Bachelor Project

Czech
Technical
University
in Prague

F3

Faculty of Electrical Engineering
Department of Cybernetics

Recognition of Road Traffic Participants in LiDAR Point Clouds

Josef Čech

Supervisor: doc. Dr. Ing. Radim Šára
Field of study: Pattern Recognition and Machine Learning
May 2019

BACHELOR'S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Čech Josef** Personal ID number: **466176**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Cybernetics**

Study program: **Cybernetics and Robotics**

II. Bachelor's thesis details

Bachelor's thesis title in English:

Recognition of Road Traffic Participants in LiDAR Point Clouds

Bachelor's thesis title in Czech:

Rozpoznávání účastníků silničního provozu v bodových množinách z LiDARů

Guidelines:

Implement, experimentally validate and evaluate an object classification method for segmented LiDAR point clouds. The goal is to assign class labels to point cloud segments corresponding to individual traffic participants in street scenes captured from the vantage point of an egovehicle that is taking part in the traffic. Consider the following preliminary choice of classes: cars, trucks or busses, cyclists, pedestrians. Assume also a universal class to which all possible segments belong.
The segments will be small point clouds of about tens to hundreds of points. Segmented input data will be provided. The output form the method will be the most probable class label or the „outlier" label corresponding to the universal class. The method should cope with a variable number of points per object (depending on the distance from the observer) and with a variable rotation angle about the vertical axis.
1. Collect an annotated set of segment samples, analyze it and revise the initial choice of object classes.
2. Propose features suitable for statistical pattern recognition of segment classes.
3. Select a suitable classifier that will use those features.
4. Test the classifier on a suitable annotated dataset and compare the results with published methods.

Bibliography / sources:

[1] Bishop, Christopher M. Pattern Recognition and Machine Learning. Springer 2006.
[2] Schlesinger, M.I., Hlaváč, V. Ten Lectures on Statistical and Structural Pattern Recognition. Kluwer Academic Publishers, 2002.
[3] Matti Lehtomäki, M. et al.Object Classification and Recognition From Mobile Laser Scanning Point Clouds in a Road Environment, IEEE Trans on Geoscience and Remote Sensing, Vol. 54, No. 2, 2016.
[4] Hackel, T. Large-scale Machine Learning for Point Cloud Processing. Ph.D Thesis, Institute of Geodesy and Photogrammetry, ETH Zurich, 2018.

Name and workplace of bachelor's thesis supervisor:

doc. Dr. Ing. Radim Šára, Vision for Robotics and Autonomous Systems, FEE

Name and workplace of second bachelor's thesis supervisor or consultant:

Date of bachelor's thesis assignment: **11.02.2019** Deadline for bachelor thesis submission: **24.05.2019**

Assignment valid until: **20.09.2020**

doc. Dr. Ing. Radim Šára Supervisor's signature	doc. Ing. Tomáš Svoboda, Ph.D. Head of department's signature	prof. Ing. Pavel Ripka, CSc. Dean's signature

III. Assignment receipt

The student acknowledges that the bachelor's thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the bachelor's thesis, the author must state the names of consultants and include a list of references.

.	
Date of assignment receipt	Student's signature

Acknowledgements

Foremost, I would like to express my sincere gratitude to my supervisor doc. Dr. Ing. Radim Šára for the continuous support, patience and motivation during the research and writing of this thesis.

Special thanks go to Ing. Dominik Fiala who provided the segmentation and helped to create the Wolfsburg dataset.

Also, I would like to thank to my parents and girlfriend whose love and support made this work a pleasant undertaking.

This work has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688652.

This work was also supported by the OP VVV MEYS funded project CZ.02.1.01/0.0/0.0/16 019/0000765 "Research Center for Informatics."

Declaration

I declare that the presented work was developed independently and that I have listed all sources of information used within it in accordance with the methodical instructions for observing the ethical principles in the preparation of university theses.

Signature:

Prague, May 20, 2019

Abstract

Road traffic participants captured by LiDAR sensor can be recognized even by the human eye.

In this thesis we present several classification models for automatic recognition of dynamic objects in the urban environment. All the mechanisms used are based on traditional machine learning, i.e., k-Nearest Neighbors rule, Gaussian Mixture Model and Random Forest.

The features applied are essentially divided into moment family and 3D Haar-like feature family.

Experiments are performed with the 4-fold cross-validation on a public Sydney dataset (347 segments) as well as on an internal Wolfsburg dataset (3203 segments; not yet public). Both scanned by Velodyne LiDAR.

Segments are labeled into seven classes: *pedestrian*, *cyclist*, *biker*, *car*, *van*, *bus* and *truck*. In case of uncertainty about the class, objects are labeled as an *outlier*. In addition, a small portion of undefined objects is added to the validation dataset and is used for a part of the experiments.

The classification performance on seven specific classes reaches a state-of-the-art accuracy of over 96 %.

Keywords: computer vision, object recognition, LiDAR data, road traffic

Supervisor: doc. Dr. Ing. Radim Šára

Abstrakt

Účastníky silničního provozu nasnímané senzorem LiDAR lze rozeznat i pouhým okem.

V této práci představíme několik klasifikačních modelů pro automatické rozpoznávání dynamických objektů v městském prostředí. Všechny použité mechanismy jsou založené na tradičním strojovém učení, tj. k nejbližších sousedů, směs Gaussových rozdělení a náhodné lesy.

Příznaky jsou rozděleny na momentovou rodinu a 3D Haarovu příznakovou rodinu.

Experimenty se čtyřnásobnou křížovou validací byly provedeny na veřejném Sydney datasetu (347 segmentů) i na interním Wolfsburg datasetu (3203 segmentů; zatím není zvěřejněný). Oba dva nasnímané LiDARem Velodyne.

Segmenty jsou rozdělené do sedmi tříd: *chodec*, *cyklista*, *motocyklista*, *osobní automobil*, *dodávka*, *autobus* a *kamion*. V případě nejistoty o třídě je objektům přiřazena univerzální třída. Část validačního datasetu je navíc tvořena těmito univerzálními objekty a je použita pro část experimentů.

Výkon klasifikace na sedmi konkrétních třídách dosahuje state-of-the-art přesnosti přes 96 %.

Klíčová slova: počítačové vidění, rozpoznávání objektů, LiDAR data, silniční provoz

Překlad názvu: Rozpoznávání účastníků silničního provozu v bodových množinách z LiDARů

Contents

Figures

Tables

Chapter 1

Introduction

1.1 Motivation and Task

A 3D object recognition is a central task in any intelligent or autonomous vehicle. A correct recognition of road traffic participants can be used to avoid accidents or to predict their future behavior which would serve valuable information for an optimal control and decision-making systems. Nonetheless, 3D object recognition in the challenging urban environment still remains an open research problem.

The aim of this thesis is to find a precise and fast classification method for real-time recognition of road traffic participants from a vehicle that is taking part in the traffic as well.

Although we can benefit from a combination of multiple sensors (e.g., a fisheye camera, a radar or an ultrasonic sensor), for the purpose of this thesis we have decided to use only segmented LiDAR data. In Figure 1.1 there is an example of how the autonomous vehicle can perceive. As you can see, the object captured by the LiDAR sensor can be recognized by the human eye just as well as the RGB image.

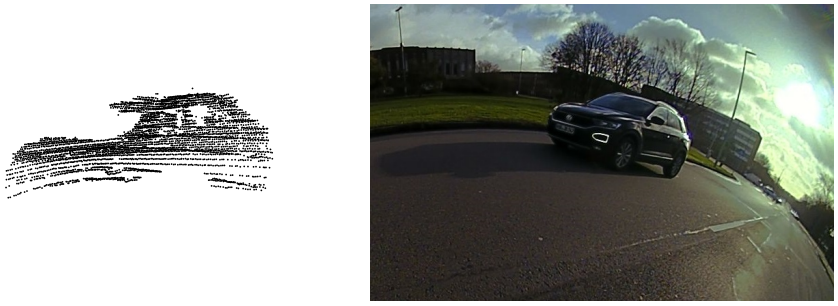
The LiDAR is a time-of-flight sensor. The light photon is emitted, hits the object and bounces back to the device at which point time and distance is measured.

The LiDAR segments are sparse point clouds of about tens to thousands of points (see Table 3.3 for per class average). Several point cloud segments of the same object captured over time will be referred to as a track.

In these days, an autonomous vehicle phenomenon is one of the fastest growing field of practical application of LiDAR point cloud recognition algorithms with over 4000 papers published from 2018[1].

In order to control an autonomous vehicle it is crucial to ensure accurate perception of the surrounding world (especially fast-moving obstacles).

[1] Papers from 2018 to 2019 related to *LiDAR point cloud recognition in autonomous vehicle environment*, according to Google Scholar.



(a): LiDAR segment. **(b):** RGB camera image.

Figure 1.1: Side-by-side comparison of car captured by both the LiDAR (a) and the RGB camera (b).

We seek a classification model that will be invariant to

1. the number of points per object (density varies by distance from the observer),
2. a rotation around the vertical axis, i.e., orientation.

Both are needed for safe and optimal control of an autonomous vehicle. Recognition of distant objects is essential for early decision making. Nevertheless, distant objects are naturally hard to read.

This raises great questions. How accurate a classification of objects using only the sparse plain depth LiDAR data can be? And is the accuracy sufficient for reliable use in the real traffic?

1.2 Related Work

The recognition of the road traffic participants in the LiDAR point clouds is an active research area. There have been a wide variety of papers written on this topic in the recent past and some of them have run into closely related problems.

Approaches basically vary in class selection, classification method and feature choice.

Traditional Machine Learning. Similarly to our approach, in [GKF09] they use shape features (e.g., the number of points, average height, standard deviation in height, etc.) and spin image descriptor with k-Nearest Neighbor classifier, Random Forest and SVM to distinguish between 16 classes (e.g., car, traffic light, fire hydrant, mailing box, etc.).

Promising results are presented by [TLT11b]. Besides individual segment features, they have also developed holistic descriptors of whole track such as mean speed or maximum acceleration. Well-segmented objects are classified into car, pedestrian, bicyclist, and background classes using an augmented discrete Bayes filter.

Classification by the multiclass SVM is also done by [LJH+16] using local descriptor histograms (LDHs), spin images and general shape and point distribution features (e.g., the number of voxels, the height, the mean height or the standard deviation of the z-coordinate). The urban objects of interest in this paper are trees, lamp posts, traffic signs, cars, pedestrians and hoardings.

The paper [DDQHD13] proposes an alternative to hand-crafted features, the unsupervised feature learning. They focus on scans produced by an outdoor 3D mobile laser scanner (e.g., 3D Velodyne LiDAR on a ground vehicle), just like us. Among the 14 classes are car, bus, pillar, tree, or truck, to name a few.

These methods are synergistic to ours, and will be used for comparison with our results in Section 5.1.

Convolutional Neural Networks. Nowadays, we can also see much effort in using deep learning for the point cloud segmentation and classification [WSL+18], [MS15], [Pro10].

