On the event of template update, the values of the template are altered by weighted average of the former template and new template extracted from the currently processed frame, where the former template has the weight of 0.95 and new the template has the weight of 0.05. This way we achieve plasticity of the target representation over the period of time while still being stable enough to keep the tracker from drifting away from the target.

5.2 Target state and motion model

In order to cover every movement of the target, a weak motion modelling was utilized. This approach brings an enhanced capability to overcome high perturbations and therefore may be more robust in situations where the movement modelling is cumbersome (typically at intersections). The particle approximation of the i-th target state consists of $\mathcal{X}_{(i)} = \{(x, y, s) | x, y, s \in \mathbb{R}\}$ where (x, y) is the location of the target in the intersection coordinate system, and s is the size of the rectangular bounding box. The components of the state are assumed to follow independent Gaussian random walk models with variances $(\sigma_x^2, \sigma_y^2, \sigma_s^2)$. The estimated target state is represented by the highest-weighted particle (*maximum a posteriori*), i.e. $\mathcal{E}(i) = \underset{p \in \mathcal{X}_{(i)}}{\operatorname{argmax}}(\mathcal{W}(p, i))$.

5.3 Importance weight computation

The proposed importance weight computation is composed of two parts: appearance similarity and AdaBoost attraction factor, and it is defined as:

$$\mathcal{W}(p, i) = e^{App(p,i) \cdot Ada(p,i)} \quad (2)$$

Let $\varphi(p, I)$ be a function which returns the descriptor template for image patch of interest in the current registered frame I parametrised by particle p and resized to the size of internal representation $T \in M_{(i)}$. Then, the appearance similarity between the internal representation and the obtained template is computed as a sum of weighted absolute differences:

$$App\,(p, i) = \frac{\sum_{(x,y) \in T} (1 - \text{absdiff}(\varphi(p, I)_{(x,y)}, T_{(x,y)})) W_{(x,y)}}{n|T|} \quad (3)$$

where W is a circular weighted mask, n denotes the number of descriptor template channels and $|.|$ returns the number of pixels of the template (all intensity values of channels are normalized to the interval of $[0, 1]$).

AdaBoost attraction factor substantially helps to overcome the situations in which the background is heavily cluttered. In such cases, the measure of the appearance similarity is not discriminative enough and the tracker may be prone to failure. To alleviate this difficulty, the detections produced by the weak classifier are used as clues during the tracking. Let $\mathcal{D}_{weak}$ be a set of detections returned by the weak classifier; then the attraction factor is defined as follows:

$$Ada(p, i) = \sum_{d \in \mathcal{D}_{weak}} f(\mathbf{x}, \mathbf{x}_d, 1.5), \quad (4)$$

where $\mathbf{x}$ is a position of the evaluated particle p, $\mathbf{x}_d$ is a position of the detection d, and function $f(\mathbf{x}, \boldsymbol{\mu}, \sigma)$ is the same as in the Equation 1.

5.4 Tracking termination

Tracking of the i-th target is automatically terminated when $\mathcal{E}(i)$ is outside the road mask, or when there was no association with any detection produced by the strong classifier for the predefined amount of time (frames). The results are recorded only for instances for which the input and output lanes are known. If the target reached the output lane and the input lane is unknown, backward tracking is utilized from the first detection in order to try to determine the input lane. The backward tracking algorithm is basically the same as the forward version; the frames are processed in a reversed order and the target is only tracked until the input lane is discovered, or until any termination condition is met.

6. EXPERIMENTS

The system presented in this paper has been evaluated on two sequences of video data captured by action camera GoPro Hero3 Black Edition mounted on a UAV flown at the height of approx. $100\,\mathrm{m}$ above the road surface. The video was captured with the resolution of $1920\,\mathrm{px} \times 980\,\mathrm{px}$ at $29\,\mathrm{Hz}$. Due to utilization of ultra-wide angle lens, the diagonal field of view was $139.6°$. The spatial resolution of the captured scene was approximately $10.5\,\mathrm{cm/px}$. In the course of data acquisition the UAV was stabilized around a fixed position in the air.

The first sequence was captured near Netroufalky construction site in Bohunice, Brno, Czech Republic. The second sequence was captured at the site of roundabout junction of Hamerska road and Lipenska road near Olomouc, Czech Republic.

The evaluation was carried out against ground truth annotated by hand consisting of trajectories of all vehicles that both fully entered and exited the crossroad area during the evaluation sequence. As high level evaluation metrics we used relative number of missed targets $\mathbf{NMT_r} = \frac{\mathbf{NMT}}{|L|}$, relative number of false tracks $\mathbf{NFT_r} = \frac{\mathbf{NFT}}{|L|}$, average number of swaps in tracks $\mathbf{ANST}$ and temporal average of measure of completeness $\mathbf{MOC_a}$ which is defined as follows:

$$\mathbf{MOC_a} = \frac{\sum_{k=0}^{n_{video}} Comp(k)}{n_{video}}. \quad (5)$$

$\mathbf{NMT}$ and $\mathbf{NFT}$ are defined as in (Gorji et al., 2011) but with respect to the whole evaluation sequence, $|L|$ is a number of ground truth tracks, n_{video} is the number of images in the evaluation sequence, $\mathbf{ANST}$ and $Comp(k)$ are described in (Gorji et al., 2011) as well.

(a) Netroufalky

(b) Olomouc

Figure 3. Scenes used for evaluation.

The spatial precision of the algorithm was evaluated using root mean square error (**RMSE**) of track position averaging over all valid tracks. The estimated track $\mathcal{E}_i$ is considered corresponding to the ground truth l_j at given time moment k if the track $\mathcal{E}_i$ is the nearest track to the truth l_j at the moment k and vice versa, and the distance between them is less than threshold $t_{dist} = 3\,\mathrm{m}$. The estimated track is valid if it corresponds to the same ground truth for the whole period of the vehicle presence in the analysed area.

Sequence	Netroufalky	Olomouc
True Tracks #	165	168
Estimated Tracks #	173	153
Valid Estimated Tracks #	143	115
$\mathbf{NMT_r}$	0.054	0.197
$\mathbf{NFT_r}$	0.052	0.163
ANST	0.049	0.066
$\mathbf{MOC_a}$	0.841	0.667
RMSE $[m]$	1.045	1.102

Table 1. The results of the evaluation.

A user of the traffic analysis tool may want to inspect and fix invalid tracks. To indicate the necessary effort that is needed to fix the invalid tracks the following graph shows the dependency of the number of missed true tracks (true tracks that were not assigned to any valid estimated track) on the rate of adjustments needed to the best partially matching estimated tracks.

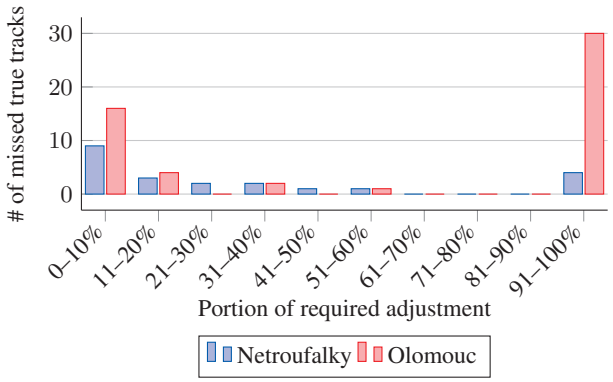


Figure 4. Histogram of missed true tracks, according to required adjustment to best matching fragments.

