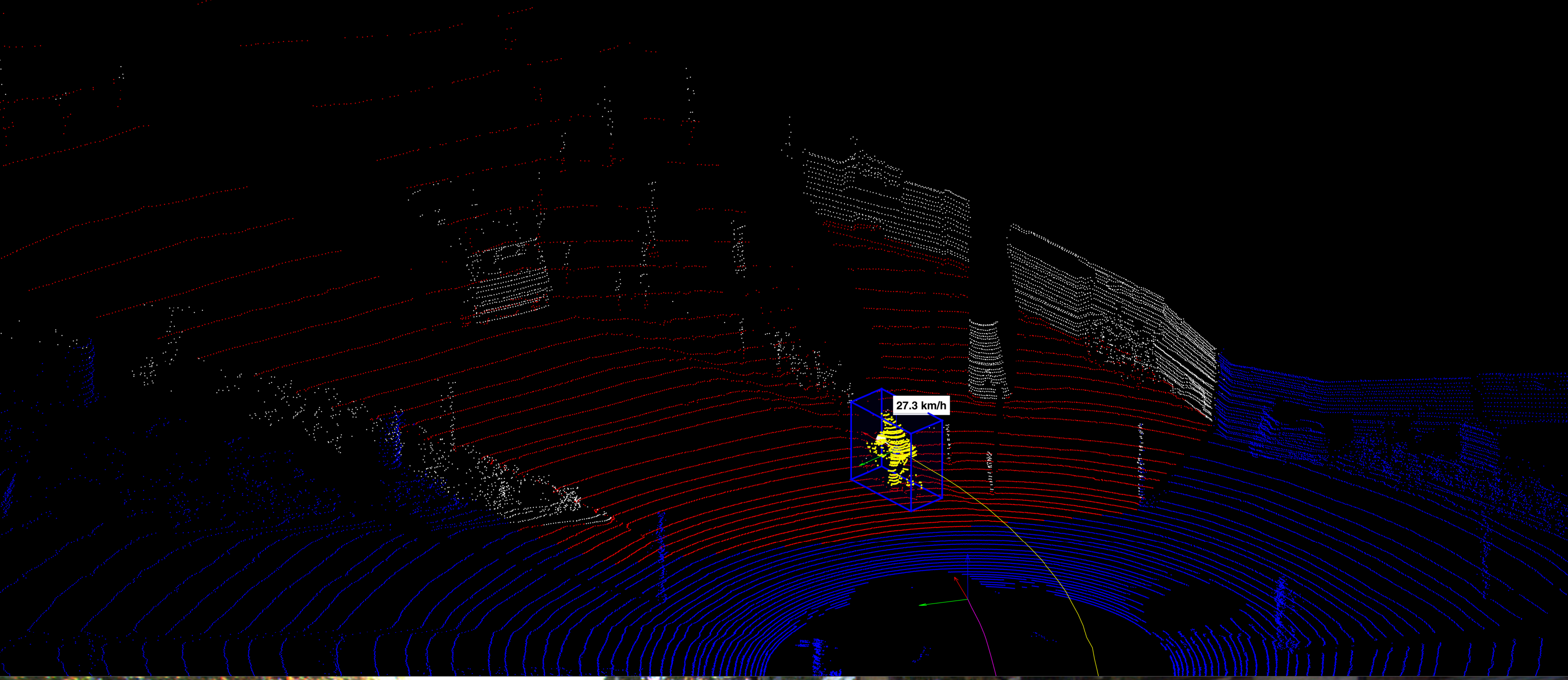
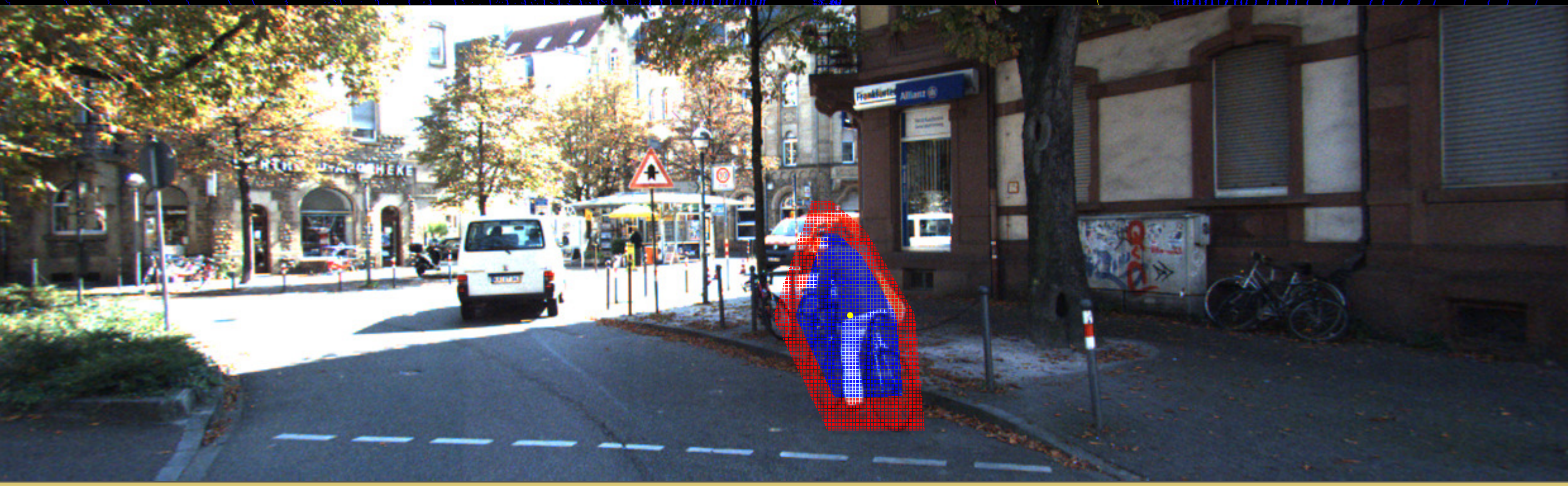




Pedro Miguel Oliveira Girão

3D Object Tracking Using RGB Camera and 3D-LiDAR Data

Dissertação submetida para a satisfação parcial dos requisitos do grau de Mestre em Engenharia Electrotécnica e de Computadores

Setembro de 2016

· U C ·

Universidade de Coimbra

· U C ·

FCTUC FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

3D Object Tracking Using RGB Camera and 3D-LiDAR Data

Pedro Miguel Oliveira Girão

Coimbra, Setembro 2016

3D Object Tracking Using RGB Camera and 3D-LiDAR Data

Supervisor:

Professor Paulo Peixoto

Co-Supervisor:

Professor Doutor Urbano Nunes

Jury:

Professor Doutor João Pedro de Almeida Barreto

Doutor Cristiano Premebida

Dissertation submitted in partial fulfillment for the degree of Master of Science in Electrical and Computer Engineering.

Coimbra, Setembro 2016

ACKNOWLEDGEMENTS

Em primeiro lugar, gostaria de demonstrar a minha gratidão ao Professor Paulo Peixoto por ter aceite o meu pedido para orientação, bem como pelo apoio e ajuda prestados ao longo da tese e por me ter apresentado à área da investigação académica (com todas as alegrias e frustrações inerentes). O encorajamento foi também uma constante ao longo destes meses, estando-lhe grato por isso.

Agradeço também ao Professor Doutor Urbano Nunes pelo *feedback* prestado nos papers técnicos submetidos no decorrer da tese.

I also want to give special thanks to Alireza Asvadi for all the knowledge, patience, insight and friendship he provided me throughout the entire process. His technical know-how and tireless guidance were key factors for the success of the thesis. By always pointing me in the right direction, never refusing any request for help, I am indebted to him.

Um abraço para o Pedro, Ricardo, Bruna e Zé pela companhia e boa disposição constante, pelos almoços e cafés, pela ajuda e pela partilha das "dores de crecimento" das dissertações. Reza a lenda que um dia ainda me vêm visitar à terra natal.

Aos meus pais agradeço ainda o apoio incondicional, não apenas nestes últimos meses mas em todo este percurso. Obrigado por acreditarem em mim e me fazerem acreditar em mim próprio, por me fazerem crescer enquanto pessoa e por estarem sempre presentes.

Por fim, o mais especial dos agradecimentos vai para a Susana. O encorajamento, paciência e acima de tudo, amor nos últimos meses tornaram esta dissertação possível. O percurso da tese foi difícil, em que nem sempre tudo correu de feição, mas o apoio mútuo fez com que tenhamos conseguido ultrapassar mais esta etapa (sem enlouquecer).

A todos, o meu obrigado.

RESUMO

Ao longo destes últimos anos, o campo da segurança rodoviária na indústria automóvel tem sido foco de muita investigação e desenvolvimento. A evolução verificada neste campo fez com que as principais empresas do ramo tenham começado a desenvolver as suas próprias soluções de segurança de modo a obter veículos mais seguros. Um dos resultados desta investigação foi o aparecimento de sistemas de assistência à condução autónoma e veículos de condução automatizada. Estes veículos dependem de sensores precisos, montados no próprio veículo, para compreender os ambientes que os rodeiam. Assim, apenas poderão ser obtidos veículos verdadeiramente autónomos quando estes dependerem apenas da informação sensorial.

No entanto, é necessário aplicar pós-processamento aos dados oriundos dos sensores de modo a que o veículo detete obstáculos e objetos enquanto navega. Entre outros, estes algoritmos permitem a extração da localização, forma e orientação do objeto. A navegação em ambientes urbanos continua a ser um problema sem soluções aceites de modo unânime, apesar das muitas propostas apresentadas. Isto prende-se principalmente com o fato dos cenários de tráfego urbano apresentarem complexidade superior a outros cenários. Os sensores existentes atualmente apresentam ainda capacidades limitadas quando usados individualmente.

O *pipeline* da perceção é um módulo crítico para o bom funcionamento destes veículos. Em particular, o seguimento de objetos num ambiente 3D dinâmico é uma das componentes chave deste *pipeline*. Ao seguir um objeto, informação sobre a sua localização, forma, orientação e até velocidade podem ser obtidas. Previsões sobre os estados futuros do objeto seguido também podem ser utilizadas para que o veículo possa planear ações futuras.

Na presente dissertação é apresentada uma abordagem para o seguimento *online* de objetos, que recorre a informação de uma imagem 2D RGB obtida por uma câmara a cores e uma nuvem de pontos 3D capturada por um sensor LiDAR, de modo a detetar e seguir um objeto em cada novo *scan*. O objeto alvo está inicialmente definido. Foi considerado um sistema de navegação inercial para a auto-localização do veículo. A integração de dados 2D e 3D pode provar-se benéfica para o seguimento de objetos, como é demonstrado pela abordagem apresentada. São usados algoritmos *mean-shift* nos espaços 2D e 3D em paralelo de modo a obter as novas localizações do objeto, e filtros Bayesianos (filtros de Kalman) para fazer a fusão da informação e, de seguida, para o próprio seguimento do objeto no espaço 3D. Este método calcula então estimativas para a localização, orientação 2D e velocidade instantânea do objeto, bem como uma estimativa da próxima localização do objeto.

O método desenvolvido foi sujeito a uma série de testes, estando os resultados qualitativos e quantitativos obtidos apresentados no documento. Estes resultados mostram que o método é capaz de estimar a localização, orientação e velocidade do objeto com boa precisão. Foi feita ainda uma comparação com dois métodos de referência (*baseline*). Os testes utilizam informação *ground truth* proveniente da base de dados KITTI para o propósito de avaliação. Tendo como base os testes efetuados, é proposto um conjunto de métricas (*benchmark*) para facilitar a avaliação de métodos de modelação da aparência de objetos. Este *benchmark* consiste em cinquenta sequências compostas pela trajetória completa (e ininterrupta) do objeto em particular, sendo categorizadas com base em quatro fatores desafiantes.

Palavras Chave

Seguimento de objetos em 3D, fusão sensorial, condução autónoma, mean-shift, filtros de Kalman

ABSTRACT

The field of vehicle safety in the automotive industry has been the focus of much research and development in recent years. This evolution in the field has made it possible for major players in the automotive business to start developing their own active safety solutions to achieve safer cars. One of the outcomes of these efforts has been the development of autonomous driving assistance systems and highly automated vehicles. These vehicles rely on accurate on-board sensor technologies to perceive their surrounding environments. Truly autonomous vehicles will only be a reality when they can reliably interpret the environment based on sensory data alone.

However, some post processing needs to be applied to the raw data given by the sensors, so that the vehicle can perceive obstacles and objects as it navigates. Such algorithms allow, among other things, the extraction of the location, shape and orientation of object. The navigation in urban environments is still an open problem, despite the many approaches presented. This is mainly due to the fact that urban traffic situations offer additional complexity. In addition, current on-board sensors also have limited capabilities when used individually.

The perception pipeline is thus a critical module of these vehicles. In particular, object tracking in a dynamic 3D environment is one of the key components of this pipeline. By tracking an object, information about location, shape, orientation and even velocity of that object can be obtained. A prediction of the future states of the tracked objects can also be obtained allowing for the planning of future actions taken by the vehicle.

In the present thesis, an online object tracking approach has been developed that used information from both 2D RGB images obtained from a colour camera and 3D point clouds captured from a LiDAR sensor to detect and track an initially defined target object in every new scan. An inertial navigation system was also considered for vehicle self-localization. Integration of 2D and 3D data can provide some benefits, as demonstrated by this approach. The tracker uses two parallel mode-seeking (mean-shift) algorithms in the 2D and 3D domains to obtain new object locations and Bayesian (Kalman) filters to fuse the output information and afterwards track the object in the 3D space. Our approach outputs the target object location, 2D orientation, and instant velocity estimates, as well as a prediction about the next location of the object.

The tracker was subject to a series of tests. The quantitative and qualitative results obtained are presented in the document. These results show that the proposed tracker is able to achieve good accuracy for location, orientation and velocity estimates. A comparison with two baseline 3D object trackers is also presented. The test cases used ground truth information from the KITTI database for evaluation purpose. A benchmark based on the KITTI database is also proposed in order to make the evaluation of object appearance modelling and tracking methods easier. This benchmark features 50 sequences labelled with object tracking ground-truth, categorized according to four different challenging factors.

Keywords

3D object tracking, sensor fusion, autonomous driving, mean-shift, Kalman filters

“I would like to die on Mars. Just not on impact."

— Elon Musk

CONTENTS

LIST OF ACRONYMS

3D BB	3D Bounding Box
ABS	Antilock Braking System
ADAS	Autonomous Driver Assistance System
CA KF	Constant Acceleration Kalman Filter
CV KF	Constant Velocity Kalman Filter
DA	Data Association
DARPA	Defense Advanced Research Projects Agency
DATMO	Detection and Tracking of Moving Objects
DEM	Digital Elevation Map
DGPS	Differential Global Positioning System
EKF	Extended Kalman Filter
ESC	Electronic Stability Control
FOV	Field of View
FPS	Frames Per Second
GNN	Generalized Neural Network
GNSS	Global Navigation Satellite System

GPS	Global Positioning System
HDR	High Dynamic Range
ICP	Iterative Closest Point
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
IR	Infrared
KDE	Kernel Density Estimation
KF	Kalman Filter
Laser	Light Amplification by Stimulated Emission of Radiation
LiDAR	Light Detection And Ranging
LLR	Log-Likelihood Ratio
MHT	Multiple Hypothesis Tracking
MS	Mean-Shift
PCD	Point Cloud
PDF	Probability Density Function
RADAR	Radio Detection And Ranging
RANSAC	Random Sample Consensus
RGB	Red Green Blue
ROI	Region of Interest
SONAR	Sound Navigation And Ranging
SNR	Signal-to-Noise Ratio
ToF	Time of Flight
UV	Ultraviolet

LIST OF FIGURES

LIST OF TABLES

CHAPTER

1 INTRODUCTION

Contents

The present document acts as the author's MSc thesis dissertation. The carried out work falls within the domain of perception systems for intelligent vehicles, applied to the particular case of single object tracking in 3D. In this chapter, the motivation behind the developed work is presented, and the context explained by stating some studies and statistics that show the importance of the perception pipeline in intelligent vehicles currently being developed. The problem at hand is then defined. Additionally, an overview of the work's key points and ideas is provided, along with the flow of information in the proposed work. Afterwards, the objectives and main contributions are presented, as well as the structure of the remainder of the document.

1.1 Motivation and Context

Injuries resulting from road transportation (or traffic) accidents claim an estimated 1.25 million lives globally each year (being the leading cause of death from young people aged from 15 to 29) [1, 2]. A recent report concerning just data from European countries states that roughly 85000 people have perished due to road traffic injuries in 2013 and, statistically speaking, for every person that dies from a road accident at least 23 people suffer non-fatal injuries that require urgent hospital care [3]. Furthermore, pedestrians, cyclists and motorcyclists compose near 40% of the total number of deaths related to these road accidents. In turn, a statistical summary released by the United States Department of Transportation projected that in 2015 the total numbers of vehicle traffic fatalities grew in comparison to previous years by almost 8% [4].

Human errors remain the main reason behind road accidents, namely through distracted driving, with all the factors typically involved: talking/texting on the phone, eating, reading, falling asleep, tuning the radio or even just talking to someone inside the car [5, 6]. According to the same study, the second and third most common causes of traffic accidents were speeding and drunk driving, which highlights the drivers responsibility and ability to avoid these unfortunate events (backed by the numbers in [7] as aforementioned, where 94% of accidents in the study were due to human errors).

As a direct consequence of this reality, safety protocols and policies have been implemented over the years, from building safer roads to enforcing the usage of a seatbelt. These safety systems can be categorized in two groups: protective and preventive [8].

Protective safety systems include the most common safety measures implemented in the last half century. Its aim is to disperse the kinetic energy in an accident in order to efficiently protect both the driver and other possible occupants of the vehicles. These mechanisms include bumpers, seat belts and air-bags.

On the other hand, preventive (or *active*) safety systems are meant to help drivers safely guide the vehicle while avoiding, mitigating or possibly even preventing accidents along the way.

Some examples of these systems are *Antilock Braking System* (ABS) and *Electronic Stability Control* (ESC). Through the monitoring and analysis of traffic scenarios surrounding the vehicle, preventive safety systems can act as a support to the driver. An additional category called integrated safety considers information that is perceived (preventive) to make better use of the protective system (seat readjustment before crash, earlier deployment of air bags, etc.).

With the goal of reducing the aforementioned number of deaths and injuries from vehicle related accidents, a lot of research has been put into preventive safety systems. One of the main ideas behind this research is the change in the driver's stance, from a direct actuator towards a spectator role, *i.e.*, a supervisor of the vehicle, only acting when needed [9]. A direct outcome of such efforts has been the creation of intelligent vehicles, capable of perceiving the environment that surrounds them and able to take appropriate decisions regarding the navigation task previously determined by the driver, being therefore safer for both the occupants of the vehicle and other agents on the road like other drivers and pedestrians.

In recent years, an increasing number of innovative technological approaches have been suggested to improve the behaviour of intelligent vehicles (otherwise referred to as *Autonomous Driver Assistance System* (ADAS)). Research in areas like computer vision and machine learning have proved the potential of autonomous driving. Nonetheless, acceptance by both governmental laws and public opinion remains divided, with several legal and policy situations being currently under evaluation [10].

Since the decision making process involved in a typical ADASs must rely on its perception of the surrounding traffic condition, nature and man-built structures and even road or weather conditions, the perception pipeline becomes of the utmost importance. In order to achieve this situational awareness, the intelligent vehicle in question (or *ego vehicle*) needs to know the location of objects of interest on a known spatial reference system (either local or absolute/world). This goal can be achieved by using on-board sensors relying on different technologies, such as vision (mono and stereo cameras) and range sensors (*Radio Detection And Ranging* (RADAR), *Sound Navigation And Ranging* (SONAR), *Light Detection And Ranging* (LiDAR)). In addition to these sensors, an *Inertial Navigation System* (INS) can also be used to directly obtain the ego vehicle's location.

Even with the successful autonomous driving tests reported on highways and urban scenarios in recent years, there are still open issues regarding the reliable perception and navigation in unconstrained environments mainly due to the limited capabilities of current sensors as well as the image processing algorithms used for the purpose of scene understanding [11].

1.2 Problem Formulation

ADASs obey to the robotic paradigms in the sense that they integrate three robot primitives: sense, plan and act [12]. The system has to gather the information (sense), evaluate it (plan) and then act accordingly. The perception pipeline belongs to the "sense" primitive, enabling a vehicle to comprehend the surrounding environment through its sensory input. Usually, perception modules are composed of, among others, an initial object detection (and possibly segmentation) module, as well as an object tracking module.

For the purpose of this thesis, we will just focus on the particular problem of 3D single object tracking. To do so, sensor data is obtained from a 3D *Light Amplification by Stimulated Emission of Radiation* (Laser) scanner and a stereo vision system for environment perception. In addition, an INS (GPS/IMU) is used to know the ego vehicle's location. For the current approach it is assumed that the initial position and characteristics of the target object are known.

Usually, perception modules are also composed of, among others, an initial object detection (and possibly segmentation) sub-module; however, for the current purpose it is assumed that the initial position and characteristics of the target object are known.

The focus of the present thesis is therefore on the modelling and tracking of a generic object in 3D, by fusion of sensor data (LiDAR and monocular camera) to enable better results when comparing with single-sensor approaches. Using an INS allows for better velocity estimates for object tracking. Through usage of *Kalman Filter* (KF) theory, the system can also predict the object location in the 3D world coordinate system in the next time-step.

1.3 System Overview

A generic overview of the proposed 3D object tracker is shown in Figure 1.1. It is assumed that the initial characteristics of the object represented in the form of a *3D Bounding Box* (3D BB) are known. A 3D BB is considered to describe the object's location, width, height, length and pose. As aforementioned, this information is normally obtained by object [13] or motion detectors [14].

The initial 3D BB is obtained in the 3D *Point Cloud* (PCD) from ground truth information. Points inside the 3D BB are projected into the image simultaneously acquired from the 2D *Red Green Blue* (RGB) and the convex hull of the projected points is calculated to define the object in the 2D domain. For the following scans, the object is located in the current PCD and image by applying a mode-seeking *Mean-Shift* (MS) algorithm in each domain. These locations are the tracker's initial estimates for the 3D location of the target object. Information obtained from both the 3D and 2D domains is fused, and another Bayesian filter effectively tracks the object in the 3D world space. The computed object 3D BB is used as the reference object information for the next scan. The overview of the process can be seen in Figure 1.1.

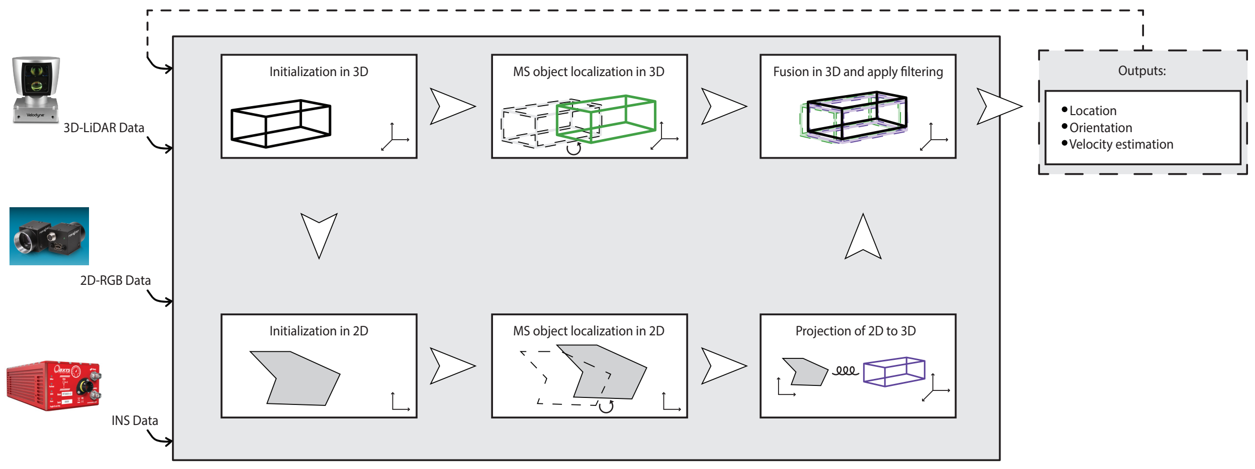


Figure 1.1: *High-level overview of the proposed 3D object tracker. A more detailed diagram and extensive description of the system behaviour can be found in Chapter 3.*

1.4 Objectives

As stated before, the main goal of this thesis is to develop an object tracker capable of tracking a single object in 3D as it moves around a scene, using information from several onboard sensor technologies. The tracker will thus track and maintain updated information about the object in both 3D and 2D spaces. The proposed approach should provide a stronger representation and understanding of the ego vehicle surroundings by handling and fusing the information perceived by each sensor. To summarize, the proposed objectives are:

- Provide a robust 2D/3D fusion-based 3D object tracking approach;
- Provide location, orientation (2D angle) and velocity estimations of the tracked object on a 3D coordinate space related to world coordinates (*i.e.* real world coordinates);
- Report both quantitative and qualitative evaluation of the proposed method;
- Create a benchmark for object appearance modelling evaluation;

1.5 Main Contributions

In this thesis, an object tracker that makes use of combined information obtained from a 3D-LiDAR and a 2D-RGB camera is proposed. The main contribution of this work is to provide a robust 2D/3D fusion-based 3D object tracking approach. Since it is hard to properly evaluate and compare most of the currently available approaches for object tracking in the literature (as these are subject to different parameters and constraints and may come from different theoretical backgrounds) another contribution of this thesis is the proposal of a benchmark to enable an easier comparison and quantitative evaluation of different 3D object trackers in driving environments (particularly those based on object appearance modelling).

1.6 Outline

The present document is organized in five chapters, beginning with the current introduction, where the problem is contextualized and a solution is proposed to mitigate it.

In Chapter 2 modern sensors and object tracking state of the art are discussed, summarizing some of the most important aspects of recent research and how they influenced the direction followed by the work presented in this document.

Chapter 3 will then focus on the detailed description of the proposed approach, showing how it handles incoming data from different sensors and how it fuses the sensory data to provide meaningful information.

Afterwards, quantitative and qualitative experimental results of the proposed tracker are presented in Chapter 4. The used dataset is also described, with some notes on how the information was extracted and how it can be used to test the developed object tracker.

Chapter 5 is reserved for the presentation of conclusions drawn from the presented work, as well as pointing towards future work that could be done to extend/improve the developed framework.

CHAPTER

2 STATE OF THE ART

Contents

Autonomous cars and ADASs need to be able to understand their surrounding environments, as aforementioned. In this sense, the quality of the perception pipeline is critical for the correct performance of these systems. This perception pipeline depends directly on the quality of perception sensors and algorithms [15]. Object tracking stands as one of the main components of the perception process.

In this chapter a discussion of current literature is provided, regarding sensing systems and object tracker approaches. Taking into consideration the number of considered approaches and studies, a table with the most salient aspects of each work will be provided.

2.1 Sensors

In order to detect and track on-road agents (such as vehicles, cyclists and pedestrians) a number of different sensor technologies can be used. Typically it is necessary to measure object properties such as position, orientation (or pose), distance to ego vehicle, speed and acceleration, among others. Algorithms that process information provided by these sensors are of utmost importance for ADASs, with the action pipeline being dependent of the output of the perception phase. The use of these technologies recently showed that there is an enormous potential for saving lives, like in the case of a Tesla vehicle owner claiming that the car alone prevented him from hitting a pedestrian during the night in low visibility conditions [16]. However, this comes in contrast to news that another vehicle by the same manufacturer was involved in a fatal death in Florida due to cameras not being able to differentiate "*the white side of* [a] *tractor trailer against a brightly lit sky.*", showing that these systems are still not perfected [17].

2.1.1 Vision Sensors

As with the retinas in human eyes, colour cameras are able to capture the colour and resolution of a scene with varying amounts of detail. Over the past decade, the improvement on vision-based perception algorithms has been noticeable; detection and tracking of moving objects can thus be obtained by equipping ADASs with on-board sensors such as monocular or stereo cameras.

Vision sensors can be categorized as passive sensors in the sense that collected information is resultant from the reception of non-emitted signals, as there is no electromagnetic energy emission but rather the measurement of light in the perceived environment (*i.e.* image capturing) [18]. Therefore, it is common practice for vision-based perception systems to install one or more cameras in the vehicle, either inside (close to rear mirror) or outside.

With the on-going improvement of vision sensors, the associated costs of camera acquisition have dropped, being one of the most interesting aspects of this approach; given the usually low *Field of View* (FOV) of such cameras, the installation of multiple sensors can be used to obtain a full

360° view of the vehicle surroundings and thus a more rich description of the latter (in comparison to those obtained from active sensors).

(a) *PointGrey Flea3* **(b)** *PointGrey Bumblebee2*

Figure 2.1: *Examples of cameras in the market. The first is a monocular colour camera [19] while the second consists on a stereo vision setup [20].*

Stereo vision systems make use of the presence of multiple cameras in order to obtain relevant range data and, in addition to the low cost, present low energy consumption whilst outputting meaningful colour data and accurate depth information. It should be noted that the performance of stereo vision approaches tends to deteriorate in regard to objects located far from the ego vehicle, which results in losing losing fidelity proportionately with the distance to scene objects. In addition to this downside, detection and tracking approaches based on monocular and stereo cameras are also directly affected by changes in lighting and weather conditions (such as fog or snow) [21] or time of the day. Some approaches try to solve these problems by utilizing specific sensors to deal with these situations. For example, Sun et al. [22] showed that the usage of *High Dynamic Range* (HDR) or low-light cameras made it possible to employ the same detection and recognition models to both day and night time.

Stereo vision-based sensors tend to produce very dense point clouds (seeing as the basis for their generation are rich images as opposed to active sensors); however these point clouds tend to be a little noisy. Some problems with stereo matching algorithms (and thus with generated stereo vision point clouds) are mainly due to sensor noise distorting images (particularly problematic in poorly textured regions due to low *Signal-to-Noise Ratio* (SNR)), lack of correspondence between pixels of half occluded regions in the two images (incorrect or no matching) and the fact that the constant brightness assumption is not commonly satisfied in practice [23].

2.1.2 Range Sensors

As opposed to vision-based sensors, range sensors are considered active as they receive and measure signals transmitted by them that are reflected by the surrounding objects and/or scene. RADAR, SONAR and LiDAR sensors fall within this category; LiDAR systems transmit and receive Ultraviolet (UV), Infrared (IR) and visible waves of the electromagnetic spectrum and, through *Time of Flight* (ToF) theory, are received and interpreted as a function of time. Paired with the knowledge of the speed of light, the systems then calculate the distance travelled by the emitted particles (forth and back).

In direct comparison with passive sensors, some active sensors like LiDAR are able to provide a 360° view of the area around the ego vehicle by using just one sensor. The output of these sensors comes in the form of a dense point cloud (very dense for objects or parts of the surrounding scene closer to ego vehicle, but sparser as distance to objects increases).

(a) *BOSCH LRR3* **(b)** *SICK LMS 210* **(c)** *HOKUYO URG-04LX* **(d)** *Velodyne HDL-64E*

Figure 2.2: *Examples of range sensors in the market. The first sensor is based on RADAR technology [24], whilst the rest are based on LiDAR technology [25–27].*

Active sensors provide a viable option for real-time detection applications, countering some of the vision-based problems (robustness under different weather conditions, for example) and measuring object characteristics such as location with little computational power needed [21]. Among themselves, RADAR presents more reliability than others sensors for greater distances, albeit the presence of less expensive LiDAR sensors in the market (but with lower range as well). In addition, most LiDAR sensors are cheaper and are easier to apply than RADAR. The latter sensors also retrieve less information than Laser based sensors and are more prone to misreadings, given the fact that environments might be dynamic and/or noisy and that there might be road traffic as well.