Both the machine learning and the deep learning methods are investigated in [Hac18] and it is shown that the deep learning approach outperforms the traditional machine learning. However, it is extremely computationally demanding.

3D Haar-Like Feature Family. The 3D Haar-like feature family motivated by successful results in digital photography [VJ+01] has been used in previous studies [EDP06] for bottles recognition and [PN13] for static engineering parts (e.g., valve, tube, ladder, etc.) and street objects (e.g., car, tree, stop sign, etc.) recognition without a priori object segmentation.

Public LiDAR Datasets. For benchmarking there is a few annotated urban LiDAR object datasets available on the internet such as [HSL+17] (over 4 billion points), [MBVH09] (over 1.3 million segments), [TLT11a] (1.3 million segments), [SMGD14] (642 segments), [QUD13] (631 segments). The last one will be used for a direct comparison with results presented in [DDQHD13] and [Qua13].

1.3 Our Approach

Features and Methods. There is not a general consensus about the best classification method. Since the deep learning methods are demanding massive computational resources that are not available in the vehicle, our approach relies on traditional machine learning. In this thesis, we compare three methods for point cloud segment classification (i.e., k-Nearest Neighbor, Gaussian Mixture Model and Random Forest) using four descriptor types (i.e., a zero-order moment, second-order moment, rescale factor $\varkappa$ and 3D Haar-like features).

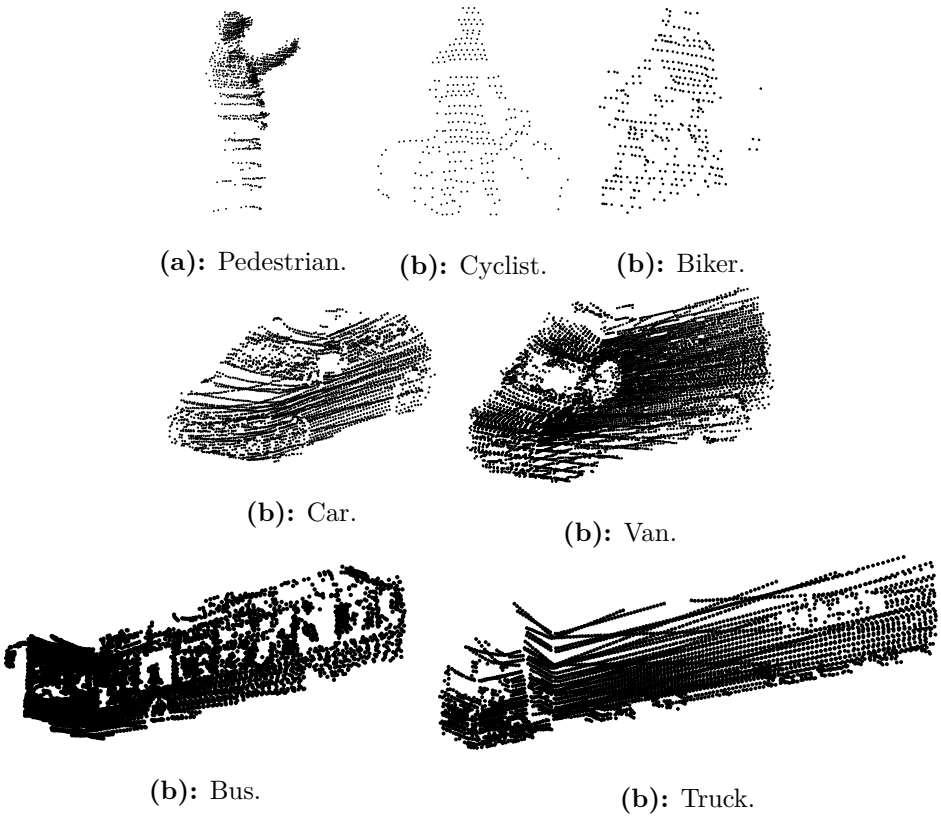


Figure 1.2: LiDAR segment example of each class.

Classes Selection. It is essential to design a reasonable classes division.

For instance, it is worth distinguishing a bus from a truck. Both vehicles are among the largest on the road but behave differently. Correct recognition of a bus with a map of its stops can significantly increase the traffic flow.

On the other hand, it is not necessary to divide small two-track vehicles into a 4WD, sport car, pickup or utility vehicle.

We definitely involve pedestrians because they are the most vulnerable group of the road traffic participants and must not be harmed.

Finally, we distinguish between seven specific classes[2] *pedestrian*, *cyclist*, *biker* (i.e., motorbike rider), *car*, *van*, *bus*, *truck* and a universal class *outlier* for an unspecified object. Figure 1.2 shows a representative example of each specific class.

The *outlier* class serves as a mechanism for an incorrect detection controlling. In case of uncertainty about the class, the segment will be labeled as the *outlier*. The effect of an uncertainty threshold will be shown in Chapter 4.

Data Processing Structure. For reader's convenience, we present a full recognition workflow diagram in Figure 1.3.

As a part of this project we have completed an annotated LiDAR dataset

[2] If they are present in the selected dataset.

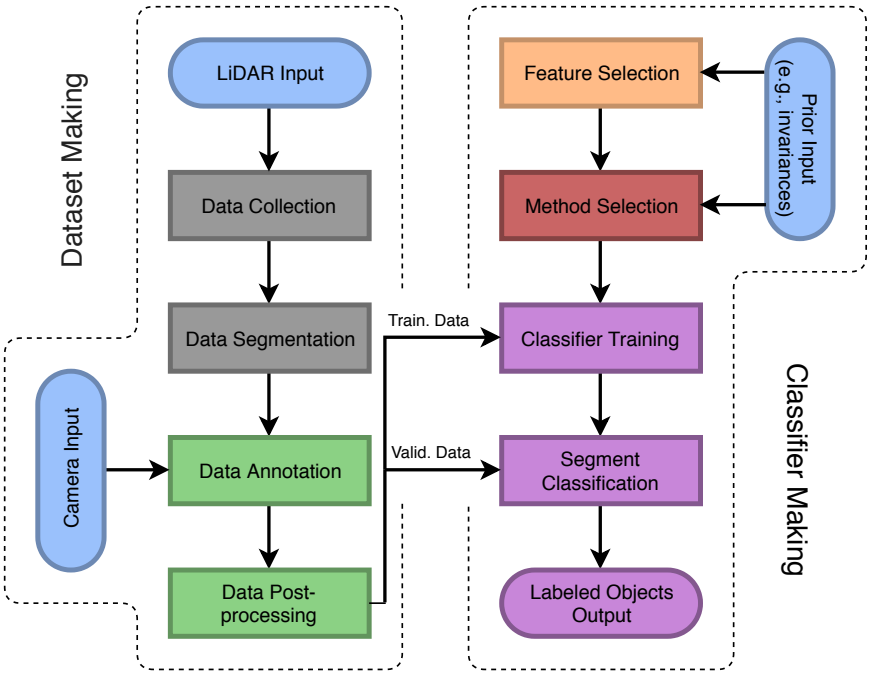


Figure 1.3: The recognition workflow diagram.

of urban objects[3]. Our contribution to the dataset creation is highlighted in green on the left side of the diagram and further described in Section 3.1.

The right side of the diagram shows the classifier design and testing.

The first two steps, feature selection (highlighted in orange) and method selection (highlighted in red), are introduced in Section 2.1 and Section 2.2, respectively.

All classification models were evaluated on our internal Wolfsburg dataset as well as public Sydney dataset in Chapter 4. This phase is shown in purple.

Our results will be broadly discussed in final Chapter 5.

The main contributions of the thesis are

1. creation of a simple classifier without complicated mechanisms,
2. inspection of the accuracy of 3D object classification captured simultaneously by four LiDAR sensors,
3. generalization of a Haar wavelet and making use of 3D Haar-like features in the combination with common geometrical features,
4. completion of the Wolfsburg dataset.

[3]The first step (i.e., data collection from LiDAR input) was done by Volkswagen AG Group Research in Wolfsburg. The second one (i.e., data segmentation) was provided by Ing. Dominik Fiala from Czech Technical University in Prague [Fia18].

Chapter 2

Theory

In this chapter, we describe the theoretical basis for the upcoming chapters, including the mathematical apparatus used.

First of all, we describe feature families in detail. After that, we will take a glance at classification methods. In the end of this chapter we will present measurement methods of the classification performance.

2.1 Feature Families for Feature Selection

The purpose of feature extraction is to reduce redundancy in input data and discard information unrelated to the class recognition task. In the feature extraction stage the initial point cloud segment is projected into a more manageable feature space. In that space it is easier to assign an appropriate class label of the point cloud segment.

To select an informative descriptor in the feature selection stage, we need sufficiently rich feature families.

Note that the z-axis increases from bottom to top and that $z = 0$ is the ground plane.

2.1.1 The Moment Family

Let's define moments up to the second order as follows.

Definition 2.1. Consider a 3D point cloud represented by a set of points $\boldsymbol{x}_i \in \mathbb{R}^3$. Each point contains random variables x-, y- and z-coordinate in the world Cartesian coordinate system, respectively. Let $N \in \mathbb{R}$ be a total number of points in the point cloud and let M_i be an i-th moment of the

point cloud, then:

$$M_0 = \sum_{i=1}^{N} 1 = N, \tag{2.1}$$

$$\boldsymbol{M}_1 = \sum_{i=1}^{N} \boldsymbol{x}_i, \tag{2.2}$$

$$\mathbf{M}_2 = \sum_{i=1}^{N} \boldsymbol{x}_i \, \boldsymbol{x}_i^{\mathrm{T}}, \tag{2.3}$$

are zero, first and second order moments, in the order given.

2.1.1.1 The Zero-order Moment

The zero-order moment M_0 (Equation 2.1) is simple and fast to extract feature. On its own it is a too naive feature and it is supposed to fail in many cases. However, it can be beneficial in combination with other features with a sufficiently sophisticated model that can be trained to recognize how objects typically look like with a different number of points.

2.1.1.2 The Second-order Moment

The main objective of this feature family is satisfaction of the rotational invariance. Note, however, that we only consider the positions of vehicles with chassis parallel to the ground, i.e., rotation only about the vertical axis.

Another intention is a possibility of the feature space visualization. Given the three-dimensional plotting limit, the aim was to select a maximum of three features to keep the feature space simple and preserve important information at the same time.

Definition 2.2. Consider again the 3D point cloud represented by a set of points $\boldsymbol{x}_i \in \mathbb{R}^3$. Using Equations (2.1), (2.2) and (2.3) we write the center of mass $\boldsymbol{\mu} \in \mathbb{R}^3$ of the point cloud as

$$\boldsymbol{\mu} = \frac{\boldsymbol{M}_1}{M_0}, \tag{2.4}$$

and the covariance matrix $\mathbf{S}$ of the point cloud as[4]

$$\mathbf{S} = \frac{\mathbf{M}_2}{M_0} - \boldsymbol{\mu}\,\boldsymbol{\mu}^{\mathrm{T}}. \tag{2.5}$$

Remark 2.3. The covariance matrix $\mathbf{S}$ is symmetric and positive semidefinite. It implies that its diagonal elements are the variances of the particular coordinates, which can never be negative.