7. CONCLUSION

In this paper, we proposed a system for vehicles' trajectories extraction from aerial video data captured by a UAV. The functionality of the system was demonstrated on two extensive hand annotated data sets. Our approach shows sufficient performance for automatic extraction of vehicles' trajectories for further traffic inspection. Moreover, we illustrated that the manual effort needed for the correction of most of the missed trajectories to get more accurate results is negligible. Several questions remain open for future research. It would be interesting to handle the road junctions with grade separations, as well as using data fusion from more UAVs.

ACKNOWLEDGMENT

This work was supported by project "Statistical evaluation of intersection movements trajectory", identification number FAST-J-14-2461, provided by the internal research program of Brno University of Technology.

The publication was supported by project OP VK CZ.1.07/2.3.00/20.0226 "Support networks of excellence for research and academic staff in the field of transport", which is co-financed by the European Social Fund and the state budget of the Czech Republic.

REFERENCES

Beauchemin, S. S. and Bajcsy, R., 2001. Modelling and removing radial and tangential distortions in spherical lenses. In: *Multi-Image Analysis*, Lecture Notes in Computer Science, Vol. 2032, Springer Berlin Heidelberg, Berlin, pp. 1–21.

Cheng, H.-Y., Weng, C.-C. and Chen, Y.-Y., 2012. Vehicle detection in aerial surveillance using dynamic bayesian networks. *IEEE Transactions on Image Processing*, 21(4), pp. 2152–2159.

Danescu, R., Oniga, F., Nedevschi, S. and Meinecke, M., 2009. Tracking multiple objects using particle filters and digital elevation maps. In: *2009 IEEE Intelligent Vehicles Symposium*, pp. 88–93.

Fischler, M. A. and Bolles, R. C., 1981. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6), pp. 381–395.

Gleason, J., Nefian, A., Bouyssounousse, X., Fong, T. and Bebis, G., 2011. Vehicle detection from aerial imagery. In: *2011 IEEE International Conference on Robotics and Automation*, pp. 2065–2070.

Gorji, A. A., Tharmarasa, R. and Kirubarajan, T., 2011. Performance measures for multiple target tracking problems. In: *Proceedings of the 14th International Conference on Information Fusion*.

Hess, R. and Fern, A., 2009. Discriminatively trained particle filters for complex multi-object tracking. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 240–247.

Hue, C., Le Cadre, J.-P. and Perez, P., 2002. Sequential monte carlo methods for multiple target tracking and data fusion. *IEEE Transactions on Signal Processing*, 50(2), pp. 309–325.

Isard, M. and Blake, A., 1998. Condensation — conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1), pp. 5–28.

Ju, Y., Zhang, H. and Xue, Y., 2013. Research of Feature Selection and Comparison in AdaBoost based Object Detection System. *Journal of Computational Information Systems*.

Khan, Z., Balch, T. R. and Dellaert, F., 2003. Efficient particle filter-based tracking of multiple interacting targets using an mrf-based motion model. In: *Proceedings of 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, pp. 254–259.

Koller, D., Weber, J. and Malik, J., 1993. Robust multiple car tracking with occlusion reasoning. Technical Report UCB/CSD-93-780, EECS Department, University of California, Berkeley, Berkeley.

Kozempel, K. and Reulke, R., 2009. Fast vehicle detection and tracking in aerial image bursts. *International Society for Photogrammetry and Remote Sensing*, 38(3), pp. 175 – 180.

Lee, J.-N. and Kwak, K.-C., 2014. A trends analysis of image processing in unmanned aerial vehicle. *International Journal of Computer, Information Science and Engineering*, 8(2), pp. 2 – 5.

Liao, S., Zhu, X., Lei, Z., Zhang, L. and Li, S.-Z., 2007. Learning multi-scale block local binary patterns for face recognition. In: *Advances in Biometrics*, Lecture Notes in Computer Science, Vol. 4642, Springer Berlin Heidelberg, pp. 828–837.

Lienhart, R. and Maydt, J., 2002. An extended set of haar-like features for rapid object detection. In: *Proceedings of 2002 International Conference on Image Processing*, Vol. 1, pp. 900–903.

Lin, R., Cao, X., Xu, X., Wu, C. and Qiao, H., 2009. Airborne moving vehicle detection for video surveillance of urban traffic. In: *2009 IEEE Intelligent Vehicles Symposium*, pp. 203–208.

Mallon, J. and Whelan, P., 2004. Precise radial un-distortion of images. In: *Proceedings of the 17th International Conference on Pattern Recognition*, Vol. 1, pp. 18–21 Vol.1.

Moon, H., Chellapa, R. and Rosenfeld, A., 2002. Performance analysis of a simple vehicle detection algorithm. *Image and Vision Computing*, 20, pp. 1 – 13.

Moranduzzo, T. and Melgani, F., 2014. Automatic car counting method for unmanned aerial vehicle images. *IEEE Transactions on Geoscience and Remote Sensing*, 52(3), pp. 1635–1647.

Nguyen, T. T., Grabner, H., Gruber, B. and Bischof, H., 2007. On-line boosting for car detection from aerial images. In: *IEEE International Conference on Reasearch, Innovation and Vision for the Future*, pp. 87–95.

Pacher, G., Kluckner, S. and Bischof, H., 2008. An improved car detection using street layer extraction. In: *Proceedings of the 13th Computer Vision Winter Workshop*, Ljubljana.

Reilly, V., Idrees, H. and Shah, M., 2010. Detection and tracking of large number of targets in wide area surveillance. In: *11th European Conference on Computer Vision*, Vol. 6313, Springer-Verlag Berlin Heidelberg, Berlin, pp. 186–199.

Rothrock, R. L. and Drummond, O. E., 2000. Performance metrics for multiple-sensor multiple-target tracking. In: *Signal and Data Processing of Small Targets 2000*, Proceedings of SPIE, Vol. 4048, pp. 521–531.

Rublee, E., Rabaud, V., Konolige, K. and Bradski, G., 2011. Orb: An efficient alternative to sift or surf. In: *2011 IEEE International Conference on Computer Vision*, pp. 2564–2571.

Saleemi, I. and Shah, M., 2013. Multiframe many-many point correspondence for vehicle tracking in high density wide area aerial videos. *International Journal of Computer Vision*, 104(2), pp. 198–219.

Schulz, D., Burgard, W., Fox, D. and Cremers, A. B., 2001. Tracking multiple moving targets with a mobile robot using particle filters and statistical data association. In: *IEEE International Conference on Robotics and Automation.*, Vol. 2, pp. 1665–1670 vol.2.

Selvakumar, S. and Kalaivani, S., 2013. Comparative study on vehicle detection techniques in aerial surveillance. *International Journal on Cybernetics & Informatics*.

Sindoori, R., Ravichandran, K. and Santhi, B., 2013. Adaboost technique for vehicle detection in aerial surveillance. *International Journal of Engineering and Technology*, 5(2), pp. 765 – 769.

Tuermer, S., Leitloff, J., Reinartz, P. and Stilla, U., 2010. Automatic vehicle detection in aerial image sequences of urban areas using 3d hog features. *Photogrammetric Computer Vision and Image Analysis*, 28, pp. 50–54.

Tuermer, S., Leitloff, J., Reinartz, P. and Stilla, U., 2011. Motion component supported boosted classifier for car detection in aerial imagery. In: *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Science*, Vol. 38 (3/W22), pp. 215–220.

Viola, P. and Jones, M., 2001. Rapid object detection using a boosted cascade of simple features. In: *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol. 1, pp. I–511–I–518 vol.1.

Wang, H., Li, M. and Zhang, L., 2011. The distortion correction of large view wide-angle lens for image mosaic based on opencv. In: *International Conference on Mechatronic Science, Electric Engineering and Computer*, pp. 1074–1077.