3D Laser based technology has gained attention in the past few years, being the main contenders in challenges such as *Defense Advanced Research Projects Agency* (DARPA) Urban Challenge. In 2007, most approaches were based on high-end LiDAR systems, such as the works presented by Montemerlo et al. [28] or Kammel et al. [29], with the latter becoming the working basis for the KITTI Vision Benchmark Suite [30].

Some of the disadvantages of LiDAR systems are that no colour data is directly obtained, and reflective and foggy environments tend to distort the results. In addition, darker coloured objects have lower reflection values since they absorb light, meaning the sensor could not get a return from the emitted Laser. A LiDAR sensor tends to receive less information in these cases; studies such as Petrovskaya et al. [31] show that looking at the absent data (no points being detected in a range of vertical angles in a given direction) can represent space that is likely occupied by a black (or darker) objects, since otherwise the emitted rays would have hit an obstacle or the ground.

Velodyne [32] has recently created a smaller 3D LiDAR sensor, the Velodyne HDL-64E. Details about the sensor can be found in Chapter 4.1. This sensor has become an interesting option for obstacle detection and navigation in urban and highway areas, being used in setups such as KITTI [30] and Google's Self-Driving Car Project (in the latter, as their main sensor) [33].

2.1.3 Sensor Fusion

Taking into account the pros and cons of passive and active sensors, multiple sensor approaches have been studied with the goal of yielding better results (hence safer ADASs) than using a single sensor. Sensor fusion is achieved by combining data from several sensors to overcome the deficiencies of individual usage. This can either be done to achieve simultaneous data capture for detection and tracking purposes, with each sensor validating the results produced by the others, or having one sensor detect and track while other sensors validate the results.

With prices for LiDAR systems getting lower, fusion of vision and LiDAR systems has been the focus of recent research. Premebida et al. [34] initially used LiDAR data for detection and tracking (to obtain a reliable object detection), but later in the process simultaneous accessed both LiDAR and vision sensory data along with a probability model for object classification.

Zhao et al. [35] presented an approach which performed scene parsing and data fusion for a 3D LiDAR scanner and a monocular video camera. Although the focus of the approach was not based on object tracking, one of the conclusion of the work was that fused results provided more reliability than those provided by individual sensors.

2.1.4 Point Clouds

A common representation for the surrounding scene when captured through LiDAR based sensors (or stereo vision setups after processing) is through a point cloud. A point cloud is thus a set of points in a given 3D coordinate system which are intended to represent the external surfaces (physical world surfaces) of surrounding objects. Some examples of points clouds obtained through two different technologies can be seen in Figures 2.3 and 2.4.

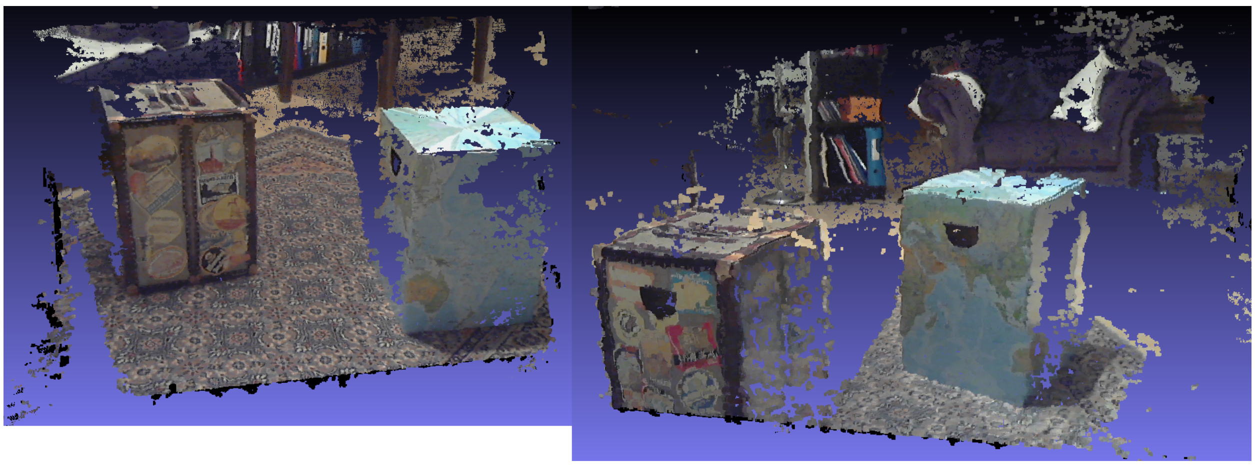


Figure 2.3: *Point clouds produced by stereo vision setup [36].*

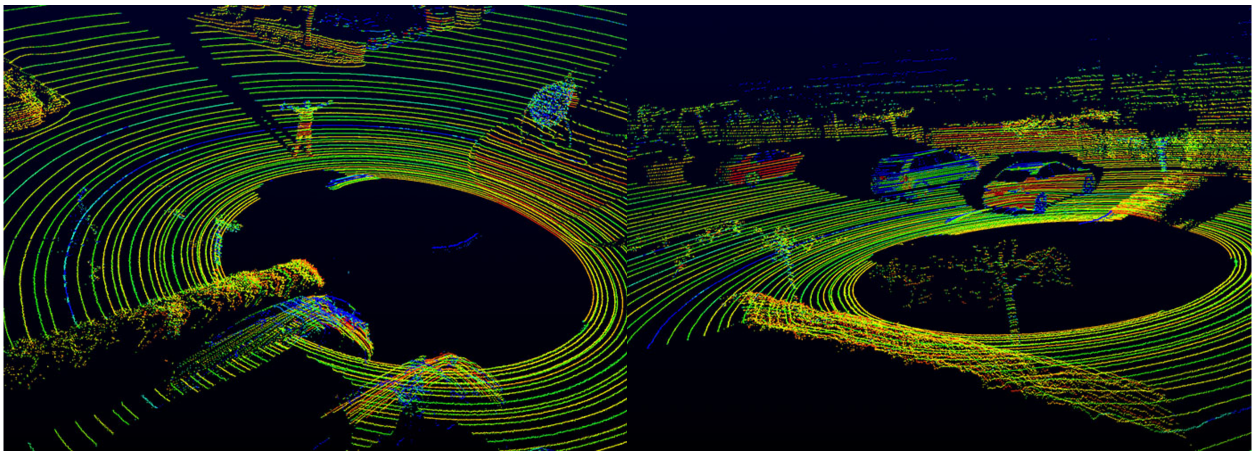


Figure 2.4: *Point clouds produced by Velodyne HDL-64E [27]. Colour is displayed for better representation as LiDAR based sensors have no colour information directly available.*

2.1.5 Summary (Sensors)

In this section a summary of sensor technologies is presented in Table 2.1. Here, a concise comparison between the considered sensor types is done in the shape of a brief description of the pros and cons of each sensor technology as a whole without looking into particular devices.

Type	*Energy*	*Data*	*Cost*	*Advantages*	*Disadvantages*
LiDAR	Laser	Distance (m)	Medium / High	• Dense high resolution 3D data in form of PCD; • Very precise; • Small beam-width; • Fast acquisition and processing rates; • Reflectance information can be useful for object recognition purposes;	• Sensible to outdoor conditions (refraction of laser might occur); • More costly than other sensors; • Higher power consumption; • Sensor made of moving parts; • No colour information;
RADAR	Radio wave	Distance (m)	Medium	• Low sensibility to both outdoor conditions and time of day; • Precise; • Long range; • Measurements done with less computational resources; • Good for environments with reflections;	• Still higher power consumption than vision sensors; • Larger beam-width; • Lower resolution than LiDAR; • Bigger sensor size than cameras;
Vision	None	Image (px)	Low	• Low cost with low installation/maintenance costs; • Meaningful information in captured image; • Non-intrusive data acquisition;	• Image quality highly dependant on outdoor conditions; • Heavy computational power to process images; • Low performance on texture-less environments (stereo vision);
Sensor Fusion	Depends on the sensors	Depends on the sensors	Depends on the sensors	• Increases system reliability; • More diverse information captured; • Compensation of individual sensor shortcoming;	• Algorithms to properly fuse informations; • Costs spread by several sensors;

Table 2.1: *Summary of discussed sensor technologies.*

Additionally, a literature study conducted in [37] further details characteristics of individual sensors used in ADASs and highlights the importance of combining different sensors in order to achieve better results for several tasks, including object tracking.

2.2 Object Trackers

After having sensor data available, some processing needs to be applied in order to extract the relevant information for tracking purposes. A very basic definition of object tracking would consist on estimating the trajectory of a target object as it moves around a scene (in an image plane or 3D space), inferring about its motion in a sequence of scans. Another definition (in the image domain) was provided by Smeulders et. al [38] as "*tracking* (being) *the analysis of video sequences for the purpose of establishing the location of the target over a sequence of frames (time)*".

The tracker thus labels an object in different scans, being able to provide information such as location, orientation, velocity and shape. As mentioned in Chapter 1, visual tracking is a fundamental task in the perception pipeline of intelligent vehicles and ADASs and has therefore been well studied in the last decades, especially in the image domain.

Even with the high number of proposed approaches to handle visual tracking, this topic still remains a big challenge, with the majority of difficulties arising from changes in object motion, perceived appearance, occlusion and movement of the capturing camera [39]. Given the high number of challenging variables, most approaches tend to provide robust solutions to only a given subset of problems.

There has been extensive research regarding object tracking in the image domain (with important surveys provided in [38–43], who considered the usage of monocular cameras). With new and improved stereo matching algorithms, along with the appearance of 3D sensing technologies as discussed in the previous section, object tracking in 3D environments became a viable alternative.

Usually, object trackers are composed by three major modules: defining the representation of the target object, searching for that target object in the scene and updating the correspondent object model. In order to be able to update object information (namely its model representation) Yang et. al [39] have proposed the categorization of object tracking approaches into 2 groups: generative and discriminative approaches.

The first category consists of methods tuned to learn the appearance of an object (later this appearance model will be used to search for the region that correspond to the next location of the object). The second category focuses on methods otherwise designated as "*tracking-by-detection*" (such as the approach considered in [11]) where the object identified in the first scan is described by a set of features, the background by another set of distinguishing features, and a binary classifier is used to separate them (updating the appearance model afterwards). Generative approaches are considered class-independent and are suitable for usage with a wider range of object classes.

One example of scene understanding was developed by Geiger et al. [11], where the proposed method estimates the layout of urban intersections based on a stereo vision setup alone. They required no previous knowledge, inferring needed information from several visual features describing the static environment and objects motion. One conclusion of this method was that the particular cases of occupancy grids, vehicle tracklets and 3D scene flow proved to be the most meaningful cues for that purpose. Since visual information is also subject to some ambiguities, a probabilistic model to describe the appearance of the proposed features was implemented, which in turn also helped to improve the performance of state-of-the-art object detectors (in both detection and orientation estimation).

With the focus of the present thesis being on 3D object tracking algorithms a more detailed review of current correspondent literature will thus be presented in the next section.

2.2.1 3D / Fusion Trackers

Tracking theory in the current automotive research has been subject to numerous changes, with new dynamic object tracking methods using LiDAR data becoming more common.

Detection and Tracking of Moving Objects (DATMO) (an expression originally coined by [44]) in dynamic environments has also been the subject of extensive research in a near past. Generative object tracking methods attracted more attention for object tracking in 3D environments. The considered object model is often updated online to adapt to appearance variation, as mentioned.

A comparative study on tracking modules (using LiDAR) was developed by Morton et al. [45]. The focus of this study was on object representation throughout the tracking process. As the authors noted, only the case of pedestrian tracking was considered. The considered baseline method was a centroid (or centre of gravity) representation for an object. This baseline was compared against 3D appearance as the object representation. In addition, and to verify the effect filtering had in object tracking (if any), the authors used either no filter or a *Constant Velocity Kalman Filter* (CV KF). To associate new measurements to already existing tracklets, the authors relied on the Hungarian method for centroid association and *Iterative Closest Point* (ICP) for 3D appearance representation association, respectively. According to their results, the best object representation from those considered in the study was the simplest (centroid) representation, as it provided the lowest location errors when tracking the pedestrians. It was additionally seen that applying a KF yielded better results than no filtering at all (the error in speed measurements was significantly higher when no filtering was considered).

Held et. al [46] presented a method that combined LiDAR and data from images to obtain better object velocity estimates, whether moving or stationary. In this approach, fusion was done at an earlier stage since they combined a 3D PCD with the 2D image captured by the camera (using bilinear interpolation for estimating the 3D location of each pixel) to build an upsampled coloured 3D PCD. For tracking purposes, a colour-augmented search algorithm was used for aligning successive PCDs, effectively tracking the object. A direct outcome of their research was that using an accumulated (and reconstructed) dense model of the object yields better velocity estimates, since the location error is also lower. Two versions of the approach are provided, with one of them suited for offline behavior modelling and the other one suited for real-time tracking (although less accurate). Like in the approach proposed on this thesis, [46] assumes that initial information about the object (such as position) is given. Contrary to Held's early fusion proposal, a later fusion is considered in our proposed approach, where the object is detected and localized in the raw 3D PCD and in the 2D image separately (with some connected data) and later the 2D/3D object locations are fused and tracked.

Based on a different idea, Dewan et al. [47] developed an approach for object detection and tracking that required no information about the object model nor any other source of previous information. Differently from common model-free approaches who rely on detecting dynamic objects (*i.e.* based on perceived changes in the environment), distinct objects were segmented using motion cues. In this sense, both static structures and dynamic objects can be recovered based on detected motion alone. In order to estimate motion models, *Random Sample Consensus* (RANSAC) [48] was used. A Bayesian method was proposed to segment multiple objects. One problem with this approach however is that pedestrian detection and tracking is not possible due to lower number of detected object points and the slower movement not suitable for motion cue based approaches. The focus of the proposed approach in this thesis is however on single object tracking, and strongly relies on the initial considered model for the object (namely, the 3D BB). Unlike the considered study, the presented approach is a general object tracker capable of tracking pedestrians as well as cyclists and other vehicles.

Another possible way to detect and track objects is through segmentation. Vatavu et al. [49] designed an approach to estimate the position, speed, and geometry of objects from noisy stereo depth data; in opposition to previous methods, the setup they considered does not consist of a LiDAR sensor, rather relying on a stereo vision setup. They built a model for the (static) background by combining information about ego vehicle locations (obtained from an INS) and a manipulated representation of the 3D PCDs. The model consists of a 2.5D elevation grid (or *Digital Elevation Map* (DEM)). Next, obstacles were segmented through the extraction of free-form delimiters of the objects (represented by their position and geometry). Lastly, these delimiters were tracked using particle filters. Since the delimiters are considered to have free-form, and thus subject to slight changes between scans, KFs were also used to update the delimiters' models.

In another attempt to solve the problem of generic object tracking, Ošep et al. [50] made use of stereo depth scene information to generate generic object proposals in 3D for each scan, and keep only those that can be tracked consistently over a sequence of scans. The considered 3D PCD was generated from the disparity map obtained from the stereo camera setup. To do so, a two-stage segmentation method was suggested to extract aforementioned (multi-scale) object proposals followed by multi-hypothesis tracking of these proposals. The two steps in the segmentation are a coarse supervised segmentation (removing non-object regions corresponding to known background categories) and a fine unsupervised multi-scale segmentation (extracting scale-stable proposals from the remaining regions in the scene).

Another approach was proposed by Asvadi et al. [51], where generic moving objects are detected and tracked based on their motion. The considered scene is obtained from a 3D LiDAR sensor mounted on top of a vehicle that can either be stationary or moving. Data points corresponding to the road are removed, and the remaining data is mapped into a static 2.5D grid map of the local environment. Another motion 2.5D grid is obtained through comparison between the last generated 2.5D elevation map and the static grid map. A mechanism based on spatial properties was also implemented that suppressed false detections due to small localization errors. In addition, and taking into consideration the 2.5D motion grids, tracking of a given object could then be obtained by fitting a 3D BB to segmented motions (detected motions were segmented into objects using connected component morphology) and keeping track of that 3D BB using data association and applying KF.

In Choi et. al [52] a tracker was proposed that took into consideration only LiDAR data. Similarly to other works such as [51] ground removal was applied so that only on-ground points were processed. A *Region of Interest* (ROI) was defined so that only objects inside that region were considered candidates for tracking. The remaining information is then clustered into segments, representing the different types of objects in the environment. Since the approach was suited for multi-target tracking, the problem of interconnected dependency between geometric and dynamic properties was solved by using a model based approach that used object geometry to eliminate the ambiguity between shape and motion of the object (instead of typical approaches where it is used for classification). Shape and motion properties for a tracked object were estimated from the LiDAR data and interpreted as a 2D virtual scan paired with a Rao-Blackwellized particle filter based on [53]. However, due to computational costs, the particle filter was later substituted by a KF.

Miyasaka et. al [54] also relied on LiDAR information to estimate ego vehicle motion, build and update a 3D local grid map and detect and track moving objects. To estimate ego-motion, an initial estimation is provided by ICP along with motion parameters. These parameters are used as an initial guess in the second step. The last available range scan is matched onto the local (denser) map and map-matching provides the final parameters. Candidate moving object points are extracted as either outlier in the ICP method, points in free-space in the occupancy grid map or points derived from the unknown space. To divide the points into possible objects, a clustering algorithm was applied and dynamic points can be considered to produce candidate dynamic objects. Tracking is then applied by matching and association algorithms, where an *Extended Kalman Filter* (EKF) was used. An object tracked for more than a minimum threshold of frames was considered to be moving.

Competitions such as the DARPA Urban Challenge have contributed to the arrival of interesting vehicles capable of navigating urban environments autonomously. To compete in the challenge, teams underwent a series of tests, with one such test being the development of a good technical paper describing how they would implement their approach. From an initial pool of 53 teams, only 11 would qualify for the Urban Challenge Final Event. Some of the best competitors work can be seen in [28] and [55], whose setups are presented in the following figure.

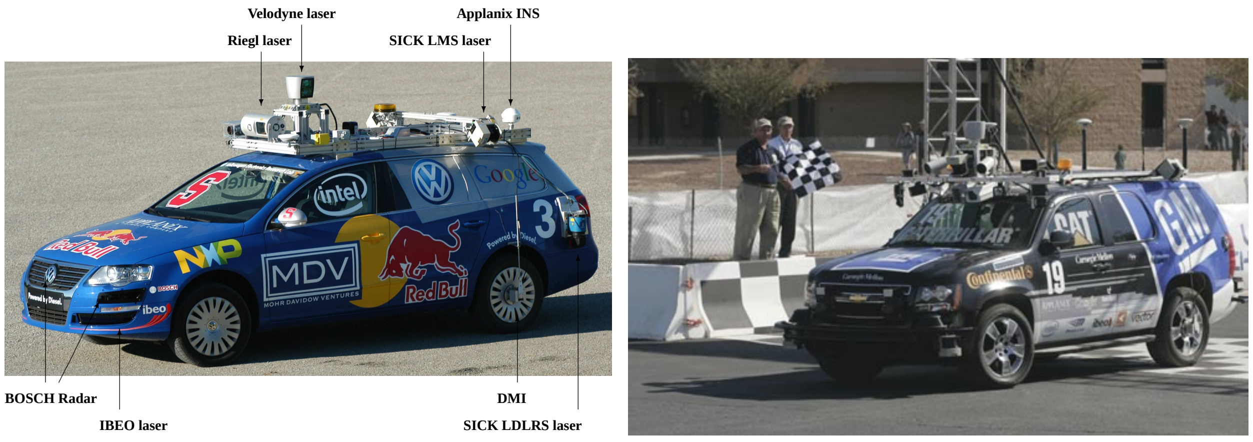


Figure 2.5: *Autonomous vehicle examples from the DARPA Urban Challenge in 2007. The setup on the left is Junior, presented in [28], and the setup on the right is Boss, the winner, presented in [55].*

2.2.2 Summary (Object Trackers)

In the current section a brief comparison between the considered approaches is present in Table 2.2, specifying characteristics such as tracking approaches and objects representation and update.

Approach	*3D Perception Sensor*	*Ego-motion Estimation*	*Tracking Approach*	*Object Representation*	*Object Search Mechanism*	*Object Model Update*
[46]	3D LiDAR, Camera	INS	Generative	Coloured PCD	CV KF	ICP, Accumulation
[47]	3D LiDAR	DGPS/IMU	Generative	PCD	Bayesian approach	No Update
[49]	Stereo Vision	GNSS, INS	Generative	Object Delimiters	Particle Filter	KF
[50]	Stereo Vision	V-Odometry	Generative	Voxel	KF and MHT	Weighted ICP
[51]	3D LiDAR	INS	Generative	Elevation Grid	CV KF and Gating	No Update
[52]	3D LiDAR	-	Discriminative	Voxel	KF and MHT	KF
[54]	Multi-Layer LiDAR	ICP	Generative	Voxel	EKF	No Update
[56]	3D LiDAR, GIS Map	INS	Generative	PCD	CV KF	No Update

Table 2.2: *Summary on considered 3D object trackers.*

CHAPTER

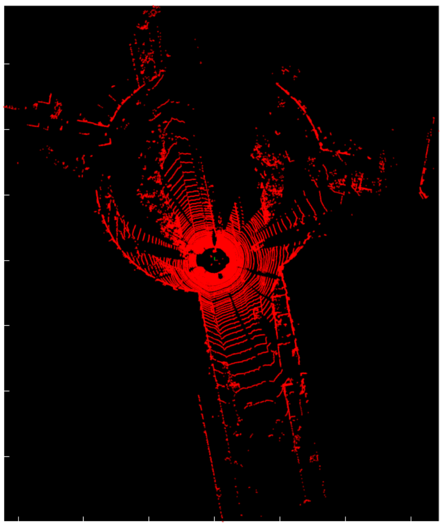
3

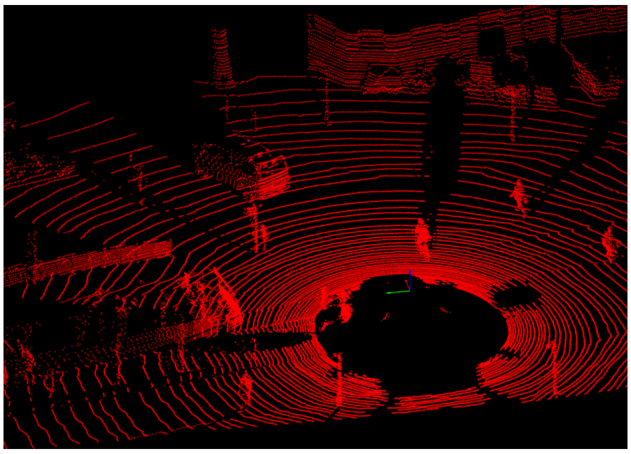
PROPOSED METHOD

Contents

In the previous chapter an overview of the current object trackers literature was given, as well as for sensor technologies. The presented studies highlighted the fact that 3D spatial data processing has gained attention in fields such as computer vision and robotics, powered by the arrival of new 3D sensing alternatives and new capable stereo matching algorithms. The aforementioned 3D spatial data comes in the shape of dense 3D PCDs.

In order for intelligent vehicles to understand their relevance and how exactly they characterize the scene, developed perception systems need to interpret the surrounding environment as well as perceive objects physical properties.


(a) *Sample output point cloud.*


(b) *Zoomed in point cloud.*

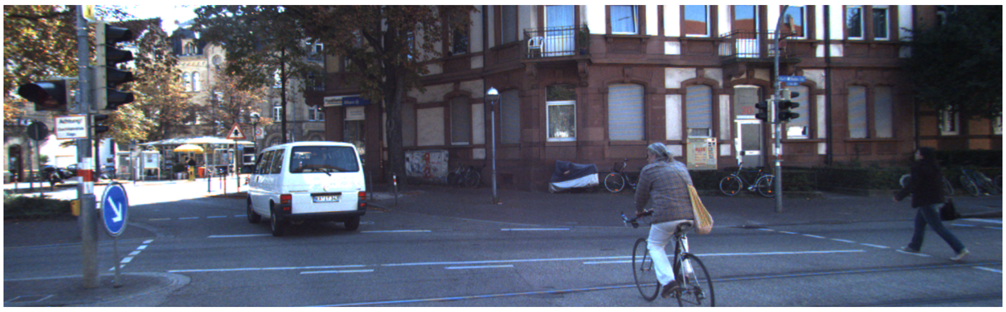

(c) *Sample output RGB image.*

Figure 3.1: *Input data from 3D and 2D sensors. Both PCD and RGB image correspond to the same scan in a given sequence from the KITTI dataset. The RGB arrows represent the local (car) reference system.*

From the presented sensors, 2D cameras have been widely used for the goal of scene representation; one of its main advantages is that it provides a very rich and high resolution colour representation. This in turn provides a very good complement to 3D LiDAR sensors, whose lack of colour information is one of their main disadvantages.

The proposed framework has thus used both of these technologies, looking into combining highly accurate 3D LiDAR PCDs with rich and dense 2D RGB corresponding camera images. This data comes from the conjugation of two sensors: a Velodyne HDL-64E and a PointGrey Flea 2 colour camera. These sensors are mounted on the vehicle that provided the KITTI Dataset as explained in Section 4.1. Additionally, an INS is also used to obtain the location of the ego vehicle. Figure 3.2 shows the considered reference system axes.

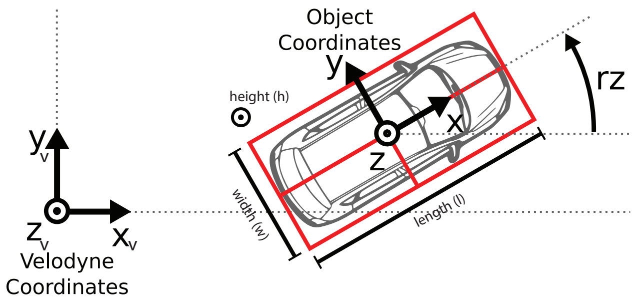


Figure 3.2: *Object coordinate system in the 3D space [30].*

3.1 Approach Formulation

The task of 3D single object tracking can be defined as the estimation of the trajectory of an object as it moves around a scene (with the ego vehicle either static or moving), with an object being identified as a 3D BB (with a given width, height and length).

As mentioned in Section 1.2 it is assumed that an initial 3D BB is known, *i.e.*, that a previous 3D object detection or motion detection method has been applied in the reference 3D PCD and that object characteristics such as location and orientation are provided for the initial reference scan.

3.1.1 Relevant Notation

Input data for the proposed framework will then consist of:

- $3D_{BB}$: 3D BB (initial is assumed to be known);
- I_i: RGB image obtained from camera in scan i;
- L_i: 3D PCD from LiDAR in scan i;
- L_{G_i}: 3D PCD in scan i with ground points removed;
- P_i: Set of 3D points projected in 2D image plane in scan i;
- Ω: Convex-hull of points in P;

With this information, the tracker will be able to estimate the trajectory of the object, representing this trajectory as $X_i = \{X_1, \cdots, X_N\}$ in the 3D world coordinate system. Each element of X will hence be the $\{x\ y\ z\}$ coordinates of the considered object centroid in a certain instant of time.

3.2 System Overview

In the following figure, a diagram representing the conceptual pipeline of the approach is shown. It follows the same flow as the diagram presented in Figure 1.1 but presents a more detailed insight.

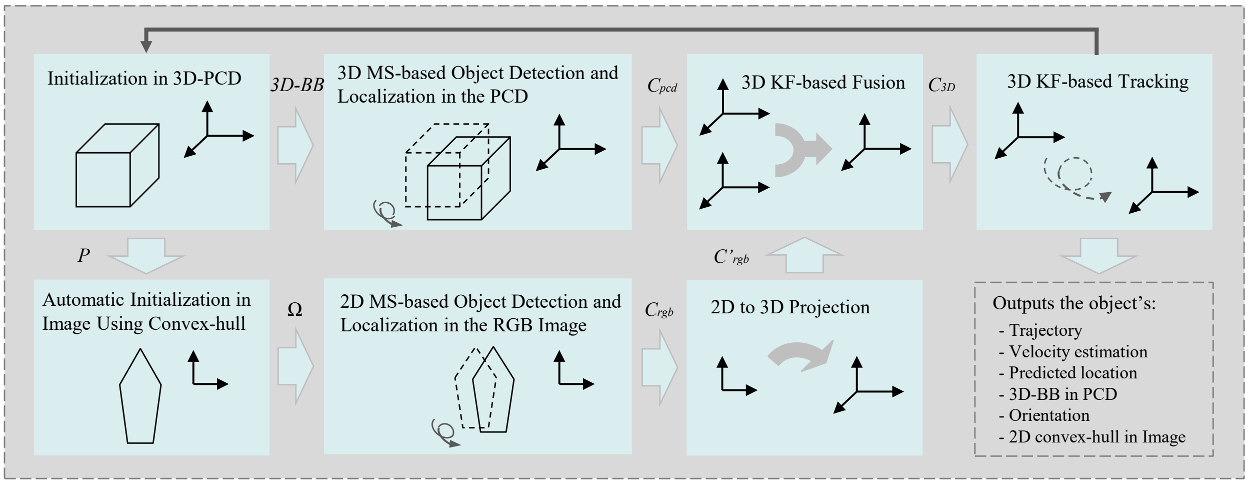


Figure 3.3: *Major pipeline of the proposed approach. 2D and 3D information is fused and used for 3D tracking of the objects.*

The raw PCD in any given scan is of a form similar to the one present in Figure 3.1b. For the purpose of object detection, recognition and tracking, it is not desired to take ground points into consideration, as these can severely hinder the correct results provided by the proposed algorithms. In the particular case of the presented object tracking approach, removal of ground points is a particularly important process for the construction of the object model. Without this process, the constructed object model would be degraded.

In this framework, the process of object tracking starts with a known $3D_{BB}$ in L_1. A new PCD with no ground points L_{G_1} is obtained. The remaining object points (points from L_{G_1} inside $3D_{BB}$) are projected onto the image plane (I_1) resulting in a set of projected points, P^*. The corresponding 2D convex-hull (Ω) is calculated. It is useful to have Ω since the convex-hull effectively and accurately segments what are object pixels from non-object (or background) pixels.

Having this information, it is possible to initialize the tracking in both the 3D PCD and 2D image plane. For tracking purposes a *Mean-Shift* (MS) based object localizer [57] is run for each information domain (3D PCD and 2D image) in order to obtain the new object location. Each MS has different characteristics: to localize an object in the 3D domain, a MS gradient estimation of those points inside the 3D BB is considered; to localize the object in 2D, and adaptive color based MS algorithm was applied. After obtaining the new 2D location of the object, its projection is calculated back onto the 3D domain using bilinear interpolation.

In this approach, and in order to fuse data from two sources of information, a *Constant Acceleration Kalman Filter* (CA KF) is considered (measurement fusion model as discussed in [58]). After this process, another CA KF is used for the purpose of object tracking, as the newly detected location and orientation are used to initialize the 3D BB in the next scanned PCD.

3.3 3D Tracking

For the purpose of 3D object tracking, it is common to apply a ground estimation algorithm to determine what points in the PCD belong to the road and not to objects in the scene. As aforementioned, the construction of an object model without removal of ground points would result in a degraded result, since the inclusion of road or sidewalk points would directly impact the centroid estimation.