[4] We avoid the $\boldsymbol{\Sigma}$-notation of covariance for the rest of the thesis not to confuse with the sum symbol.

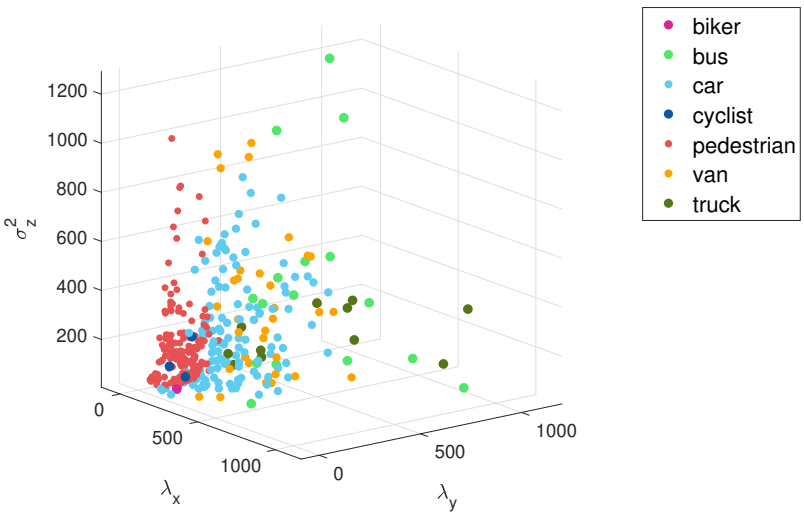


Figure 2.1: Feature space view of $[\lambda_x, \lambda_y, \sigma_z^2]$.

Next, we split the matrix $\mathbf{S}$ to

$$\mathbf{S} = \begin{bmatrix} & \mathbf{S}_A & & s_{\mathrm{M}_{13}} \\ & & & s_{\mathrm{M}_{23}} \\ s_{\mathrm{M}_{31}} & & s_{\mathrm{M}_{32}} & s_{\mathrm{M}_{33}} \end{bmatrix}. \tag{2.6}$$

Eigenvalues of $\mathbf{S}_A$, i.e., λ_x, λ_y, are the first two features. As we have observed (Remark 2.3), the diagonal element $s_{\mathrm{M}_{33}}$ is equivalent to variance in z-coordinate σ_z^2 which yields the third feature.

Figure 2.1 shows a view of the $[\lambda_x, \lambda_y, \sigma_z^2]$ feature space. As we can see, classes are interspersed and are not linearly separable with this basic geometric descriptor.

2.1.2 3D Haar-Like Feature Family

More advanced classifiers (especially Random Forest), are able to handle higher-dimensional feature space efficiently [AG97]. We introduce the Haar wavelet, to extend the number of feature families in this thesis.

The one-dimensional Haar wavelet is a three-valued piecewise continuous function defined as follows.

Definition 2.4. [RB98]
Let the mother wavelet be

$$\psi(t) = \begin{cases} 1 & 0 \leq t < \frac{1}{2}, \\ -1 & \frac{1}{2} \leq t < 1, \\ 0 & \text{otherwise.} \end{cases} \tag{2.7}$$

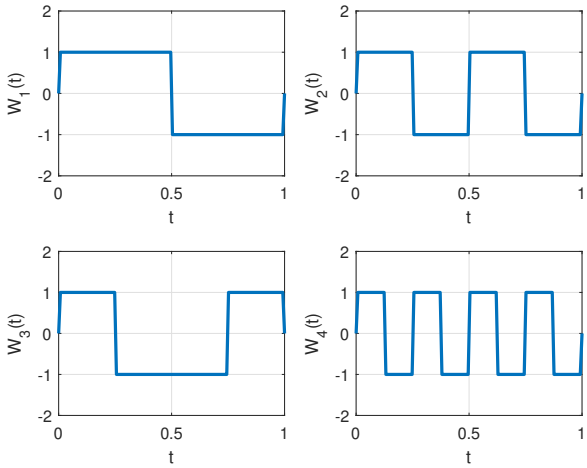


Figure 2.2: Haar mother wavelet (W_1) and three derived wavelets (W_2, W_3, W_4).

Its time domain scaling function is

$$\phi(t) = \begin{cases} 1 & 0 \leq t < 1, \\ 0 & \text{otherwise.} \end{cases} \tag{2.8}$$

Remark 2.5. The Haar mother wavelet satisfies following properties [RB98]:

1. The function integrates to zero:

$$\int_{-\infty}^{\infty} \psi(t)\mathrm{d}t = 0. \tag{2.9}$$

2. It is square integrable or, equivalently, has finite energy:

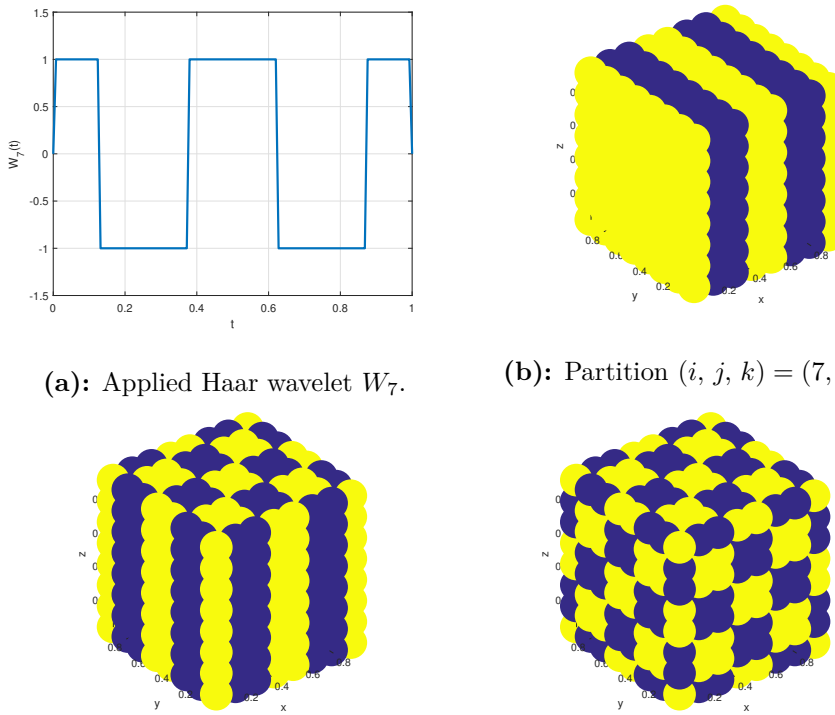
$$\int_{-\infty}^{\infty} |\psi(t)|^2 \,\mathrm{d}t = 1. \tag{2.10}$$

Definition 2.6. Taking $W_0(t) = \phi(t)$ and $W_1(t) = \psi(t)$ a complete Haar wavelet sequence for $n \in \mathbb{Z}^0$ is defined by induction

$$W_{2n}(t) = W_n(2t) + W_n(2t-1), \tag{2.11}$$
$$W_{2n+1}(t) = W_n(2t) - W_n(2t-1). \tag{2.12}$$
$$\tag{2.13}$$

Roughly speaking the one-dimensional Haar wavelets divide the unit interval to several subintervals and assigns ± 1 to each of them. At the endpoints and outside of the unit interval it is equal to zero (see Figure 2.2).

The three-dimensional Haar wavelet is a straightforward generalization of the one-dimensional Haar wavelet as long as we simply apply it according to each axis.



(a): Applied Haar wavelet W_7.

(b): Partition $(i,\, j,\, k) = (7,\, 0,\, 0)$.

(c): Partition $(i,\, j,\, k) = (7,\, 7,\, 0)$.

(d): Partition $(i,\, j,\, k) = (7,\, 7,\, 7)$.

Figure 2.3: The unit cube partitioned by the single wavelet.

Definition 2.7. Let W_i, W_j, W_k be the 1D Haar wavelets (or the scaling function itself) defined by Equations (2.11) or (2.12). A 3D Haar wavelet is defined by an intersection

$$C_{i,j,k}(x,\, y,\, z) = W_i(x)\, W_j(y)\, W_k(z), \qquad i,\, j,\, k \geq 0, \tag{2.14}$$

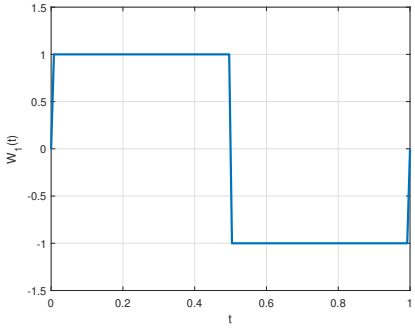
where the domain of W_i (resp. W_j and W_k) is the unit interval on x-axis (resp. y-axis and z-axis).

The intersection divides the unit cube to several cuboids (as shown in Figure 2.3). The value of each cuboid is $\{\pm 1\}$.

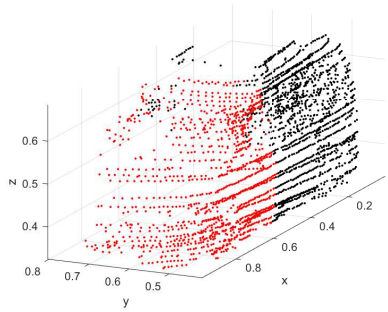
The singular case $(i,\, j,\, k) = (0,\, 0,\, 0)$ (i.e., the intersection of three unit pulses) is identical to zero-order moment M_0 defined within the moment family and will be excluded from the 3D Haar wavelet sequence.

It might seem that we have not gained much by doing so. However, consider a normalized point cloud segment (see Section 2.1.2.1).

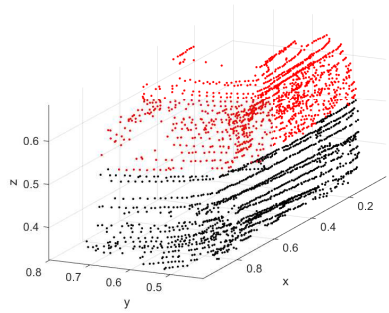
Each point of the segment belongs to some cuboid (Figure 2.4 shows a car segment in three simple configurations). It sufficies to count up all points in each cuboid and multiply by the corresponding sign. The resulting feature is obtained by summing up all those numbers, i.e., the total number of points in cuboids with -1 is subtracted from the total number of points in cuboids with $+1$.



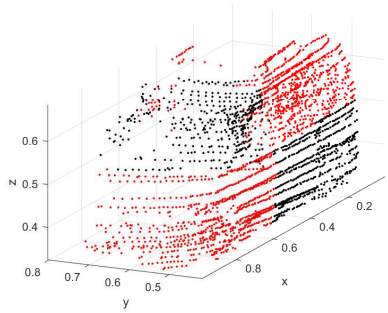
(a): Applied Haar wavelet W_1.