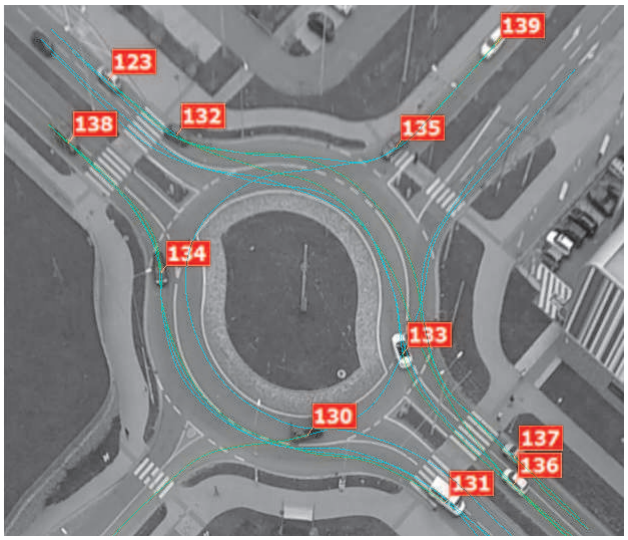
Xiao, J., Cheng, H., Sawhney, H. and Han, F., 2010. Vehicle detection and tracking in wide field-of-view aerial video. In: *2010 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 679–684.

Yilmaz, A., Javed, O. and Shah, M., 2006. Object tracking: A survey. *ACM Computing Surveys*.

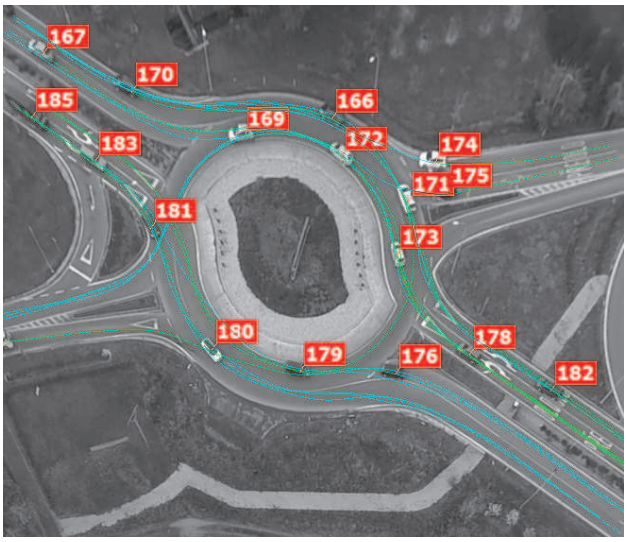
Zhao, T. and Nevatia, R., 2001. Car detection in low resolution aerial images. In: *Eight IEEE International Conference on Computer Vision*, IEEE, Vancouver, pp. 710–717.

Zivkovic, Z., 2004. Improved adaptive gaussian mixture model for background subtraction. In: *Proceedings of the 17th International Conference on Pattern Recognition*, Vol. 2, pp. 28–31 Vol.2.

APPENDIX



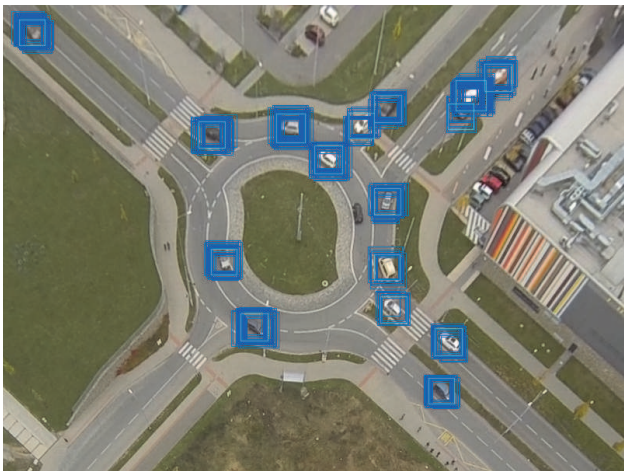
(a) Netroufalky



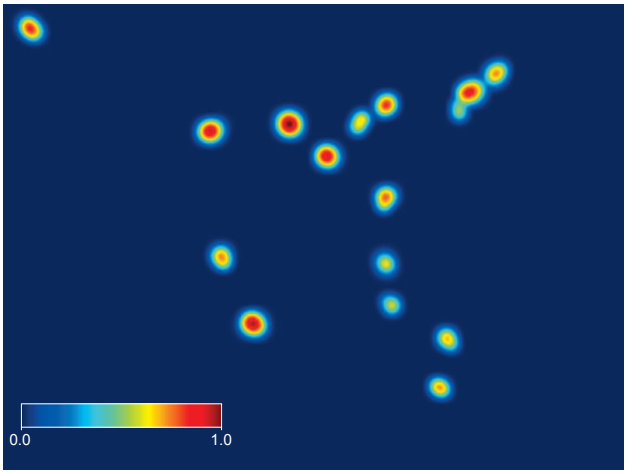
(b) Olomouc

Figure 5. Illustration of extracted vehicle trajectories in evaluated traffic scenes. The images are converted to grayscale for better readability of overlay graphics: blue-green curves represent vehicle trajectories and red labels with white numbers represent unique identificators of tracked objects.

(a) Geo-registered input image



(b) Overlaid detections: dark blue rectangles represent weak detections, light blue rectangles represent strong detections.



(c) Colourised map representing normalised values of attraction factor expressed over whole area of the scene, according to Equation 4.

Figure 6. Illustration of selected steps of single image analysis.

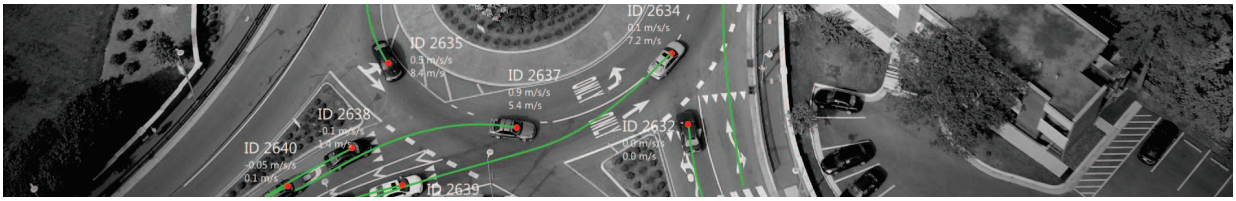
Appendix D

Paper: Excel@FIT 2015

Excel@FIT 2015
http://excel.fit.vutbr.cz

Automatic Vehicle Trajectory Extraction from Aerial Video Data

Adam Babinec*



Abstract
In this paper we present a complex solution to automatic vehicle trajectory extraction from aerial video data, providing a basis for a cost-effective and flexible way to gather detailed vehicle trajectory data in traffic scenes. The video sequences are captured using an action camera mounted on a UAV flying above the traffic scene and processed off-line. The system utilizes video stabilisation algorithm and geo-registration based on RANSAC guided transformation estimation of ORB image feature sets. Vehicles are detected in scene using AdaBoost classifier constructed of Multi-Scale Block Local Binary Patterns features. The vehicle tracking is carried out by multi-target tracker based upon set of intra-independent Bayesian bootstrap particle filters specialized to deal with environmental occlusion, multi-target overlap, low resolution and feature salinity of targets and their appearance changes. The performance of the presented system was evaluated against hand-annotated video sequences captured in distinct traffic scenes. The analysis show promising results with average target miss ratio of 22.5% while keeping incorrect tracking ratio down to 20.4%.