3.3.1 Ground Removal

A significant number of points in a typical LiDAR PCD are ground points. By selecting an appropriate set of features that characterize these points, the corresponding *Probability Density Function* (PDF) can be calculated. The notion of a given variable being close to a target value can be captured by a PDF; the highest (or peak) value indicates the highest probability that the variable is close to the target value. Hence, the peak value from the computed PDF will indicate the ground points in the current scan.

To take advantage of this information, a *Kernel Density Estimation* (KDE) was also applied. This estimator gauges the PDF of all angles that lie between the ground (XY plane) and the set of lines passing through the *origin point* (projection of the LiDAR center of mass onto the ground) and the *end points* (remaining points that belong to the PCD).

Considering $\Theta = \{\theta_1, \ldots, \theta_N\}$ 1D angles in the XZ plane for a given PCD (where $\theta_i = arctan(z_i/x_i)$) a univariate KDE estimated can be obtained from

$$f(\theta) = \frac{1}{N} \sum_{i=1}^{N} K_\sigma(\theta - \theta_i) \tag{3.1}$$

In this expression, $K_\sigma(\cdot)$ is a Gaussian kernel function that has σ bandwidth, and N is the total number of points in the PCD. For this model, the ground is assumed to be plane, and the pitch angle of the ground plane is identified as the peak value θ_ρ in the KDE. Having identified the ground plane, all points whose z coordinate (height) fall under a given threshold d_{min} are thus considered ground points. d_{min} is chosen as an arbitrarily low value designed to represent a safety boundary for the process.

To smooth the process of determining the ground plane pitch angle in consecutive scans, an additional CA KF was used for this estimation. This provides additional robustness and accuracy to the ground removal process. Given the size of the PCD, outliers are also eliminated by limiting the angle search area to a gate in the neighbourhood of the predicted value from the KF in the previous step. If no measurements become available inside the provided gate, the predicted value from the filter is used. The ground removal process is illustrated in Figure 3.4.

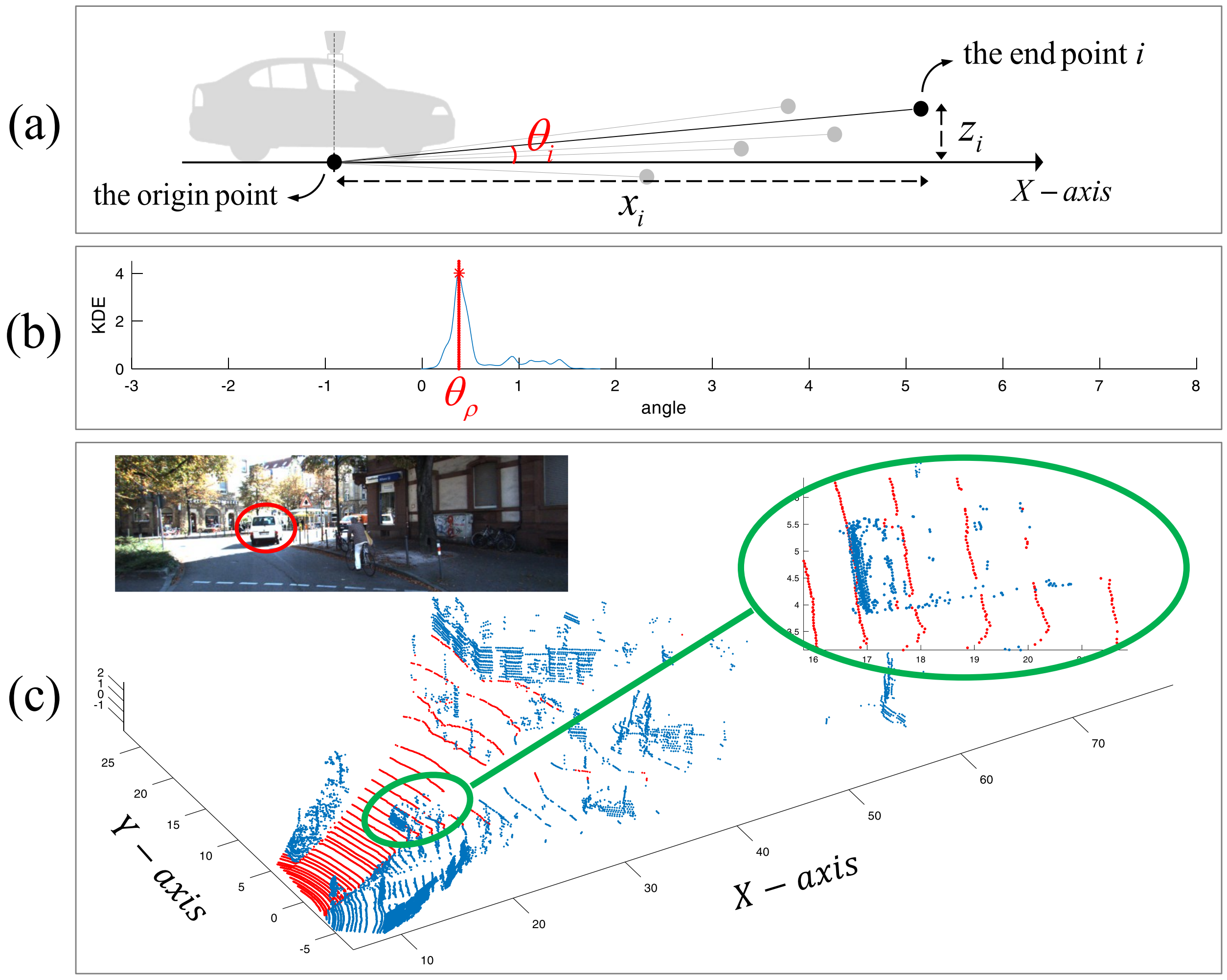


Figure 3.4: *Ground removal process. (a) Computation of the θ_i angle. (b) KDE of the set of 1D angles Θ with pitch angle of the ground plane θ_ρ. (c) End result, with detected ground points (red) and points above ground (blue). The green ellipse shows an example of the ground points extraction for a car. The corresponding car in the image is shown with a red ellipse.*

3.3.2 Detection and Localization

After removing ground points, detection and tracking of the target object can be achieved. To do so, a MS approach was considered. This iterative procedure, based on a KDE, shift point(s) in a set by an amount equal to the average of the data points in their neighbourhood.

The MS procedure can be summarized as:

1. **Computing the *shift vector*:** Given the known object centroid (centre of the last known 3D BB) as χ, the shift vector between χ and the point set P' (points from the current scan PCD inside previous time step 3D BB) is computed using

$$m_k = \chi_k - \mu(P') \tag{3.2}$$

where $\mu(.)$ indicates the "*mean*" function, and k is the iteration index.

2. **Translating the 3D BB:** The 3D BB is translated using the shift vector

$$\chi_{k+1} = \chi_k + m_k \tag{3.3}$$

with the shift vector always pointing toward the direction of the maximum increase in the density.

3. **Iterating steps 1. and 2. until convergence:** The MS iteratively shifts the 3D BB until the object is placed entirely within the 3D BB. A centroid movement $|m_k|$ less than 5 cm or a maximum number of iterations equal to 5 are considered as the MS convergence. These values were chosen after testing performance with different scenarios.

An example of the MS in action can be seen in the figure below: From the initial 3D BB (from previous scan) and set P', a MS was employed to obtain the 3D BB that defines the object in the current scan.

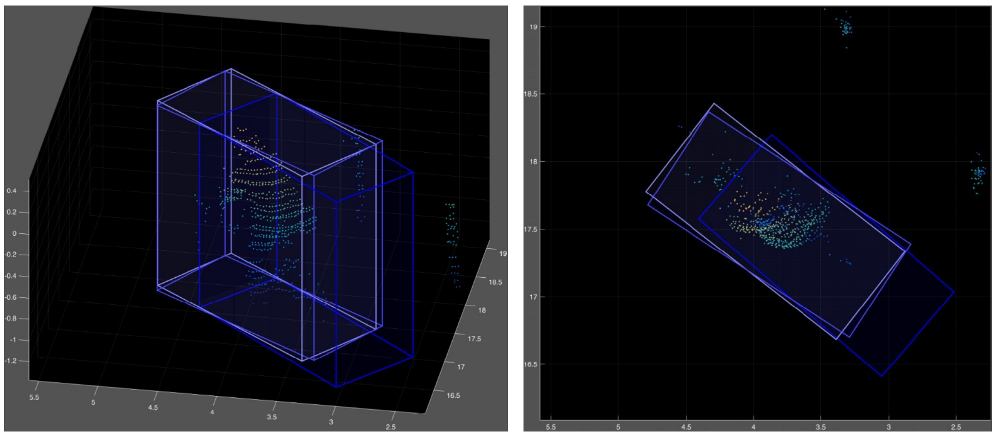


Figure 3.5: *MS procedure in the PCD. The figure on the left represents a bird's-eye view of the object (cyclist) and the right figure shows the top view. In this example, the MS convergence (the centroid movement less than 5 cm) was achieved in three iterations. The darker blue colour represents the initial 3D BB whilst the brighter blue colour 3D BBs show the most recent iterations (with brightest blue equalling the 3D BB when convergence was met).*

As mentioned earlier in this chapter, current object position is represented by the centroid of object points inside the 3D BB. This point model representation is feasible even with a low number of detected object points, which happens in sparse 3D LiDAR PCDs, specially for objects located further away from the LiDAR. The resulting centroid after convergence is denoted by C_{pcd} and outputted to the projection module.

3.4 2D Tracking

To initialize the object tracking process in the 2D image, the object initially needs to be detected. To do so, the 3D object points P after ground removal are projected onto the image plane. In the same manner that a 3D BB was used to describe the 3D object, a 2D convex hull (Ω) was chosen to identify the 2D object in the image. Therefore, the convex hull Ω of the projected point set $P^* = \{p_1^*, \cdots, p_n^*\}$ is then computed. Ω can be defined as the smallest 2D convex polygon that encloses P^*. By considering a 2D convex hull it is possible to more accurately segment the object from the background in comparison with a traditional 2D bounding box. A surrounding area $\Omega^\dagger$ is defined automatically by expanding Ω by a factor equal to $\sqrt{2}$ with respect to its centroid, so that the number of pixels in $(\Omega^\dagger - \Omega)$ (the region between Ω and $\Omega^\dagger$) is approximately equal to the number of pixels inside Ω.

3.4.1 Discriminant Colour Model

For tracking the object in the image domain a discriminant colour model of the object is created and updated in every new scan. To do so, an RGB histogram of the object is calculated. The considered pixels for the calculation lie within the Ω region and the $(\Omega^\dagger - \Omega)$ region. Two colour models are then calculated: one for Ω and one for $(\Omega^\dagger - \Omega)$.

The next step is to calculate the *Log-Likelihood Ratio* (LLR) of the computed RGB histograms. The LLR expresses how many times more likely the data are under one model than the other. In our application, the LLR expresses how much more likely each histogram bin is under the Ω colour model than the $(\Omega^\dagger - \Omega)$ colour model. This is achieved by knowing that in the LLR positive bins likely belong to Ω, with negative valued bins likely belonging to $(\Omega^\dagger - \Omega)$. Bins that are shared by both Ω and $(\Omega^\dagger - \Omega)$ tend towards zero. The positive bins in the LLR are thus used to represent the discriminant object colour model. The discriminant object colour model can be represented by

$$\Re(i) = \max\left\{\log \frac{\max\{H_\Omega(i), \epsilon\}}{\max\{H_{\Omega^\dagger}(i), \epsilon\}},\ 0\right\} \tag{3.4}$$

In this representation, H_Ω and $H_{\Omega^\dagger}$ are the computed histograms from Ω and $(\Omega^\dagger - \Omega)$ regions, respectively, and ϵ is a small value that prevents either divisions by zero or log of zero operations. The variable i ranges from 1 to the total number of histogram bins. The discriminant colour model of the object ($\Re$) is normalized and used for object detection and localization in the next frame using the MS procedure described in the next section.

3.4.2 Detection and Localization

With the location of the object in the previous scan known and described as the shape of a convex hull (Ω), object detection in the current scan can be achieved using a MS based object detection and localization from previous known information.

Initially, a confidence map (represented by $\mho$) of the Ω region in the previous frame is calculated. The centroid of this region is obtained and MS process start in this point. The confidence map can be achieved by replacing the colour value of each pixel in the Ω region by its corresponding bin value in the colour model $\Re$.

In each iteration the centre of Ω from the previous step is shifted to the centroid of $\mho$ (the current confidence map of the object). This centroid can be calculated by

$$C_{iter} = \frac{1}{m} \sum_{i=1}^{m} \mho_i \; C_i \tag{3.5}$$

In this notation, $C_i = \{r_i, c_i\}$ denotes the pixel position (row and column) in Ω and m is the total number of pixels inside the Ω region.

An empirical value of 4 was chosen as the threshold for MS iterations needed to achieve convergence. If the method achieves centroid movement smaller than a pixel then it is also considered to have achieved convergence. The computed 2D object centroid after convergence is denoted by C_{rgb} and outputted to the fusion module. An overview of the detection and tracking process in 2D can be observed in Figure 3.6.

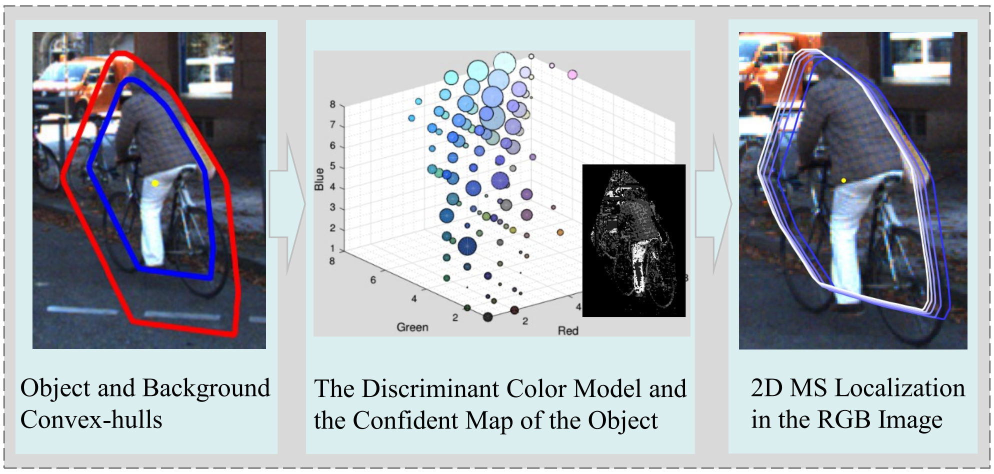

Blue
Green
Red
Object and Background Convex-hulls
The Discriminant Color Model and the Confident Map of the Object
2D MS Localization in the RGB Image

Figure 3.6: *MS procedure in the 2D RGB image. The left figure represents both Ω (in blue) and $\Omega^{\dagger}$ (in red). The middle represents the computed $\Re$ and the correspondent confident map ($\mho$). Each non-empty bin in the $\Re$ is shown with a circle with a value correspondent to the area of its circle. Circle locations represent the correspondent colour in the joint RGB histogram. The right figure shows the MS localization procedure, with the brighter blue Ω indicating that it was computed in the most recent MS iteration. The final Ω centroid is shown as a yellow point.*

3.4.3 Adaptive Updating of $\Re$ Bins

As mentioned in Chapter 2, dense 2D RGB images are very informative. However, they are sensitive to variations in illumination conditions. In order to adapt the object colour model with illumination variations (surmounting changes in the object colour appearance during tracking), a bank of 1D CA KFs model was applied.

The goal of these KFs is to estimate and predict $\Re$ bin values for the next frames. A new 1D CA KF is initialized and associated for each newly observed colour bin. When the bin values become zero or negative, the corresponding CA KFs are removed. Based on a series of tests where

$8 \times 8 \times 8$ histograms (512 bins) were considered, the average number of utilized KFs in each frame were about 70 (or 14% of the total number of bins).

Histogram KF The following is the considered CA KF formulation used for the updating process of the object histogram. Each time a new colour is observed a CA KF is initialized. The corresponding state vector is based on the value of that bin (between 0 and 1)

$$x_k = [bin_{value}\ 0\ 0]^T \tag{3.6}$$

The state transition matrix A becomes

$$A = \begin{pmatrix} 1 & d_t & \frac{1}{2}d_t^2 \\ 0 & 1 & d_t \\ 0 & 0 & 1 \end{pmatrix} \tag{3.7}$$

H was set to

$$H = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix} \tag{3.8}$$

The histogram CA KFs covariance matrices R, Q and P were set to

$$R = 0.5; \tag{3.9}$$

$$Q = \begin{pmatrix} 5 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 5 \end{pmatrix} \tag{3.10}$$

$$P = \begin{pmatrix} 15 & 0 & 0 \\ 0 & 15 & 0 \\ 0 & 0 & 15 \end{pmatrix} \tag{3.11}$$

Bin Updating For the first scan (and object detection) non empty histogram bins are found, with a KF initialized for each. In the next scans, a search for non empty histogram bins is also applied. If an empty bin in the previous scan has a value in the current scan, a KF is initialized for that bin. If a non empty bin in the current scan was also not empty in the bin before, its value must be updated through an iteration of the KF, with the current value in the bin used as a measurement. A last case may happen when a previously non empty given bin no longer has a value, resulting in the corresponding KF being deleted.

3.5 2D/3D Projection

The computed 2D location of the object in the RGB image (C_{rgb}) is projected back to 3D (C'_{rgb}). To do so, a method described by Held et. al [46] for PCD upsampling was used. In our approach however, the method is used solely to project C_{rgb} back to the 3D LiDAR space.

Having found the 2D centroid, the nearest projected points (inside the convex hull) in each of the four quadrants are found (f_1, f_2, f_3 and f_4 for upper left, upper right, lower left and lower right quadrants, respectively). Bilinear interpolation using these projected points is obtained by first linearly interpolating between the points on top (p_t), then the points below (p_b) and finally interpolating between the previous two interpolated positions, obtaining the position of the image pixel p (see Figure 3.7).

$$p_t = \frac{f_1 + f_2}{2}$$

$$p_t = \frac{f_3 + f_4}{2}$$

$$p = \frac{p_t + p_b}{2}$$

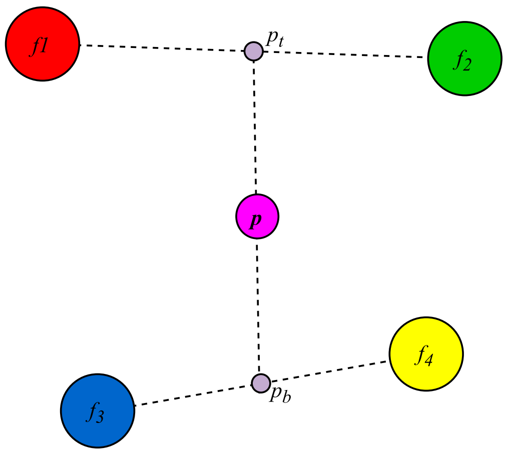


Figure 3.7: *Bilinear interpolation diagram.*

Fractions of the horizontal distances (s_1 and s_2) between p and each neighbouring pixel pair are obtained from

$$s_1 = \frac{p_u - f_{1,u}}{f_{2,u} - f_{1,u}}$$

$$s_2 = \frac{p_u - f_{3,u}}{f_{4,u} - f_{3,u}}$$

with (u,v) being the pixel horizontal and vertical coordinates. Bilinear interpolation can then be defined as

$$p_t = f_1 + s_1(f_2 - f_1)$$

$$p_b = f_3 + s_2(f_4 - f_3)$$

$$s_3 = \frac{p - p_b}{p_t - p_b}$$

$$p_{interpolated} = p_b + s_3(p_t - p_b)$$

The resulting $p_{interpolated}$ is calculated in both pixel space and in 3D coordinates. $p_{interpolated}$ is close to p in pixel space and the corresponding 3D location is then considered to be the 3D location

(C'_{rgb}) of the 2D computed centroid (C_{rgb}).

One example of the bilinear interpolation process can be seen in Figure 3.8 for a tracked van.

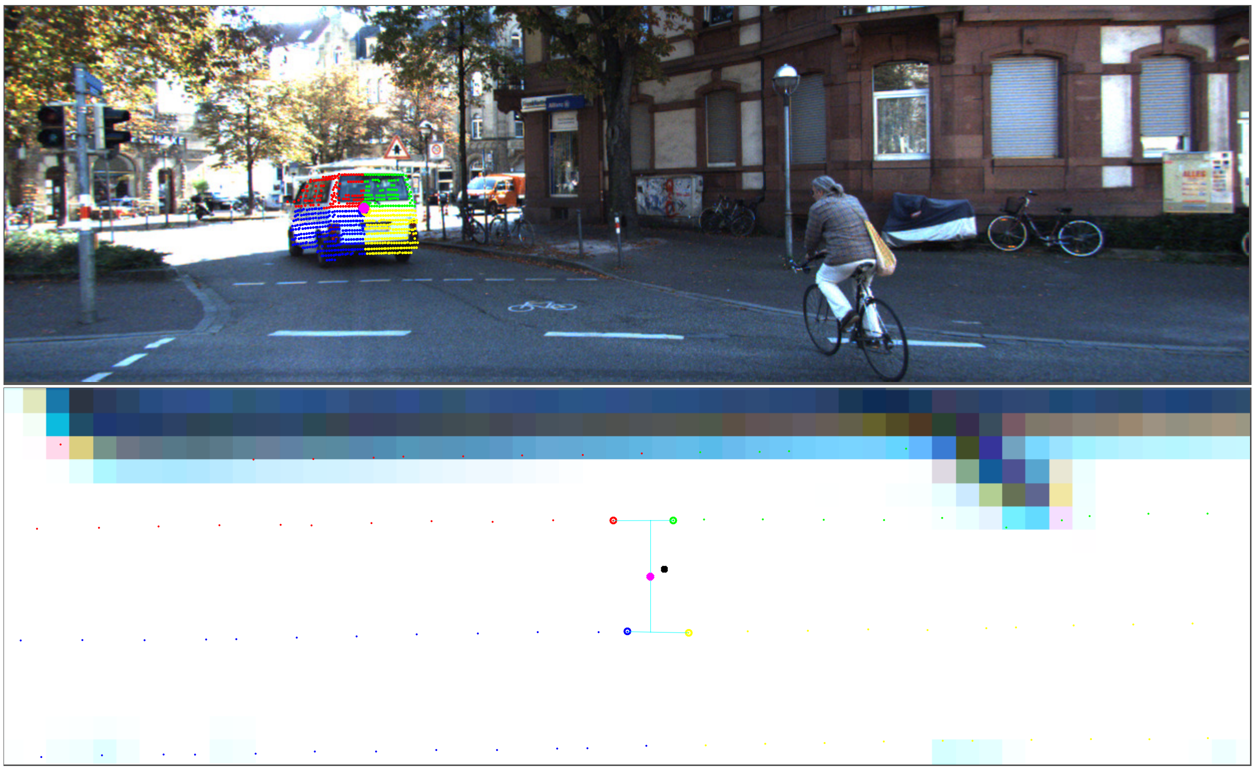


Figure 3.8: *Bilinear interpolation process. The black dot represents C_{rgb}, with the four quadrants represented in red, green, blue and yellow points. The dot in magenta is the calculated interpolation point (in 2D), with the corresponding 3D coordinates obtained from the previous formulation.*

3.6 Fusion and Tracking

After converting the object 2D location in the RGB image into the 3D space, two different 3D centroids are available. To fuse both centroids, a CA KF was applied. After the fusion, another CA KF was used to filter and track the resulting fused 3D centroid along time. A visual representation of the data flow (centroid wise) is presented in Figure 3.9, taking into consideration the two centroids outputted from previous modules. On the following section both fusion and tracking KFs are explained.

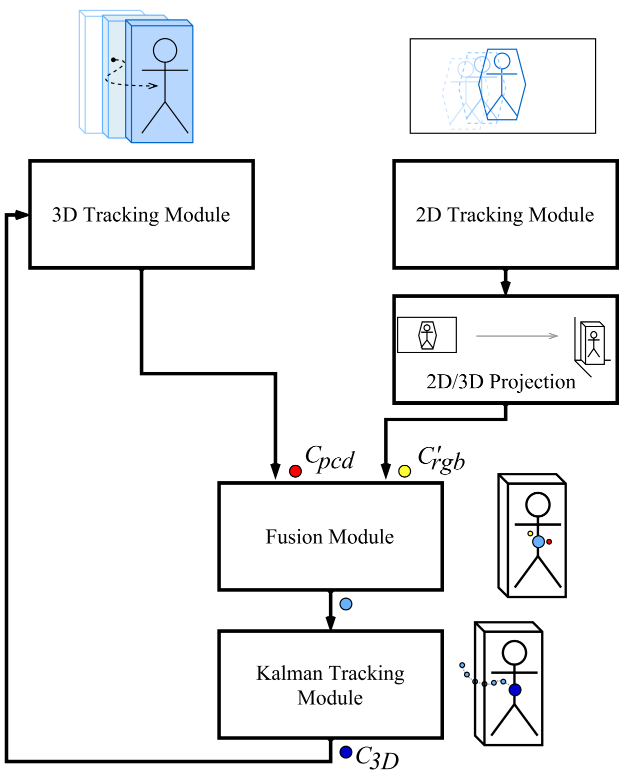


Figure 3.9: *Overview of the tracking system. The fusion and Kalman tracking modules are based on CA KFs.*

3.6.1 2D/3D Fusion for Improved Localization

To integrate information from two different sensors (after some processing) a KF based fusion was considered. Based on the measurement fusion model from Gao et. al [58], a KF is applied to integrate the centroid from the 2D tracking module C'_{rgb} with the computed 3D object location in the PCD C_{pcd}. An assumption of identical sample rates between sensors was made. The sensors are also assumed as independent of each other. To define the fusion model, object dynamics is first represented as

$$x_{k+1} = F_k x_k + \Gamma_k v_k$$

where x_k is the state vector at time k and v_k the state noise such that $E[v_k] = 0$ and $E[v_k v_l^T] = Q_k \delta_{kl}$. The corresponding measurements are

$$z_k^m = H_k^m x_k + w_k^m, m = 1, 2$$

with z_k^m being the measurement from sensor m at time k and measurement noises w_k^m being zero mean, white noise with covariances R_k^m and mutually independent.

The measurement fusion is done at state vector level as an augmented observation vector, combining z_k^1 and z_k^2 into the augmented observation vector

$$z_k = [(z_k^1)^T (z_k^2)^T]^T$$

Assuming that $H_k = [(H_k^1)^T (H_k^2)^T]^T$ and $w_k = [(w_k^1)^T (w_k^2)^T]^T$ a new measurement equation can be obtained as

$$z_k = H_k x_k + w_k$$

Finally, and since statistical independence was assumed, the merged measurement noise w_k is defined as

$$R_k = \begin{pmatrix} R_k^1 & 0 \\ 0 & R_k^2 \end{pmatrix}$$

The estimate of the state vector is obtained by conventional Kalman filtering.

The considered dynamics of the object and the fused measurement model of the object localizers (both in 3D and 2D in the presented approach), as well as the used notation and variable meaning, are described below.

Fusion CA KF The following is the formulation considered for the CA KF used for fusing information (centroids provided by 3D MS and re projected 2D MS). The state vector composed as

$$x_k = [x_{proj2D_k}\ y_{proj2D_k}\ z_{proj2D_k}\ x_{3D_k}\ y_{3D_k}\ z_{3D_k}]^T \tag{3.12}$$

The state transition matrix A becomes

$$A = \begin{pmatrix} 1 & d_t & \frac{1}{2}d_t^2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & d_t & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & d_t & \frac{1}{2}d_t^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & d_t & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & d_t & \frac{1}{2}d_t^2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & d_t \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.13}$$

H was set to

$$H = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \tag{3.14}$$

Fusion CA KF covariance matrices R, Q and P were set to

$$R = \begin{pmatrix} 5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 5 \end{pmatrix} \tag{3.15}$$

$$Q = P = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.16}$$

In order to obtain a better initial prediction and to maintain the internal coherence of the system, a linear least squares method is used to estimate the initial state based on the initial observation. Doing so is equivalent to solve a system of linear equation in the shape of $Ax = B$ for x with left matrix division (in this case, solve $Hx_k = x_{k_{initial}}$ for x_k).

The dynamics of the object and the fused measurement model are then given by Equations (3.17) and (3.18) respectively.

$$x_f = A_F \cdot x_{f-1} \tag{3.17}$$

$$z_f = H_F \cdot x_f \tag{3.18}$$

For fusion purposes, the augmented measurement vector z_f is then given by

$$z_t = \begin{bmatrix} (C_{pcd})^T & (C'_{rgb})^T \end{bmatrix}^T \tag{3.19}$$

The end result of this integration is a fused 3D centroid (noted as C_{3D}). By having an approach based on a KF, it is possible to give a higher weight to a method that performed better in the recent past iterations, thus providing a more accurate estimate than applying each method individually.

3.6.2 3D Object Tracking

For the purpose of tracking the final object estimation, another CA KF was used. This assumption is useful to model target motion when it is believed to be smooth in both position and velocity changes. For the purpose of object tracking in urban scenarios, 3D objects (read pedestrians, cyclists or vehicles) are subject to changes in instant velocity during a sequence of scans.

Object Tracking CA KF The following is the formulation considered for the CA KF used for object tracking. In this case, object representation is defined in a state vector composed as

$$x_k = [x_k \; y_k \; z_k \; \dot{x}_k \; \dot{y}_k \; \dot{z}_k \; \ddot{x}_k \; \ddot{y}_k \; \ddot{z}_k]^T \tag{3.20}$$

Since object representation is done as a point model, the centroid of the object is located in the $\{x \; y \; z\}$ coordinates. $\{\dot{x}_k \; \dot{y}_k \; \dot{z}_k\}$ and $\{\ddot{x}_k \; \ddot{y}_k \; \ddot{z}_k\}$ represent object velocity and acceleration in each coordinate, respectively.

The state transition matrix A becomes

$$A = \begin{pmatrix} 1 & d_t & \frac{1}{2}d_t^2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & d_t & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & d_t & \frac{1}{2}d_t^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & d_t & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & d_t & \frac{1}{2}d_t^2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & d_t \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.21}$$

with d_t being the sampling rate. In the case of Velodyne LiDAR this sampling rate is 10 Hz (ten scans every second). The PointGrey camera has been calibrated to acquire images at the same rate.

H was set to

$$H = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \tag{3.22}$$

And covariance matrices R, Q and P were set to

$$R = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.23}$$

$$Q = P = \begin{pmatrix} 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 \end{pmatrix} \tag{3.24}$$

Equations (3.25) and (3.26) represent the discrete time process model and the measurement model of the system, respectively

$$x_t = A_T \cdot x_{t-1} \tag{3.25}$$

$$z_t = H_T \cdot x_t \tag{3.26}$$

To eliminate outliers and increase robustness of the process, the search area is limited to a gate in the vicinity of the predicted KF location (available from $x_t = A_T \times x_{t-1}$). If no measurement is available inside the gate area, the predicted value given by the KF is used.

The result of the proposed algorithm is the estimated trajectory of an object in the 3D world coordinate system, its current estimated velocity (obtained directly from the tracking Kalman filter), and the predicted location of the object in the next time step. The object orientation is achieved by subtracting the current estimated location from its previous location. The object region in the image is obtained by computing the 2D convex-hull of the projected object points (the points inside the 3D-BB) into the image.