(b): Partition $(i,\, j,\, k) = (1,\, 0,\, 0)$.



(c): Partition $(i,\, j,\, k) = (0,\, 0,\, 1)$.



(d): Partition $(i,\, j,\, k) = (1,\, 0,\, 1)$.

Figure 2.4: The car segment partitioned by the single wavelet.

Wavelets (α)	1	2	3	4	5	6	7	8
Features ($\mathcal{H}_\alpha$)	7	26	63	124	215	342	511	728

Table 2.1: The number of wavelets depending on α.

We will denote the 3D Haar-like features by $\mathcal{H}_\alpha$, where the subscript $\alpha \in \mathbb{N}$ indicates the number of used wavelets.

Table 2.1 shows how rich the Haar feature family is. The size of $\mathcal{H}_\alpha$ is one less[5] than a number of 3-element variations of $(\alpha + 1)$ elements[6] with repetition allowed, i.e., $V'(\alpha + 1,\, 3) - 1 = (\alpha + 1)^3 - 1$. For example, $\mathcal{H}_1 = \{(0,0,1),\, (0,1,0),\, (0,1,1),\, (1,0,0),\, (1,0,1),\, (1,1,0),\, (1,1,1)\}$, $\mathcal{H}_2 = \{(0,0,1),\, (0,0,2),\, (0,1,0),\, (0,1,1),\, \ldots,\, (2,2,2)\}$, etc.

Iterating over α, Haar wavelets brings a large amount of distinct features[7].

2.1.2.1 Pre-processing of the Point Cloud Segment

The method described above assumes a normalized point cloud segment. For that reason, it is necessary to perform several linear transformations to the

[5] We subtract the singular case of the three unit pulses intersection.
[6] To simplify the formula, we consider the unit pulse to be a wavelet too.
[7] Although, not an infinity on a discrete point cloud segment.

data.

At first, the point cloud segment is shifted so that the center of mass is at the origin.

Next, in order to reduce the number of the object orientations it is rotated so that dominant eigenvector of the covariance matrix of the point cloud segment is parallel to the x-axis.

Once this is done, the segment is rescaled by the factor $\varkappa$ that is equal to the distance of the farthest point from the origin.

Finally, the segment is shifted so that the center of mass is in the $(\frac{1}{2}, \frac{1}{2}, \frac{1}{2})$ and it fits in the unit cube.

It is important to pay attention to the transformations order. For example, rotating the point cloud after the rescale and the second shifting can cause some points lying out of the unit cube.

After the normalization process it is straightforward to divide the point cloud into a regular voxel structure as a basis for the feature computation.

Additionally, the scaling factor $\varkappa$ can be used as a feature as well.

2.2 Classification Methods

2.2.1 k-Nearest Neighbors Rule

The k-Nearest Neighbors rule classifier [DHS01] serves as a baseline classifier as it is a non-parametric[8] and a distribution flexible method.

Quoting essentially verbatim from [FH04], we formally define a k-Nearest Neighbor rule as follows.

Definition 2.8. Let $\mathcal{T}_{XY} = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_l, y_l)\}$ be a set of prototype vectors $\boldsymbol{x}_i \in \mathcal{X} \subseteq \mathbb{R}^n$ and the corresponding classes be $y_i \in \mathcal{Y} = \{1, \ldots, c\}$. Let $\boldsymbol{x} \in \mathbb{R}^n$ be an incoming vector with an unknown class label.

Let $\mathbb{B}^n(\boldsymbol{x}) = \{\boldsymbol{x}' : \| \boldsymbol{x} - \boldsymbol{x}' \| \leq r^2\}$ be a n-dimensional ball of radius $r \in \mathbb{R}^n$ centered at the vector $\boldsymbol{x}$ in which k prototype vectors $\boldsymbol{x}_i$, $i \in \{1, \ldots, l\}$ lies, i.e., $|\{\boldsymbol{x}_i : \boldsymbol{x}_i \in \mathbb{B}^n(\boldsymbol{x})\}| = k$.

Assuming uniform weights of all classes, we compute a posterior probability of class y

$$p(y \,|\, \boldsymbol{x}) = \frac{v(\boldsymbol{x}, y)}{k}, \tag{2.15}$$

where $v(\boldsymbol{x}, y)$ is the number of prototype vectors $\boldsymbol{x}_i$ with class $y_i = y$ which lie in the n-dimensional ball $\mathbb{B}^n(\boldsymbol{x})$.

The k-Nearest Neighbor (k-NN) classification rule $q : \mathcal{X} \mapsto \mathcal{Y}$ is then defined as

$$q(\boldsymbol{x}) = \operatorname*{argmax}_{y \in \mathcal{Y}} p(y \,|\, \boldsymbol{x}). \tag{2.16}$$

Running the k parameter optimization, we found the $k = 7$ to be the best value in both datasets. A MATLAB implementation `fitcknn` has been used to perform the k-Nearest Neighbor classification.

[8] Except for the number k.

2.2.2 Gaussian Mixture Model

The general multivariate Gaussian density function in n dimensions is given by

$$p(\boldsymbol{x}; \boldsymbol{\mu}, \mathbf{S}) = \frac{1}{\sqrt{\det(2\pi\,\mathbf{S})}} \exp\left\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^{\mathrm{T}}\mathbf{S}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right\}, \tag{2.17}$$

where $\boldsymbol{\mu}$ is a n-dimensional mean vector and $\mathbf{S} \in \mathbb{R}^{n\times n}$ is a covariance matrix that is symmetric and positive-definite.

Taking the prototype vector $\boldsymbol{x}_i$ as a mean vector, we represent it with a kernel function $p(\boldsymbol{x}; \boldsymbol{x}_i, \mathbf{S})$, where $\mathbf{S}$ is a diagonal covariance matrix corresponding to the class of the prototype vector $\boldsymbol{x}_i$.

Following the core idea from [Bis06] we formulate an adjusted Gaussian Mixture Model classifier as follows.

Definition 2.9. Let $\mathcal{T}_{XY} = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_l, y_l)\}$ be a set of prototype vectors $\boldsymbol{x}_i \in \mathcal{X} \subseteq \mathbb{R}^n$ and the corresponding classes be $y_i \in \mathcal{Y} = \{1, \ldots, c\}$. Let $\mathcal{X}_j \subset \mathcal{X}$ be a subset of all prototype vectors from class y_j. Let $\boldsymbol{x} \in \mathbb{R}^n$ be an incoming vector with an unknown class label.

Class y_j is represented by a mixture of Gaussian components in the form

$$p(\boldsymbol{x} \,|\, y_j) = \frac{1}{|\mathcal{X}_j|} \sum_{\boldsymbol{x}_i \in \mathcal{X}_j} p(\boldsymbol{x}; \boldsymbol{x}_i, \mathbf{S}_j), \tag{2.18}$$

normalized by the number of elements in the class y_j.

The Gaussian Mixture Model (GMM) classifier $q : \mathcal{X} \mapsto \mathcal{Y}$ is defined as

$$q(\boldsymbol{x}) = \operatorname*{argmax}_{y \in \mathcal{Y}} p(\boldsymbol{x} \,|\, y). \tag{2.19}$$

The classifier (2.19) will be evaluated using only the three-dimensional feature space of $[\lambda_x,\, \lambda_y,\, \sigma_z^2]$.

2.2.3 Random Forest

A classification using a binary decision tree is a predictive model based on a sequential decision making. One feature criterion is selected as a root node. Every node (except of the last level) splits into two branches according to the feature criterion chosen. Nodes in the last level of the decision tree are called leaves and contain an output class decision [Bis06]. For illustration, a simple binary tree is shown in Figure 2.5. However, individual decision trees tend to overfit.

We follow [Bre01] in Definition 2.10.

Definition 2.10. Let $\mathcal{T}_{XY} = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_l, y_l)\}$ be a set of prototype vectors $\boldsymbol{x}_i \in \mathcal{X} \subseteq \mathbb{R}^n$ and the corresponding classes be $y_i \in \mathcal{Y} = \{1, \ldots, c\}$. Let $\boldsymbol{x} \in \mathbb{R}^n$ be an incoming vector[9] with an unknown class label.

[9] Note that the space of $\mathbb{R}^n$ implies $n \in \mathbb{N}$ features.

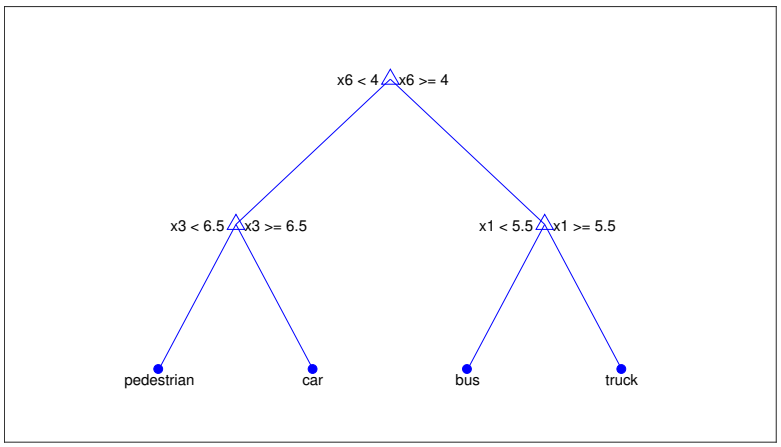


Figure 2.5: One simple decision tree.

Let $\Theta_1, \ldots, \Theta_N$ are $N \in \mathbb{N}$ sets of the same size as the original set $\mathcal{T}_{XY}$ created from a random picking of the prototype vectors from the $\mathcal{T}_{XY}$ with replacement.

The Θ_i serves as a training set for growing the tree $h_i(\Theta_i, s)$, where $s \in \mathbb{N}$, $s < n$ is a number of features (selected at random out of n features) to find the best split at each node in the sense of the Gini index [Bis06]. Each tree is grown to the largest extent possible without pruning and the value of s is the same for each tree.

A Random Forest is an ensemble of N classifiers $\{h_1(\Theta_1, s), \ldots, h_N(\Theta_N, s)\}$.

The incoming vector $\boldsymbol{x}$ is passed down from a root node to a leaf node of each classification tree. Since there are N trees in the ensemble, we get a N-dimensional vector of predicted classes $\mathcal{K} = \{k_1, \ldots, k_N\}$.

Assuming uniform weights of all classes and ensemble trees as well, we define the posterior probability of class y as

$$p(y \mid \boldsymbol{x}) = \frac{\sum_{i=1}^{N} u_i(\boldsymbol{x}, y)}{N}, \tag{2.20}$$

where $u_i(\boldsymbol{x}, y)$ is 1 when output class of i-th decision tree $k_i = y$ at input $\boldsymbol{x}$ and 0 otherwise.