Keywords: Vehicle Detection — Vehicle Tracking — UAV — Aerial Imagery — Particle Filter

Supplementary Material: Demonstration Video TBD

*xbabin02@stud.fit.vutbr.cz, *Faculty of Information Technology, Brno University of Technology*

1. Introduction

Traffic congestion data and information about traffic participants, their trajectory and dynamics during their passage through a traffic scene render to be crucial in numerous applications, from transport infrastructure design analysis and improvement, through analysis of driver's behaviour in various situations (such as unusual intersections, changes in road signs, or weather/lighting conditions) to traffic management (dynamic traffic flow redirection, collision detection, navigation assistance) [1].

The current primary sources of traffic statistics are measurement stations based on induction loops and ultrasonic sensors, which count vehicles that pass a given point on the road. These conventional solutions typically provide data only in the form of basic frequency statistics. Further solution is to employ fixed cameras mounted on ground which can provide additional data, such as vehicle identification and speed estimation. Such solutions however tent to require massive investments and so alternatives are needed. Aerial video surveillance using a wide field-of-view camera sensors offer new opportunities in traffic monitoring, especially in latest years thank to the boom of multicopter UAVs and action cameras.

The utilisation of UAVs operating in low-altitude for traffic inspection has been a major research interest in the past decade; an introduction to the current trends

can be found in this brief survey [2]. Generally, the task of vehicle trajectory extraction can be divided into two essential parts: vehicle detection and vehicle tracking.

Vehicle detection methods can be classified into two categories depending on whether an explicit or implicit model is utilized [3]. Explicit model approach requires a user provided description of detected object – a generic 2D or 3D model of vehicle, which relies on geometric features, such as edges and surfaces of vehicle body or cast shadows, as seen in works of Moon et al. [4] (detection of rectangular body boundaries and windscreens in response of Canny Edge detector), Zhao and Nevatia [5] (adding possibility of shadow presence and incorporating Bayessian network in decision making step) and ZuWhan and Malik [6] (three-dimensional line features based upon detailed model of possible vehicle shapes and probablistic distributions of its dimensions).

Implicit models are derived through collecting statistics over the extracted features, such as Histogram of Oriented Gradients, Local Binary Patterns, specialised pixel-wise features and so forth. The detection for candidate image regions is performed by computing the feature vectors and classifying them against the internal representation usually built up by a cascade classifier training algorithm, such as AdaBoost [7] or more complex dynamic Bayesian networks [8]. The main idea when using implicit models is to use two-stage detection, when in the first stage, the detection candidates are filtered according to various clues such as colour-spatial profiles, density of Harris corners [9], SIFT feature salinity [10] or street extraction techniques [11, 12]. Additionally, temporal information from motion analysis and background subtraction can be incorporated, as seen in [13, 14].

Object tracking algorithms employed in traffic analysis may be divided in two groups: algorithms using Bayesian filters and off-line data association algorithms. One of the most prominent Bayesian filtering algorithm – Kalman filter has been proposed for object tracking as early, as in 1970 [15], and its variations has been used in vehicle tracking in aerial data already in 1993 [16]. To deal with non-linearity of target behaviour model, the extended Kalman filter is suggested by Obolensky in [17]. However, Kalman filter is suited only for tasks modelled by single-modal Gaussian probability density, which is limiting factor in tasks of vehicle tracking in aerial imagery. Therefore, in latest years, particle filter algorithms are being employed, as seen in works of Karlsson and Gustafsson [18] (with its modification using Bayessian Bootstrap algorithm), Samuelsson [19] and Hess et al [20] (pseudo-independent particle filters parametrized by log-linear models with error-driven discriminative filter training). Offline data association tracking algorithms may be based upon graph matching techniques with edges weighted according to spatial proximity and velocity orientation components [21] or kinematic measures, shape and appearance matching [14]. Alternative approaches can be based upon hierarchical connecting of shorter estimations of parts of trajectories – tracklets, into longer ones, eventually forming whole trajectories [22] or maintaining multiple possible candidate tracks per object using a context-aware association (vehicle leading model, avoidance of track intersection) [23].

In this paper we would like to present a system for automatic trajectory extraction from aerial video data. The system utilizes video stabilisation algorithm and geo-registration based on RANSAC [24] guided transformation estimation of ORB image features [25] extracted from annotated reference image and video sequence frames. To produce vehicle detection candidates, we apply background modelling algorithm based on Gaussian Mixture Model [26] which output (foreground mask) is fused together with road reference mask and map of currently tracked vehicles. To keep the system easily modifiable for different target types (pedestrians, animals,...), for vehicle detection, we have utilized implicit model description based on Multi-Scale Block Local Binary Pattern features [27] trained by Viola and Jones's AdaBoost algorithm. The system incorporates two classifiers — *strong classifier*, which detections are considered as significant indication of vehicle presence, and *weak classifier* with higher false alarm rate and very low target miss rate, which output is used to aid vehicle tracking stage. For tracking algorithm we have employed set of fully independent Bayesian bootstrap particle filters [18], one per each target. The algorithm was modified to cope with nature of aerial video data — environmental occlusion, multi-target overlap, low resolution and feature salinity of targets and their appearance changes.

2. Vehicle Detection

The purpose of vehicle detection stage is to provide new targets for tracking stage and aid tracking stage by giving clues about the positions of already tracked vehicles. The preceding step of vehicle detection is transformation of input image into the real world coordinate system, removing perspective effect. This transformation is derived from known transformation of reference image into real world coordinate system

and estimated transformation between reference image and current image. This step leads to orthographic representation of the scene, reducing the range of possible candidate sizes. Candidate generation is limited to road surface area retrieved from annotated reference image and the candidate must fulfil at least one of the following conditions:

- Center of candidate area exhibits the signs of motion. To detect the motion, we employed background cubtration method based on Gaussian Mixture Model as presented in [26].
- Candidate area is overlapped significantly by currently tracked vehicle. To test this condition, the position of the tracked vehicle has to be predicted – using the motion model of vehicle as described in section 3.2.

For the selection of appropriate features and construction of a robust detector, we have utilized Viola and Jones's Adaboost algorithm [7], but instead of HAAR features, we have employed Multi-Scale Block Local Binary Patterns (MB-LBP) features, which calculation is computationally less expensive, and due to the ability to encode both microstructures and macrostructures of the image area, they are shown to be more robust to illumination changes [28] and have significantly smaller false alarm rate than HAAR features, while keeping comparable hit rate [28].

The classifier was trained on hand annotated training dataset with 20000 positive and 20000 negative samples taken from aerial videos. The size of every sample is 32×32 pixels. Positive samples contain motor vehicles of different types, colours and orientations. The negative samples were created from surroundings of the intersections, as well as from the road surface with the emphasis on horizontal traffic signs. The resulting trained classifier consists of 18 stages of boosting cascade of simple MB-LBP classifiers. For the further improvement of detection assistance for tracking algorithm, the trained detector was split to two classifiers:

- *strong vehicle classifier* – consisting of all 18 stages of trained classifier, having small false alarm rate. The detections signalled by this classifier, further referenced as *strong detections*, are considered as very significant indication of vehicle presence in detection candidate area, and are treated as such further in tracking stage of the algorithm, especially as basis for new tracking targets.
- *weak vehicle classifier* – consisting of first 11 stages of trained classifier, having higher false alarm rate. This classifier is more benevolent than *strong vehicle classifier* accepting much more detection candidates and its output is used to aid vehicle tracking, acting as tracking attractors. These detections, further referenced as *weak detections*, form basis for *heat function*.