CHAPTER

4

EXPERIMENTAL RESULTS

Contents

This chapter outlines the considered dataset for testing purposes, as well as the experimental results obtained during the development of this thesis. In the previous chapter a detailed description of the proposed approach was made, highlighting the behaviour of each system module and its outputs. The results obtained from running this approach on selected test cases (from the KITTI dataset) are presented in this chapter. Section 4.1 details the KITTI dataset and how data acquisition was performed and presents the considered test sequences. Section 4.2 explains the evaluations metrics used to obtain the practical results of the implementation. Section 4.3 presents two baseline methods that the proposed method was tested against. Finally, Section 4.4 presents the resulting data obtained by applying each of the considered methods and makes a critical analysis of the results.

4.1 Dataset

The developed approach was tested on sequences extracted from the KITTI Dataset [30]. This dataset was obtained by driving around the mid-size city of Karlsruhe, in both rural areas and on highways. The information present in the dataset was obtained using a Velodyne 3D LiDAR and a high-precision INS (GPS/IMU). The Velodyne HDL-64E [27] rotates at 10 *Frames Per Second* (FPS) counter-clockwise with vertical resolution of 64 layers, angular resolution of 0.09° and 2cm distance accuracy. Both the horizontal and vertical fields of view are 360° and 26.8°, respectively, and the maximum range is 120m. The Velodyne point cloud is compensated for the vehicle ego-motion. The inertial navigation system is a OXTS RT3003 and GPS navigation system [59], with a frame rate of 100Hz and a resolution of 0.02m/0.1°. The color cameras are PointGrey Flea 2 global shutter cameras [19] with a maximum resolution of 1384×1032 and a maximum capture rate of 15 FPS.

When spinning at 10 Hz, Velodyne's HDL-64E model produces about 2.5 million distance estimations per second. Sensor coverage is dependant on the mounting position, which directly influences the size of the blind area around the vehicle, the fixed angular resolution, and the reflectance of the measured targets. It should also be noted that the HDL-64E rolling shutter causes a deformation in the resulting data as noted by [60]; when travelling straight at 10 m/s (*i.e.* 36 km/h) and operating at 10 Hz, the reference system moves 1 m between the start and the end of the scan, producing even larger errors when other objects move as well.

4.1.1 Synchronization

With every sensor mounted on the vehicle having different acquisition rates, it was necessary to have the information coming from the different sensors synchronized. In [30] it is explained that

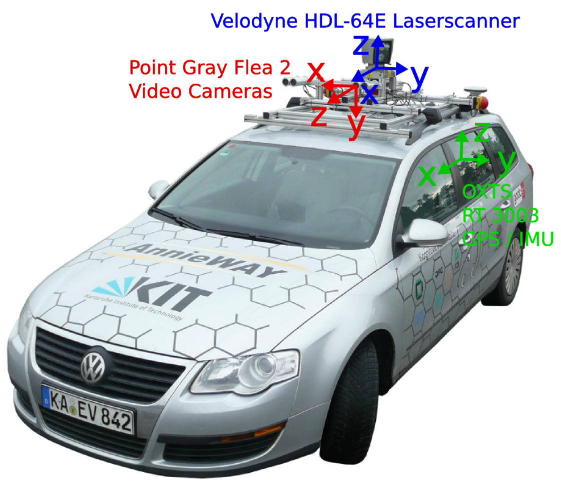

Velodyne HDL-64E Laserscanner
Point Gray Flea 2 Video Cameras
OXTS RT 3003 GPS / IMU
x
y
z

Figure 4.1: *Recording platform for the KITTI Dataset. The base platform is a Volkswagen Passat B6, which has been modified with actuators for the pedals (acceleration and brake) and the steering wheel. The data is recorded using an eight core i7 computer equipped with a RAID system, running Ubuntu Linux and a real-time database [30].*

timestamps of the Velodyne 3D Laser scanner were used in order to set it as the reference and thus consider each LiDAR spin as a frame. The cameras were also triggered when the LiDAR is facing forward. For INS synchronization a collection of information with the closest time stamp to the LiDAR time stamp was done (worst-case time difference of 5 ms between data from the two sensors).

4.1.2 Dataset Sequences

In the KITTI Dataset objects are identified by a tracklet. This is mainly due to the fact that the dataset is more focused towards the evaluation of data association approaches. To evaluate the considered approach (as well as two generative baseline approaches described in literature for comparison) tracklet information alone is not useful on its own. Hence, and instead of using the plain tracklets provided, the tracks of a selected number of objects were considered (for as much frames in the sequence without full occlusions as possible). For the purpose of evaluation 20 annotated sequences were selected to evaluate the performance of the mentioned approaches. The general specifications of each sequence as well as the challenging factors they face are reported in Table 4.1.

In this table, information for each sequence is described based on a number of characteristics. These include existing number of frames, object type (C for car, Y for cyclist and P for pedestrian), object and ego vehicle status (M for moving and S for stopped) and the scene conditions (either being characterized as either roads on urban (U) or alleys/avenues in downtown (D) areas). Information about the initial 2D BB width and height (in pixels) and 3D BB width, height and length (in meters) is also present. Regarding the challenging factors of each sequence, four major categories were chosen: objects undergo occlusion (OCC), object pose varies (POS), the distance between ego vehicle and target object varies (DIS) and the relative velocity between ego vehicle and target

ID	Num. Frames	Object Type	Object Status	Ego Status	Scene Condition	2D BB Width	2D BB Height	3D BB Height	3D BB Width	3D BB Length	OCC	POS	DIS	RVL
1	154	C	M	M	U	178	208	1.73	0.82	1.78	*	*	*	
2	154	Y	M	M	U	154	127	2.00	1.82	4.43		*	*	
3	144	P	S	M	U	16	42	1.72	0.73	0.55			*	
4	51	C	M	M	U	52	34	1.57	1.65	4.10		*	*	
5	31	C	S	M	U	52	34	1.45	1.60	4.22			*	
6	24	C	M	M	U	30	32	3.43	2.81	7.02			*	
7	63	P	M	M	U	16	30	1.63	0.40	0.83			*	
8	99	Y	M	M	D	39	39	1.81	0.59	1.89		*	*	*
9	41	P	M	M	D	25	42	1.53	0.61	0.73			*	
10	323	Y	S	M	U	25	38	1.72	0.78	1.70			*	
11	188	C	M	M	U	30	21	1.44	1.74	4.23	*	*	*	
12	41	P	M	S	D	70	105	1.63	0.66	0.89	*			
13	132	P	M	S	D	43	72	1.89	0.84	1.05	*	*		
14	112	P	M	S	D	33	66	1.84	0.78	1.03	*	*		
15	45	P	M	S	D	196	224	1.64	0.55	0.94	*			
16	264	C	M	M	U	28	24	1.40	1.54	3.36	*		*	
17	71	P	M	M	D	89	124	1.61	0.91	0.91	*			
18	125	P	M	M	D	36	62	1.64	0.88	0.49	*		*	
19	45	P	M	M	D	29	58	1.67	0.70	0.94	*		*	
20	188	P	M	M	D	31	67	1.76	0.76	1.01		*	*	

Table 4.1: *Detailed information and challenging factors for each sequence.*

object is also subject to changes (RVL).

The first frame of each selected sequence can be seen in Figure 4.2. The initial 2D BB (obtained from the projection of the initial 3D BB onto the image) is also present to identify the target object.

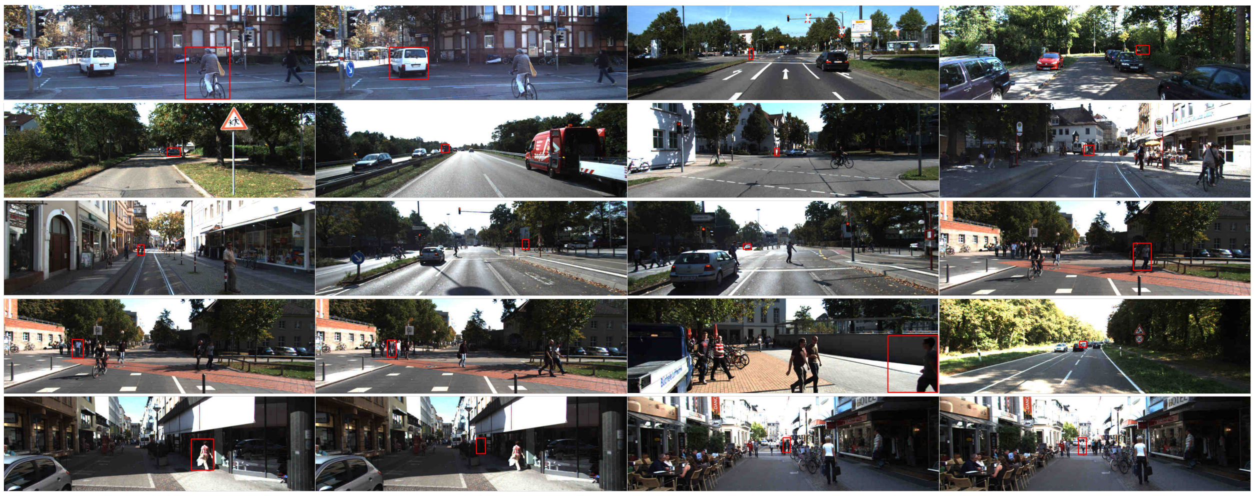


Figure 4.2: *Initial frames from the selected sequences. In a left-right, top-bottom manner, this figure represents the initial frame from each of the 20 sequences presented in Table 4.1. The displayed 2D BB in red is the projection of the initial ground truth 3D BB onto the image.*

4.2 Evaluation

In order to evaluate the presented method and the baseline methods some metrics were defined. They contemplate the error distance between the calculated and ground-truth centroids (in 3D and 2D), as well as the difference in pose (2D angle since only *yaw* rotation is considered) and instant velocity estimations. Additionally, and since all methods output objects 3D BBs, an overlap success metric is also presented.

4.2.1 Quantitative Evaluation Metrics

Localization Estimation Evaluation In the 3D case, the average centroid location error was computed for three methods: 3D-KF, 3D-MS and the proposed method (3D-Fusion). To evaluate the centroid location error in 2D, the resulting 3D BBs of the 3D tracking methods were projected into the image in order to compute and compare the corresponding 2D BBs. This metric is computed as the Euclidean distance between the calculated 3D (or 2D) centroids from the 3D (or 2D) ground truth. The average centroid location errors in 3D and 2D are given by Equation (4.1) and Equation (4.2), respectively

$$E_{3D} = \frac{1}{N}\sum_{i=1}^{N}\sqrt{(x_i - x_i^g)^2 + (y_i - y_i^g)^2 + (z_i - z_i^g)^2} \tag{4.1}$$

$$E_{2D} = \frac{1}{N}\sum_{i=1}^{N}\sqrt{(r_i - r_i^g)^2 + (c_i - c_i^g)^2} \tag{4.2}$$

In this representation, $\{r_i\ c_i\}$ refer to the 2D computed centroid and $\{r_i^g\ c_i^g\}$ denotes the 2D ground truth centroid. $\{x_i\ y_i\ z_i\}$ indicates the 3D centroid of the detected 3D BB. Similarly, $\{x_i^g\ y_i^g\ z_i^g\}$ represents the 3D centroid obtained from ground truth. N is the total number of frames/scans in the given sequence.

Orientation Estimation Evaluation The ground truth orientation of the object is given only by the *yaw* angle (The yaw angle describes the direction of the object, given by rotation around Y-axis in camera coordinates in the KITTI dataset [30]). The orientation error can be computed by

$$E_{\varphi} = \frac{1}{N}\sum_{i=1}^{N}\left|\arctan|\frac{\overrightarrow{\varphi} \times \overrightarrow{\varphi}^g}{\overrightarrow{\varphi} \cdot \overrightarrow{\varphi}^g}|\right| \tag{4.3}$$

$\overrightarrow{\varphi}^g$ is the ground-truth orientation of the object as seen in Figure 4.3.

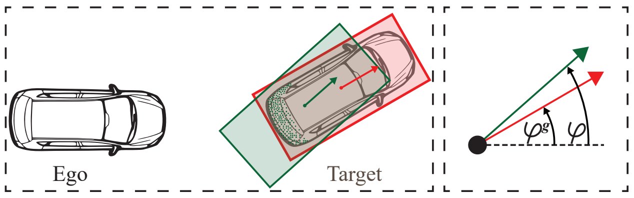


Figure 4.3: *Pose error representation.*

Velocity Estimation Evaluation To estimate the instant velocity error, a simple subtraction between the current velocity estimated as the norm of the three velocity components and the ground truth velocity in the same frame is calculated (either on $^{km}/_{h}$ or $^{m}/_{s}$). The average absolute error can be thus represented by

$$E_{\vartheta} = \frac{1}{N}\sum_{i=1}^{N} |\ \vartheta - \vartheta^{g}| \tag{4.4}$$

Overlap Success Evaluation The overlap success rate (also referred to as the intersection-over-union metric) in 3D can be defined as the volume between the computed 3D BB and the ground truth 3D BB. A visual representation can be found in Figure 4.4.

To calculate the intersection between two 3D BBs, the overlapping area between the top-down 2D projection of the 3D BBs is obtained ($overlap_A$). Afterwards, and given that each 3D BB base might be at different heights, the difference between the height of the obtained 3D BB (f_{ob}) and the ground truth 3D BB (f_{gt}) is calculated. The considered height is then $\hat{h} = h - |f_{ob} - f_{gt}|$. The intersection volume is obtained by having $Inter_{vol} = overlap_A \times \hat{h}$. The union volume can be calculated based on the total and intersection volumes as $Union_{vol} = (2 \times \text{3D BB}_{vol}) - Isect_{vol}$.

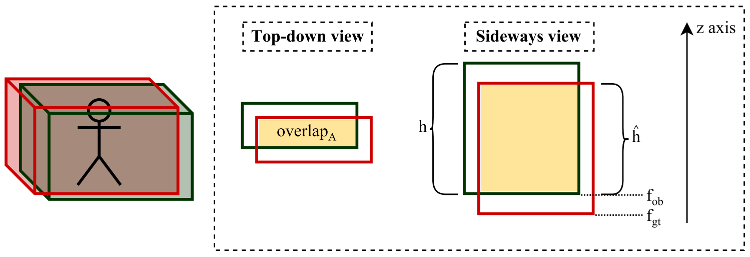


Figure 4.4: *Bounding boxes overlap calculation.*

The overlap percentage in a given frame can be given by

$$O_{3D} = \frac{Inter_{vol}}{Union_{vol}} \times 100 \tag{4.5}$$

In order for an object tracker to be considered correct, the overlap ratio O_{3D} must exceed a standard threshold of 25% [61–63]. The average of the overlap ratio along the sequence can be used as a metric for the tracking performance of each approach.

4.3 Baseline Methods

For evaluation purposes two generative 3D object tracking methods were implemented as baselines for the evaluation purpose. These baseline methods take LiDAR PCDs as the input and, like in the proposed approach, the ground points are removed. The initial position of the object 3D BB is again also known and its size assumed constant. Additionally, the point model representation was also considered.

4.3.1 Baseline KF 3D Object Tracker (3D-KF)

A 3D CA KF with a gating *Data Association* (DA) method is used for tracking of the object centroid in the consecutive PCDs. Like in the proposed approach, the state of the filter is $x_k = [x_k\ y_k\ z_k\ \dot{x}_k\ \dot{y}_k\ \dot{z}_k\ \ddot{x}_k\ \ddot{y}_k\ \ddot{z}_k]^T$. The discrete time process model and the measurement model of the system are given by Equations (3.25) and (3.26), respectively. To further eliminate outliers and provide more reliable information, the search area is limited to a gate (defined as twice the area of the projection of the object 3D BB onto the ground) near the the predicted KF location from the previous-step from $x_t = A_t \times x_{t-1}$). If no measurement is available inside the gate area, the predicted KF value is used.

4.3.2 Baseline MS 3D Object Tracker (3D-MS)

In turn, the baseline 3D MS approach works by iterative repetition of the procedures described in the cycle present in Section 3.4.2. For the baseline MS method a maximum of 3 iterations was chosen to be the threshold.

In both cases, the object orientation is achieved by subtracting the estimated current and the previous location of the object (resulting in a vector pointing towards the direction of movement), with the velocity estimation being obtained by subtraction of the current and previous centroid over the time step.

4.4 Results

In the previous sections, the KITTI dataset was described, and information about the selected sequences was detailed in Table 4.1 and Figure 4.2. The evaluations metrics used for the evaluation of the proposed methods were also described. In the present section, the evaluation results are presented in Table 4.2 for the proposed fusion, baseline KF and baseline MS approaches, respectively. In these tables, for each sequence ($S_{\#}$), the errors are discriminated as distance to object ground truth location in 3D (E_{3D}) in meters and 2D (E_{2D}) in pixels, difference to ground truth pose (E_{φ}) in radians and to velocity (E_{ϑ}) in $^{km}/_{h}$. The average 3D BB overlap success ($\bar{O}_{3D}$) in percentage and data processing rate or *runtime* per frame (*Rtime*) in FPS are also displayed. These 20 sequences are composed by a total of 2295 scans.

In the table, for the baseline KF method, a dashed row (—) means that in addition to losing track of the target object, no points from the PCD were detected inside the gating area and thus numerical calculations inside the tracker failed. In these cases, no error data is considered. Consequently, such sequences for the baseline KF method will not be represented in the figures.

$S_{\#}$	E_{3D}	E_{2D}	E_{φ}	E_{ϑ}	$\bar{O}_{3D}$	*Rtime*	E_{3D}	E_{2D}	E_{φ}	E_{ϑ}	$\bar{O}_{3D}$	*Rtime*	E_{3D}	E_{2D}	E_{φ}	E_{ϑ}	$\bar{O}_{3D}$	*Rtime*
1	0.304	9.663	0.412	0.561	61.040	4.857	0.210	9.186	0.287	5.118	68.208	11.647	—	—	—	—	—	—
2	1.979	11.762	0.413	1.555	29.831	5.650	1.845	19.196	0.443	5.940	30.556	12.183	16.794	465.322	0.799	28.774	1.432	12.346
3	0.233	6.414	1.467	0.434	40.310	5.193	0.155	7.221	1.885	0.091	52.023	6.271	0.182	29.133	1.696	0.902	91.832	6.610
4	1.002	758.314	1.281	3.291	34.849	2.408	0.910	1340.386	2.079	3.161	33.276	6.060	13.063	1549.315	1.365	16.968	7.522	5.964
5	0.784	11.190	1.333	2.090	25.418	3.742	0.771	19.650	1.721	-2.477	25.098	6.384	11.754	316.650	1.427	3.041	4.887	6.394
6	1.715	29.824	0.109	24.895	37.620	4.065	2.239	25.029	2.837	3.366	32.808	6.913	8.900	164.293	2.090	53.390	29.794	6.511
7	0.311	4.784	0.147	0.700	40.977	5.632	0.249	4.580	1.583	-0.002	46.584	7.552	0.500	11.811	1.439	5.394	56.609	7.689
8	0.626	10.918	0.190	19.934	41.914	4.457	0.528	12.840	0.452	23.549	45.937	6.451	—	—	—	—	—	—
9	0.227	10.764	0.485	1.421	49.123	4.231	0.340	19.309	0.317	0.773	32.324	6.561	0.274	12.194	0.219	5.765	82.287	6.175
10	0.596	21.767	1.483	-0.571	25.427	5.135	0.571	22.888	1.639	-2.591	26.498	6.907	9.531	242.133	1.648	0.451	1.214	6.837
11	1.891	15.824	0.837	21.273	19.037	7.622	19.293	315.599	1.777	19.996	1.283	13.404	27.349	459.618	1.683	32.227	0.731	12.910
12	0.407	13.727	0.222	1.377	39.937	3.853	0.186	15.129	2.502	1.390	58.118	6.426	0.632	46.630	2.536	5.914	52.303	6.838
13	0.227	12.201	0.220	0.395	58.834	4.092	0.233	17.468	2.289	1.943	59.366	8.114	1.290	52.518	2.317	7.575	19.287	13.229
14	0.234	9.532	0.175	0.284	58.225	6.398	0.181	9.486	2.396	1.504	65.096	12.777	2.956	182.408	2.341	6.933	5.216	7.516
15	0.184	12.702	0.215	2.112	55.894	3.255	0.104	8.964	1.098	1.745	71.949	6.330	—	—	—	—	—	—
16	1.483	14.441	0.230	0.811	27.938	3.403	1.368	8.101	0.054	3.543	30.713	6.614	—	—	—	—	—	—
17	0.233	17.806	0.200	0.565	57.456	3.436	0.161	13.633	0.167	0.646	67.605	6.110	2.229	254.426	0.646	5.896	14.849	6.297
18	0.204	12.005	0.240	0.802	55.966	4.976	0.189	10.834	0.223	1.243	58.793	8.148	—	—	—	—	—	—
19	0.140	4.822	0.123	1.575	69.513	4.345	0.103	4.515	0.183	0.707	75.319	5.787	2.424	24.296	0.425	5.964	3.548	5.422
20	0.179	24.363	0.151	0.279	64.255	3.386	0.159	17.848	0.332	1.021	67.532	10.641	0.819	46.243	0.347	5.713	50.177	5.744

Table 4.2: *Error results for the proposed approach (left), baseline MS (middle) and baseline KF (right).*

As stated in Equation (4.2) the 2D location error is based on an Euclidean distance. While low on average, the proposed method presents a higher error value in sequence 4. This is due to the fact the 2D estimated and ground truth centroids are obtained by projecting the estimated and ground truth 3D BB onto the image and then obtaining the centroid of the corresponding 2D BBs. However, in sequence 4, the target object moves close to the ego vehicle in the opposite direction, and the projection of the 3D BBs is not correctly done, leading to the calculated 2D centroids not correctly representing the output of the tracker.

The proposed method processes incoming data at an average rate of 4.5 FPS, and thus is not suited for real-time applications (data acquisition is done at a rate of 10 FPS). This data was obtained by running the approach on a Windows 10, 64-bit, 2.3 GHz Intel® Core™ i7 3610QM processor laptop with 6GB of RAM in Matlab R2015b (in a simulation environment). However, both baseline methods also perform data processing, on average, below the 10 FPS threshold to belong to real-time processing applications (8.06 FPS for the baseline MS and 7.77 FPS for the baseline KF when it does not fail). A dedicated implementation in a language such as `C++` could potentially accelerate the process up to a point where the framework became usable in real-time systems.

The critical module of the proposed approach in terms of computational power is the one responsible for the 2D tracking, because it has to keep track of the object colour histograms, initializing, updating or removing them for each occupied cell KF, recalculating the confidence map for each frame. In addition, and instead of the 2D MS tracking module for the image domain, well known computer vision based object trackers such as SCM [64] and ASLA [65] were considered. These approaches have also been used in benchmarks such as the Visual Tracker Benchmark [66]. However, SCM and ASLA alone run at about 1 and 6.5 FPS, respectively; 3D LiDAR based solution generally have superior performance over approaches in the computer vision field since both the ego vehicle and target object(s) are often moving. These size and pose changes, also reflected on the captured RGB images, can mislead purely visual object tracker approaches. By mapping the 3D object points onto the image, colour information about the object is kept and a robust 2D object tracker was implemented. In addition, some of the computation time is also due to utilizing non-optimized built-in Matlab functions such as `inpolygon` that require more computation time.

In Figures 4.5 to 4.8 a graphical comparison of each result for each sequence is done. Figure 4.9 represents the percentage of scans for which the proposed method was considered to be successful according to the overlap success metric given in Equation (4.5).

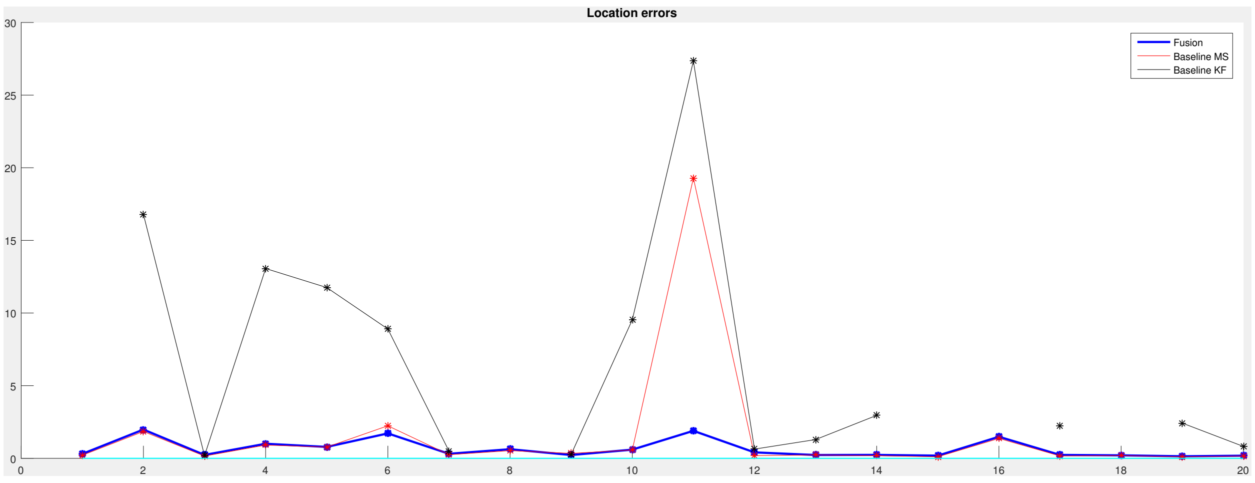


Figure 4.5: *Location error comparison plot (in meters). The proposed fusion approach produces in most cases a smaller location error than the baseline methods (with few exceptions for the baseline MS).*

The proposed approach is able to provide relatively low average location errors for all sequence, with the highest (average) location error of 1.979m for sequence 2 (in which the target object is a van). In this same sequence, the lowest average error was obtained by the baseline MS at 1.845m, with the highest average error being 16.794m provided by the baseline KF.

In addition to the cases identified by a "—" in Table 4.2, the baseline KF provided the worst results all around, with the exception being sequence 9 where the average location error was lower than the one provided by the baseline MS (as well as the average overlap success for that sequence); however, the difference is not significant. One of the reasons as to why this might happen can be related to the gating area; having a fixed size gate (either small or big) does not suit all cases. By being too small, the associated KF might not have enough target object points to detect, and thus the predictions would tend to suffer from a "drag" effect, lagging behind the tracked object until it effectively loses track of the object. On the other hand, a gate too big might produce a different effect, since the centroid is calculated as the average of all the detected points inside the gating area.

In turn, the proposed approach applies a MS algorithm to keep track of object points, and further corrects the estimated location based on this observation. Errors based on the proposed approach and the baseline MS tend to be low and close to each other due to this, with some exceptions such as presented in sequence 11. In this case, some challenges were verified: the target object, initially stopped at a stop light and partially occluded, starts moving and turning; the planar ground assumption made for the ground removal function is not well suited for the slanted road present in this sequence. However, due to the use of the applied CA KF the proposed approach is able to track the car, even though with a higher than the average associated location error (with the average overlap success percentage falling below the defined threshold for all cases).

Figure 4.6 represents the average pose errors for each method, with the proposed approach yielding the lowest errors. The associated CA KF makes the proposed approach more robust as it tends to rely less on the direct observation from the 3D MS module when the object points count is lower and thus avoiding sudden pose changes in between consecutive scans. Stopped vehicles and pedestrians offer the greatest challenges when estimating their orientation.

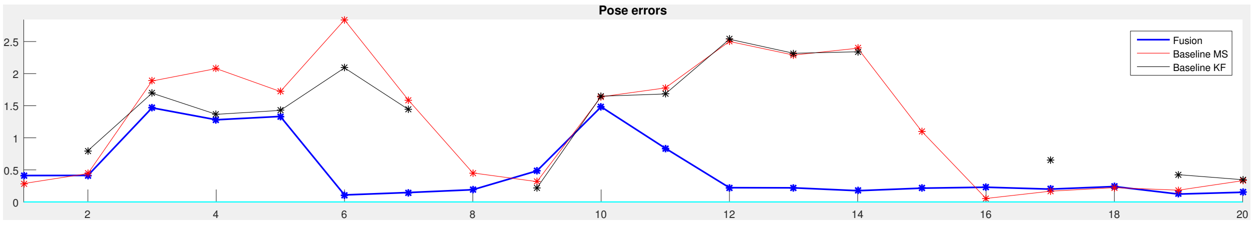


Figure 4.6: *Pose error comparison plot (in radians). The proposed fusion approach handles pose variation better than the baseline methods.*

Figure 4.7 shows the average velocity errors. Both instant velocity values, v and v^g, were obtained by having the spatial shift between two consecutive ground truth 3D centroids over the elapsed time step. Since a point model object representation was considered, changes in the perceived object pose influence the measured instant velocity estimation even with the application of the CA KF.

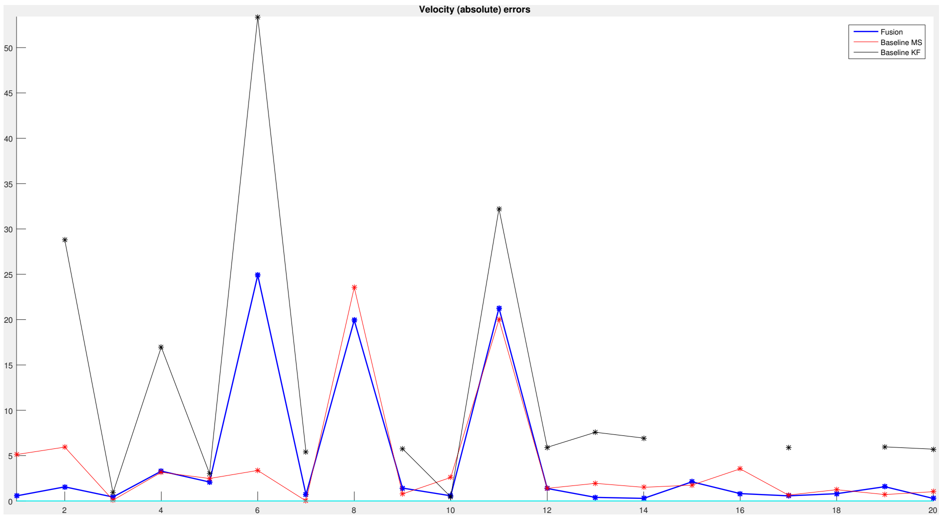


Figure 4.7: *Velocity error comparison plots (in $^{km}/_{h}$). The plot represents the difference between the measured and ground-truth as an absolute difference. The proposed fusion approach yields a better instant velocity estimation when compared to the baseline methods.*

Figure 4.8 shows the average 3D BB overlap ratios for each method. For this particular metric, the MS baseline method obtained an overall average better than the other cases; however, just as demonstrated by the average localization error, in sequence 11 the MS baseline tracker loses the object.

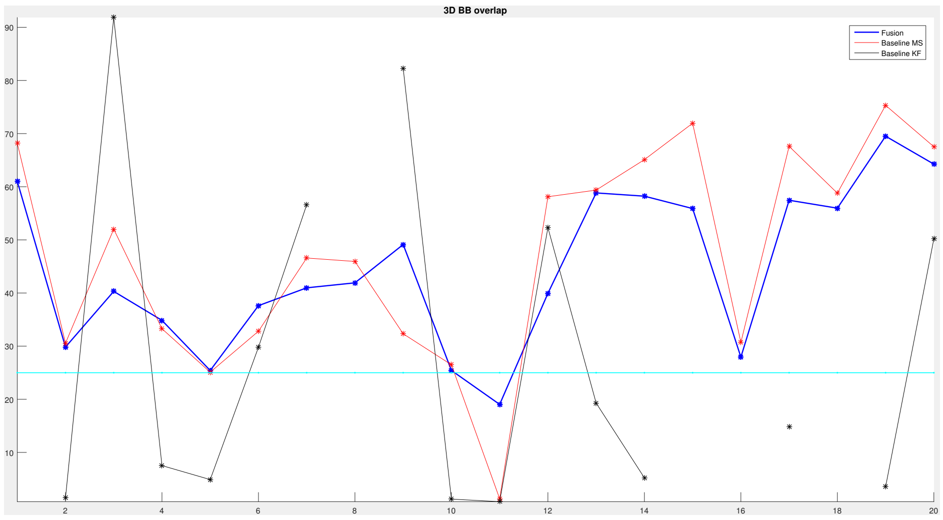


Figure 4.8: *3D BB overlap comparison plot (in percentage). An approach yielding an average overlap ratio greater or equal to 25% is considered as successfully tracking an object.*

In Figure 4.9 the 3D BB precision plot is present. Based on the information present in Figure 4.8, the precision plot specifies the percentage of scans for which the method provides an overlap percentage O_{3D} greater than the defined threshold by cumulatively summing each time that it happens. In doing so, the method was able to achieve a ratio of $O_{3D} > 25\%$ for 77.69% of the scans.