The Random Forest (RF) classifier $q : \mathcal{X} \mapsto \mathcal{Y}$ is then defined as

$$q(\boldsymbol{x}) = \operatorname*{argmax}_{y \in \mathcal{Y}} p(y \mid \boldsymbol{x}). \tag{2.21}$$

Figure 2.6 shows a qualitative characteristics of an out-of-bag classification error [Bre01] convergence. The out-of-bag error decreases with the number of grown trees almost monotonously. Running a few experiments, we have found out that it always takes no more than 500 trees to converge. Therefore, we will keep it as a fixed value for all experiments. Assume $d^2 \in \mathbb{N}$ is a number of features, then $s = d$ features (selected at random) are used in each split.

A MATLAB implementation `TreeBagger` has been used to perform the Random Forest classification.

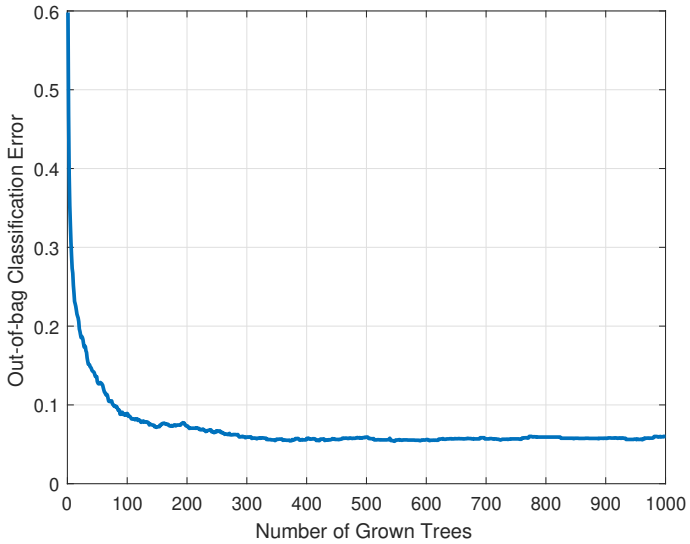


Figure 2.6: Out-of-bag classification error over the number of grown trees.

2.2.4 Assignment of the Outlier Class

The question to be answered is when and how the universal class is assigned to the object.

There is a posterior probability of class defined for each classification method. That allows us to generalize all classifiers with the following rule.

Definition 2.11. Let $\mathcal{T}_{XY} = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_l, y_l)\}$ be a set of prototype vectors $\boldsymbol{x}_i \in \mathcal{X} \subseteq \mathbb{R}^n$ and the corresponding classes be $y_i \in \mathcal{Y}^* = \{1, \ldots, c_{\text{out}}\}$, i.e., a set of specific classes $\mathcal{Y}$ extended by the outlier class c_{out}. Let $\boldsymbol{x} \in \mathbb{R}^n$ be an incoming vector with an unknown class label.

Assume the posterior probability of the class y given $\boldsymbol{x}$ is $p(y \,|\, \boldsymbol{x})$. We compute two most probable classes

$$\widetilde{y}_1 = \operatorname*{argmax}_{y \in \mathcal{Y}} p(y \,|\, \boldsymbol{x}), \tag{2.22}$$

$$\widetilde{y}_2 = \operatorname*{argmax}_{y \in \mathcal{Y} \setminus \{\widetilde{y}_1\}} p(y \,|\, \boldsymbol{x}). \tag{2.23}$$

The generalized classification rule $q : \mathcal{X} \mapsto \mathcal{Y}$ is then defined as

$$q(\boldsymbol{x}) = \begin{cases} c_{out} & \frac{p(\widetilde{y}_2 \,|\, \boldsymbol{x})}{p(\widetilde{y}_1 \,|\, \boldsymbol{x})} > T, \\ \widetilde{y}_1 & \text{otherwise}, \end{cases} \tag{2.24}$$

where $T \in \mathbb{R}$ is a probability threshold. Smaller T is more restrictive.

The probability threshold T will be further investigated in Chapter 4.

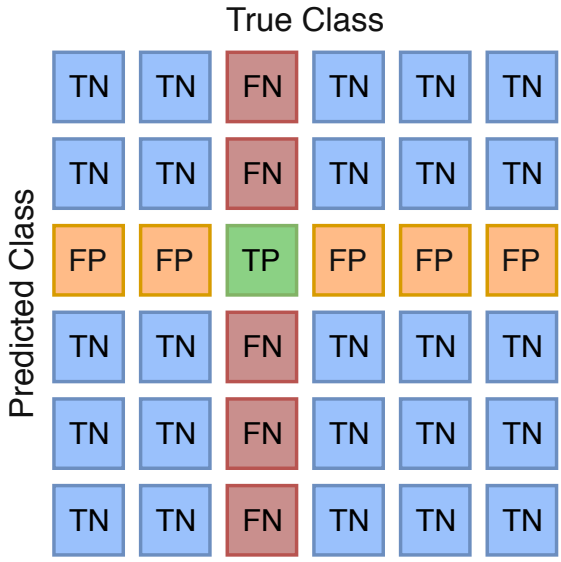


Figure 2.7: Confusion matrix sections.

2.3 Metric Models of Classification Performance

A conventional confusion matrix will be used for the multi-class classification performance measurement. Let's consider a confusion matrix in which the predicted classes are in rows and the true classes are in the columns. In the confusion matrix the outlier class is treated the same way as the specific classes.

Given the confusion matrix we generalize a commonly used two-class metric model of true positive, true negative, false positive and false negative for multi-class classification.

Without loss of generality consider one particular class c. A true positive (TP_c) is simply the number of correctly classified segments from class c, a false positive (FP_c) is the number of segments from a non-c class that were labeled as a class c, a false negative (FN_c) is the number of segments from a class c that were not labeled as a class c and a true negative (TN_c) is the number of segments which do not belong to any previous group mentioned above (i.e., the segments out of class c which were correctly not classified as a class c).

Confusion matrix partitions for one particular class are clearly shown in Figure 2.7, which is motivated by [Krü16].

Further we define mean precision (PPV), mean recall (TPR), mean F_1 score and mean accuracy (ACC) in Equation (2.26), (2.27), (2.28) and (2.29), respectively. Assume there are $\mathcal{C}$ classes in the confusion matrix. All the parameters are weighted by the number of instances I_i in the corresponding class i as defined in Equation (2.25).

$$w_i = \frac{I_i}{\sum_{j=1}^{\mathcal{C}} I_j}, \tag{2.25}$$

$$\mathrm{PPV} = \sum_{i=1}^{\mathcal{C}} w_i \frac{\mathrm{TP_i}}{\mathrm{TP_i} + \mathrm{FP_i}}, \tag{2.26}$$

$$\mathrm{TPR} = \sum_{i=1}^{\mathcal{C}} w_i \frac{\mathrm{TP_i}}{\mathrm{TP_i} + \mathrm{FN_i}}, \tag{2.27}$$

$$\mathrm{F_1} = 2 \sum_{i=1}^{\mathcal{C}} w_i \frac{\mathrm{PPV_i} \cdot \mathrm{TPR_i}}{\mathrm{PPV_i} + \mathrm{TPR_i}}, \tag{2.28}$$

$$\mathrm{ACC} = \sum_{i=1}^{\mathcal{C}} w_i \frac{\mathrm{TP_i} + \mathrm{TN_i}}{\mathrm{TP_i} + \mathrm{TN_i} + \mathrm{FP_i} + \mathrm{FN_i}}. \tag{2.29}$$

The goal is to maximize precision and recall. Consequently, accuracy and $\mathrm{F_1}$ score as well.

Chapter 3

Datasets

A dataset choice is crucial for an evaluation of developed classifiers.

There are several public datasets available on the internet listed in Section 1.2 but only a Sydney dataset [QUD13] was picked of them all to evaluate our models (see Section 3.2 for detail motivation). The remaining datasets suffered from common drawbacks such as too few or complete lack of instances in a class of interest or acquisition by a static LiDAR sensor that provides different properties than the mobile laser scanner.

We also completed the internal Wolfsburg dataset under the UP-Drive project[10] funded by the European Commision. The Wolfsburg dataset is not yet public.

3.1 Wolfsburg Dataset

3.1.1 Data Origin and Pre-processing

Data were collected in Wolfsburg city, Germany, on November 26, 2018 (4 driving sessions) and on January 14, 2019 (3 driving sessions).

Four LiDAR sensors were mounted on the roof of the egovehicle (Figure 3.1 shows an UP-Drive vehicle with fifth extra LiDAR sensor at the back). That yields four independent point clouds at a time. The corresponding point clouds were segmented and aggregated into one scan by Dominik Fiala [Fia18]. As a result, scans are approximately four times denser (see Table 3.3 for a comparison with the Sydney single scanner).

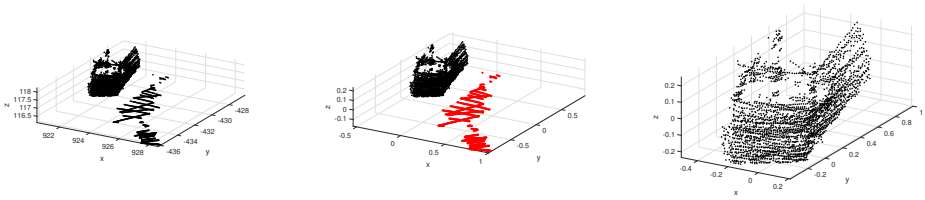
3.1.2 Hand-crafted Annotation

The ground truth annotation of individual segments was achieved by the human workforce. We went through 23458 scans and collected a total of 3203 segments (see Table 3.1 for details).

Sometimes it was hard to decide which class a segment belongs to. Thus, camera images were used to verify the ground truth annotation.

[10] See www.up-drive.eu.

Figure 3.1: UP-Drive vehicle.



(a): Input segment. **(b):** Outlier separation. **(c):** Well-segmented van.

Figure 3.2: Correction of an undersegmented van.

3.1.2.1 Post-processing Phase

Several objects were not segmented properly and contained points of another class or background (e.g., ground or wall). Similarly, a few semantically separate objects (e.g., two pedestrians walking side by side) were labeled as a single segment. All of these occurrences have been splitted into its correct class (resp. erased as shown in Figure 3.2) in post-processing phase.

On the contrary, oversegmented objects have been merged and labeled according to their class as well.

Due to this approach, we have reached a higher number of well-segmented samples in the dataset[11].

3.1.3 Semi-synthetic Outliers

The original segmentation did not provide sufficient amount of undefined objects that could be labeled as an *outlier*. For that reason, additional 1200 segments (selected at random) were extracted from the background.

WOLFSBURG dataset								
			Class					
	acq. date	scans	bus	car	pedestrian	truck	van	Σ
1	2018-11-26 11:04	3008	12	211	88	99	86	496
2	2018-11-26 11:10	3703	47	148	99	79	72	445
3	2018-11-26 11:18	3209	2	94	142	45	23	306
4	2018-11-26 11:25	4257	22	285	141	57	107	612
5	2019-01-14 10:38	3461	40	140	91	58	78	407
6	2019-01-14 11:01	3466	20	153	113	65	129	480
7	2019-01-14 11:29	2354	14	157	133	77	76	457
	Σ	23458	157	1188	807	480	571	3203

Table 3.1: Class instances in Wolfsburg dataset.