2.1 Heat Function

To speed up the operations with *weak detections*, we proposed following function which for given point in two-dimensional real-world coordinate system returns the significance of this point according to the set of *weak detections* :

$$h(\mathbf{x}) = 1 + \sum_{d \in \mathscr{D}_{weak}} f(\mathbf{x}, \mathbf{x}_d, \sigma_d) \quad , \tag{1}$$

where $\mathscr{D}_{weak}$ is a set of current *weak detections*, $\mathbf{x}_d$ is a position of the detection d in real-world coordinate system, σ_d is size of detection d in real-world coordinate system, $f(\mathbf{x}, \mu, \sigma)$ represents the value of a multivariate normal distribution $\mathscr{N}(\mu, \Sigma)$ expressed at point $\mathbf{x}$. Matrix Σ is constructed as follows:

$$\Sigma = \begin{bmatrix} (0.16\sigma)^2 & 0 \\ 0 & (0.16\sigma)^2 \end{bmatrix} \quad . \tag{2}$$

3. Vehicle Tracking

Tracking of vehicle targets in the scene is carried out using the detections generated by both strong and weak vehicle classifiers and data extracted from geo-registered input frame. It is divided into three steps: Detection-Track Association, Tracks Update and Tracking Termination, which all will be described in following subsections.

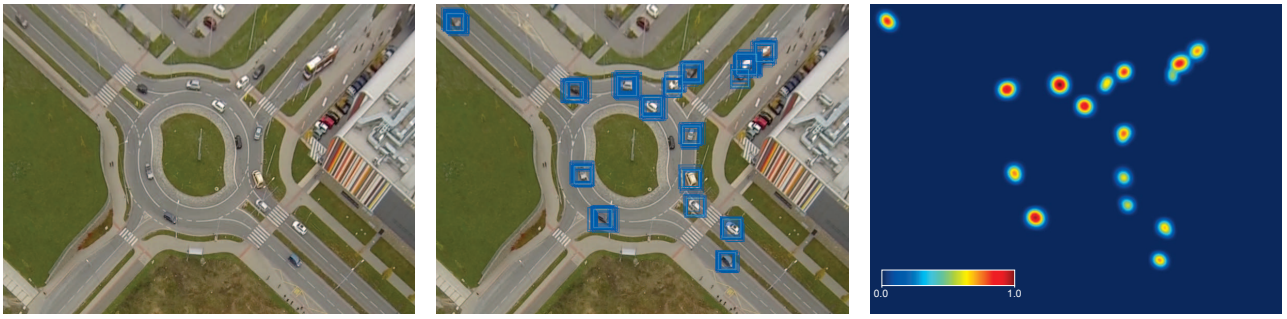
3.1 Detection-Track Association

For each detection from both sets of weak and strong detections, the best fitting tracked object is found and vice versa, while strong detections being favoured. Formally, let $\mathscr{D}_{weak}$ be a set of weak detections, $\mathscr{D}_{strong}$ be a set of strong detections, $\mathscr{D} = \mathscr{D}_{weak} \cup \mathscr{D}_{strong}$ be a set of all detections, $\mathscr{T}$ be a set of all currently tracked objects and $\mathscr{A}$ is set of constructed associations. Then for every constructed association $(d,t) \in \mathscr{A}$, where $d \in \mathscr{D}$ and $t \in \mathscr{T}$, it must be that:

$$\begin{aligned} \forall d' \in \mathscr{D}, d' \neq d : \operatorname{diff}(d',t) \geq \operatorname{diff}(d,t) \vee \\ d \in \mathscr{D}_{strong} \wedge d' \in \mathscr{D}_{weak} \end{aligned} \tag{3}$$

and

$$\begin{aligned} \forall t' \in \mathscr{T}, t' \neq t : \operatorname{diff}(d,t') \geq \operatorname{diff}(d,t) \wedge \\ \operatorname{diff}(d,t) \leq diff_{max} \quad , \end{aligned} \tag{4}$$



(a) Geo-registered input image

(b) Overlaid detections: dark blue rectangles represent weak detections, light blue rectangles represent strong detections.

(c) Colourised representation of heat function values normalised into interval $< 0, 1 >$.

Figure 1. Example of *heat function* defined in equation (1) constructed from a set of weak detections.

where $diff_{max}$ is arbitrary constant and $\mathrm{diff}(d,t)$ is function, which returns the difference of rectangles representing detection d and tracked object t. It is defined as follows:

$$\mathrm{diff}(d,t) = \frac{|r_d| + |r_t| - 2|r_d \,\&\, r_d|}{|r_d| + |r_t|} \quad , \tag{5}$$

where r_d is rectangle representing detection d, r_t is rectangle representing tracked object t, function $|r|$ returns area of rectangle r and binary operator & represents intersection of given operands.

Every strong detection that has not been associated to any of currently tracked objects forms basis for new tracking target – its initial size, position and target model is derived from detection, while velocity is set to zero.

3.2 Tracks Update

Tracks update step engages tracking algorithm itself to estimate the position of tracked targets in the current image, according to sequence of all video frames up to this time moment. For this purpose, a set of intra-independent Bayesian bootstrap particle filters has been employed – one per each target, similarly as in [29, 30]. Bootstrap filter uses transition density as a proposal density and performs resampling step in each iteration [18].

For the algorithm to be able to track its target, the description of tracked object is necessary – target model. Our system uses a rectangular descriptor template derived from detection area of the target, consisting of 3 colour channels (RGB) and edge map channel – sum of absolute response of Scharr operator in both x and y directions in the image. This template is extracted from geo-registered image bounded by area defined by initial detection, and it is resized to uniform size 32×32 pixels. To achieve the plasticity of target model, the template $\mathbf{T}_t$ of tracked object t is updated if one of the following events happen:

- There is currently associated strong detection to object t.
- There is currently associated weak detection to object t and the value of *heat* function $h(\mathbf{x})$ as described in equation (1), evaluated at the estimated position of tracked object, is greater than threshold T_{heat}.

In the case of template update, the values of template are altered by weighted average of the former template and new template extracted from currently processed image, where former template has weight of 0.95 and new template has weight of 0.05. Additionally, to prevent undesirable swaps between targets, the template update is disabled if multiple targets overlap.

The particle filter uses particles which states are defined by target position vector $\mathbf{x}$, target velocity vector $\mathbf{v}$ and target size s, all in real-world coordinate system, forming together particle state vector:

$$\mathbf{p} = \begin{bmatrix} \mathbf{x}(0) \\ \mathbf{x}(1) \\ \mathbf{v}(0) \\ \mathbf{v}(1) \\ s \end{bmatrix} \quad . \tag{6}$$

As the transition model, we consider target position is an integration of target velocity, and therefore it is represented by following matrix:

$$\mathbf{D} = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad , \tag{7}$$

and state transition equation:

$$\mathbf{p}^{(i+1)} = \mathbf{D}\left(\mathbf{p}^{(i)} + \mathbf{n}\right) \quad , \tag{8}$$

where $\mathbf{n}$ is 5D noise vector, which elements are generated randomly by normal distribution $\mathcal{N}\left(\mu, \sigma^2\right)$,

where $\mu = 0$ and value of σ is user defined parameter for each element of noise vector. In case of velocity noise and size noise, the parameter σ is constant during the whole course of tracking, whilst for position noise, the value of parameter σ evolves according to following equation:

$$\sigma_{\mathbf{x}}^{(i)} = \sigma_{\mathbf{x}} \left(1 + f^i m\right) \quad , \tag{9}$$

where, i is sequential order of currently processed image from target's tracking inception, $\sigma_{\mathbf{x}}$ is target value of $\sigma_{\mathbf{x}}^{(i)}$ for $i \to \infty$, f is falloff ratio and m is initial multiplier of covariance. The application of this approach causes the position of particle to be affected more by random noise than by its velocity during the early stage of tracking, and afterwards slowly elevating the effect of velocity. This way, the particle's velocity may slowly adapt while elevating its effect on particle behaviour.