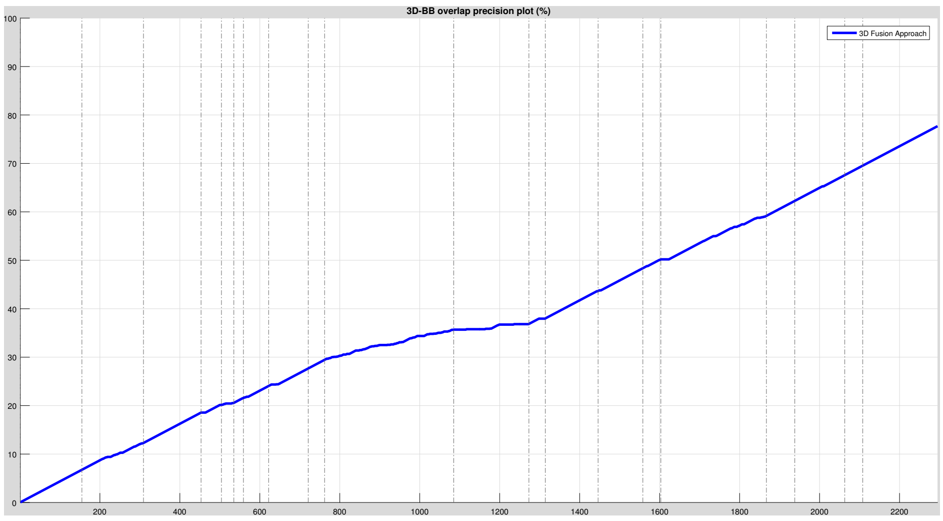


Figure 4.9: *3D BB overlap precision plot.*

4.4.1 Qualitative Evaluation

In the current section some examples of the proposed approach in action are presented. In a given scan in a sequence, each figure (composed by top and bottom sub-figures) represents the 2D and 3D tracking modules, respectively.

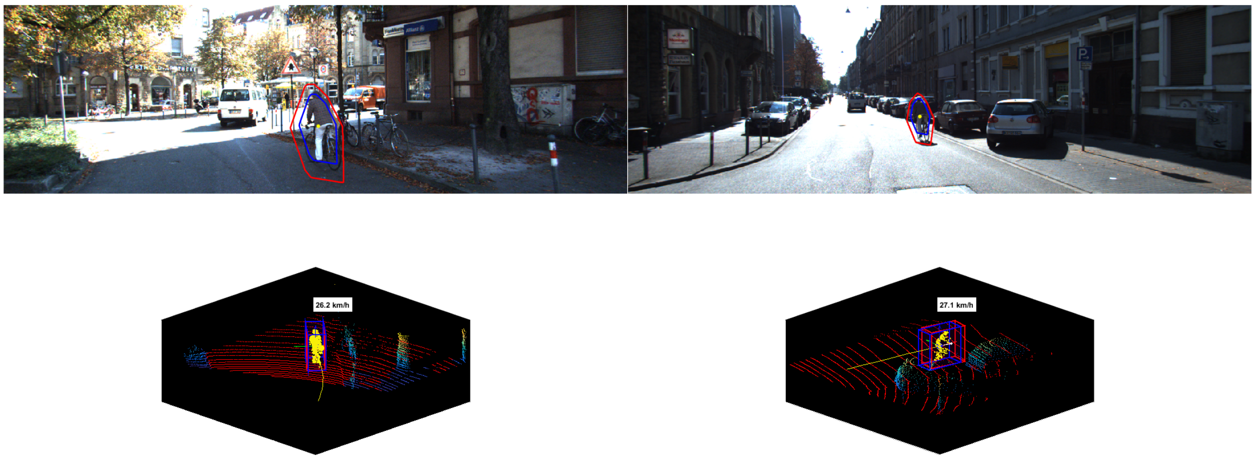


Figure 4.10: *Sequence 1, tracking modules in scans 40 (left) & 130 (right).*

In sequence 1 the cyclist is visible in all scans, is never occluded and undergoes a change in terms of lighting conditions. From the 3D tracker point of view, this sequence represents a simple scenario that does not offer much of a challenge.

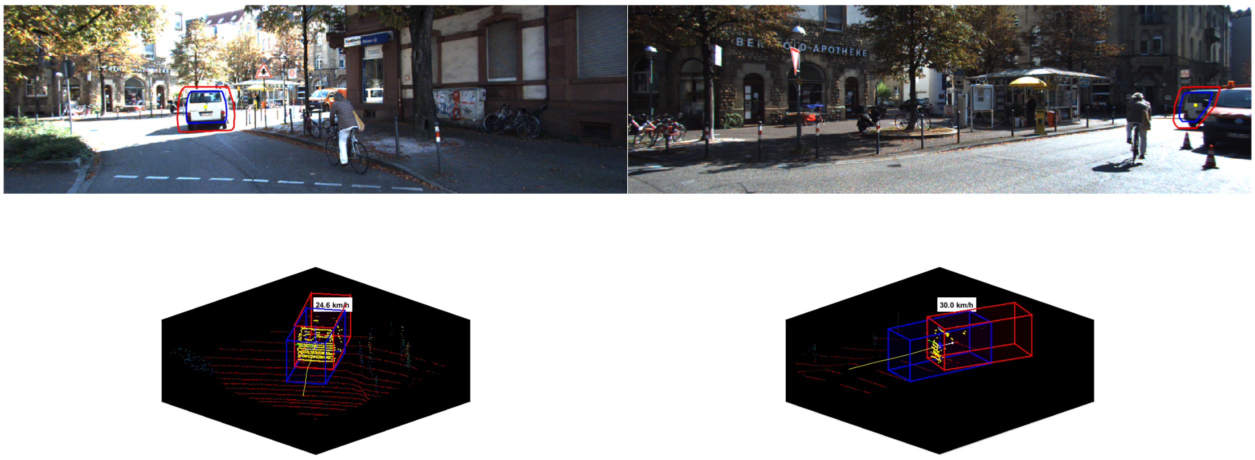


Figure 4.11: *Sequence 1, tracking modules in scans 34 (left) & 80 (right).*

Sequence 2 represents a moving van as the target object. The tracked van is initially estimated to be moving at a speed lower than its actual speed, seeing as the associated tracking CA KF takes some time to adjust and provide good velocity estimations. In addition, the van also undergoes lighting changes and is severely occluded by a parked van. Both in 3D and 2D the proposed method is able to keep tracking the object, even with a low number of object points detected. In the bottom right sub-figure a white dot can be seen, representing the CA KF estimation for the object centroid in the next scan.

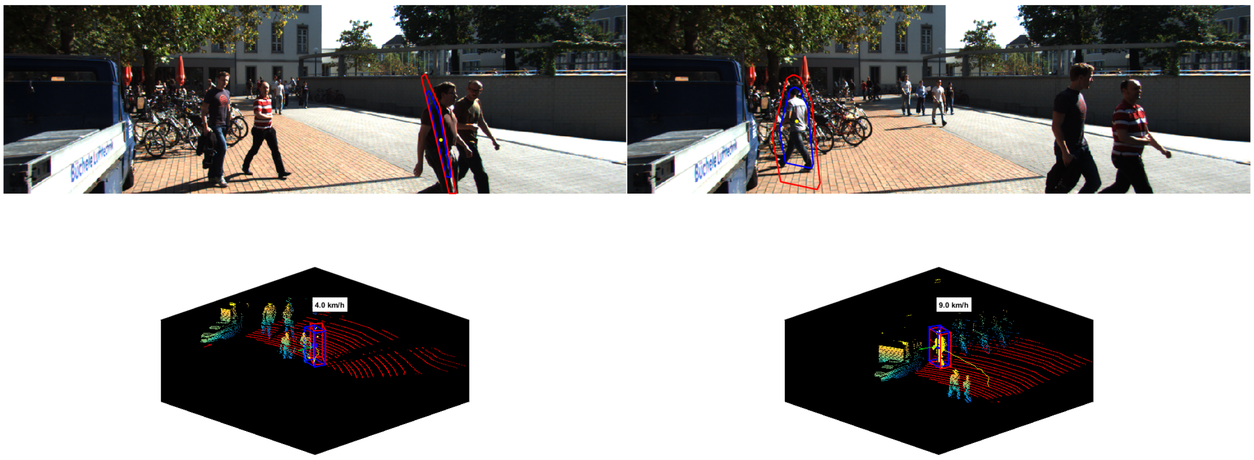


Figure 4.12: *Sequence 15, tracking modules in scans 30 (left) & 57 (right).*

In sequence 15, the target object is a walking pedestrian that undergoes nearly full occlusion due to a pedestrian closer to the ego vehicle walking in the opposite direction. Such a scenario is frequent in downtown and rural areas. In this particular example, the proposed tracker can be seen making use of the prediction for the next frame to balance out the low number of detected object points, both maintaining the correct pose as well as not jumping to the nearby pedestrians.

4.5 Benchmark

The 20 considered sequences for testing were selected amongst a group of 50 sequences extracted from the KITTI dataset. These total 50 sequences compose the 3D Object Tracking in Driving Environments (or 3D-OTD) proposed benchmark. Most of the existing object tracking benchmark tools available are based on monocular or stereo cameras approaches or focused on data association problems. Hence, the created benchmark is aimed at 3D object appearance modelling.

As mentioned in Section 4.1.2, instead of using object tracklets the full track of each object was extracted. Since information in the dataset is given as tracklets, the selection of the sequences had to be based on objects who were never fully occluded, *i.e.*, from the first to the last frame of an extracted sequence, the respective target object will never go under full occlusion. In addition, if a given scenario in the dataset features several candidate target objects, then a sequence will be extracted for each object. Based on these considerations, 50 sequences were extracted to create the benchmark. The general specification of these sequences and the considered challenging factors are as defined and described as in Section 4.1.2. As a starting point, the two baseline methods here presented were also considered the baseline evaluators in the benchmark.

The full benchmark, with the total 50 sequences, can be observed in Appendix C as a workshop paper. This paper has been accepted in IEEE Intelligent Transportation Systems Conference (ITSC) 2016, in the Int. Workshop on Advanced Perception, Machine Learning and Datasets workshop.

CHAPTER

5 CONCLUSIONS

Contents

With the increasing effort being put in developing robust ADASs, autonomous driving has proved to be a successful concept with a lot of direct benefits, from safety (pedestrian detection and emergency braking) and Quality of Life (parking assistance) to energy savings. Big automotive companies are making efforts to make driverless cars available to the public in a near future. An ADAS system can be composed of several modules, from perception systems to actuators. By moving from research and experimental environments to the real world, perception modules must become more robust and able to deal with data acquisition, modelling and scene interpretation. Object tracking is one of the critical components of these perception modules, responsible by the task of following each target object in the environment. Future predictions of the target object trajectory are very useful since the ego vehicle can understand what is most likely to happen in the next time step and plan to act accordingly.

Thus, the main goal of this thesis was to develop a fusion based 3D object tracker. The fusion of information from different sensor technologies aimed to maximize the benefits of the dense information present in the RGB image as well as the inherent spatial information provided by the LiDAR sensor. This resulted in the two main contributions of this thesis:

- A robust, online, 2D/3D fusion based approach was developed. It takes both sequential 2D RGB images and 3D PCDs as inputs as well as the location of the ego vehicle, given by an INS. An assumption was made to consider that an initial 3D BB of the target object is known known and outputted by a previous object detection and/or segmentation module. From this data two independent and parallel MS algorithms are applied to each considered data space (2D image and 3D PCD) to detect and track the target object. Following this step, a robust CA KF is applied to fuse information from both 2D and 3D tracking modules, and a final CA KF is considered for 3D tracking purposes.

- To test the developed work, a series of experiments were designed with different types of target objects, scene conditions and associated challenging factors (inherent or relative to the surrounding scene) to create a benchmark. The benchmark can be used to evaluate 3D object trackers, even if the base assumptions or theoretical backgrounds are different. Based on this benchmark, a subset of 20 sequence were selected. An analysis of the obtained results of testing the proposed approach with these sequences was provided, demonstrating the viability of the proposed object tracker.

5.1 Future Work

The proposed approach is able to achieve satisfying results in its current state. However, and as aforementioned, the execution times are not high enough to process incoming data in real-time. One direction of future work could be the improvement of the existing code as well as the study of different methodologies or functions that could further reduce computation times. Taking advantage of the fact that the 3D and 2D trackers run independently and in parallel, exploration of parallel computing architectures could also be considered.

In the current method, the point model was used for object representation. Even though the applied KFs were able to apply trajectory correction when the information was sparse, different object representation models could also be considered. One such hypothesis could be to apply 3D reconstruction of the shape of the target object. In consecutive frames, since target object location and orientation are known in real world coordinates in each scan, reconstruction of the object shape can be obtained by aligning the detected object points (or the current 3D BB) with reference information (*e.g.* last scan's 3D BB). Initial attempts at this reconstructions were made, with promising results obtained as observable in Figure 5.1. However, due to time constraints and the lack of robustness of the reconstruction module (in some cases, the module failed to correctly reconstruct the shape due to wrong rotation values in the alignment process), it was not included in the final framework.

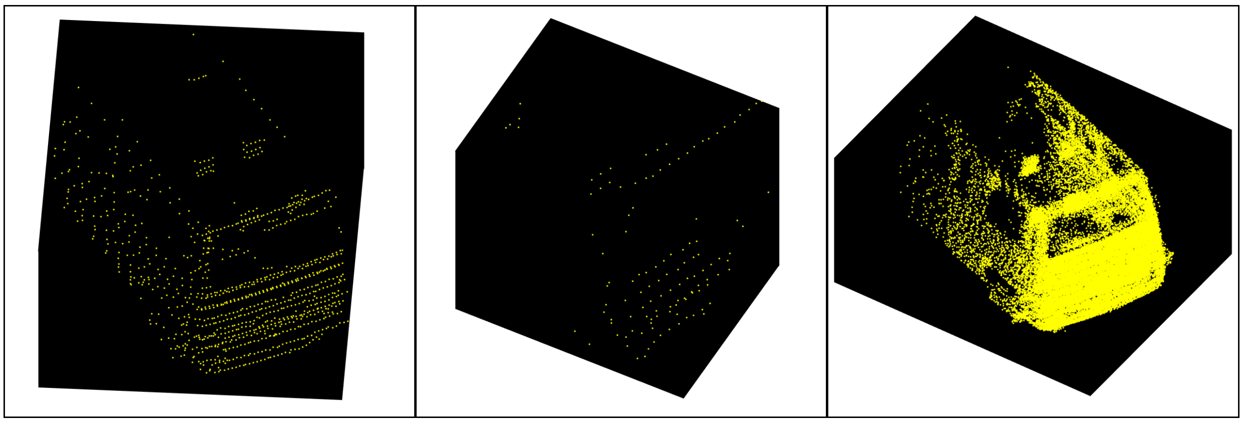


Figure 5.1: *Object shape reconstruction result. The sub-figure on the left represents the detected object points in the first scan from sequence 2 (940 total). The middle sub-figure shows the detect object points for scan 80 in the same sequence without object reconstruction. Without reconstruction, the direct readings from the LiDAR provide only 49 points in the entire PCD. By registration and accumulation of all the detected object points from previous frames without any further processing, the accumulated model of the object can be used when less points are detected to overcome occlusion problems. This is highlighted in the right sub-figure; for the same scan as the middle sub-figure (scan 80), the accumulated object model has much more information present (38163 points) and can be used in its stead.*

CHAPTER

6

BIBLIOGRAPHY

[1] World Health Organization (WHO), *Global status report on road safety 2015.* World Health Organization, 2015, no. 978 92 4 156506 6, accessed: 18-07-2016. [Online]. Available: http://www.who.int/violence_injury_prevention/road_safety_status/2015/en/

[2] ASIRT, “Annual global road crash statistics,” Accessed: 04-08-2016. [Online]. Available: http://asirt.org/initiatives/informing-road-users/road-safety-facts/road-crash-statistics

[3] J. Jackisch, D. Sethi, F. Mitis, T. Szymañski, and I. Arra, “European facts and the global status report on road safety 2015,” World Health Organization, Tech. Rep., 2015, accessed: 18-07-2016. [Online]. Available: http://www.euro.who.int/___data/assets/pdf_file/0006/293082/European-facts-Global-Status-Report-road-safety-en.pdf

[4] National Highway Traffic Safety Administration (NHTSA), “Early estimate of motor vehicle traffic fatalities in 2015,” 2015, accessed: 18-07-2016. [Online]. Available: https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812269

[5] M. Pines, “Top 3 causes of car accidents,” Accessed: 04-08-2016. [Online]. Available: http://www.drivers.com/article/1173/

[6] S. G. Klauer, T. A. Dingus, V. L. Neale, J. D. Sudweeks, and D. J. Ramsey, “The impact of driver inattention on near-crash/crash risk: An analysis using the 100-car naturalistic driving study data,” Virginia Tech Transportation Institute, Tech. Rep., April 2006.

[7] National Highway Traffic Safety Administration (NHTSA), “Critical reasons for crashes investigated in the national motor vehicle crash causation survey,” 2015, accessed: 18-07-2016. [Online]. Available: https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812115

[8] L. Danielsson, *Tracking and radar sensor modelling for automotive safety systems*, ser. Doktorsavhandlingar vid Chalmers tekniska högskola. Ny serie, no: 3064. Institutionen för signaler och system, Signalbehandling, Chalmers tekniska högskola, 2010, 279.

[9] J. Thalen, “ADAS for the Car of the Future ,” Universiteit Twente, Technical Report, April/June 2006.

[10] S. A. Beiker, “Legal aspects of autonomous driving,” Santa Clara University, Tech. Rep., 12 December 2012.

[11] A. Geiger, M. Lauer, C. Wojek, C. Stiller, and R. Urtasun, “3D Traffic Scene Understanding From Movable Platforms,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 5, pp. 1012–1025, May 2014.

[12] R. R. Murphy, *Introduction to AI Robotics.* The MIT Press, 2000, ISBN: 9780262133838.

[13] X. Chen, K. Kundu, Y. Zhu, A. G. Berneshawi, H. Ma, S. Fidler, and R. Urtasun, “3d object proposals for accurate object class detection,” in *Advances in Neural Information Processing Systems*, 2015, pp. 424–432.

[14] A. Asvadi, P. Peixoto, and U. Nunes, *Two-Stage Static/Dynamic Environment Modeling Using Voxel Representation.* Cham: Springer International Publishing, 2016, pp. 465–476. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-27146-0_36

[15] M. Brahmi, K.-H. Siedersberger, A. Siegel, and M. Maurer, “Reference systems for environmental perception: Requirements, validation and metric-based evaluation,” *Automotive Systems Engineering*, pp. 205–211, 2013.

[16] P. E. Ross, “Tesla’s Autopilot May Have Saved A Life,” July 2016, Accessed: 04-08-2016. [Online]. Available: http://spectrum.ieee.org/cars-that-think/transportation/self-driving/teslas-autopilot-may-have-saved-a-life-cars-that-think

[17] E. Ackerman, “Fatal Tesla Self-Driving Car Crash Reminds Us That Robots Aren’t Perfect,” July 2016, Accessed: 04-08-2016. [Online]. Available: http://spectrum.ieee.org/cars-that-think/transportation/self-driving/fatal-tesla-autopilot-crash-reminds-us-that-robots-arent-perfect

[18] S. Sivaraman and M. M. Trivedi, “Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 4, pp. 1773–1795, Dec 2013.

[19] PointGrey, “Flea 3,” Accessed: 04-08-2016. [Online]. Available: https://www.ptgrey.com/flea3-usb3-vision-cameras

[20] ——, “Bumbleebee 2,” Accessed: 04-08-2016. [Online]. Available: https://www.ptgrey.com/bumblebee2-firewire-stereo-vision-camera-systems

[21] A. Mukhtar, L. Xia, and T. B. Tang, “Vehicle detection techniques for collision avoidance systems: A review,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 5, pp. 2318–2338, Oct 2015.

[22] Z. Sun, G. Bebis, and R. Miller, “Monocular precrash vehicle detection: features and classifiers,” *IEEE Transactions on Image Processing*, vol. 15, no. 7, pp. 2019–2034, July 2006.

[23] A. Klaus, M. Sormann, and K. Karner, “Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure,” in *18th International Conference on Pattern Recognition (ICPR’06)*, vol. 3, 2006, pp. 15–18.

[24] BOSCH, “Bosch Engineering Long-Range-Radar LRR3,” Accessed: 04-08-2016. [Online]. Available: http://www.bosch-engineering.de/media/de/pdfs/einsatzgebiete_1/produktdatenblaetter/120903_LRR3_EN_V05_final.pdf

[25] SICK, “Lms200/211/221/291 laser measurement systems,” Accessed: 04-08-2016. [Online]. Available: http://sicktoolbox.sourceforge.net/docs/sick-lms-technical-description.pdf

[26] HOKUYO, “Scanning range finder (sokuiki sensor),” Accessed: 04-08-2016. [Online]. Available: https://www.hokuyo-aut.jp/02sensor/07scanner/urg_04lx.html

[27] Velodyne, “Velodyne LiDAR™ HDL-64E,” Accessed: 04-08-2016. [Online]. Available: http://velodynelidar.com/hdl-64e.html

[28] M. Montemerlo, J. Becker, S. Bhat, H. Dahlkamp, D. Dolgov, S. Ettinger, D. Haehnel, T. Hilden, G. Hoffmann, B. Huhnke, D. Johnston, S. Klumpp, D. Langer, A. Levandowski, J. Levinson, J. Marcil, D. Orenstein, J. Paefgen, I. Penny, A. Petrovskaya, M. Pflueger, G. Stanek, D. Stavens, A. Vogt, and S. Thrun, *Junior: The Stanford Entry in the Urban Challenge*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 91–123. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-03991-1_3

[29] S. Kammel, J. Ziegler, B. Pitzer, M. Werling, T. Gindele, D. Jagzent, J. Schöder, M. Thuy, M. Goebl, F. von Hundelshausen, O. Pink, C. Frese, and C. Stiller, *Team AnnieWAY’s Autonomous System for the DARPA Urban Challenge 2007*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 359–391. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-03991-1_9

[30] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, “Vision meets robotics: The KITTI dataset,” *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, aug 2013. [Online]. Available: http://dx.doi.org/10.1177/0278364913491297

[31] A. Petrovskaya and S. Thrun, “Model based vehicle detection and tracking for autonomous urban driving,” *Autonomous Robots*, vol. 26, no. 2, pp. 123–139, 2009. [Online]. Available: http://dx.doi.org/10.1007/s10514-009-9115-1

[32] V. LiDAR™, Accessed: 04-08-2016. [Online]. Available: http://velodynelidar.com/

[33] Google, “Google Self-Driving Car Project,” Accessed: 04-08-2016. [Online]. Available: https://www.google.com/selfdrivingcar/

[34] C. Premebida, G. Monteiro, U. Nunes, and P. Peixoto, “A lidar and vision-based approach for pedestrian and vehicle detection and tracking,” in *2007 IEEE Intelligent Transportation Systems Conference*, Sept 2007, pp. 1044–1049.

[35] G. Zhao, X. Xiao, J. Yuan, and G. W. Ng, “Fusion of 3d-lidar and camera data for scene parsing,” *Journal of Visual Communication and Image Representation*, vol. 25, no. 1, pp. 165 – 183, 2014, visual Understanding and Applications with RGB-D Cameras. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1047320313001211

[36] D. Lee, “Producing 3d point clouds with a stereo camera in opencv,” April 2014, Accessed: 04-08-2016. [Online]. Available: https://erget.wordpress.com/2014/04/27/producing-3d-point-clouds-with-a-stereo-camera-in-opencv/

[37] M. Pijpers, “Sensors in ADAS,” Universiteit Twente, Technical Report, January 2007.

[38] A. W. M. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah, “Visual tracking: An experimental survey,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 7, pp. 1442–1468, July 2014.

[39] H. Yang, L. Shao, F. Zheng, L. Wang, and Z. Song, “Recent advances and trends in visual tracking: A review,” *Neurocomputing*, vol. 74, no. 18, pp. 3823 – 3831, 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0925231211004668

[40] A. Yilmaz, O. Javed, and M. Shah, “Object tracking: A survey,” *ACM Comput. Surv.*, vol. 38, no. 4, Dec. 2006. [Online]. Available: http://doi.acm.org/10.1145/1177352.1177355

[41] X. Li, W. Hu, C. Shen, Z. Zhang, A. Dick, and A. V. D. Hengel, “A survey of appearance models in visual object tracking,” *ACM Trans. Intell. Syst. Technol.*, vol. 4, no. 4, pp. 58:1–58:48, Oct. 2013. [Online]. Available: http://doi.acm.org/10.1145/2508037.2508039

[42] M. Kristan, J. Matas, A. Leonardis, M. Felsberg, L. Cehovin, G. Fernandez, T. Vojir, G. Hager, G. Nebehay, R. Pflugfelder, A. Gupta, A. Bibi, A. Lukezic, A. Garcia-Martin, A. Saffari, A. Petrosino, and A. S. Montero, "The visual object tracking vot2015 challenge results," in *2015 IEEE International Conference on Computer Vision Workshop (ICCVW)*, Dec 2015, pp. 564–586.

[43] Y. Wu, J. Lim, and M. H. Yang, "Object tracking benchmark," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1834–1848, Sept 2015.

[44] C.-C. Wang, C. Thorpe, and S. Thrun, "Online simultaneous localization and mapping with detection and tracking of moving objects: Theory and results from a ground vehicle in crowded urban areas," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2003.

[45] P. Morton, B. Douillard, and J. Underwood, "An evaluation of dynamic object tracking with 3d lidar," in *Proceedings of Australasian Conference on Robotics and Automation (ACRA)*, 2011.

[46] D. Held, J. Levinson, and S. Thrun, "Precision tracking with sparse 3d and dense color 2d data," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, May 2013, pp. 1138–1145.

[47] A. Dewan, T. Caselitz, G. D. Tipaldi, and W. Burgard, "Motion-based detection and tracking in 3d lidar scans," in *ICRA*, 2016.

[48] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, Jun. 1981. [Online]. Available: http://doi.acm.org/10.1145/358669.358692

[49] A. Vatavu, R. Danescu, and S. Nedevschi, "Stereovision-based multiple object tracking in traffic scenarios using free-form obstacle delimiters and particle filters," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 1, pp. 498–511, Feb 2015.

[50] A. Ošep, A. Hermans, F. Engelmann, D. Klostermann, M. Mathias, and B. Leibe, "Multi-scale object candidates for generic object tracking in street scenes," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 3180–3187.

[51] A. Asvadi, P. Peixoto, and U. Nunes, "Detection and tracking of moving objects using 2.5d motion grids," in *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, Sept 2015, pp. 788–793.

[52] J. Choi, S. Ulbrich, B. Lichte, and M. Maurer, “Multi-target tracking using a 3d-lidar sensor for autonomous vehicles,” in *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*, Oct 2013, pp. 881–886.

[53] A. Elfes, “Using occupancy grids for mobile robot perception and navigation,” *Computer*, vol. 22, no. 6, pp. 46–57, June 1989.

[54] T. Miyasaka, Y. Ohama, and Y. Ninomiya, “Ego-motion estimation and moving object tracking using multi-layer lidar,” in *Intelligent Vehicles Symposium, 2009 IEEE*, June 2009, pp. 151–156.

[55] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. N. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer, M. Gittleman, S. Harbaugh, M. Hebert, T. M. Howard, S. Kolski, A. Kelly, M. Likhachev, M. McNaughton, N. Miller, K. Peterson, B. Pilnick, R. Rajkumar, P. Rybski, B. Salesky, Y.-W. Seo, S. Singh, J. Snider, A. Stentz, W. R. Whittaker, Z. Wolkowicki, J. Ziglar, H. Bae, T. Brown, D. Demitrish, B. Litkouhi, J. Nickolaou, V. Sadekar, W. Zhang, J. Struble, M. Taylor, M. Darms, and D. Ferguson, “Autonomous driving in urban environments: Boss and the urban challenge,” *Journal of Field Robotics*, vol. 25, no. 8, pp. 425–466, 2008. [Online]. Available: http://dx.doi.org/10.1002/rob.20255

[56] S. Hosseinyalamdary, Y. Balazadegan, and C. Toth, “Tracking 3d moving objects based on gps/imu navigation solution, laser scanner point cloud and gis data,” *ISPRS International Journal of Geo-Information*, vol. 4, no. 3, pp. 1301–1316, 2015.

[57] Y. Cheng, “Mean shift, mode seeking, and clustering,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 8, pp. 790–799, Aug 1995.

[58] J. Gao and C. Harris, “Some remarks on kalman filters for the multisensor fusion,” *Information Fusion*, vol. 3, no. 3, pp. 191 – 201, 2002. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1566253502000702

[59] Oxford Technical Solutions, “RT3000 v2 Family.” [Online]. Available: http://www.oxts.com/products/rt3000-family/

[60] A. Broggi, P. Grisleri, and P. Zani, “Sensors technologies for intelligent vehicles perception systems: A comparison between vision and 3d-lidar,” in *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*, Oct 2013, pp. 887–892.

[61] C. J. Olsson, “Scene Category Context for 3d Object Detection with RGBD Cameras,” 2016.

[62] S. Song and J. Xiao, *Sliding Shapes for 3D Object Detection in Depth Images*. Cham: Springer International Publishing, 2014, pp. 634–651. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-10599-4_41

[63] Z. Ren and E. B. Sudderth, “Three-dimensional object detection and layout prediction using clouds of oriented gradients,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[64] M.-H. Y. W. Zhong, H. Lu, “Robust object tracking via sparse collaborative appearance model,” *IEEE Transactions on Image Processing*, vol. 23, no. 5, pp. 2356–2368, 2014.

[65] X. Jia, H. Lu, and M. H. Yang, “Visual tracking via adaptive structural local sparse appearance model,” in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, June 2012, pp. 1822–1829.

[66] “Visual tracker benchmark.” [Online]. Available: http://cvlab.hanyang.ac.kr/tracker_benchmark/

[67] R. E. Kalman, “A New Approach to Linear Filtering and Prediction Problems,” *Transactions of the ASME – Journal of Basic Engineering*, no. 82 (Series D), pp. 35–45, 1960. [Online]. Available: http://www.cs.unc.edu/~{}welch/kalman/media/pdf/Kalman1960.pdf

APPENDIX

A
KALMAN FILTER THEORY

Although more than 50 years old, KF and its variants such as EKF still remain popular choices in information processing and fusion. Presented by R. E. Kalman in 1960 [67], the algorithm described a recursive approach to the problem of discrete-data linear filtering. The KF is a set of mathematical equations used to recursively estimate the state of a given process, typically derived by vector algebra as a minimum mean square error estimator. Furthermore, the filter supports estimation of past, present and future states even when the system model is unknown.