3.1.4 Dataset Overview and Analysis

Unfortunately, there are no cyclists nor bikers in the Wolfsburg dataset. Most likely it has occurred due to

- strong influence of the industrial environment,
- winter off-season, (see acquisition dates in Table 3.1),
- unsuitable time (before noon),
- cycling unfriendly routes.

Wolfsburg is a location of Volkswagen AG's headquarters and the biggest factory. Hence, the majority of cars and vans are Volkswagen and the variety of vehicle shapes is limited.

3.2 Sydney Dataset

In order to be more data independent, we want to select at least one public dataset. The Sydney dataset seem to be the most useful option.

Motivation of Dataset Choice. While the traffic in Wolfsburg is quite monotonous, the city of Sydney in Australia provides a wide variety of vehicles because it is not influenced by the connection to the only one automobile manufacturer.

Geodetic distance from Sydney to Wolfsburg[12] is about 16250 km. In this regard, Sydney is the farthest[13] city from Wolfsburg that provides a public dataset [QUD13] of common road traffic participants scanned with the Velodyne LiDAR.

[11] Segmentation-based inaccuracy is not of interest in this thesis. Therefore, all presented hand-edits are justified. Beyond that, it is a common practice [DDQHD13], [SMGD14].
[12] According to Google Maps.
[13] To the author's knowledge.

SYDNEY dataset								
Class	biker	bus	car	cyclist	pedestrian	truck	van	Σ
Instances	4	16	125	3	152	12	35	347

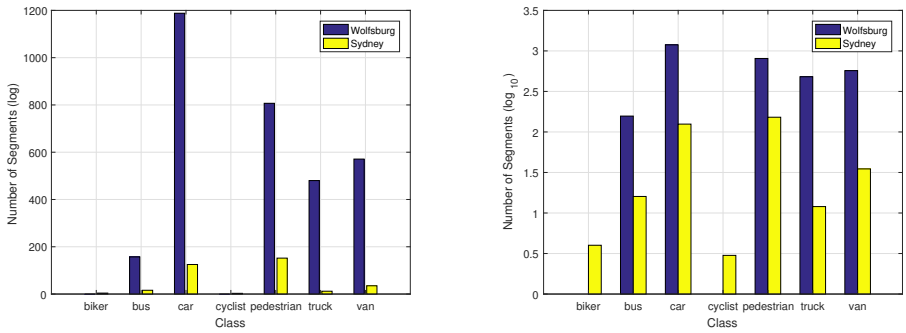
Table 3.2: Class instances in Sydney dataset.

Dataset Description and Class Choice. Sydney dataset was generated from several sequences of Velodyne scans. As in our case, inaccurate segmentations[14] had been manually corrected.

The Sydney dataset contains a total number of 631 objects divided into 26 classes, but only 9 classes of interest. Besides those 9 classes, we have selected 5 more classes (used in [DDQHD13]) as outliers. In addition, we combine small two-track vehicles (4WD, car and ute) into one class *car*. See Table 3.2 for final distribution.

3.3 Datasets Comparison

In Figure 3.3 there is a comparison of the number of instances in both datasets. Table 3.3 shows a point cloud density comparison. The segments in Wolfsburg dataset are about two to four times denser due to the aggregation of four LiDAR scans.



(a): Linear scale. **(b):** Logarithmic scale (decadic).

Figure 3.3: Comparison of the number of segments in both datasets.

WOLFSBURG dataset							
Class	biker	bus	car	cyclist	pedestrian	truck	van
Points	~	7999	2088	~	264	6925	3191
SYDNEY dataset							
Class	biker	bus	car	cyclist	pedestrian	truck	van
Points	311	1847	582	124	111	2489	1152

Table 3.3: Average number of points per segment.

[14] The segmentation techniques used are available in [DUK+11].

WOLFSBURG dataset							
	bus	car	pedestrian	truck	van	outlier	Σ
Fold A	22	285	141	57	107	315	927
Fold B	52	351	179	157	164	315	1218
Fold C	61	305	232	156	148	315	1217
Fold D	22	247	255	110	152	315	1101

(a) : The number of instances in folds.

WOLFSBURG dataset							
	bus	car	pedestrian	truck	van	outlier	$\overline{\mathrm{w}}$
Fold A	0.14	0.24	0.17	0.12	0.19	0.25	0.22
Fold B	0.33	0.30	0.22	0.33	0.29	0.25	0.28
Fold C	0.39	0.26	0.29	0.33	0.26	0.25	0.28
Fold D	0.14	0.21	0.32	0.23	0.27	0.25	0.25

(b) : The instances ratio in folds.

Table 3.4: The four fold division of Wolfsburg Dataset.

3.4 The k-Fold Cross-Validation

To ensure valid results, the k-fold cross-validation [Bis06] with the $k = 4$ for the purpose of the classification model training and testing has been used.

The k-fold cross-validation yields k confusion matrices. We will add them together and get a single confusion matrix that will be used for the evaluation.

Note that outliers will be used solely for validation purposes and will not be part of the training set.

3.4.1 Wolfsburg Dataset

Since the presence of segments from the same track in two distinct folds is flawed, we need to preserve the acquisition date granularity.

We want to divide the dataset into four even parts with the even class representation in each fold. Table 3.4 shows the final four fold division.

3.4.2 Sydney Dataset

In Sydney dataset we follow the pre-designed folding[15].

The only inconvenience is too few cyclists in the dataset to be divided into four batches (see Table 3.2). In this case, which is not mentioned in the four fold recommendation, we have decided to omit cyclists in the last fold (i.e., no samples in class *cyclist*). We consider this the best solution that will not harm the evaluation.

[15] The $k = 4$ as well.

SYDNEY dataset									
	biker	bus	car	cyclist	pedestrian	truck	van	outlier	Σ
Fold A	1	5	32	1	37	3	9	58	146
Fold B	1	3	30	1	36	3	11	71	156
Fold C	1	3	29	1	34	3	8	54	133
Fold D	1	5	34	0	45	3	7	58	153

(a) : The number of instances in folds.

SYDNEY dataset									
	biker	bus	car	cyclist	pedestrian	truck	van	outlier	$\overline{w}$
Fold A	0.25	0.31	0.26	0.33	0.24	0.25	0.26	0.24	0.25
Fold B	0.25	0.19	0.24	0.33	0.24	0.25	0.31	0.30	0.27
Fold C	0.25	0.19	0.23	0.33	0.22	0.25	0.23	0.22	0.23
Fold D	0.25	0.31	0.27	0.00	0.30	0.25	0.20	0.24	0.27

(b) : The instances ratio in folds.

Table 3.5: The four fold division of Sydney Dataset.

Chapter 4

Experimental Results

In this chapter, we will test the classifiers described in Section 2.2 on both datasets from Chapter 3.

The entire framework was implemented using MATLAB R2016b with dependencies on a Statistics and Machine Learning Toolbox and a Wavelet Toolbox.

Let's consider a set of the specific classes (hereinafter referred to as SC) and a set of the specific classes extended by the *outlier* class (hereinafter referred to as EC). We provide two ways of evaluation. The first one works solely with SC and the second one uses whole EC, which is used for the posterior probability thresholding.

We will be using three combinations of feature families declared in Section 2.1. Let us use a special symbol $\mathcal{F}_i$ to save the space in tables and graphs

$$\mathcal{F}_1 = [\lambda_x,\, \lambda_y,\, \sigma_z^2], \tag{4.1}$$

$$\mathcal{F}_2(\alpha) = [\mathcal{F}_1,\, M_0,\, \varkappa,\, \mathcal{H}_\alpha]. \tag{4.2}$$

WOLFSBURG dataset				
Classification Features	Mean Accuracy (%)	Mean Precision (%)	Mean Recall (%)	Mean F_1 score (%)
$\mathcal{F}_1$	94.6	89.7	88.1	87.5
SYDNEY dataset				
Classification Features	Mean Accuracy (%)	Mean Precision (%)	Mean Recall (%)	Mean F_1 score (%)
$\mathcal{F}_1$	95.8	87.9	87.3	87.5

Table 4.1: The GMM method results on the SC.

WOLFSBURG dataset				
Classification Features	Mean Accuracy (%)	Mean Precision (%)	Mean Recall (%)	Mean F_1 score (%)
$\mathcal{F}_1$	93.2	87.7	85.8	86.2
$\boldsymbol{\mathcal{F}_2(\alpha = 0)}$	**96.0**	**91.8**	**91.4**	**91.5**
$\mathcal{F}_2(\alpha = 1)$	94.7	88.7	88.1	88.1
$\mathcal{F}_2(\alpha = 2)$	93.4	86.1	85.3	85.3
$\mathcal{F}_2(\alpha = 3)$	92.9	85.3	84.2	84.2
SYDNEY dataset				
Classification Features	Mean Accuracy (%)	Mean Precision (%)	Mean Recall (%)	Mean F_1 score (%)
$\mathcal{F}_1$	92.3	84.1	77.8	80.1
$\boldsymbol{\mathcal{F}_2(\alpha = 0)}$	**95.1**	**89.1**	**85.3**	**86.9**
$\mathcal{F}_2(\alpha = 1)$	93.9	83.8	81.8	82.7
$\mathcal{F}_2(\alpha = 2)$	92.2	81.7	78.4	79.7
$\mathcal{F}_2(\alpha = 3)$	90.9	79.7	75.8	77.2

Table 4.2: The 7-NN method results on the SC.

WOLFSBURG dataset				
Classification Features	Mean Accuracy (%)	Mean Precision (%)	Mean Recall (%)	Mean F_1 score (%)
$\mathcal{F}_1$	95.0	89.5	89.3	89.4
$\mathcal{F}_2(\alpha = 0)$	96.3	92.2	92.1	92.2
$\mathcal{F}_2(\alpha = 1)$	96.3	92.3	92.1	92.1
$\boldsymbol{\mathcal{F}_2(\alpha = 2)}$	**96.4**	**92.5**	**92.4**	**92.4**
$\mathcal{F}_2(\alpha = 3)$	96.4	92.1	92.0	92.1
SYDNEY dataset				
Classification Features	Mean Accuracy (%)	Mean Precision (%)	Mean Recall (%)	Mean F_1 score (%)
$\mathcal{F}_1$	94.8	85.8	83.6	84.4
$\boldsymbol{\mathcal{F}_2(\alpha = 0)}$	**96.0**	**88.8**	**87.3**	**87.6**
$\mathcal{F}_2(\alpha = 1)$	95.4	86.3	85.9	86.0
$\mathcal{F}_2(\alpha = 2)$	95.6	85.5	87.0	86.1
$\mathcal{F}_2(\alpha = 3)$	94.1	82.7	84.4	83.3

Table 4.3: RF method results on the SC (mean values with approximate deviation of ± 1 % are shown).

