Lappeenranta University of Technology
School of Industrial Engineering and Management
Degree Program in Computer Science

Master's Thesis

Petri Hienonen

AUTOMATIC TRAFFIC SIGN INVENTORY- AND CONDITION ANALYSIS

Examiners: Professor Heikki Kälviäinen
M.Sc. (Eng.), Master of Military Science Markus Melander

Supervisor: Professor Lasse Lensu

ABSTRACT

Lappeenranta University of Technology
School of Industrial Engineering and Management
Degree Program in Computer Science

Petri Hienonen

Automatic traffic sign inventory- and condition analysis

Master's Thesis

2014

91 pages, 29 figures, 12 tables, 10 algorithms.

Examiners: Professor Heikki Kälviäinen
M.Sc. (Eng.), Master of Military Science Markus Melander

Keywords: machine vision, pattern recognition, traffic sign, object detection, object classification, tracking, road maintenance, visual inspection

This thesis researches automatic traffic sign inventory and condition analysis using machine vision and pattern recognition methods. Automatic traffic sign inventory and condition analysis can be used to more efficient road maintenance, improving the maintenance processes, and to enable intelligent driving systems. Automatic traffic sign detection and classification has been researched before from the viewpoint of self-driving vehicles, driver assistance systems, and the use of signs in mapping services. Machine vision based inventory of traffic signs consists of detection, classification, localization, and condition analysis of traffic signs. The produced machine vision system performance is estimated with three datasets, from which two of have been been collected for this thesis. Based on the experiments almost all traffic signs can be detected, classified, and located and their condition analysed. In future, the inventory system performance has to be verified in challenging conditions and the system has to be pilot tested.

TIIVISTELMÄ

Lappeenrannan teknillinen yliopisto
Tuotantotalouden tiedekunta
Tietotekniikan koulutusohjelma

Petri Hienonen

Automaattinen liikennemerkkien inventointi ja kunnon arviointi

Diplomityö

2014

91 sivua, 29 kuvaa, 12 taulukkoa, 10 algoritmiä.

Tarkastajat: Professori Heikki Kälviäinen
Diplomi-insinööri, sotatieteiden maisteri Markus Melander

Hakusanat: konenäkö, hahmontunnistus, liikennemerkki, kohteen tunnistaminen, kohteen luokittelu, seuranta, liikenneväylien kunnossapito, visuaalinen tarkastus

Tämä diplomityö tutkii liikennemerkkien automaattista inventointia sekä kunnon arviointia käyttäen konenäkö- ja hahmontunnistusmenetelmiä. Automaattista liikennemerkkien inventointia ja kunnon arvioimista voidaan soveltaa tehokkaampaan liikenneväylien kunnossapitoon, kunnossapitoprosessien kehittämiseen ja älyliikenteen tarpeiden täyttämiseen. Automaattista liikennemerkkien havaitsemista ja luokittelua on tutkittu aiemmin itseajavien autojen, kuljettajan apujärjestelmien ja karttatietopalveluiden tarpeiden näkökulmasta. Konenäköön perustuva liikennemerkkien inventointi koostuu liikennemerkin havaitsemisesta, luokittelusta, paikantamisesta sekä kuntoarviosta. Toteutetun järjestelmän toimivuus arvioidaan käyttäen kolmea eri testiaineistoa, joista kaksi on kerätty tätä työtä varten. Tulosten perusteella lähes kaikki liikennemerkit voidaan havaita, tunnistaa, paikallistaa ja niiden kunto arvioida. Tulevaisuudessa inventoinnin toimivuus tulee varmistaa haastavissa olosuhteissa ja järjestelmälle toteuttaa pilottitestaus.

PREFACE

I wish to thank my supervisor Professor Lasse Lensu and examiners Professor Heikki Kälviäinen and Markus Melander. I also wish to thank the Finnish Transportation Agency for funding this project.

Lappeenranta, 19 Sept, 2014

Petri Hienonen

CONTENTS

ABBREVIATIONS

ACF	Aggregated Channel Features.
AUC	Area Under Curve.
BB	Bounding Box.
CCD	Charge-Coupled Device.
CIE LUV	Lightness and chromaticity coordinates U and V.
FPPI	False Positives Per Image.
FPS	Frames Per Second.
FTA	Finnish Transport Agency.
gPB	Berkley Boundary Detector.
GPS	Global Postitioning System.
GPX	GPS Exchange Format.
GT	Ground Truth.
HOG	Histogram of Oriented Gradients.
HSV	Hue, Saturation, and Value.
ICF	Integrated Channel Features.
JPDAF	Joint Probabilistic Data Association Filter.
KNN	K-Nearest Neighbours.
L1	L1 norm, corresponding to absolute distance.
L2	L2 norm, corresponding to euclidean distance.
LDA	Linear Discriminant Analysis.
LED	Light Emiting Diode.
MAP	Maximum A Posteriori Estimation.
MCMC	Markov Chain Monte Carlo.
NN	Neural Networks.
PCA	Principal Component Analysis.
RGB	Red, Green, Blue.
ROC	Receiver Operating Characteristic.
SIFT	Scale-Invariant Feature Transform.
SSE2	Streaming Single instruction stream multiple data Streams Extensions 2.
SVM	Support Vector Machine.
TSC	Traffic Sign Classification.
TSD	Traffic Sign Detection.
TSI	Traffic Sign Inventory.
TSR	Traffic Sign Recognition.