As highlighted on the previous chapter, most object tracking approaches are based on some sort of Bayesian tracking theory. The main idea when using these filters is to set up a discrete time model with a state vector to describe a given object with the goal of estimating the state $x \in \Re^n$ of the process based on a measurement $z \in \Re^m$. The object motion model can be seen as a linear stochastic difference equation

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1}$$

and a measurement model in the shape of

$$z_k = Hx_k + v_k$$

with k representing the current time step. Time between consecutive time steps can be variable but in most cases is consistent, since observations (or measurements) are made in specific and well-behaved time intervals.

The $n \times n$ A matrix (*state transition matrix*) corresponds to the relationship between the state at time steps $k-1$ and k (assumed constant), applying the effect of system parameters at $k-1$ on the system at time k. The $n \times l$ matrix B (optional) relates the control input $u \in \Re^l$ to the state x. The random variables w and v contain the process noise and measurement noise terms (respectively) for the parameters in the state vector. These variables are assumed independent of each other, as being white noise and having a normal probability distribution. In this approach, they are also considered constant (zero-mean Gaussians). The $m \times n$ H matrix (*transformation matrix*) relates the current state to the measurement z_k. Even though H can change with each time step, it is also assumed constant.

To estimate a process, KF uses feedback control. First, it estimates the process state and then obtains feedback in the form of noisy measurements. The equations for the KF can be categorized as either being time update or measurement update equations. Time update equations project the current state forward in time as to acquire the *a priori* estimates for the next time step. Measurement update equations goal is to provide the new measurement to the existing *a priori* estimate to get a better estimate (also called the *a posteriori* estimate). In this sense, the first set of equations can be considered the predictor equations, with the other being the corrector equations.

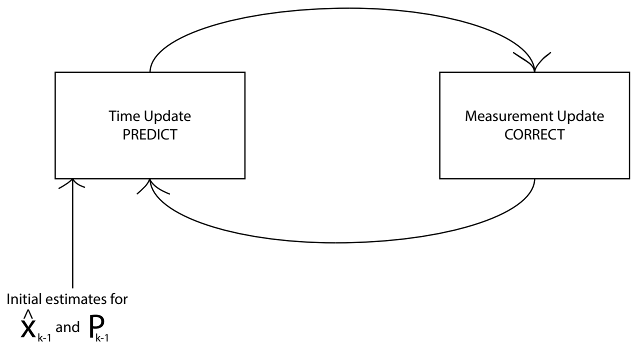


Figure A.1: *Kalman filter overview. The prediction projects the current state forward; the correction fits the projected estimate to the measurement taken at that time.*

After prediction, computation of the Kalman gain is done and afterwards a new process measurement is obtained. An *a posteriori* estimate for the state is generated by taking into account the obtained measurement and afterwards an *a posteriori* estimate for the error covariance is obtained as well. The recursive nature of the KF lies on the fact that after each prediction and correction, the whole process is repeated with the previous *a posteriori* estimates as new *a priori* estimates. In the presented notation, R is the measurement error covariance, Q is the process noise covariance and P *a priori* covariance matrix.

Time update (*prediction*) equations	**Measurement update (*correction*) equations**
Projection of the current state $$\hat{x}_k^- = A\hat{x}_{k-1} \quad \text{(A.1)}$$	*Computation of Kalman gain* $$\hat{K}_k = P_k^- H^T (HP_k^- H^T + R)^{-1} \quad \text{(A.2)}$$
Projection of error covariance $$P_k^- = AP_{k-1}A^T + Q \quad \text{(A.3)}$$	*Updating estimate with measurement* $$\hat{x}_k = \hat{x}_k^- + K_z(z_k - H\hat{x}_k^-) \quad \text{(A.4)}$$
	Updating error covariance $$P_k = (I - K_k H)P_k^- \quad \text{(A.5)}$$

Table A.1: *Discrete KF filter time and measurement update equations. From initial estimates $\hat{x}_{k-1}$ and P_{k-1} a prediction is made, which is then corrected and fed back to the prediction equations in a recursive fashion (as seen in Figure A.1).*

APPENDIX

B

SUBMITTED PAPER: 3D OBJECT TRACKING USING RGB AND LIDAR DATA

3D Object Tracking using RGB and LIDAR Data

Alireza Asvadi*†, Pedro Girão*, Paulo Peixoto*†, and Urbano Nunes*†
*Department of Electrical and Computer Engineering (DEEC)
†Institute of Systems and Robotics (ISR)
University of Coimbra, Portugal
{asvadi, peixoto, urbano}@isr.uc.pt

***Abstract*—Object tracking is one of the key components of the perception system of autonomous cars and ADASs. Using tracking, an ego-vehicle can make a prediction about the location of surrounding objects in the next time epoch and plan for next actions. Object tracking algorithms typically relay on sensory data either from RGB cameras or LIDAR. In fact, integration of 2D-RGB image from camera and 3D-LIDAR data can provide some distinct benefits. This paper proposes a 3D object tracking algorithm using a 3D-LIDAR, a RGB camera, and INS (GPS/IMU) data. The proposed method analyzes sequential 2D-RGB, 3D point-cloud, and the ego-vehicle's localization data and outputs the trajectory of the tracked object, an estimation of its current velocity, and its predicted location in the 3D world coordinate system in the next time-step. Tracking starts with a known initial 3D bounding box for the object. Two parallel mean-shift algorithms are applied for object detection and localization in the 2D image and 3D point-cloud, followed by a robust 2D/3D Kalman filter based fusion and tracking. Reported results, from quantitative and qualitative experiments using the KITTI database, demonstrate the applicability and efficiency of the proposed approach in driving environments.**

I. Introduction and Motivation

Visual object tracking is one of the rapidly evolving research fields in the computer vision community and became a key-feature of perception system of autonomous cars and advanced driver assistance systems (ADASs). The goal of object tracking is to estimate the trajectory of an object in a sequence of images given a known initial position [1]. Object tracking algorithms in image sequences have been studied comprehensively in the last few decades [1], [2], [3]. These methods mainly differ in the way they model the motion or appearance of the object. Considering the intended application each approach has strengths and weaknesses. However, most of these methods work only on monocular image sequences.

3D spatial data processing has recently gained much attention in computer vision and robotics. Recently, with the arrival of modern stereo matching algorithms and new 3D sensing technologies, providing an accurate and dense 3D point-cloud (PCD), perception systems of intelligent/autonomous vehicles became able to interpret surrounding environment in 3D and perceive objects physical properties. Stereo vision and 3D LIDAR are the first options to acquire 3D spatial information in the IV/ITS industries. The major limitations of stereo vision approaches are their poor performance in texture-less environments and their dependency on the calibration quality. The main disadvantages with LIDARs are the costly price and the moving parts of the sensor. In comparison with stereo cameras, LIDAR sensors have higher precision but do not acquire color data [4].

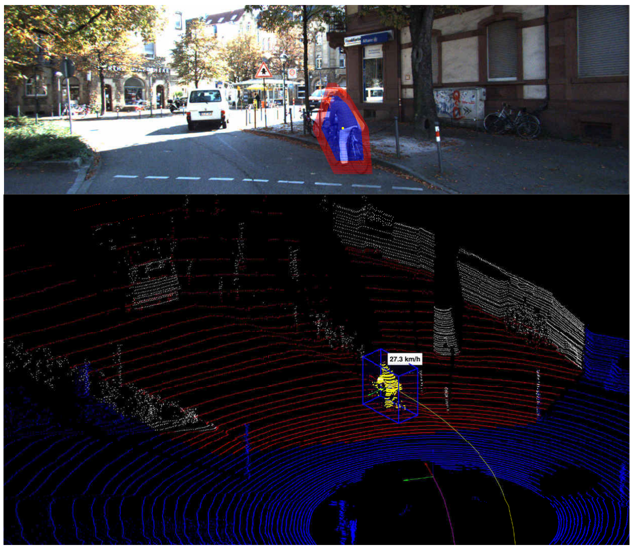


Fig. 1. The results of the proposed object tracking method for a given frame from the KITTI dataset. The bottom figure shows the result in the 3D-PCD, where the 3D Bounding Box (3*D-BB*) of the tracked object (here a cyclist) is shown in blue, object trajectory is represented with a yellow-curve, object points in 3*D-BB* are shown in yellow, and the current estimated velocity of the object is denoted with a text-box (27.3 km/h). The ego-vehicle trajectory, given by an INS system, is represented by a magenta-curve. The current position of the ego-vehicle and the tracked object are represented with red-green-blue vectors. Parts of the 3D-LIDAR PCD that are in the field of view of the camera are shown in white and red, where red denotes the detected ground points and white indicates the on-ground obstacle points. LIDAR points outside the image field are shown in blue. The top figure represent the tracking result in the 2D-RGB image for the current frame, where the detected object region and its surrounding area are shown in blue and red, respectively. Please refer to the PDF version for a high-resolution color representation of the figure.

2D cameras have been the most common sensor used for perceiving the environment. The high-spatial-resolution color data provided by 2D cameras can be used as a complement of 3D-LIDARs. The combined use of 3D-LIDAR with 2D-RGB camera, has attracted a lot of attention recently, to name a few, in applications such as object detection [5], road detection [6], scene parsing [7] and dense depth mapping [8].

This paper proposes an on-board 3D object tracking system benefiting from '*highly accurate 3D-LIDAR PCDs*' and '*dense 2D-RGB camera images*' with application in ADASs and autonomous driving (see Fig. 1). Specifically, the main contributions of this paper are as follows:

– A robust 2D/3D fusion-based 3D object tracking.

– A new ground plane estimation method.

Preprint submitted to 19th IEEE Intelligent Transportation Systems Conference.
Received June 16, 2016.

In addition, the proposed method provides a velocity estimation of the object, using the ego-vehicle localization data given by an INS (GPS/IMU). This enables the ego-vehicle to predict the object location in the 3D world coordinate system in the next time-step. The proposed method can be used in a wide range of applications from object behavior modeling to collision avoidance and planning systems. Extensive quantitative and qualitative experiments with different criteria were performed to evaluate the performance of the proposed approach.

The rest of the paper is organized as follows: Section II reviews the related work in object tracking for autonomous vehicles applications using 3D-LIDARs or stereo systems. Section III describes the proposed adaptive fused 3D object tracker. Experimental results are presented in Section IV, and Section V brings some concluding remarks.

II. Related Work

Object tracking algorithms can be divided into two categories [2]: Tracking-by-detection methods (or discriminative methods) and generative methods. Tracking-by-detection methods such as [9] detect an object in every time instant using a pre-trained detector (*e.g.*, DPM [10]) and link-up the detected positions over time to have the object track. The requirement of having all object categories being previously trained limits the application of this approach. Recently, generative object tracking methods attracted more attention for object tracking in driving environments. These approaches usually learn the appearance of an object. The object model is often updated online to adapt to appearance variation. Due to their class-independent generic nature, these approaches can be used for a wide range of object classes. Next, we will present a brief survey on generative 3D object tracking algorithms based on 3D sensing technologies (stereo vision and 3D LIDAR) including when they are fused with other sensors.

Held et al. [11] combined a 3D-PCD with a 2D camera image to construct an up-sampled colored 3D-PCD. They used a color-augmented search algorithm to align the colored PCDs from successive time frames. They utilized 3D shape and color data to perform joint 3D reconstruction and tracking of an object. They showed that the accumulated dense model of the object leads to a better object velocity estimate. In contrast with their early fusion of a 3D-PCD with a 2D-RGB data, we used a later fusion. In our approach, the object is detected and localized in the 2D-RGB image and 3D-PCD data spaces (with some interconnected data flow), and the 2D/3D object locations are then fused and tracked.

Some approaches try to use segmentation for object detection and tracking. Ošep et al. [12] used the PCD generated from a disparity map (obtained from a stereo camera pair) to find and track generic objects. They suggested a two-stage segmentation approach for extracting multi-scale object proposal, followed by multi-hypothesis tracking at the level of object proposals. Their two-stage segmentation method consists of a coarse supervised segmentation to removes non-object regions (*e.g.*, road and buildings) and a fine unsupervised multi-scale segmentation to extract object proposals. Vatavu et al. [13] built a Digital Elevation Map (DEM) from PCD obtained from a stereo vision system. They segmented on-ground obstacles by extracting the object delimiters. The object delimiters are represented by their positions and geometries, and then tracked using particle filters.

Other approaches detect and track objects based on their motion. Asvadi et al. [14] used 3D-LIDAR data to find and track generic moving objects based on their motion. In their approach, a static 2.5D grid map of the local environment surrounding the ego-vehicle is built and maintained. The 2.5D motion grid is achieved by comparing the last generated 2.5D grid with the static grid map. They fitted 3*D-BB* object models to 2.5D motion grids, followed by tracking of 3*D-BB*s using Kalman Filters (KFs). Dewan et al. [15] detect motions between consecutive scans by sequentially using RANSAC and proposed a Bayesian approach to segment and track multiple objects in 3D-LIDAR data.

In contrast to the segmentation-based tracking and motion-based tracking methods, we assume that the initial position of the object is given (similar to [11]), and hence the focus is on the robust appearance-based modeling and tracking of a generic object. In particular, this paper proposes a general fusion framework for integrating data from a 3D-LIDAR and a RGB camera for object tracking.

III. Adaptive Fused 3D Object Tracker

A. Problem Formulation and System Overview

In this paper, and considering sensory inputs from a color camera, a Velodyne LIDAR and an Inertial Navigation System (INS) mounted on-board an ego-vehicle, the 3D single-object tracking is defined as: Given a set of input data, RGB image I_i, 3D-LIDAR PCD L_i, and an ego-vehicle pose E_i and given an object's 3D Bounding Box (3*D-BB*) initial position in the first PCD, estimate the trajectory of the object, $\mathbf{X} = \{X_1, \cdots, X_N\}$, in the 3D world coordinate system as the ego-vehicle and object move around a scene.

The conceptual model of the proposed adaptive fused 3D object tracker is shown in Fig 2. Typically object tracking is one of the components of the perception pipeline, that should also include an object segmentation or detection module. In this paper, we assume that an initial 3*D-BB* is given by a 3D object detection [16] or a 3D motion detection method [17]. Object tracking starts with the known 3*D-BB* in L_1. Next, the ground plane is estimated, and ground points are eliminated from the points inside the 3*D-BB*. The remaining object points P are projected into the image plane and the 2D convex-hull (Ω) of the projected point set P^* is computed. The convex-hull Ω accurately segments object pixels from the rest of the image. The pairs of object points inside the 3*D-BB* and its corresponding Ω are used for the initialization of the tracking in the PCD and in the RGB image. In the next time-step, two Mean-Shift (MS) [18] based object localizers are run individually to estimate the 2D and 3D object locations in the RGB image and PCD, respectively. An adaptive color-based MS localizer is used to identify and localize the object in the RGB image, while the object localization in PCD is performed using the MS gradient estimation of the object points in the 3*D-BB*. The 2D location of the object in the RGB image is projected back to 3D. Kalman Filters (KFs) [19] with Constant Acceleration (CA) model are used for the fusion and tracking

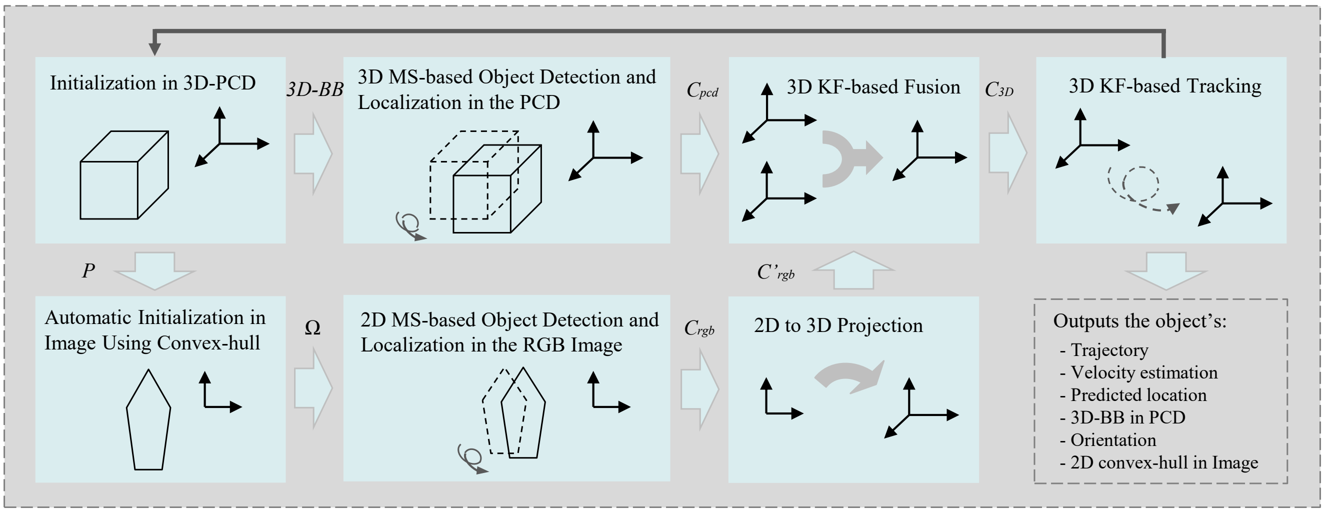


Fig. 2. The diagram of the major pipeline of the proposed 2D/3D fusion-based 3D object tracking algorithm. For details, please refer to the text.

of detected object locations in the 2D-RGB image and 3D-PCD. The predicted object's location and orientation are used for the initialization of the *3D-BB* in the next PCD.

B. 3D Object Detection and Localization in the PCD

Incoming PCD first need to be processed to remove the ground points. This step is essential to remove object points inside the *3D-BB* that correspond to ground points, avoiding degrading the quality of the object model.

1) Removing the Ground Points: Ground points typically constitute a large portion of a 3D-LIDAR's PCD. Therefore, if an appropriate feature is selected and the corresponding PDF is computed, the peak-value in the PDF indicates the ground points. Leveraging this fact, a Kernel Density Estimation (KDE) is used to estimate the PDF of the angles between the X-Y plane and the set of lines that pass through the center of the ego-vehicle (the origin-point) and every point belonging to the PCD (end-points). Let $\Theta = \{\theta_1, \cdots, \theta_N\}$ denotes the set of 1-dimensional angle values (measured in X-Z plane) for a certain PCD, where $\theta_i = arctan(z_i/x_i)$. Given this, the univariate KDE estimate is obtained using (1).

$$f(\theta) = \frac{1}{N}\sum_{i=1}^{N} K_\sigma(\theta - \theta_i) \qquad (1)$$

where $K_\sigma(.)$ is a Gaussian kernel function with bandwidth σ, and N is the number of points in the PCD. The planar ground assumption is adopted, and the pitch angle of the ground plane is identified as the peak value θ_p in the KDE. The points under a height d_{min} from the estimated ground plane are considered as ground points and removed, where d_{min} is a small value indicating a safe boundary for the ground removal process. The remaining points in the PCD represent obstacles' points (see Fig. 3).

In order to increase the robustness and accuracy of the ground removal process, a KF with Constant Acceleration (CA) model is used for the estimation of the ground plane's pitch angle in consecutive PCDs. To eliminate outliers, the angle search area is limited to a gate in the vicinity of the

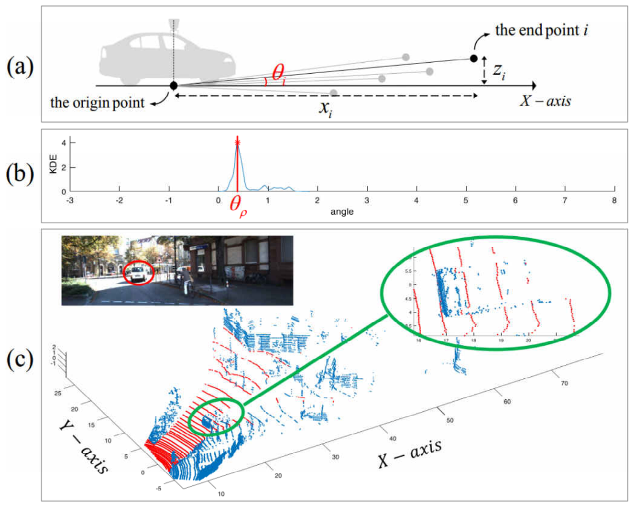


Fig. 3. The ground removal process. (a) Computation of the angle value θ_i for a point i in the PCD. (b) The Kernel Density Estimation (KDE) of the set Θ. The detected pitch angle of the ground plane is denoted by θ_p. (c) The ground removal result where red denotes the detected ground points and blue indicates the obstacle points. The green ellipse shows an example of the ground points extraction for a car. The green ellipse in the right shows the car in the top view. For a better illustration, the corresponding car in the image is shown with a red ellipse. Please refer to the PDF version for a high-resolution color representation of the figure.

predicted KF value from the previous-step. If no measurements are available inside the gate area, the predicted KF value is used.

2) MS-based Detection and Localization in the PCD: The object detection and localization in the PCD is performed using MS procedure, as follow:

1) Computing the shift-vector: Given the center of the *3D-BB* as χ, the shift-vector between χ and the point set P' inside the *3D-BB* is computed using,

$$m_k = \chi_k - \mu(P') \qquad (2)$$

Preprint submitted to 19th IEEE Intelligent Transportation Systems Conference.
Received June 16, 2016.

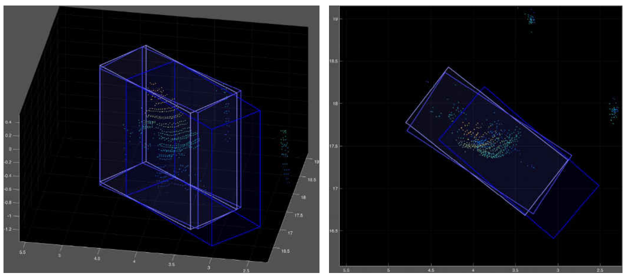


Fig. 4. The MS procedure in the PCD. The left figure represents a bird's-eye view and the right figure shows the top view. The MS convergence (the centroid movement less than 5 cm) is achieved in three iterations. The brighter blue color of the *3D-BB* shows the more recent iteration.

where $\mu(.)$ indicates the '*mean*' function, and k is the iteration index.

2) Translating *3D-BB*: The *3D-BB* is translated using the shift-vector,

$$\chi_{k+1} = \chi_k + m_k \tag{3}$$

The shift-vector always points toward the direction of the maximum increase in the density.

3) Iterating steps 1 and 2 until convergence: The MS iteratively shifts the *3D-BB* until the object is placed entirely within the *3D-BB*. A centroid movement $|m_k|$ less than 5 cm or a maximum number of iterations equal to 5 are considered as the MS convergence.

The MS process in PCD is shown in Fig. 4. The object position is represented by the centroid of object points (so-called '*point model*') inside the *3D-BB*. The '*point model*' is feasible even with a few number of object points which is the case in sparse 3D-LIDAR's PCD, specially for far objects. The centroid after convergence is denoted by C_{pcd} and outputted to the fusion module.

C. 2D Object Detection and Localization in the RGB Image

The object points P (after ground removal) are projected onto the image plane and the corresponding 2D convex-hull (Ω) of the projected point set $P^* = \{p_1^*, \cdots, p_n^*\}$ is computed. The 2D convex-hull can be defined as the smallest 2D convex polygon that encloses P^*. The 2D convex-hull more accurately segments the object from the background in comparison with the traditional 2D Bounding Box (*2D-BB*). A surrounding area $\Omega^\dagger$ is computed automatically by expanding Ω by a factor equal to $\sqrt{2}$ with respect to its centroid, so that the number of pixels in $\Omega^\dagger - \Omega$ (the region between Ω and $\Omega^\dagger$) is approximately equal to the number of pixels within the Ω.

1) Discriminant Color Model of the Object: Two joint RGB histograms are calculated from the pixels within Ω and $\Omega^\dagger - \Omega$ regions. The Log-Likelihood Ratio (LLR) of the computed RGB histograms expresses how much more likely each bin is under Ω color model than $\Omega^\dagger - \Omega$ color model. In the computed LLR, positive bins are more likely belonging to Ω, bins with negative values are more likely part of $\Omega^\dagger - \Omega$, and bins shared by both Ω and $\Omega^\dagger - \Omega$ tend towards zero. Therefore, the positive part of the LLR is used to represent the discriminant object color model,

$$\Re(i) = \max\left\{\log\frac{\max\{H_\Omega(i), \varepsilon\}}{\max\{H_{\Omega^\dagger}(i), \varepsilon\}},\ 0\right\} \tag{4}$$

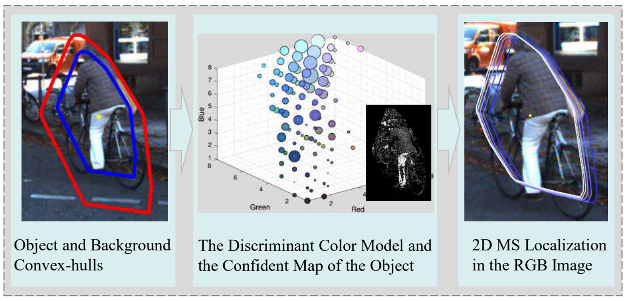

Object and Background Convex-hulls
The Discriminant Color Model and the Confident Map of the Object
2D MS Localization in the RGB Image
Blue
Green
Red

Fig. 5. The MS procedure in the RGB image. The left figure represents Ω and $\Omega^\dagger$ in blue and red, respectively. The computed $\Re$ and the confident map ($\mho$) are shown in the middle. Each non-empty bin in the $\Re$ is shown with a circle. A bin-value is represented as the area of the circle. Each circle's location represents a color in the joint RGB histogram (the same as its face color). The right figure shows the MS localization procedure. The brighter blue Ω indicates that it is computed in the more recent MS iteration. The final Ω centroid is shown by a yellow point.

where H_Ω and $H_{\Omega^\dagger}$ are the computed histograms from Ω and $\Omega^\dagger - \Omega$ regions, respectively, and ε is a small value that prevents dividing by zero or taking the log of zero. The variable i ranges from 1 to the number of histogram bins. The discriminant color model of the object ($\Re$) is normalized and used for the object detection and localization in the next frame using the MS procedure.

2) MS-based Detection and Localization in the RGB Image: MS-based object detection and localization for the next frame starts at the centroid of the confident map ($\mho$) of the Ω region in the current frame. This confidence map is computed by replacing the color value of each pixel in Ω region by its corresponding bin value in $\Re$. In each iteration, the center of Ω, from the previous step, is shifted to the centroid of $\mho$ (the current confidence map of the object) computed by Equation 5.

$$C_{new} = \frac{1}{m}\sum_{i=1}^{m} \mho_i\ C_i \tag{5}$$

where $C_i = \{r_i, c_i\}$ denotes pixel positions in Ω, and m is the total number of pixels in Ω. The maximum number of MS iterations needed to achieve convergence is limited to 4 unless the centroid movement is smaller than a pixel (see Fig. 5). The computed 2D object centroid after convergence is denoted by C_{rgb} and outputted to the fusion module.

3) Adaptive Updating of $\Re$ Bins: The dense 2D-RGB images obtained from cameras are very informative but they are sensitive to light and illumination conditions. To adapt the object color model with illumination variations and to overcome changes in the object color appearance during tracking, a bank of 1D-KFs with a CA model is applied. KFs estimate and predict $\Re$ bin values for next frames. A new 1D-KF is initialized and associated for each newly observed color bin, and when the bin values become zero or negative the corresponding KFs are removed. We conducted an experiment with $8 \times 8 \times 8$ histogram (512 bins) and observed that the average number of utilized KFs in each frame were 70 (about 14% of bins).

D. KF-based 2D/3D Fusion and Tracking

A 3D CA-KF is applied for the integration of the computed object centroids from 2D-RGB image and 3D-LIDAR PCD, followed by another 3D CA-KF for the robust tracking of the fused 3D location of the object over time.

1) 2D/3D Fusion for Improved Localization: The computed 2D location of the object in the RGB image (C_{rgb}) is projected back to 3D (C'_{rgb}) using a method described in [11]. Although this method was originally used for up-sampling a PCD, we employ it for projecting the computed object centroid in the 2D-RGB image back to the 3D-LIDAR space. The PCD is projected to the RGB image and the nearest projected points in each of the 4 quadrants (upper left, upper right, lower left, and lower right) surrounding C_{rgb} are found. The bilinear interpolation on the corresponding 4 points in the PCD (before the projection to the RGB image) is computed to estimate C'_{rgb}.

A KF-based fusion (the measurement fusion model [20]) is applied to integrate C'_{rgb} with the computed 3D object location in the PCD (C_{pcd}) and estimate the fused 3D centroid C_{3D}. The main idea is to give a higher weight to a method that performs better, thus providing a more accurate estimate than each method individually. The dynamics of the object and the fused measurement model of the object localizers in the 3D-PCD and the 2D-RGB image are given by Equations (6) and (7) respectively.

$$x_t = A_F \cdot x_{t-1} + w_t \tag{6}$$

$$z_t = H_F \cdot x_t + v_t \tag{7}$$

where w_t and v_t represent the process and measurement noise, respectively, A_F is the fusion state transition matrix, and H_F is the fusion transformation matrix. The augmented measurement vector z_t is given by,

$$z_t = \left[(C_{pcd})^T \quad (C'_{rgb})^T\right]^T \tag{8}$$

2) 3D Object Tracking: A 3D CA-KF is used for the robust tracking of the fused centroid C_{3D}. Let the state of the filter be $x = [x, \dot{x}, \ddot{x}, y, \dot{y}, \ddot{y}, z, \dot{z}, \ddot{z}]^T$, where $\dot{x}, \dot{y}, \dot{z}$ and $\ddot{x}, \ddot{y}, \ddot{z}$ define the velocity and acceleration corresponding to the x, y, z location, respectively. The discrete time process model and the measurement model of the system are given by Equations (9) and (10), respectively.

$$x_t = A_T \cdot x_{t-1} + w_t \tag{9}$$

$$z_t = H_T \cdot x_t + v_t \tag{10}$$

where A_T and H_T are the state transition matrix and the transformation matrix for object tracking, respectively. To eliminate outliers and increase the robustness of the process, the search area is limited to a gate in the vicinity of the predicted KF location (accessible from: $x_t = A_T \times x_{t-1}$). If no measurement is available inside the gate area, the predicted value given by the KF is used.

The result of the proposed algorithm is the estimated trajectory of an object in the 3D world coordinate system, its current estimated velocity, and the predicted location of the object in the next time-step. The object orientation is achieved by subtracting the current estimated location from its previous location. The object region in the image is obtained by computing the 2D convex-hull of the projected object points (the points inside the 3D-BB) into the image.

IV. Experiments and Analysis

A. Experimental Setup

1) Dataset - Object Tracking Evaluation: The KITTI dataset [21] was captured using RGB color cameras, a Velodyne 3D-LIDAR and a high-precision GPS/IMU inertial navigation system. The Velodyne HDL-64E spins at 10 frames per second and consists of 64 layers. The maximum recording range is 120 m. Front facing cameras have the resolution of 1.4-Megapixel and are synchronized with the LIDAR. The inertial navigation system has a 100 Hz sampling rate and a resolution of 0.02 m / 0.1°.

For the evaluation purpose, we extracted a variety of challenging object sequences from the 'Object Tracking Evaluation' set of the KITTI Vision Benchmark Suite. We conducted experiments on the extracted sequences. The 8 most representative sequences are reported in this paper. The details of each sequence and the challenging factors are summarized in Table I. When there are multiple entries, the order is related to the temporal occurrence, and each entry is denoted by its first letter. For example in the case of *distance to the object* (in the 3D-LIDAR's challenging factors columns), the entry N-M-F express that the object was first near to the ego-vehicle, next went to the middle range and after went far.