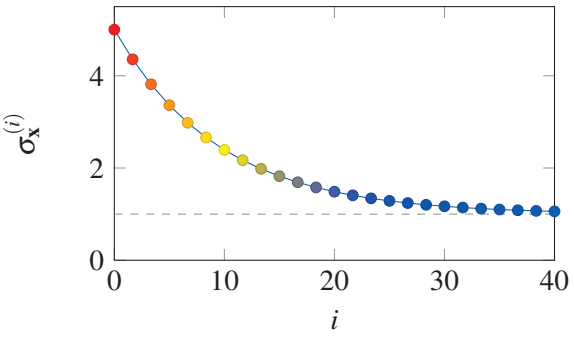


Figure 2. Graph of function $\sigma_{\mathbf{x}}^{(i)}$ defined by equation (9) according to values of i for parameters $\sigma_{\mathbf{x}} = 1$, $f = 0.9$ and $m = 4$.

The evaluation and resampling step of the particle filter is based upon importance weight $\mathscr{W}(p,t)$ of particle p for tracked target t which is defined as:

$$\mathscr{W}(p,t) = e^{App(p,t)^2 \cdot Att(p)} \quad . \tag{10}$$

The appearance similarity function $App(p,t)$ of particle p to target t is evaluated as follows:

$$App(p,t) = \frac{1}{1 + \mathrm{SAD}_C(T_t, T_p)} \quad , \tag{11}$$

where T_t is model template of target object t, T_p is model template of fictitious target object based on particle p and $\mathrm{SAD}_C(T_1, T_2)$ is sum of absolute differences of templates T_1 and T_2 across all their channels spatially weighted by circular mask around the centre of the templates at point $c = (16, 16)$ px, with radius of $16\,px$. The attraction factor function $Att(p)$ of particle p is defined as:

$$Att(p) = h(\mathbf{x}_p) \quad , \tag{12}$$

where $h(\mathbf{x})$ is *heat function* as defined in equation (1) and $\mathbf{x}_p$ is position of particle p in real world coordinate system. The estimated state $\mathscr{E}(t)$ of the target t is represented as the highest-weighted particle (*maximum a posteriori*), i.e.:

$$\mathscr{E}(t) = \arg\max_{p \in \mathscr{X}_{(t)}} \left(\mathscr{W}(p,t)\right) \quad , \tag{13}$$

where $\mathscr{X}_{(t)}$ is set of all particles of particle filter modelling tracked target t. Resampling step of particle filter for particle set $\mathscr{X}_{(t)}$ is carried out using weight proportionate random selection, also known as roulette wheel principle, according to values of particle's importance weight.

3.3 Tracking Termination

Tracking of target $t^\dagger$ is terminated when one of the following conditions are met:

- Target $t^\dagger$ leaves the area defined by annotated road surface.
- Target model of target $t^\dagger$ has not been updated for certain amount of time steps.
- Target t is overlapping with another target t' for certain amount of time steps and following condition is met:

$$App\left(p^*_{t'}, t'\right) > App\left(p^*_{t^\dagger}, t^\dagger\right) \quad , \tag{14}$$

  where $App(p,t)$ is appearance similarity function as described in equation (11), $p^*_{t'}$ is highest weighted particle of particle set $\mathscr{X}_{(t')}$ tied with target t' and $p^*_{t^\dagger}$ is highest weighted particle of particle set $\mathscr{X}_{(t^\dagger)}$ tied with target $t^\dagger$.

In case of tracking termination of target $t^\dagger$, its generated trajectory is analysed. If it is found that the target $t^\dagger$ has fully entered the analysed area of the scene, passed through it and left it, in that order, it is considered as successful tracking. Otherwise, the tracking is considered as unsuccessful and is rejected.

4. Experiments

The system presented in this paper has been evaluated on two sequences of video data captured by action camera GoPro Hero3 Black Edition mounted on a UAV flown at the height of approx. 100 m above the road surface. The video was captured with the resolution of 1920 px × 980 px at 29 Hz. Due to utilization of ultra-wide angle lens, the diagonal field of view was 139.6°. The spatial resolution of the captured scene was approximately 10.5 cm/px. In the course of data acquisition the UAV was stabilized around a fixed position in the air.

The first sequence was captured near Netroufalky construction site in Bohunice, Brno, Czech Republic. The second sequence was captured at the site of roundabout junction of Hamerska road and Lipenska road near Olomouc, Czech Republic.

(a) Netroufalky

(b) Olomouc

Figure 3. Scenes used for evaluation.

The evaluation was carried out against ground truth annotated by hand consisting of trajectories of all vehicles that both fully entered and exited the crossroad area during the evaluation sequence. As high level evaluation metrics we used relative number of missed targets $\mathbf{NMT_r} = \frac{\mathbf{NMT}}{|L|}$, relative number of false tracks $\mathbf{NFT_r} = \frac{\mathbf{NFT}}{|L|}$, average number of swaps in tracks **ANST** and temporal average of measure of completeness $\mathbf{MOC_a}$ which is defined as follows:

$$\mathbf{MOC_a} = \frac{\sum_{k=0}^{n_{video}} Comp(k)}{n_{video}}. \tag{15}$$

Metrics **NMT** and **NFT** are defined as in [31], averaged by arithmetic mean across the whole evaluation sequence, $|L|$ is a number of ground truth tracks, n_{video} is the number of images in the evaluation sequence, **ANST** and $Comp(k)$ are described in [31] as well.

The spatial precision of the algorithm was evaluated using root mean square error (**RMSE**) of track position averaging over all valid tracks. The estimated track $\mathscr{E}_i$ is considered corresponding to the ground truth l_j at given time moment k if the track $\mathscr{E}_i$ is the nearest track to the truth l_j at the moment k and vice versa, and the distance between them is less than threshold $t_{dist} = 3\,\mathrm{m}$. The estimated track is valid if it corresponds to the same ground truth for the whole period of the vehicle presence in the analysed area.

Table 1. Results of the evaluation.

Sequence	Netroufalky	Olomouc
True Tracks #	165	168
Estimated Tracks #	173	153
Valid Estimated Tracks #	143	115
$\mathbf{NMT_r}$	0.054	0.197
$\mathbf{NFT_r}$	0.052	0.163
ANST	0.049	0.066
$\mathbf{MOC_a}$	0.841	0.667
RMSE $[m]$	1.045	1.102

Incorrect or missed targets in the sequence can be easily noticed and fixed by a human user. To indicate the necessary effort that is needed to fix the invalid tracks the following graph shows the dependency of the number of missed true tracks (true tracks that were not assigned to any valid estimated track) on the rate of adjustments needed to the best partially matching estimated tracks.

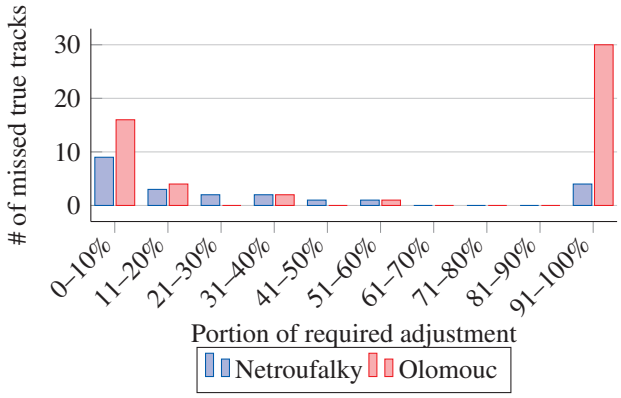


Figure 4. Histogram of missed true tracks, according to required adjustment to best matching fragments.