The probability threshold on the x-axis of the following graphs corresponds to $1 - T$, where T is defined in Section 2.2.4.

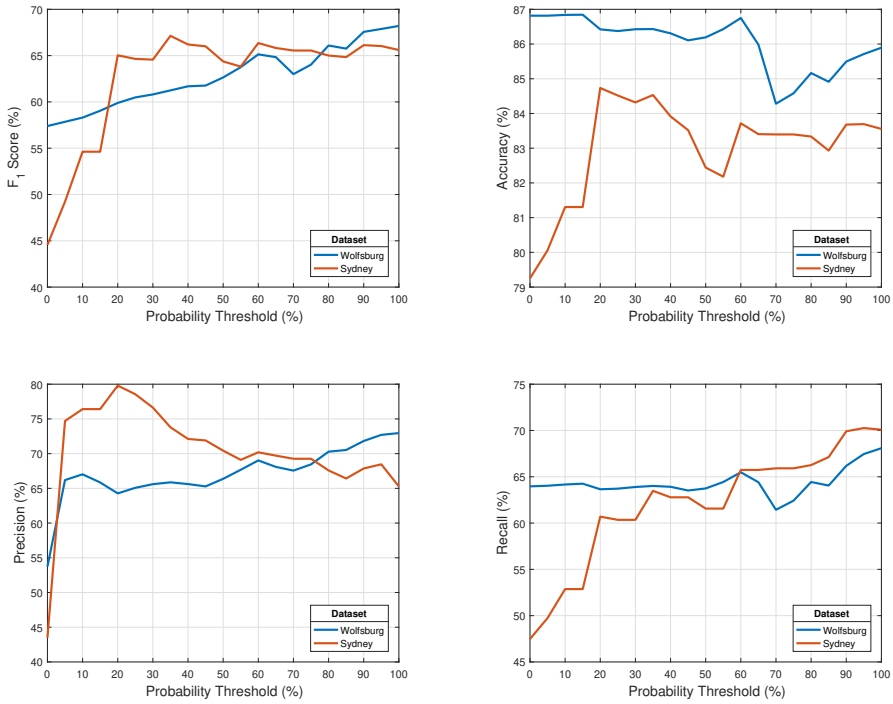


Figure 4.1: The 7-NN method with $\mathcal{F}_1$ features on the EC.

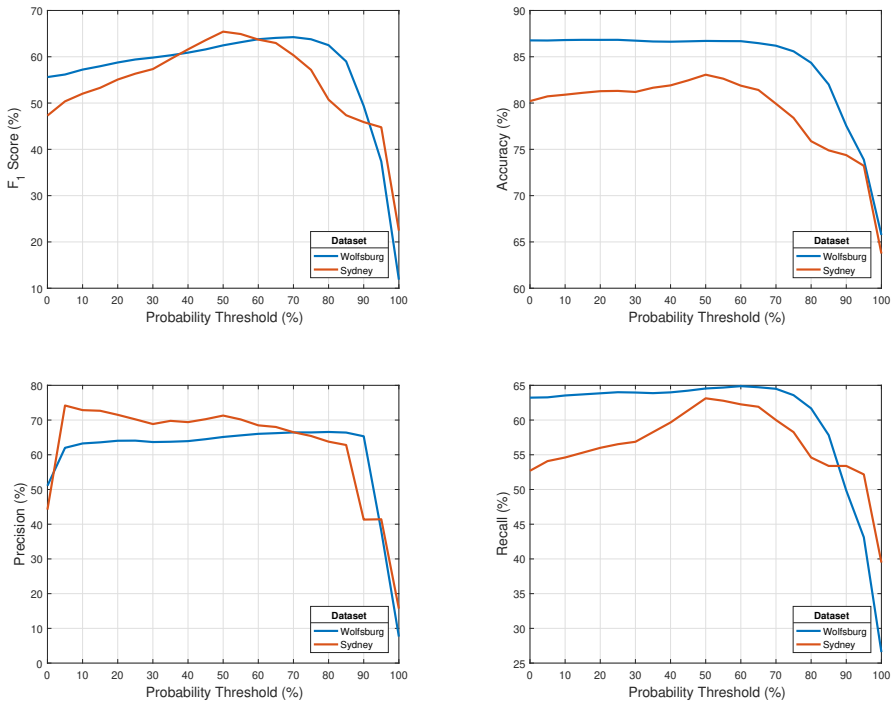
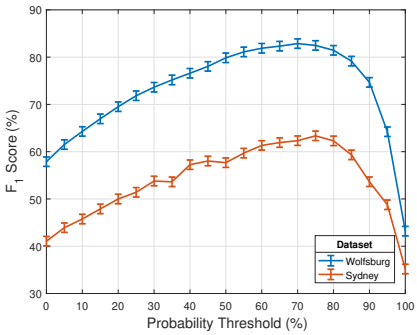
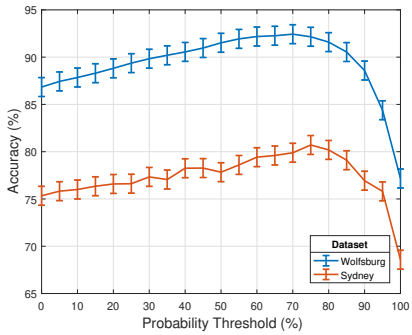


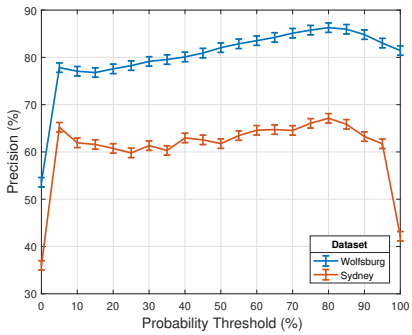
Figure 4.2: The GMM method with $\mathcal{F}_1$ features on the EC.



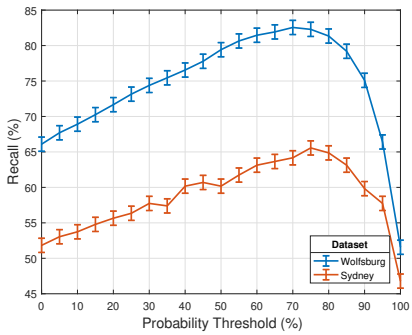
(a): F_1 score graph.



(b): Accuracy graph.



(c): Precision graph.



(d): Recall graph.

Figure 4.3: The RF method with $\mathcal{F}_2(\alpha = 1)$ features on the EC.

4.1 Best Results

This section is dedicated to the best classifier on the SC in this thesis so far.

The best parameters (weighted average of F_1 score, accuracy, precision and recall across all classes) are generated by Random Forest with $\mathcal{F}_2(\alpha = 2)$ for the Wolfsburg dataset (confusion matrix in Table 4.4) and $\mathcal{F}_2(\alpha = 0)$ for the Sydney dataset (confusion matrix in Table 4.5). Both confusion matrices are generated with a random seed.

WOLFSBURG dataset						
		True				
		pedestrian	car	van	bus	truck
Predicted	pedestrian	809	0	0	0	0
	car	0	1089	81	0	0
	van	0	99	479	0	17
	bus	0	0	2	124	14
	truck	0	0	9	33	449

Table 4.4: Confusion matrix for Wolfsburg dataset.

SYDNEY dataset							
		True					
		pedestrian	cyclist	biker	car	van	bus	truck
Predicted	pedestrian	147	1	0	0	0	0	0
	cyclist	5	2	0	0	0	0	0
	biker	0	0	4	1	1	0	0
	car	0	0	0	110	5	0	0
	van	0	0	0	13	26	2	4
	bus	0	0	0	1	2	12	6
	truck	0	0	0	0	1	2	2

Table 4.5: Confusion matrix for Sydney dataset.

4.1.1 Misclassification Insights

(a): Car labeled as a van.

(b): Van labeled as a car.

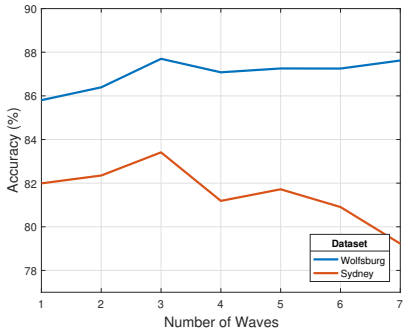
(c): Bus labeled as a truck.

(d): Truck labeled as a bus.

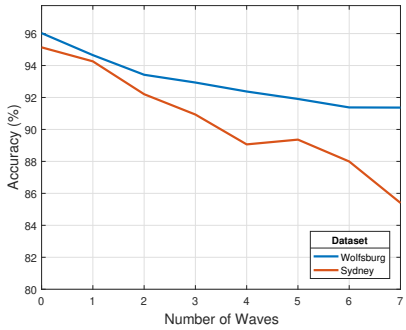
Figure 4.4: Four examples of misclassified segments.

4.2 Influence of 3D Haar-Like Features

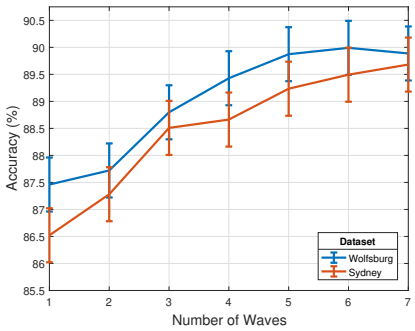
The number of waves on the x-axis of the following graphs represents the α.



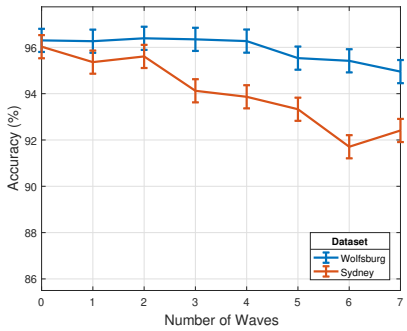
(a): k-NN trend of $\mathcal{H}_\alpha$.



(b): k-NN trend of $\mathcal{F}_2(\alpha)$.



(c): RF trend of $\mathcal{H}_\alpha$.



(d): RF trend of $\mathcal{F}_2(\alpha)$.

Figure 4.5: Comparison of 3D Haar-like features influence.

Chapter 5

Discussion and Conclusions

5.1 Discussion

The most successful classification models are reviewed in Table 5.1.

Misclassified Segments Observations. Section 4.1 presents two confusion matrices 4.4, 4.5 of the classification performance. Besides this overall visualization, Section 4.1.1 shows several cases where the model failed (see Figure 4.4). In this paragraph we will try to identify common factors in data that were classified incorrectly.

Most often we have confused the car with the van and vice versa. Figure 4.4(a) and 4.4(b) shows the similarity between Volkswagen Touran (the class *car*) and Volkswagen Caddy (the class *van*) which caused a lot of classification faults.

Typical attributes of the bus are large windows on both sides. In Figure 4.4(c) there is a LiDAR segment of a bus with advertising stickers in the windows which may look like a truck.