2) Implementation Details: The presented approach was implemented in MATLAB, and the experiments were carried out using a quad core 3.4 GHz processor with 8 GB RAM under MATLAB R2015a. The parameter values used in the implementation of the proposed method are reported in Table II. The first two parameters η and η' are the maximum number of MS iterations in the PCD and RGB image, respectively. The next two ($\delta\ell$ and $\delta\ell'$) are also related to the termination conditions of the MS. A displacement in the MS less than 5 cm ($\delta\ell$) in the PCD and less than 1 pixel ($\delta\ell'$) in the image are considered to guarantee convergence. The value d_{min} is a threshold in centimeters for the ground removal process. Points with heights lower than d_{min} from the estimated ground plane are considered as part of the ground and removed. The last parameter $b = 8$ is the selected number of histogram bins for each color channel. With these parameters the proposed algorithm un-optimized implementation runs at about 4 fps.

3) Description of the Evaluated Methods: The proposed method was evaluated against five object tracking methods. The selected methods operate on the image or PCD or on the fusion of both of them. Two of them are publicly available MS variants that take a RGB image as the input: (1) The original MS (MS) [22] and (2) MS with Corrected Background Weighted Histogram (CBWH) [23]. MS uses a color histogram regularized by a kernel to describe the appearance of an object. CBWH uses a color histogram with new weights to pixels in the target candidate region to reduce background's interference in object localization. Two PCD-based methods were implemented for evaluation: (3) The first one is the base-line KF-based object tracking method that uses the '*point model*' and 3D CA-KF with a Gating DA (3D-KF). (4) The second method is actually the first part of the proposed method, which is the MS-based object detection and localization in 3D-PCD (3D-MS). (5) A color-based 3D object tracker (3D-MS-KF) is also implemented that uses the colored PCD obtained by combining PCD and RGB data. It uses a color-based MS

TABLE I. DETAILED INFORMATION ABOUT EACH SEQUENCE.

	general specifications				RGB camera's challenging factors				3D-LIDAR's challenging factors		
Seq. name	No. of frames	Scene condition	Ego-vehicle situation	Obj. type	Occlusion	Illumination variations	Obj. pose variations	Change in the obj. size	No. of obj. points	Distance to the obj.	Velocity variations
(1)	154	urban	moving	cyclist	no	yes	yes	yes	high	N-M	no
(2)	154	urban	moving	car	partial	yes	yes	yes	high-low	M-F	yes
(3)	373	urban	S-M	car	no	yes	no	yes	high-low	N-M-F	yes
(4)	41	urban	stopped	cyclist	partial	no	no	no	high	near	no
(5)	149	downtown	stopped	pedestrian	partial	no	no	yes	high	near	no
(6)	45	downtown	stopped	pedestrian	F-P	yes	no	yes	high-low	near	no
(7)	71	downtown	S-M	pedestrian	P-F	yes	no	yes	low-high	near	no
(8)	188	downtown	M-S	pedestrian	no	yes	yes	yes	low-high	M-N	yes

TABLE II. MAIN PARAMETER VALUES USED IN THE PROPOSED ALGORITHM.

η	η'	$\delta\ell$	$\delta\ell'$	d_{min}	b
5	4	5	1	20	8

localization for the object detection and localization in the colored PCD, and a CA-KF for the object tracking.

B. Quantitative Evaluation

To assess the proposed method's performance, the center location errors in 2D and 3D, as well as the orientation in 3D are used.

1) Localization Estimation Evaluation: In the 3D case, the average center location error were computed for four methods: 3D-KF, 3D-MS, 3D-MS-KF and the proposed method (3D-Fusion). For the 2D case, all methods were considered for the evaluation (the resulting 3*D-BB*s of the 3D tracking methods were projected into the image to compute and compare the 2*D-BB*s). This metric is computed as the Euclidean distance of the center of the computed 2*D-BB* or 3*D-BB* from the 2D/3D ground-truth (the ground-truth is extracted from the KITTI dataset). The average center location errors in 2D and 3D are given by Equations (11) and (12), respectively.

$$E_{2D} = \frac{1}{N}\sum_{i=1}^{N}\sqrt{(r_i - r_i^g)^2 + (c_i - c_i^g)^2} \tag{11}$$

$$E_{3D} = \frac{1}{N}\sum_{i=1}^{N}\sqrt{(x_i - x_i^g)^2 + (y_i - y_i^g)^2 + (z_i - z_i^g)^2} \tag{12}$$

where r_i, c_i shows the detected object location (the center of the 2*D-BB*) in the image and r_i^g, c_i^g denotes the center of the ground-truth 2*D-BB*, x_i, y_i, z_i indicates the center of the detected 3*D-BB* in the PCD, x_i^g, y_i^g, z_i^g is the center of the ground-truth 3*D-BB*, and N is the total number of frames/scans.

Table III represents the results in terms of E_{2D} and E_{3D}, where a dash entry (-) represents a failure of the correspondent algorithm to track the selected object. The 3D-MS provides the smallest error when it does not fails, however, it is prone to errors, mostly because it starts diverging to the nearest object or obstacle in cluttered environments (*e.g.*, a group of pedestrians). It can be seen that the proposed method is the only one with stable results (without failures), while keeping the center location error low. The MS and CBWH methods (that work only on images) are very fragile because essentially they are not designed to overcome most challenging factors in driving environments as evidenced in Table I.

2) Orientation Estimation Evaluation: The orientation evaluation was only performed for the 3D approaches (3D-KF, 3D-MS, 3D-MS-KF, and 3D-Fusion). The ground-truth orientation of the object is given only in Yaw angle (Yaw angle describes the heading of the object, and in the KITTI dataset it is given by rotation around Y-axis in camera coordinates. For more information please refer to [21]). The orientation error can be computed by,

$$E_{\varphi} = \frac{1}{N}\sum_{i=1}^{N}\left|\arctan\left|\frac{\overrightarrow{\varphi} \times \overrightarrow{\varphi}^g}{\overrightarrow{\varphi} \cdot \overrightarrow{\varphi}^g}\right|\right| \tag{13}$$

where $\overrightarrow{\varphi}^g$ is the ground-truth orientation of the object (see Table IV).

C. Qualitative Evaluation

Sample results of the application of the proposed algorithm on the selected sequences are shown in Fig. 6, and the most challenging factors are discussed here. The proposed method successfully tracks the objects throughout the considered sequences.

1) Occlusion: The objects in sequences (2), (5), (6) and (7) undergo partial or full occlusions. In sequence (2) the tracked car goes under severe occlusion caused by a red parked van, and in sequences (5) and (6) the tracked pedestrians are occluded by other moving pedestrians. In sequence (7) the pedestrian was partially occluded by bushes for a number of frames.

2) Illumination Change: In sequences (1), (2), (3), (6), (7) and (8), the tracked objects go through illumination changes, mainly due to moving from shadow to sunlight areas or vice-versa.

3) Velocity Change: In sequences (2), (3) and (8), the object of interest moves with variable velocity throughout the sequence. For example in sequence (3) the red car accelerates and then stops in a crossroad and a crosswalk. In the remaining sequences the object moves with a fairly constant speed.

4) Pose/Size Changes: In sequences (2), (3) and (8), large variations in the object size occur, mainly because the distance to the ego-vehicle is also changing. In addition, the object pose varies in almost all sequences. For example in the second sequence, the car size and orientation with respect to the ego-vehicle changes constantly.

V. CONCLUDING REMARKS AND FUTURE WORK

In this paper, we developed an on-board object tracking system using data from 3D-LIDAR and RGB camera with

TABLE III. THE AVERAGE CENTER LOCATION ERRORS IN 2D (PIXELS) AND 3D (METERS).

Seq. no.	The average center location errors (3D)				The average center location errors (2D)					
	3D-Fus.	3D-KF	3D-MS	3D-MS-KF	3D-Fus.	3D-KF	3D-MS	3D-MS-KF	MS	CBWH
(1)	0.306	-	0.215	0.250	9.237	-	8.439	11.749	263.680	298.199
(2)	1.983	17.693	1.844	-	12.095	407.186	15.190	-	208.392	306.135
(3)	1.672	-	1.540	1.622	3.975	-	7.270	8.617	16.351	37.508
(4)	0.395	-	-	-	5.799	-	-	-	217.836	333.193
(5)	0.229	2.903	0.187	1.442	12.443	157.052	12.119	53.626	279.050	128.952
(6)	0.191	-	0.114	0.153	13.645	-	10.257	15.890	420.643	418.760
(7)	0.264	2.301	0.198	0.181	19.581	186.937	16.356	14.585	118.836	192.681
(8)	0.174	0.820	0.153	0.207	22.843	51.262	17.172	25.837	225.073	162.179

TABLE IV. ORIENTATION ESTIMATION EVALUATION (IN RADIAN)

Methods Seq. no.	3D-Fus.	3D-KF	3D-MS	3D-MS-KF
(1)	0.411	-	0.393	0.393
(2)	0.416	1.255	0.427	-
(3)	0.118	-	0.135	0.145
(4)	0.108	-	-	-
(5)	0.208	0.569	0.136	0.242
(6)	0.203	-	0.149	0.163
(7)	0.260	0.921	0.155	0.168
(8)	0.155	0.297	0.148	0.159

application in automotive driver assistance systems (ADASs) and autonomous driving. A fusion scheme was proposed to maximize the benefits of using dense 2D-RGB images and sparse 3D-PCDs as inputs. The ego-vehicle localization data, given by an INS (GPS/IMU), was applied to achieve a velocity estimate of the object. KFs with Constant Acceleration (CA) models were used to keep track of object changes in location and color appearance. The proposed method can be used in a wide range of applications from object behavior modeling to collision avoidance and planning for future actions. A set of assorted experiments, and corresponding result analysis, aimed at evaluating the performance of the proposed approach were performed.

A planar ground assumption was adopted by the proposed method, with only a pitch angle (which is the rotation around the X-axis, uphill/downhill ground cases). This assumption works fine with most of the cases available in the KITTI dataset. In addition to the pitch angle θ, the Y values of the PCD could also be considered in the KDE estimation process to take into account the rolling angle of the ground.

ACKNOWLEDGMENTS

This work has been supported by the FCT project "AMS-HMI2012 - RECI/EEIAUT/0181/2012" and project "ProjB-Diagnosis and Assisted Mobility - Centro-07-ST24-FEDER- 002028" with FEDER funding, programs QREN and COMPETE.

REFERENCES

[1] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *Acm computing surveys (CSUR)*, vol. 38, no. 4, p. 13, 2006.

[2] H. Yang, L. Shao, F. Zheng, L. Wang, and Z. Song, "Recent advances and trends in visual tracking: A review," *Neurocomputing*, vol. 74, no. 18, pp. 3823–3831, 2011.

[3] X. Li, W. Hu, C. Shen, Z. Zhang, A. Dick, and A. V. D. Hengel, "A survey of appearance models in visual object tracking," *ACM transactions on Intelligent Systems and Technology (TIST)*, vol. 4, no. 4, p. 58, 2013.

[4] A. Broggi, P. Grisleri, and P. Zani, "Sensors technologies for intelligent vehicles perception systems: A comparison between vision and 3d-lidar," in *Intelligent Transportation Systems-(ITSC), 2013 16th International IEEE Conference on*. IEEE, 2013, pp. 887–892.

[5] C. Premebida, J. Carreira, J. Batista, and U. Nunes, "Pedestrian detection combining rgb and dense lidar data," in *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*. IEEE, 2014, pp. 4112–4117.

[6] L. Xiao, B. Dai, D. Liu, T. Hu, and T. Wu, "Crf based road detection with multi-sensor fusion," in *Intelligent Vehicles Symposium (IV), 2015 IEEE*. IEEE, 2015, pp. 192–198.

[7] G. Zhao, X. Xiao, J. Yuan, and G. W. Ng, "Fusion of 3d-lidar and camera data for scene parsing," *Journal of Visual Communication and Image Representation*, vol. 25, no. 1, pp. 165–183, 2014.

[8] Y. He, L. Chen, and M. Li, "Sparse depth map upsampling with rgb image and anisotropic diffusion tensor," in *Intelligent Vehicles Symposium (IV), 2015 IEEE*. IEEE, 2015, pp. 205–210.

[9] A. Geiger, M. Lauer, C. Wojek, C. Stiller, and R. Urtasun, "3d traffic scene understanding from movable platforms," *Pattern Analysis and Machine Intelligence (PAMI)*, 2014.

[10] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 32, no. 9, pp. 1627–1645, 2010.

[11] D. Held, J. Levinson, and S. Thrun, "Precision tracking with sparse 3d and dense color 2d data," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 1138–1145.

[12] A. Ošep, A. Hermans, F. Engelmann, D. Klostermann, , M. Mathias, and B. Leibe, "Multi-scale object candidates for generic object tracking in street scenes," in *ICRA*, 2016.

[13] A. Vatavu, R. Danescu, and S. Nedevschi, "Stereovision-based multiple object tracking in traffic scenarios using free-form obstacle delimiters and particle filters," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 16, no. 1, pp. 498–511, 2015.

[14] A. Asvadi, P. Peixoto, and U. Nunes, "Detection and tracking of moving objects using 2.5 d motion grids," in *Intelligent Transportation Systems (ITSC), 2015 IEEE 18th International Conference on*. IEEE, 2015, pp. 788–793.

[15] A. Dewan, T. Caselitz, G. D. Tipaldi, and W. Burgard, "Motion-based detection and tracking in 3d lidar scans," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, Stockholm, Sweden, 2016.

[16] X. Chen, K. Kundu, Y. Zhu, A. G. Berneshawi, H. Ma, S. Fidler, and R. Urtasun, "3d object proposals for accurate object class detection," in *Advances in Neural Information Processing Systems*, 2015, pp. 424–432.

[17] A. Asvadi, P. Peixoto, and U. Nunes, "Two-stage static/dynamic environment modeling using voxel representation," in *Robot 2015: Second Iberian Robotics Conference*. Springer, 2016, pp. 465–476.

[18] Y. Cheng, "Mean shift, mode seeking, and clustering," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 17, no. 8, pp. 790–799, 1995.

[19] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[20] J. Gao and C. J. Harris, "Some remarks on kalman filters for the multisensor fusion," *Information Fusion*, vol. 3, no. 3, pp. 191–201, 2002.

Preprint submitted to 19th IEEE Intelligent Transportation Systems Conference.
Received June 16, 2016.

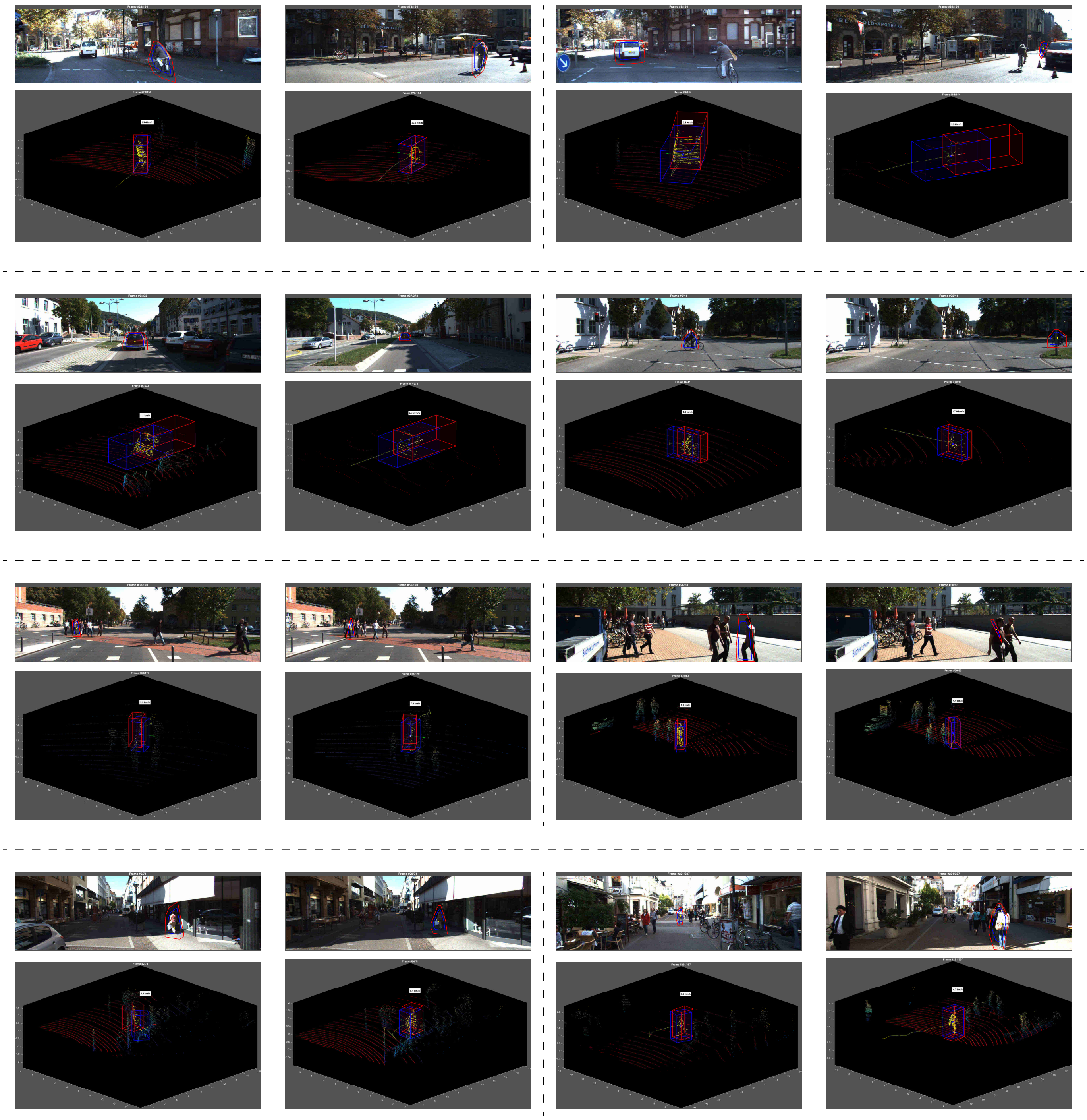


Fig. 6. Sample screenshots of the object tracking results obtained for sequences 1 to 8 as listed in Table I and its corresponding representation in the 3D space. Each of the right/left halves of each row represents one sequence (with two images and two corresponding PCDs for each sequence). The tracking results in the 2D-RGB image are shown by blue and red polygons, where they denote the detected object region and its surrounding area, respectively. In the 3D-PCD, the object trajectory is represented with a yellow-curve, the 3*D-BB* of the tracked object is shown in blue, and the 3*D-BB* of the ground-truth data is shown in red. The object points in 3*D-BB* are represented in yellow and the current velocity estimation of the object is denoted with a text-box over the 3*D-BB* of the object. The detected ground points are shown in red. Please refer to the PDF version for a high-resolution color representation of the figure.

[21] A. Geiger, P. Lenz, and R. Urtasun, “Are we ready for autonomous driving? the kitti vision benchmark suite,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[22] D. Comaniciu, V. Ramesh, and P. Meer, “Kernel-based object tracking,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 25, no. 5, pp. 564–577, 2003.

[23] J. Ning, L. Zhang, D. Zhang, and C. Wu, “Robust mean-shift tracking with corrected background-weighted histogram,” *Computer Vision, IET*, vol. 6, no. 1, pp. 62–69, 2012.

Preprint submitted to 19th IEEE Intelligent Transportation Systems Conference.
Received June 16, 2016.

APPENDIX

C

SUBMITTED PAPER: 3D OBJECT TRACKING IN DRIVING ENVIRONMENT: A SHORT REVIEW AND A BENCHMARK DATASET

3D Object Tracking in Driving Environment: a short review and a benchmark dataset

Pedro Girão, Alireza Asvadi, Paulo Peixoto, and Urbano Nunes

Abstract— **Research in autonomous driving has matured significantly with major automotive companies making important efforts to have their autonomous or highly automated cars available in the near future. As driverless cars move from laboratories to complex real-world environments, the perception capabilities of these systems to acquire, model and interpret the 3D spatial information must become more powerful. Object tracking is one of the challenging problems of autonomous driving in 3D dynamic environments. Although different approaches are proposed for object tracking with demonstrated success in driving environments, it is very difficult to evaluate and compare them because they are defined with various constraints and boundary conditions. The appearance modeling for object tracking in the driving environments, using a multimodal perception system of autonomous cars and advanced driver assistance systems (ADASs), and the evaluation of such object trackers are the research focus of this paper. A benchmark dataset, called 3D Object Tracking in Driving Environments (3D-OTD), is also proposed to facilitate the assessment of object appearance modeling in object tracking methods.**

I. Introduction

Object tracking is an essential component in the perception pipeline of autonomous cars and ADASs. Using tracking, an ego-vehicle can make a prediction about its surrounding objects locations and behaviors, and based-on that make proper decisions and plan next actions.

There is an extensive research literature on object tracking in image sequences [1–6]. These surveys are mainly focused on 2D object tracking algorithms, which work with images from monocular cameras. 3D spatial data processing has gained much attention in computer vision and robotics in the last few years. Recently, with the arrival of modern stereo matching algorithms and new 3D sensing technologies, providing an accurate and dense 3D point-cloud (PCD), perception systems of intelligent/autonomous vehicles became able to interpret surrounding environment in 3D and perceive objects physical properties. Generally, autonomous vehicles [7–9] are equipped with a number of on-board sensors *e.g.*, mono and stereo cameras, thermal, night vision, LIDAR, Radar, Inertial Navigation System (INS), Global Positioning System (GPS), and Inertial Measurement Unit (IMU), to have a multimodal robust observation of the scene. Such a sensor setup makes autonomous cars capable with a more robust sensory perception in comparison with a single monocular observation. This paper reviews different object tracking techniques and approaches that have been developed for autonomous driving with a focus on using stereo vision and 3D LIDAR, which are the first options to acquire 3D spatial information in the IV/ITS industries, including when they are fused with other sensors. The major limitations of stereo vision approaches are their poor performance in texture-less environments and their dependency on the calibration quality. The main disadvantages with LIDARs are the costly price and the moving parts of the sensor. In comparison with stereo cameras, LIDAR sensors are robust against illumination changes and have higher precision but do not acquire color data [10]. 2D cameras have been the most common sensor used for perceiving the environment. The high-spatial-resolution color data provided by 2D cameras can be used as a complement of 3D-LIDARs.

This paper gives a brief survey of the 3D object tracking methods, focusing on algorithms actually designed and used for ADASs and autonomous driving. In this work, a benchmark dataset is constructed out of the '*KITTI Object Tracking Evaluation*', and the sequence attributes and challenging factors are extracted. Two baseline object trackers were implemented. Experiments with various evaluation criteria were performed for the performance analysis. The evaluation scripts, source codes for the baseline object trackers and the ground-truth data corresponding to this work are available online[1].

The remainder of this paper is organized as follows, in the next section, a brief overview of object tracking algorithms for autonomous driving using 3D sensors is presented, Section III is devoted to describe the benchmark dataset and its characteristics. Section IV presents the baseline 3D object trackers, while Section V focuses on the selected metrics and evaluation methodology. Evaluation results are detailed in Section VI, and finally Section VII brings some concluding remarks.

II. Overview of 3D Object Tracking Methods for Autonomous Driving

This section gives an overview 3D object tracking methods for purposes of autonomous driving and relevant concepts. Fig. 1 presents a taxonomy of 3D object tracking methods in driving environments in terms of approaches for object representation and the main tracking components. The details are discussed in the following subsections.

A. Object Tracking Approaches

Object tracking algorithms can be divided into two categories based-on representation scheme for the object [2]:

Institute of Systems and Robotics (ISR), Department of Electrical and Computer Engineering (DEEC), University of Coimbra, Portugal pedro.girao@student.uc.pt

[1]http://a-asvadi.ir/3d-object-tracking-in-driving-environments/

Preprint submitted to 19th IEEE Intelligent Transportation Systems Conference. Received July 11, 2016.

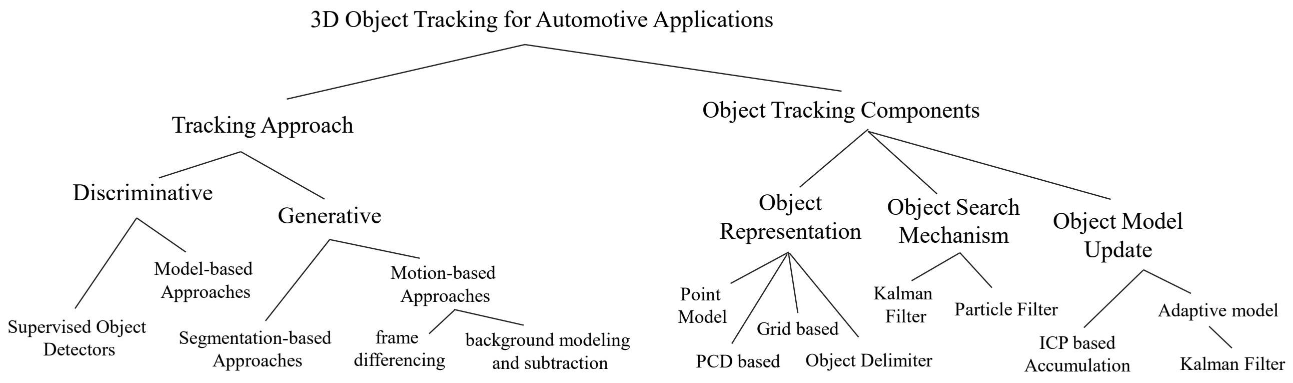


Fig. 1: Taxonomy of object tracking using 3D sensors for autonomous driving.

1) Tracking-by-detection (or Discriminative) Approaches: Discriminative object trackers localize the object using a pre-trained detector (*e.g.*, DPM [22]) that learns a decision boundary between the appearance of the target object and other objects and obstacles, and next link-up the detected positions over time. Many approaches [23–28] have been proposed for discriminative object tracking, with most of them focused on the data association problem. An overview of such approaches is given in the '*KITTI Object Tracking Evaluation Benchmark*'[2] [9]. In MOT15 [29] and MOT16[3] [30] another review of multi-object tracking methods, and their evaluation is given. However, the requirement of having all object categories being known and previously trained limits the application of discriminative approaches. Moreover, most of these approaches are only based-on monocular cameras.

2) Model-free (or Generative) Approaches: In order to have a reliable perception system for autonomous cars in real-world driving scenarios a generic object tracker [14], [19], [31–33] is also required. A generic tracker should be able to track all kinds of objects, even if their existence is not previously predicted or trained. Generative methods build a model to describe the appearance of an object and then looks for the next occurrence of the object by searching for the region most similar to the model. To handle the appearance variations of the target object, the object model is often updated online.

B. Object Tracking Components

Object tracking algorithms are composed of three main components: object representation, search mechanism, and model update. The object appearance model can be depend solely on the target object (generative approaches) or can be built by considering its discriminative power with respect to the appearance of other objects or obstacles (discriminative approaches). A search mechanism is needed to localize object in the next sensor input (frame/scan) using their appearance models. It is essential to update the target object model to compensate changes caused by observing objects in different view points, scene conditions, etc.

[2]http://www.cvlibs.net/datasets/kitti/eval_tracking.php
[3]https://motchallenge.net/

There are many approaches for object tracking using monocular cameras or 2D-LIDARs [35], [36]. Recently, most of them have been revisited based-on stereo vision and 3D-LIDARs. These 3D sensors enables the 3D appearance modeling of the objects of interest. The simplest representation of a target object is by the centroid of object points, so-called the '*point model*'. The point model is feasible even with a few number of object points, however, a richer appearance modeling can be exploited to improve tracking performance. Other approaches were proposed for the appearance modeling of the objects, like 2D rectangular shape [12], 2.5D box [13], 2.5D grid [14], 3D voxel grid [15], [16], octree data structure-based representation [17], [18], object delimiter representation [19], and 3D reconstruction of the shape of the target object [20], [21]. The main advantage of the later one is a higher robustness against occlusion (See Fig. 2).

Some approaches [12], [14–17] detect and track generic objects based-on their motion. This group of methods are the most widely used and are closely related to the Detection and Tracking of Moving Object (DATMO) [37] approaches. These methods are unable to detect stationary objects which potentially can move. Moving object detection can be achieved by detecting changes that occur between two or three consecutive observations (which can be interpreted as '*frame differencing*'). Generally, such methods can be used for detecting generic moving objects. However, Petrovskaya and Thrun [12] narrow it to vehicle tracking by fitting a 2D rectangular object model to detected motions.

Moving object detection can also be achieved by building a consistent static model of the scene, called the background model, and then finding deviations from the model for each incoming frame [14], [17]. This process can be referenced as '*background modeling and subtraction*'. The background model is usually a short-term map of the surrounding environment of the ego-vehicle. Generally, the static background model is built by combining the ego-vehicle localization data and a representation of 3D sensor inputs such as: PCD [12], 2.5D elevation grid [14], [19], 3D voxel grid [15], [16] or

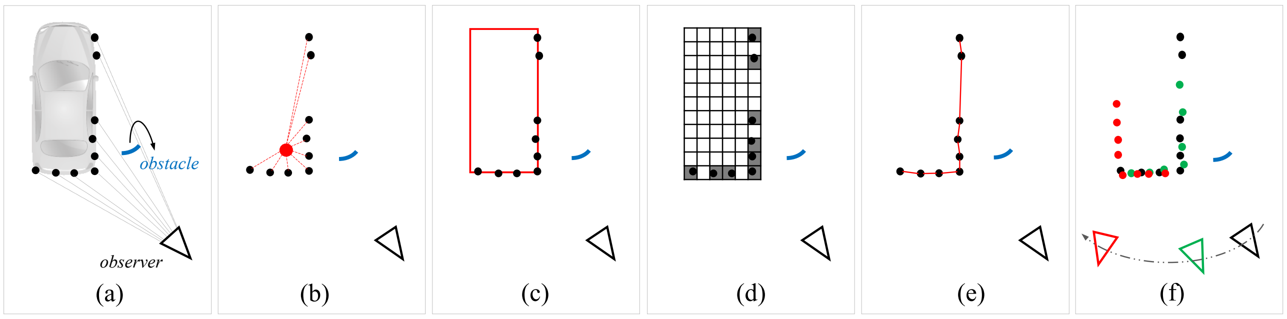


Fig. 2: Some approaches for the appearance modeling of a target object. (a) represents a scan of a vehicle which is split up by an occlusion from top view [11] , (b) the centroid (the point model) representation of the target object, (c) 2D rectangular [12] or 2.5D box [13] shape based representations, (d) 2.5D grid [14], 3D voxel grid [15], [16] or octree data structure-based representation [17], [18], (e) object delimiter-based representation [19], and (f) 3D reconstruction of the shape of the target object [20], [21].

TABLE I: Some of the recent 3D object tracking methods for autonomous driving applications.