5. Conclusions

In this paper, we have presented a system for vehicles' trajectories extraction from aerial video data captured by a UAV. The most important part of the system is its tracking algorithm based upon set of intra-independent Bayesian bootstrap particle filters which were modified

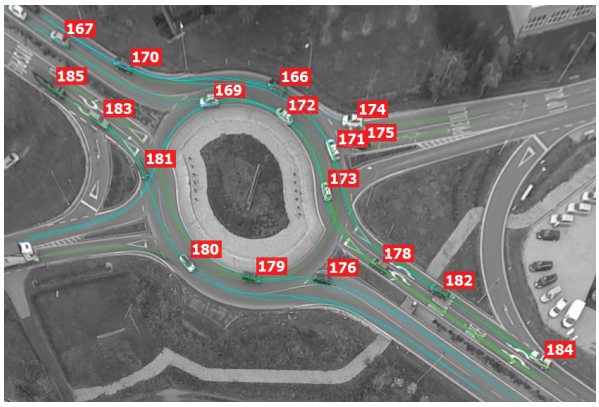


Figure 5. Output example of the presented system.

to deal with special caveats which are derived from the nature of vehicle tracking in aerial video data - huge data amount, low spatial resolution and temporal variance of camera position and tracked objects.

The system's accuracy and performance is at present stage not suitable for wide application, but we believe, that it can be improved further. Both the detection algorithm and tracking algorithm were implemented with plans of further improvements and possible specialisation. The detection cascades can be retrained for different object types and their performance improved by collecting much more training data. Alternatively, we consider change to deep learning methods for target detection, as they render to be superior when huge set of training data is provided [32]. Also, the tracking algorithm can be improved by introducing intra-dependency, driver model, target shape estimation, occlusion ordering and other usual approaches in multi-object tracking. Further discussion, improvements and tests will be part of my master thesis.

The potential of the system is also indicated by the fact, that it is already being used as assisting tool in process of intersection analysis and design at Institute of Road Structures under Faculty of Civil Engineering, Brno University of Technology.

Acknowledgement

This work was done as part of master thesis under supervision of Ing. Jaroslav Rozman, PhD.[1] and assistance of Ing. David Herman[2]. I would also like to thank Ing. Jiří Apeltauer[3] for providing the necessary video sequences for evaluation and development.

[1]Faculty of Information Technology, Brno University of Technology

[2]RCE Systems, s.r.o., Brno, Czech Republic

[3]Faculty of Civil Engineering, Brno University of Technology

References

[1] *Highway Capacity Manual: Practical Applications of Research*. U.S. Dept. of Commerce, Bureau of Public Roads, 2000.

[2] J.-N. Lee and K.-C. Kwak. A trends analysis of image processing in unmanned aerial vehicle. *International Journal of Computer, Information Science and Engineering*, 8(2):2 – 5, 2014.

[3] T.T. Nguyen, Grabner H., Gruber B., and Bischof H. On-line boosting for car detection from aerial images. In *IEEE International Conference on Reasearch, Innovation and Vision for the Future*, pages 87–95, 2007.

[4] H. Moon, R. Chellapa, and A. Rosenfeld. Performance analysis of a simple vehicle detection algorithm. *Image and Vision Computing*, 20:1 – 13, January 2002.

[5] T. Zhao and R. Nevatia. Car detection in low resolution aerial images. In *Eight IEEE International Conference on Computer Vision*, pages 710–717, Vancouver, 2001. IEEE.

[6] K. ZuWhan and J. Malik. Fast vehicle detection with probabilistic feature grouping and its application to vehicle tracking. In *Proceedings on Ninth IEEE International Conference on Computer Vision*, pages 524–531, October 2003.

[7] Y. Freund and R.E. Schapire. A short introduction to boosting. In *Journal of Japanese Society for Artificial Intelligence*, pages 771–780, 1999.

[8] K.P. Murphy. *Dynamic bayesian networks: representation, inference and learning*. PhD thesis, University of California, Berkeley, 2002.

[9] J. Gleason, A.V. Nefian, X. Bouyssounousse, T. Fong, and G. Bebis. Vehicle detection from aerial imagery. In *2011 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2065–2070, May 2011.

[10] T. Moranduzzo and F. Melgani. Automatic car counting method for unmanned aerial vehicle images. *IEEE Transactions on Geoscience and Remote Sensing*, 52(3):1635–1647, March 2014.

[11] G. Pacher, S. Kluckner, and H. Bischof. An improved car detection using street layer extraction. In *Proceedings of the 13th Computer Vision Winter Workshop*, Ljubljana, 2008.

[12] S. Tuermer, J. Leitloff, P. Reinartz, and U. Stilla. Automatic vehicle detection in aerial image sequences of urban areas using 3d hog features.

Photogrammetric Computer Vision and Image Analysis, 28:50–54, 2010.

[13] S. Tuermer, J. Leitloff, P. Reinartz, and U. Stilla. Motion component supported boosted classifier for car detection in aerial imagery. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Science*, 2011.

[14] J. Xiao, H. Cheng, H. Sawhney, and F. Han. Vehicle detection and tracking in wide field-of-view aerial video. In *2010 IEEE Conference on Computer Vision and Pattern Recognition*, pages 679–684, June 2010.

[15] R.A. Singer. Estimating optimal tracking filter performance for manned maneuvering targets. *IEEE Transactions on Aerospace and Electronic Systems*, AES-6(4):473–483, July 1970.

[16] D. Koller, J. Weber, and J. Malik. Robust multiple car tracking with occlusion reasoning. Technical Report UCB/CSD-93-780, EECS Department, University of California, Berkeley, Berkeley, November 1993.

[17] N. Obolensky. Kalman filtering methods for moving vehicle tracking. Master's thesis, University of Florida, 2002.

[18] R. Karlsson and F. Gustafsson. Monte carlo data association for multiple target tracking. In *Target Tracking: Algorithms and Applications*, volume 1, pages 13/1–13/5 vol.1, October 2001.

[19] O. Samuelsson. *Video Tracking Algorithm for Unmanned Aerial Vehicle Surveillance*. PhD thesis, KTH Royal Institute of Technology, School of Electrical Engineering, 2012. Master's Degree Project.

[20] R. Hess and A Fern. Discriminatively trained particle filters for complex multi-object tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 240–247, June 2009.

[21] Idrees H. Reilly V. and Shah M. Detection and tracking of large number of targets in wide area surveillance. In *11th European Conference on Computer Vision*, volume 6313, pages 186–199, Berlin, 2010. Springer-Verlag Berlin Heidelberg.

[22] J. Prokaj, X. Zhao, and G. Medioni. Tracking many vehicles in wide area aerial surveillance. In *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 37–43, June 2012.

[23] I. Saleemi and M. Shah. Multiframe many-many point correspondence for vehicle tracking in high density wide area aerial videos. *International Journal of Computer Vision*, 104(2):198–219, 2013.

[24] M.A. Fischler and R.C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communication ACM*, 24(6):381–395, June 1981.

[25] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. ORB: An Efficient Alternative to SIFT or SURF. In *International Conference on Computer Vision*, Barcelona, 2011. Willow Garage.

[26] Z. Zivkovic. Improved adaptive gaussian mixture model for background subtraction. In *Proceedings of the 17th International Conference on Pattern Recognition*, volume 2, pages 28–31 Vol.2, August 2004.