Very sparse point cloud segments of trucks (see Figure 4.4(d)) were confused with buses or vans.

Classification of pedestrians was almost flawless. The only confusion was with the *cyclist* class. Both false positives and false negatives have probably a simple explanation. Two of the three cyclists in the Sydney dataset led the bike side by side. Thus, the segment represents a pedestrian and a bike rather than a cyclist.

Influence of 3D Haar-Like Features. In Section 4.2 we have presented an effect of the size of the 3D Haar-like feature family to classification performance (see Figure 4.5).

While the $\mathcal{H}_\alpha$ on its own for small α increases the accuracy, the presence of more 3D Haar-like features in the $\mathcal{F}_2(\alpha)$ did not improve the performance.

The computation of 3D Haar-like features is highly dependent on the position of the segment in the unit cube. Therefore, one of the main reasons for the 3D Haar-like features failure could be just the imperfect segment preprocessing that can be enhanced in the future work.

Dataset	Classification Method	Classification Features	Probability Threshold (%)	Mean Accuracy (%)	Mean Precision (%)	Mean Recall (%)	Mean F_1 score (%)
Wolfsburg - SC	RF	$\mathcal{F}_2(\alpha = 2)$	~	96.4	92.5	92.4	92.4
Sydney - SC	RF	$\mathcal{F}_2(\alpha = 0)$	~	96.0	88.8	87.3	87.6
Wolfsburg - EC	RF	$\mathcal{F}_2(\alpha = 1)$	70	92.4	85.1	82.6	82.9
Sydney - EC	k-NN	$\mathcal{F}_1$	35	84.5	73.8	63.5	67.1

Table 5.1: Review of the best classification models (mean values with approximate deviation of ±1 % are shown for RF).

On the other hand, objects in classes are perhaps too diverse (e.g., front-view, side-view, rear-view, distance based shape, shading with another object, etc.) to be separable by simply adding and subtracting the number of points in fixed voxel grid.

Comparison with Previous Studies. Comparison with other studies is always difficult when the conditions are uneven (e.g., different datasets, different object classes, etc.). In case of undefined objects, it is usually unclear whether they were used for training or not. In addition, different papers provide different performance parameters. The F_1 score ranges from 47.2 % to 70.0 %, accuracy from 77.6 % to 98.5 %, precision from 73.1 % to 92.8 % and recall from 44.3 % to 81.4 %.

Although [GKF09] does not present *outlier* class in results it was considered during classification together with 16 specific classes. Our EC set[16] is therefore suitable for comparison. The highest precision is 78 % and the highest recall is 65 %. Both parameters lie in the lower half of our Wolfsburg-Sydney interval but there is not the confusion matrix to do the proper class reduction.

In [TLT11b] they reached excellent accuracy of 98.5 %. However, for both training and testing dataset, the segments were prefiltered manually. From the huge dataset of 1.3 million segments they selected only well-segmented and well-tracked instances and invalidated all objects with lower frequency than 10 segments in the track or objects with the total number of points lower than 75. Those constraints may overlook common cases in practice and real traffic. They considered classes *pedestrian*, *cyclist*, *car*[17] and *background*. We merged classes *car*, *van* and *truck* and exclude the class *bus* to adjust the classification conditions for the purpose of the comparison. Our resulting accuracy is 87.5 % on EC with merged classes.

Considering the EC set, we have slightly outperformed the [LJH+16] with the mean classification accuracy from 77.6 % to 88.1 % with six specific classes and undefined class.

Much effort was spent to establish similar conditions with the [DDQHD13]. On the same dataset we got exactly the same maximal F_1 score 67.1 %. Due to missing confusion matrix, we are unable to do precise comparison with regard to our class *outlier*.

The best classification model in the PhD thesis [Qua13] did slightly better with maximal F_1 score 70 % on Sydney dataset.

[16] The set of the specific classes extended by the *outlier* class.

[17] Consisting of cars, vans and trucks.

They achieve overall accuracy over 97 % with six classes in [Hac18], but less than 25 % in the case of the *pedestrian* class where we have reached nearly the upper limit of 100 % on both datasets. It should be noted that their dataset contained only 14 pedestrians from 642 objects and that the pedestrian classification was not the main objective of that study.

5.2 Conclusions

In this thesis we described and experimentally evaluated several potential classification models for LiDAR point cloud segments of road traffic participants. All presented classification models rely on a priori segmentation and hand-annotation.

We have achieved significantly better performance with the Wolfsburg dataset created by aggregation of four independent LiDAR scanners.

Our recognition system is able to classify point cloud segments with overall accuracy up to 96 % depending on the dataset choice and used classes.

Well, is it enough for real traffic? Nothing below 100 % is not enough when human life is at stake but at least the results achieved are comparable with current state-of-the-art methods.

5.3 Future Work

The direction of the future work that would be most beneficial is the development of an automated pre-processing of the point cloud segments so that no hand-edits are needed. The pre-processing method should remove background, other objects and noise with adjustable sensitivity[18]. On the other hand, it is not necessary to split undersegmented objects. For example, one pedestrian behaves differently from a pair and a crowd acts as a whole. In the case of fast moving objects, it is important to distinguish between the segment with the cyclist near the car and the segment only with the car. To do this, it is sufficient to extend the classes by typical combinations of objects occurring in the segment. The recognition of those cases leads to higher road safety.

Further experimental studies are needed to estimate whether the neural networks can bring significant improvements in accuracy.

If the object is seen for a sufficiently long time, the core idea of holistic track classification in [TLT11b] seems to be very powerful and it could further improve existing methods for numerous tracks even without unreasonable constraints.

[18] If there is noise remaining in the segment, it is no longer suitable to define the scaling factor $\varkappa$ as the distance of the farthest point from the origin (while computing the 3D Haar-like features 2.1.2.1).

Bibliography

[AG97] Y. Amit and D. Geman, *Shape quantization and recognition with randomized trees*, Neural Computation **9** (1997), no. 7, 1545–1588.

[Bis06] Christopher M. Bishop, *Pattern recognition and machine learning*, 1 ed., Springer-Verlag, New York, Feb 2006.

[Bre01] L. Breiman, *Random forests*, Machine Learning **45** (2001), no. 1, 5–32.

[DDQHD13] M. De Deuge, A. Quadros, C. Hung, and B. Douillard, *Unsupervised feature learning for classification of outdoor 3D scans*, Australasian Conference on Robitics and Automation, vol. 2, 2013.

[DHS01] Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern classification*, 2 ed., Wiley, New York, 2001.

[DUK+11] B. Douillard, J. Underwood, N. Kuntz, V. Vlaskine, A. Quadros, P. Morton, and A. Frenkel, *On the segmentation of 3D lidar point clouds*, 2011 IEEE International Conference on Robotics and Automation, May 2011, pp. 2798–2805.

[EDP06] A. El Daher and S. Park, *Object recognition and classification from 3D point clouds*, Student Project (2006).

[FH04] V. Franc and V. Hlavac, *Statistical pattern recognition toolbox for matlab*, Center for Machine Perception, Czech Technical University in Prague, 1 ed., Jun 2004.

[Fia18] D. Fiala, *Change detection in vehicle neighbourhood using 3D data and aggregated 3D map*, Master's thesis, Czech Technical University in Prague, Department of Cybernetics, Faculty of Electrical Engineering, Czech Technical University in Prague, 6 2018.

[GKF09] A. Golovinskiy, V. G. Kim, and T. Funkhouser, *Shape-based recognition of 3D point clouds in urban environments*, 2009

IEEE 12th International Conference on Computer Vision, Sep. 2009, pp. 2154–2161.

[Hac18] T. Hackel, *Large-scale machine learning for point cloud processing*, Ph.D. thesis, ETH Zurich, https://doi.org/10.3929/ethz-b-000264691, 5 2018.

[HSL+17] T. Hackel, N Savinov, L. Ladicky, J. D. Wegner, K. Schindler, and M. Pollefeys, *Semantic3d.net: A new large-scale point cloud classification benchmark*, 2017.

[Krü16] F. Krüger, *Activity, context, and plan recognition with computational causal behaviour models*, Ph.D. thesis, University, 2016.

[LJH+16] M. Lehtomäki, A. Jaakkola, J. Hyyppä, J. Lampinen, H. Kaartinen, A. Kukko, E. Puttonen, and H. Hyyppä, *Object classification and recognition from mobile laser scanning point clouds in a road environment*, IEEE Transactions on Geoscience and Remote Sensing **54** (2016), no. 2, 1226–1239.

[MBVH09] D. Munoz, J. A. Bagnell, N. Vandapel, and M. Hebert, *Contextual classification with functional max-margin markov networks*, 2009 IEEE Conference on Computer Vision and Pattern Recognition, June 2009, pp. 975–982.

[MS15] D. Maturana and S. Scherer, *Voxnet: A 3D convolutional neural network for real-time object recognition*, 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Sep. 2015, pp. 922–928.

[PN13] G. Pang and U. Neumann, *Training-based object recognition in cluttered 3d point clouds*, 2013 International Conference on 3D Vision - 3DV 2013, June 2013, pp. 87–94.

[Pro10] D. Prokhorov, *A convolutional learning system for object classification in 3-D lidar data*, IEEE Transactions on Neural Networks **21** (2010), no. 5, 858–863.

[Qua13] A. J. Quadros, *Representing 3D shape in sparse range images for urban object classification*, Ph.D. thesis, University of Sydney, Oct 2013.

[QUD13] A. Quadros, J. Underwood, and B. Douillard, *Sydney urban objects dataset*, http://www.acfr.usyd.edu.au/papers/SydneyUrbanObjectsDataset.shtml, 2013.

[RB98] M. R. Raghuveer and A. S. Bopardikar, *Wavelet transforms: Introduction to theory and applications*, 1998.

[SMGD14] A. Serna, B. Marcotegui, F. Goulette, and J.-E. Deschaud, *Paris-rue-Madame Database: a 3D mobile Laser Scanner Dataset for Benchmarking Urban Detection, Segmentation and Classification Methods*, 4th International Conference on Pattern Recognition, Applications and Methods ICPRAM 2014 (Angers, France), March 2014.

[TLT11a] A. Teichman, J. Levinson, and S. Thrun, *The stanford track collection*, https://cs.stanford.edu/people/teichman/stc/, 2011.

[TLT11b] A. Teichman, J. Levinson, and S. Thrun, *Towards 3D object recognition via classification of arbitrary object tracks*, 2011 IEEE International Conference on Robotics and Automation, May 2011, pp. 4034–4041.

[VJ+01] P. Viola, M. Jones, et al., *Rapid object detection using a boosted cascade of simple features*, CVPR (1) **1** (2001), 511–518.

[WSL+18] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, *Dynamic graph CNN for learning on point clouds*, 2018.

Appendix A

Contents of attached CD

Path	Description
/ctu_thesis.pdf	This thesis as a PDF file.
/MATLAB/	All MATLAB source codes.

Table A.1: Contents of attached CD.