Ref.	3D Perception Sensor	Ego-motion Estimation	Tracking Approach	Object Representation	Object Search Mechanism	Object Model update
[19]	Stereo Vision	GNSS, INS	Generative	Object Delimiters	Particle Filter	KF
[15]	Multi-Layer LIDAR	ICP	Generative	Voxel	EKF	No Update
[11]	3D-LIDAR, GIS Map	INS	Generative	PCD	CV-KF	No Update
[31]	Stereo Vision	V-Odometry	Generative	Voxel	KF and MHT	Weighted ICP
[21]	3D-LIDAR	-	Generative	PCD	CV-KF	ICP, Accumulation
[20]	3D LIDAR, Camera	INS	Generative	Colored PCD	CV-KF	ICP, Accumulation
[34]	3D-LIDAR	DGPS/IMU	Generative	PCD	Bayesian approach	No Update
[16]	Stereo Vision	V-Odometry	Generative	Voxel	KF	No Update
[17]	3D-LIDAR	INS, ICP	Discriminative	Octree	KF and GNN	No Update
[12]	3D-LIDAR	INS	Discriminative	2D Rectangle	Particle Filter	CV-KF
[14]	3D LIDAR	INS	Generative	Elevation Grid	CV-KF and Gating	No Update

octree data structure-based representation [17], [18]. Ego-motion estimation usually is achieved using Visual Odometry [16], INS (GPS/IMU) [14], [19], variants of Iterative Closest Point (ICP) scan matching algorithm [15] or a combination of them [17]. Moosmann and Stiller [21] used a local convexity based segmentation method for object hypotheses detection. A combination of KF and ICP is used for tracking moving objects and a classification method for managing tracks. Their method includes the 3D reconstruction of the shape of moving objects. Hosseinyalamdary et al. [11] used prior Geospatial Information System (GIS) map to reject outliers. They tracked moving objects in a scene using Constant Velocity (CV) KFs and used ICP for pose estimation. Dewan et al. [34] detect motions between consecutive scans using RANSAC and use a Bayesian approach to segment and track multiple objects in 3D-LIDAR data. However, they admitted themselves that their approach does not work well with pedestrians.

The majority of these approaches have only been developed for detection and tracking of moving objects. However, in real-world applications, static objects should be taken into account. Segmentation-based approaches are proposed to partition the PCD into perceptually meaningful regions that can be used for object detection. Ošep et al. [31] used the PCD generated from a disparity map (obtained from a stereo camera pair) to find and track generic objects. They suggested a two-stage segmentation approach for multi-scale object proposal generation, followed by Multi Hypothesis Tracking (MHT) at the level of object proposals. Vatavu et al. [19] built a Digital Elevation Map (DEM) from PCD obtained from a stereo vision system. They segmented on-ground obstacles by extracting free-form object delimiters. The object delimiters are represented by their positions and geometries, and then tracked using particle filters. KFs are used for adapting object delimiter models.

In another approach, Held et al. [20] combined a PCD with a 2D camera image to construct an up-sampled colored PCD. They used a color-augmented search algorithm to align the colored PCDs from successive time frames. Assuming a known initial position of the object, they utilized 3D shape, color data and motion cues in a probabilistic framework to perform joint 3D reconstruction and tracking. They showed that the accumulated dense model of the object leads to a better object velocity estimate. A summary of the most representative tracking approaches is provided in Table I.

III. Dataset and Experimental Setup

Previous attempts to propose object tracking benchmarks for automotive applications were mostly based on monocular cameras [29], [30], or were just focused on the data association problem [23]. In this paper a new framework to evaluate the performance of the appearance modeling of a target object using 3D sensors, based on the '*KITTI Object Tracking Evaluation*' dataset is proposed.

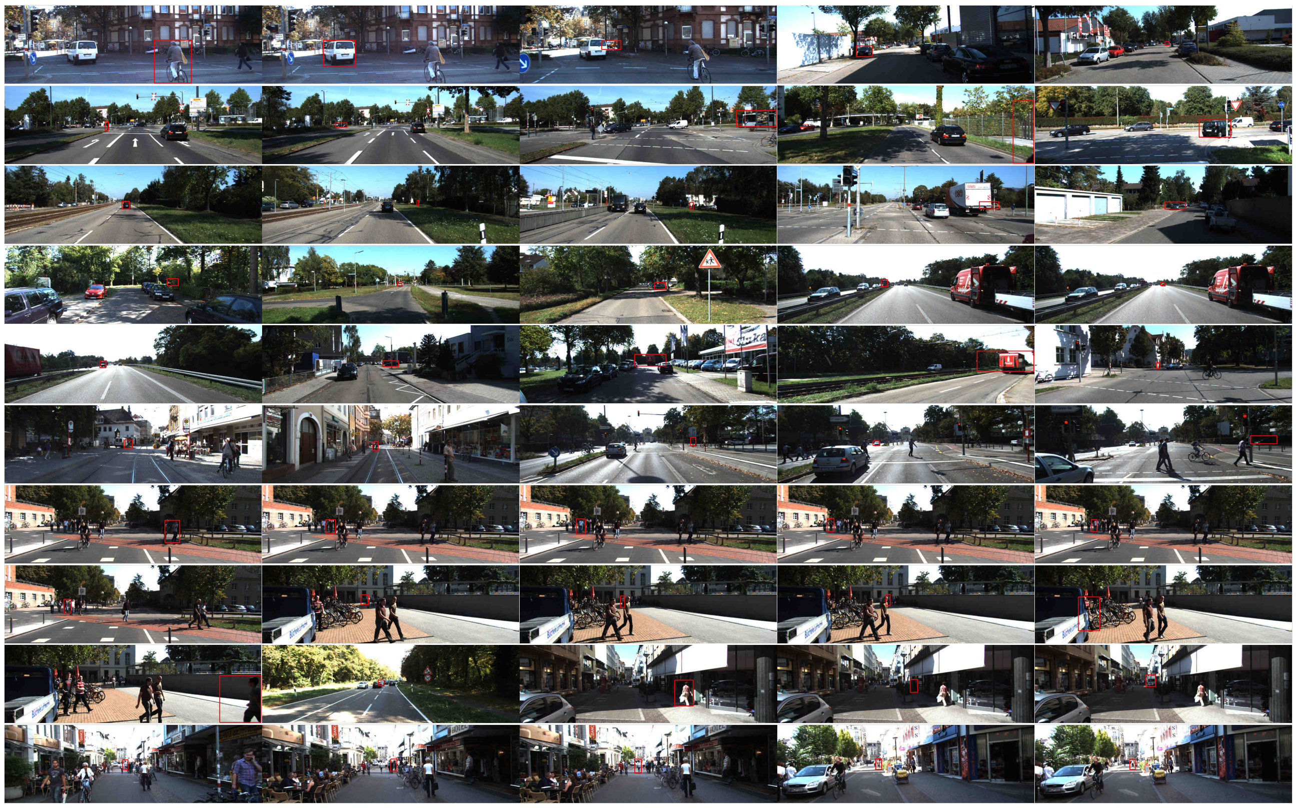


Fig. 3: From left to right and top to bottom shows initial frames of 50 sequences for performance evaluation, as listed in Table. The projected 3D Bounding Box of the target object to 2D is shown by a red rectangle. Please refer to the PDF version for a high-resolution color representation of the figure.

A. *KITTI - Object Tracking Evaluation*

The KITTI dataset [9] was captured using gray scale and RGB color cameras, a Velodyne 3D-LIDAR and a high-precision GPS/IMU inertial navigation system. The Velodyne HDL-64E spins at 10 frames per second with 26.8 degree vertical field of view ($+2°/-24.8°$ up and down), consists of 64 equally spaced angular subdivisions (approximately 0.4°) and angular resolution of 0.09 degree. The maximum recording range is 120 m. Front facing cameras with a resolution of 1.4-Megapixel are also used, with their images synchronized with the LIDAR data. The inertial navigation system is an OXTS RT3003 inertial and GPS system with 100 Hz sampling rate and a resolution of 0.02 m / 0.1°.

B. *3D-OTD Benchmark Dataset*

In the original KITTI dataset, objects are annotated with their tracklets, and generally the dataset is more focused on the evaluation of data association problem in discriminative approaches. Our goal is to provide a tool for the assessment of object appearance modeling in both the discriminative and generative methods. Therefore, instead of tracklets, full track of each object is extracted. A benchmark dataset with 50 annotated sequences is constructed out of the '*KITTI Object Tracking Evaluation*' to facilitate the performance evaluation. In the constructed benchmark dataset, each sequence denotes a trajectory of only one target object (*i.e.*, if one scenario includes two target objects, it is considered as two sequences). The initial frames of sequences are shown in Fig. 3, and the general specifications of each sequence and the most challenging factors are extracted and reported in Table II. The table contains the description of the scene, sequence, and objects including the *number of frames* for each sequence, *object type*: car '*C*', pedestrian '*P*' and cyclist '*Y*', *object and Ego-vehicle situations*: moving '*M*' or stationary '*S*', *scene condition*: roads in urban environment '*U*' or alleys and avenues in downtown '*D*'. The object width (Im-W) and height (Im-H) in the first frame (in pixels), and width (PCD-W), height (PCD-H), and length (PCD-L) in the first PCD (in meters) of each sequence are also reported. Each of the sequences are categorized according to the following challenges: *occlusion (OCC)*, *object pose (POS)* and *distance (DIS)* variations to Ego-vehicle, and changes in the *relative velocity (RVL)* of the object to the Ego-vehicle.

IV. Baseline 3D Object Tracking Algorithms

As a starting point for the benchmark, two generative 3D-LIDAR-based methods were implemented as baselines for the evaluation purpose. The baseline methods take LIDAR PCDs as the input and the ground points are removed. The initial position of the Object's 3D Bounding Box (*3D-OBB*)

TABLE II: Detailed information and challenging factors for each sequence.

ID	No. Frames	Obj.	Obj. Status	Ego. Status	Scene Cond.	Im-W	Im-H	PCD-H	PCD-W	PCD-L	OCC	POS	DIS	RVL
1	154	C	M	M	U	178	208	1.73	0.82	1.78	*	*	*	
2	154	Y	M	M	U	154	127	2.00	1.82	4.43		*	*	
3	101	C	S	M	U	93	42	2.19	1.89	5.53	*		*	
4	18	C	S	M	U	77	52	1.52	1.55	3.57			*	
5	58	C	S	M	U	19	17	1.54	1.66	4.14			*	
6	144	P	S	M	U	16	42	1.72	0.73	0.55			*	
7	78	C	M	M	U	54	21	1.48	1.59	3.46	*		*	*
8	78	C	M	M	U	193	77	3.01	2.59	11.84		*		*
9	122	C	M	M	U	100	302	1.59	1.65	3.55				*
10	314	C	M	M	U	152	87	1.64	1.67	3.63		*	*	*
11	297	C	M	M	U	36	36	1.62	1.62	4.50			*	
12	101	Y	M	M	U	8	26	1.64	0.33	1.57			*	
13	42	Y	M	M	U	15	36	1.64	0.33	1.57	*		*	
14	136	C	M	M	U	95	34	1.47	1.35	3.51	*	*	*	*
15	38	C	S	M	U	98	34	1.45	1.63	4.20	*			
16	51	C	M	M	U	52	34	1.57	1.65	4.10		*	*	
17	42	C	M	M	U	22	19	1.85	1.67	4.09			*	
18	31	C	S	M	U	52	34	1.45	1.60	4.22			*	
19	24	C	M	M	U	30	32	3.43	2.81	7.02			*	
20	390	C	M	M	U	18	13	1.25	1.59	3.55			*	
21	36	C	S	M	U	28	32	2.71	1.89	5.77			*	
22	65	C	S	M	U	76	28	1.72	1.73	4.71			*	
23	56	C	M	M	U	152	57	3.52	2.89	10.81	*	*	*	
24	474	C	M	M	U	274	97	3.52	2.89	10.81	*	*	*	*
25	63	P	M	M	U	16	30	1.63	0.40	0.83			*	
26	99	Y	M	M	D	39	39	1.81	0.59	1.89		*	*	*
27	41	P	M	M	D	25	42	1.53	0.61	0.73			*	
28	323	Y	S	M	U	25	38	1.72	0.78	1.70			*	
29	188	C	M	M	U	30	21	1.44	1.74	4.23	*	*	*	
30	51	C	M	M	U	126	37	1.50	1.54	4.09	*	*	*	*
31	41	P	M	S	D	70	105	1.63	0.66	0.89	*			
32	131	P	M	S	D	46	65	1.76	0.90	1.11	*	*		*
33	132	P	M	S	D	43	72	1.89	0.84	1.05	*	*		
34	140	P	M	S	D	33	63	1.83	0.73	1.16	*	*		
35	141	P	M	S	D	27	58	1.70	0.65	1.10	*	*		
36	112	P	M	S	D	33	66	1.84	0.78	1.03	*	*		
37	31	Y	M	S	D	35	47	1.84	0.50	1.60	*		*	
38	112	P	M	S	D	20	54	1.67	0.44	0.75	*		*	
39	145	P	M	S	D	19	53	1.95	0.62	0.74	*		*	
40	54	P	M	S	D	101	160	1.71	0.48	0.93	*			
41	45	P	M	S	D	196	224	1.64	0.55	0.94	*			
42	264	C	M	M	U	28	24	1.40	1.54	3.36	*		*	
43	71	P	M	M	D	89	124	1.61	0.91	0.91	*			
44	125	P	M	M	D	36	62	1.64	0.88	0.49	*		*	
45	146	V	S	M	D	45	56	2.56	2.05	5.86			*	
46	156	P	M	M	D	25	48	1.88	0.95	0.94			*	
47	45	P	M	M	D	29	58	1.67	0.70	0.94	*		*	
48	188	P	M	M	D	31	67	1.76	0.76	1.01		*	*	
49	359	P	M	M	D	28	51	1.80	0.90	0.94		*	*	
50	360	P	M	M	D	26	49	1.72	0.84	0.85		*	*	

is known, the size of the *3D-BB* is assumed fixed during the tracking, and the '*point model*' is used for the object representation.

A. Baseline KF 3D Object Tracker (3D-KF)

A 3D Constant Acceleration (CA) KF with a Gating Data Association (DA) is used for the robust tracking of the object centroid in the consecutive PCDs. Let the state of the filter be $x = \left[x, \dot{x}, \ddot{x}, y, \dot{y}, \ddot{y}, z, \dot{z}, \ddot{z}\right]^T$, where $\dot{x}, \dot{y}, \dot{z}$ and $\ddot{x}, \ddot{y}, \ddot{z}$ are velocity and acceleration in x, y, z location, respectively. The discrete time process model and the measurement model of the system are given by Equations (1) and (2), respectively.

$$x_t = A \cdot x_{t-1} + w_t \tag{1}$$

$$z_t = H \cdot x_t + v_t \tag{2}$$

where w_t and v_t represent the process and measurement noise, respectively, and A is the state transition matrix which applies the effect of each state parameter at time $t-1$ on the state at time t, and H is the transformation matrix that maps the state vector parameters into the measurement domain. To eliminate outliers and increase the robustness of the process, the search area is limited to a gate in the vicinity of the predicted KF location from the previous-step (accessible from: $x_t = A_T \times x_{t-1}$). If no measurement is available inside the gate area, the predicted KF value is used. Experiments with different gate sizes ($1 \times$ *3D-OBB*, $1.5 \times$ *3D-OBB* and $2 \times$ *3D-OBB*) were performed to conclude that the gate size of $1 \times$ *3D-OBB* provides a better result. The object orientation is achieved by subtraction of the current estimated location

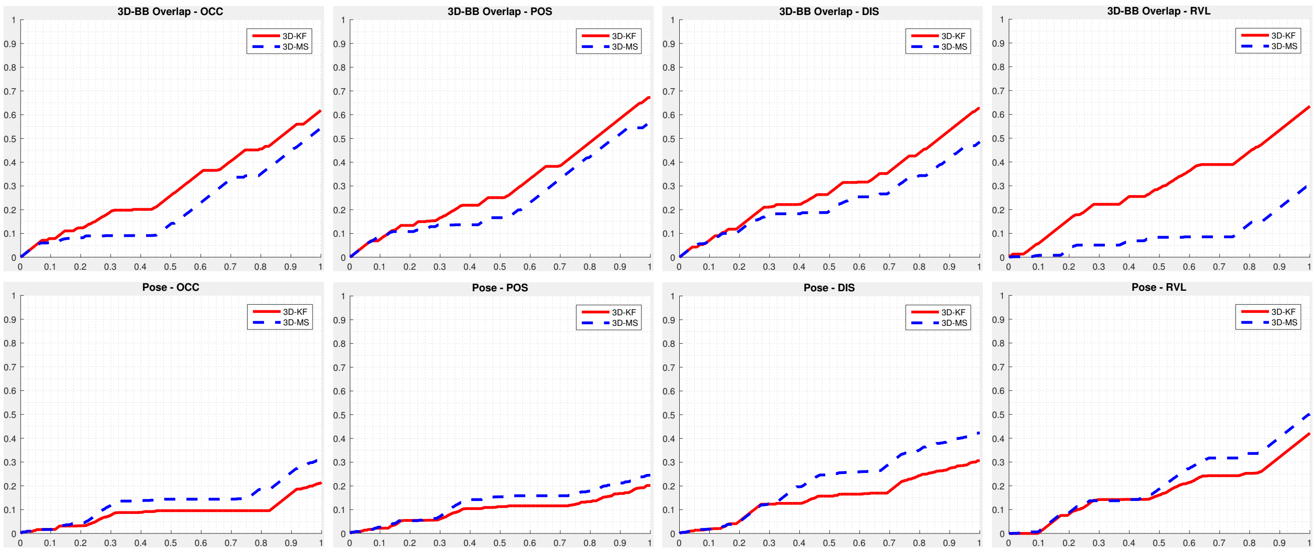


Fig. 4: The precision plot of 3D overlap rate and orientation error based-on *OCC, POS, DIS and RVL* challenges.

and the kalman prediction for the next time-step.

B. Baseline MS 3D Object Tracker (3D-MS)

In the 3D-MS approach, the Mean Shift (MS) iterative procedure is used to locate the object, as follows:

1) Compute the shift-vector: Given the center of the *3D-BB* as χ, the shift-vector between χ and the point set *P* inside the *3D-BB* is computed using,

$$m_k = \chi_k - \mu(P) \tag{3}$$

where $\mu(.)$ indicates the '*mean*' function, and k is the iteration index.

2) Translate *3D-BB*: The *3D-BB* is translated using the shift-vector,

$$\chi_{k+1} = \chi_k + m_k \tag{4}$$

The shift-vector always points toward the direction of the maximum increase in the density.

3) Iterate steps 1 and 2 until convergence: The MS iteratively shifts the *3D-BB* until the object is placed entirely within the *3D-BB*. MS is considered converged when the centroid movement $|m_k| < 0.5$m or the maximum number of iterations is met.

We conducted an experiment with different maximum number of iterations (3, 5 and 10) and observed that a maximum of 3 iterations provides a better result. The object orientation is achieved by subtracting the current estimated location and the previous location of the object.

V. Quantitative Evaluation Methodology

Different metrics have been proposed for the evaluation of object tracking methods [5], [6], [38]. For the quantitative evaluation, two assessment criteria are used: 1)- The precision plot of overlap success, and 2)- The precision plot of orientation success. The overlap rate (score or the intersection-over-union metric) in 3D is given by,

$$O_{3D} = \frac{volume(3D\text{-}BB \cap 3D\text{-}BB^g)}{volume(3D\text{-}BB \cup 3D\text{-}BB^g)} \tag{5}$$

where $3D\text{-}BB^g$ is the ground-truth 3D bounding boxes available in the KITTI dataset. The overlap rate ranges from 0 to 1. To be correct (to be considered a success), the overlap ratio O_{3D} must exceed 0.25, which is a standard threshold. The percentage of frames with successful occurrence is used as a metric to measure tracking performance.

The ground-truth for the orientation of the object in the KITTI dataset is given by the Yaw angle (Yaw angle describes the heading of the object, and corresponds to the rotation around Z-axis) The orientation error can be computed by,

$$E_\theta = |\overrightarrow{\theta} - \overrightarrow{\theta}^g| \tag{6}$$

where $\overrightarrow{\theta}^g$ is the ground-truth orientation of the object. The precision plot of orientation is given by the percentage of frames with E_θ less than certain threshold of about 10 degrees.

VI. Evaluation Results and Analysis of Metric Behaviors

The metrics for the two baseline trackers (3D-MS and 3D-KF) are computed based-on *OCC, POS, DIS and RVL* challenges and plotted in Fig. 4. The 3D-KF achieves higher success rate because the 3D-MS tracker may diverge to a denser nearby object (a local minima) instead of tracking the target object. Interestingly, 3D-KF performs much better in the *RVL* challenge because of a more accurate estimation of the object dynamics. However, the 3D-MS tracker has a higher precision in orientation estimation. The average computation time of baseline trackers is about 15 fps. The

Preprint submitted to 19th IEEE Intelligent Transportation Systems Conference. Received July 11, 2016.

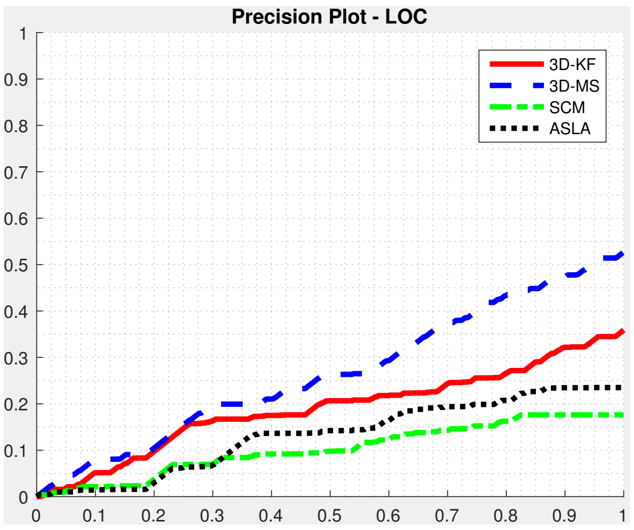


Fig. 5: The precision plot of location error.

experiment was carried out using a quad core 3.4 GHz processor with 8 GB RAM under MATLAB R2015a.

A. A Comparison of Base-line Trackers with the State-of-the-art Computer Vision based Object Trackers

3D-LIDAR sensors are opening their way for high-level perception tasks in computer vision, like object tracking, object recognition, and scene understanding. We found it interesting to compare our baseline trackers (3D-MS and 3D-KF) with two high-ranking state-of-the-art computer vision based object trackers (SCM [39], and ASLA [40]) in the Object Tracking Benchmark[4] [6]. SCM and ASLA run at about 1 fps and 6.5 fps, respectively. The precision plot is given by the percentage of success occurrence (localization error less than 20 pixels [6]), and is presented in Fig. 5.

We found that our baseline trackers, benefiting from highly reliable 3D-LIDAR data, have superior performance over the state-of-the-art approaches in Computer Vision field. This is because, in autonomous driving scenarios, ego-vehicle and objects are often moving. Therefore, object size and pose undergo severe changes (in the RGB image), which can easily mislead visual object trackers.

VII. Conclusion and Future Directions

In this work, we presented a brief survey for 3D object tracking in driving environments. A benchmark dataset based-on '*KITTI Object Tracking Evaluation*', a quantitative evaluation methodology, and two baseline trackers are provided for performance assessment.

We encourage other authors to evaluate their 3D object tracking methods using the proposed evaluation benchmark (*3D-OTD*), and make their results available in order to facilitate the quantitative comparison of future approaches. An extension of the dataset and codes to include more sequences and trackers remains an area for future work. Fusion of reliable 3D-LIDAR data with mature visual object tracker in the computer vision field could be a promising direction for future work.

[4]http://www.visual-tracking.net

Acknowledgments

This work has been supported by the FCT project "AMS-HMI2012 - RECI/EEIAUT/0181/2012" and project "ProjB-Diagnosis and Assisted Mobility - Centro-07-ST24-FEDER-002028" with FEDER funding, programs QREN and COMPETE.

References

[1] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *Acm computing surveys (CSUR)*, vol. 38, no. 4, p. 13, 2006.
[2] H. Yang, L. Shao, F. Zheng, L. Wang, and Z. Song, "Recent advances and trends in visual tracking: A review," *Neurocomputing*, vol. 74, no. 18, pp. 3823–3831, 2011.
[3] X. Li, W. Hu, C. Shen, Z. Zhang, A. Dick, and A. V. D. Hengel, "A survey of appearance models in visual object tracking," *ACM transactions on Intelligent Systems and Technology (TIST)*, vol. 4, no. 4, p. 58, 2013.
[4] A. W. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah, "Visual tracking: An experimental survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 7, pp. 1442–1468, 2014.
[5] M. Kristan, J. Matas, A. Leonardis, M. Felsberg, L. Cehovin, G. Fernandez, T. Vojir, G. Hager, G. Nebehay, and R. Pflugfelder, "The visual object tracking vot2015 challenge results," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2015, pp. 1–23.
[6] Y. Wu, J. Lim, and M.-H. Yang, "Object tracking benchmark," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1834–1848, 2015.
[7] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer *et al.*, "Autonomous driving in urban environments: Boss and the urban challenge," *Journal of Field Robotics*, vol. 25, no. 8, pp. 425–466, 2008.
[8] M. Montemerlo, J. Becker, S. Bhat, H. Dahlkamp, D. Dolgov, S. Ettinger, D. Haehnel, T. Hilden, G. Hoffmann, B. Huhnke *et al.*, "Junior: The stanford entry in the urban challenge," *Journal of field Robotics*, vol. 25, no. 9, pp. 569–597, 2008.
[9] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.
[10] A. Broggi, P. Grisleri, and P. Zani, "Sensors technologies for intelligent vehicles perception systems: A comparison between vision and 3d-lidar," in *Intelligent Transportation Systems-(ITSC), 2013 16th International IEEE Conference on.* IEEE, 2013, pp. 887–892.
[11] S. Hosseinyalamdary, Y. Balazadegan, and C. Toth, "Tracking 3d moving objects based on gps/imu navigation solution, laser scanner point cloud and gis data," *ISPRS International Journal of Geo-Information*, vol. 4, no. 3, pp. 1301–1316, 2015.
[12] A. Petrovskaya and S. Thrun, "Model based vehicle detection and tracking for autonomous urban driving," *Autonomous Robots*, vol. 26, no. 2-3, pp. 123–139, 2009.
[13] J. Choi, S. Ulbrich, B. Lichte, and M. Maurer, "Multi-target tracking using a 3d-lidar sensor for autonomous vehicles," in *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013).* IEEE, 2013, pp. 881–886.
[14] A. Asvadi, P. Peixoto, and U. Nunes, "Detection and tracking of moving objects using 2.5 d motion grids," in *Intelligent Transportation Systems (ITSC), 2015 IEEE 18th International Conference on.* IEEE, 2015, pp. 788–793.
[15] T. Miyasaka, Y. Ohama, and Y. Ninomiya, "Ego-motion estimation and moving object tracking using multi-layer lidar," in *Intelligent Vehicles Symposium, 2009 IEEE.* IEEE, 2009, pp. 151–156.
[16] A. Broggi, S. Cattani, M. Patander, M. Sabbatelli, and P. Zani, "A full-3d voxel-based dynamic obstacle detection for urban scenario using stereo vision," in *Intelligent Transportation Systems - (ITSC), 2013 16th International IEEE Conference on*, Oct 2013, pp. 71–76.
[17] A. Azim and O. Aycard, "Layer-based supervised classification of moving objects in outdoor dynamic environment using 3d laser scanner," in *Intelligent Vehicles Symposium Proceedings, 2014 IEEE*, June 2014, pp. 1408–1414.

[18] J. Ćesić, I. Marković, S. Jurić-Kavelj, and I. Petrović, "Short-term map based detection and tracking of moving objects with 3d laser on a vehicle," in *Informatics in Control, Automation and Robotics*. Springer, 2016, pp. 205–222.

[19] A. Vatavu, R. Danescu, and S. Nedevschi, "Stereovision-based multiple object tracking in traffic scenarios using free-form obstacle delimiters and particle filters," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 16, no. 1, pp. 498–511, 2015.

[20] D. Held, J. Levinson, and S. Thrun, "Precision tracking with sparse 3d and dense color 2d data," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 1138–1145.

[21] F. Moosmann and C. Stiller, "Joint self-localization and tracking of generic objects in 3d range data," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 1146–1152.

[22] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 32, no. 9, pp. 1627–1645, 2010.

[23] A. Geiger, M. Lauer, C. Wojek, C. Stiller, and R. Urtasun, "3d traffic scene understanding from movable platforms," *Pattern Analysis and Machine Intelligence (PAMI)*, 2014.

[24] L. Zhang, Y. Li, and R. Nevatia, "Global data association for multi-object tracking using network flows," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE, 2008, pp. 1–8.

[25] H. Pirsiavash, D. Ramanan, and C. C. Fowlkes, "Globally-optimal greedy algorithms for tracking a variable number of objects," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE, 2011, pp. 1201–1208.

[26] A. Milan, S. Roth, and K. Schindler, "Continuous energy minimization for multitarget tracking," *IEEE transactions on pattern analysis and machine intelligence*, vol. 36, no. 1, pp. 58–72, 2014.

[27] P. Lenz, A. Geiger, and R. Urtasun, "Followme: Efficient online min-cost flow tracking with bounded memory and computation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 4364–4372.

[28] J. Hong Yoon, C.-R. Lee, M.-H. Yang, and K.-J. Yoon, "Online multi-object tracking via structural constraint event aggregation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1392–1400.

[29] L. Leal-Taixé, A. Milan, I. Reid, S. Roth, and K. Schindler, "Motchallenge 2015: Towards a benchmark for multi-target tracking," *arXiv preprint arXiv:1504.01942*, 2015.

[30] A. Milan, L. Leal-Taixe, I. Reid, S. Roth, and K. Schindler, "Mot16: A benchmark for multi-object tracking," *arXiv preprint arXiv:1603.00831*, 2016.

[31] A. Ošep, A. Hermans, F. Engelmann, D. Klostermann, , M. Mathias, and B. Leibe, "Multi-scale object candidates for generic object tracking in street scenes," in *ICRA*, 2016.

[32] A. Teichman and S. Thrun, "Tracking-based semi-supervised learning," *The International Journal of Robotics Research*, vol. 31, no. 7, pp. 804–818, 2012.

[33] R. Kaestner, J. Maye, Y. Pilat, and R. Siegwart, "Generative object detection and tracking in 3d range data," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 3075–3081.

[34] A. Dewan, T. Caselitz, G. D. Tipaldi, and W. Burgard, "Motion-based detection and tracking in 3d lidar scans," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, Stockholm, Sweden, 2016.

[35] S. Sivaraman and M. M. Trivedi, "Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 14, no. 4, pp. 1773–1795, 2013.

[36] A. Mukhtar, L. Xia, and T. B. Tang, "Vehicle detection techniques for collision avoidance systems: A review," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 5, pp. 2318–2338, 2015.

[37] A. Petrovskaya, M. Perrollaz, L. Oliveira, L. Spinello, R. Triebel, A. Makris, J.-D. Yoder, C. Laugier, U. Nunes, and P. Bessiere, "Awareness of road scene participants for autonomous driving," in *Handbook of Intelligent Vehicles*. Springer, 2012, pp. 1383–1432.

[38] L. Čehovin, A. Leonardis, and M. Kristan, "Visual object tracking performance measures revisited," *IEEE Transactions on Image Processing*, vol. 25, no. 3, pp. 1261–1274, 2016.

[39] W. Zhong, H. Lu, and M.-H. Yang, "Robust object tracking via sparse collaborative appearance model," *IEEE Transactions on Image Processing*, vol. 23, no. 5, pp. 2356–2368, 2014.

[40] X. Jia, H. Lu, and M.-H. Yang, "Visual tracking via adaptive structural local sparse appearance model," in *Computer vision and pattern recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 1822–1829.

Preprint submitted to 19th IEEE Intelligent Transportation Systems Conference.
Received July 11, 2016.