[27] S. Liao, X. Zhu, Z. Lei, L. Zhang, and S.-Z. Li. Learning multi-scale block local binary patterns for face recognition. In *Advances in Biometrics*, volume 4642 of *Lecture Notes in Computer Science*, pages 828–837. Springer Berlin Heidelberg, 2007.

[28] Y. Ju, H. Zhang, and Y. Xue. Research of Feature Selection and Comparison in AdaBoost based Object Detection System. Number 22, 2013.

[29] R. Danescu, F. Oniga, S. Nedevschi, and M. Meinecke. Tracking multiple objects using particle filters and digital elevation maps. In *Intelligent Vehicles Symposium, 2009 IEEE*, pages 88–93, June 2009.

[30] D. Schulz, W. Burgard, D. Fox, and A.B. Cremers. Tracking multiple moving targets with a mobile robot using particle filters and statistical data association. In *IEEE International Conference on Robotics and Automation.*, volume 2, pages 1665–1670 vol.2, 2001.

[31] A. A. Gorji, R. Tharmarasa, and T. Kirubarajan. Performance measures for multiple target tracking problems. In *Proceedings of the 14th International Conference on Information Fusion*, July 2011.

[32] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov. Scalable object detection using deep neural networks. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2155–2162. IEEE, 2014.

Appendix E

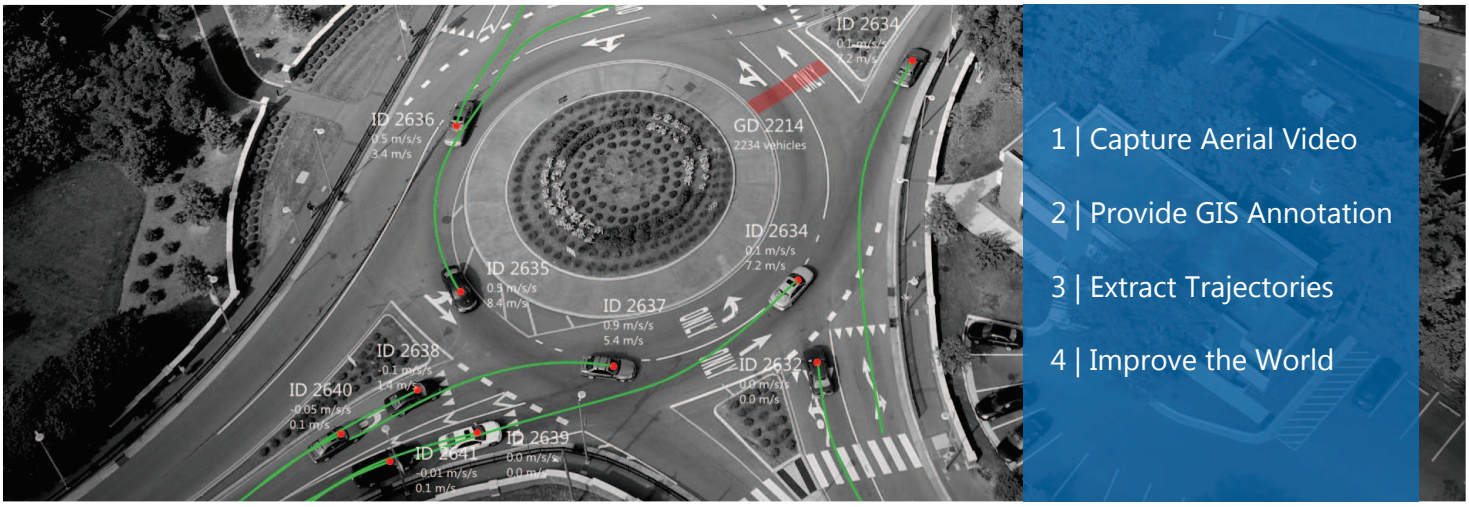
Poster

DATA FROM SKY

Automatic vehicle trajectory extraction from aerial video data

Adam Babinec

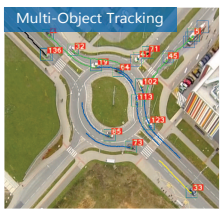
Faculty of Information Technology, Brno University of Technology, Czech Republic - xbabin02@stud.fit.vutbr.cz
RCE systems s.r.o., Svatopluka Čecha 1d, 612 00 Brno, Czech Republic - adam.babinec@rcesystems.cz

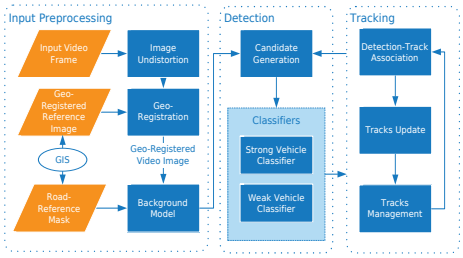


This poster presents a system for automatic vehicle trajectory extraction from aerial video data. The input video sequence can be captured by regular action camera mounted on a multicopter drone or balloon flying in altitudes from 40 to 200 m.

The geo-registration of video sequence is based upon transformation estimation of two ORB feature sets extracted from a reference image and video sequence frame. To provide the robustness of the estimation, RANSAC procedure is employed to guide the algorithm.

Vehicle detection candidates are produced by video sequence analysis in temporal dimension to detect moving objects. This is carried out by background subtraction algorithm using Gaussian Mixture Models. Its output is fused together with a road surface mask and currently tracked objects database to produce set of detection candidates.

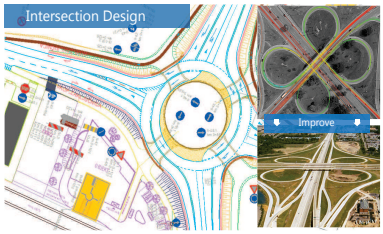


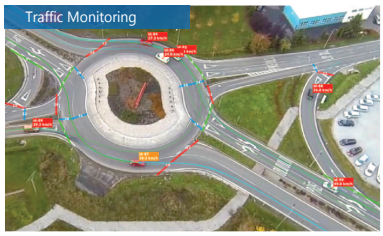


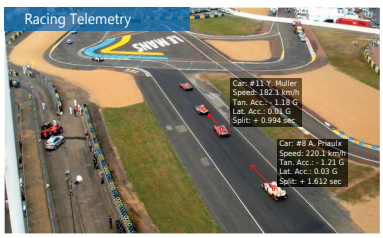
The detection candidates are analyzed by two vehicle classifiers - strong vehicle classifier which produces confident detection cues, and weak vehicle classifier which provides tracking attractors to aid multi-target tracking. The classifiers use Multi-Block Local Binary Pattern features and are constructed by AdaBoost algorithm on dataset of 20,000 hand annotated vehicles and road structures.

The tracking of vehicles is carried out in RGB+Edge image space using a set of independent Bayesian bootstrap particle filters, one for each target. The transition model of the particle filter is simple velocity model considering the vehicle position as integration of its velocity. Target model of the vehicle is represented by 32x32 RGB +Edge template with circular mask, which is extracted from the area initial vehicle detection. The plasticity of the model is achieved by 5% template update in case of overlapping strong detection and/or when the cues from the weak classifier are strong enough in the area of tracked vehicle. To prevent target swaps, the update is disabled when multiple targets overlap.

The evaluation of particle is based on its template distance to the target model and the attractor cues produced by the weak classifier. The initialisation of particle filter for moving object is guided by "falloff" algorithm, which at the beginning of the target tracking causes the position of the particles of given target to be more affected by random noise than their velocity, so the particles can slowly adapt to the target motion while elevating particles' velocity effect on their behaviour.







Partners:

www.datafromsky.com
info@datafromsky.com