LIST OF SYMBOLS

BB_s	Traffic sign bounding box.
C	Color Channel.
C_{pos}	Position of the camera.
$\rho_c(\lambda)$	Camera sensitivity function with respect to wavelenght λ and point x.
d	Distance measured in pixels on image between two points.
$E(\lambda, x)$	Illumination spectrum distribution at wavelenght λ and point x.
f	Focal length.
x	Feature.
$y_1 ... y_N$	Class labels.
$\vec{x}$	Vector of features.
$\vec{x}_{max}$	Maximum value of a feature vector.
$\vec{x}_{min}$	Minimum value of a feature vector.
$\vec{x}_{new}$	Previously unkown feature vector.
$\vec{x}_1, ..., \vec{x}_N$	Set of feature vectors.
$\hat{\vec{x}}$	Transformed feature vector.
I	Image.
Ω	Channel image.
I_{hsv}	HSV Image.
I_{rgb}	RGB Image.
I_s	Image at scale s.
I_{seed}	Black and white seed image.
I_{sign}	Cropped traffic sign image.
k	Symbol denoting chosen integer.
$L(x)$	Color of the light source in image pixel x.
M	Classification model.
$R_{3\times3}$	Rotation matrix.
$R(I, s)$	Resample function R image I at scale s.
$S(\lambda, x)$	Surface reflectance function.
s	Scale of an image.
S_{cond}	Traffic sign condition.
S_r	Traffic sign after resizing.
S_{sign}	Size of traffic sign.
$t_{3\times1}$	Translation factor.
Z	A distance that is previously known.

1 INTRODUCTION

Section 1 introduces the background, motivation, objectives-, and restrictions and summarizes the content of the rest of the thesis.

1.1 Background

In Finland the traffic signs are mandated by law to be catalogued manually every five to seven years [1], including also traffic sign condition information. The relatively long timespan causes problems to intelligent driving systems and maintenance because the information can be outdated, inaccurate, and there is no guarantee of its validity. In road maintenance there is also a new trend of being paid not from contracts but from results. This gives an incentive to shorten the inspection period and increase the reaction time to changes in roads. Machine vision offers solutions to automate the inventory and condition analysis of the signs. After the inventory and condition analysis, the information is stored in the database to be used in road maintenance. The idea behind this thesis is to automate this process.

Finland's road sign inventory information is managed by Finnish Transport Agency (FTA) [2], "Liikennevirasto" in Finnish. Currently, the inventory is based on a manually managed knot-based model, where the locations are announced as distances from the previous road intersections. The knot-based model makes the information less usable in intelligent driving system scenarios when compared to a Global Postitioning System (GPS) based location system. In some cases, the knot-based model does not fulfil the accuracy demands of modern requirements. If the traffic sign database inventory and maintenance is possible to automate, (for example, during normal road maintenance), it would offer cost savings, increase road security, open possibilities for more efficient information management in intelligent driving systems, improve competitive bidding processes in road maintenance, and ease the transition from a knot based database to a more accurate GPS - based database.

During the road maintenance contracts traffic signs are inventoried and catalogued using class, direction, position, and condition. The condition of the traffic signs is determined in FTA's instructions [1] using three parameters expressed at a categorical scale of 1 to 5: the condition of the surface, the overall condition, and the structural integrity. The three parameters include the following:

1. Structural condition: includes wear, rust marks, deflections, and distortions.
2. Appearance condition: includes colour fading, shade differences, stubborn stains, graffiti, and surface growth.
3. External damage: contains outside mechanical damage.

The word traffic sign usually includes simple geometric signs and the larger signs that used to give information about distances and roads. This thesis differentiates these by the words traffic signs and traffic sign posts. Traffic signs are designed to stand out from the environment, regardless of the weather and illumination conditions. Figure 1 shows two example images from the environment and traffic signs.

a) **b)**

Figure 1. Varying road environments: a) Summer image; b) Winter image.

Pattern recognition studies the regularities and patterns in the data. Machine vision consists of methods and their applicability for image-based inspection as input data for pattern recognition. From the machine vision point of view, the automatic Traffic Sign Inventory (TSI) and condition analysis consist of three parts: Traffic Sign Recognition (TSR), condition analysis and sign location estimation. The approach is illustrated in Figure 2. The machine vision research literature divides and defines TSR using two parts: Traffic Sign Detection (TSD) and Traffic Sign Classification (TSC). TSD is the problem of finding the traffic sign from an image. The purpose of TSC is to find out the sign class. Common-use cases for TSR are autonomous driving, assisted driving, and mobile traffic signs mapping.

1.2 Objectives and restrictions

This research is a part of a goal to develop an automatic system for TSI and condition analysis. This kind of system can be used assist local and national authorities in

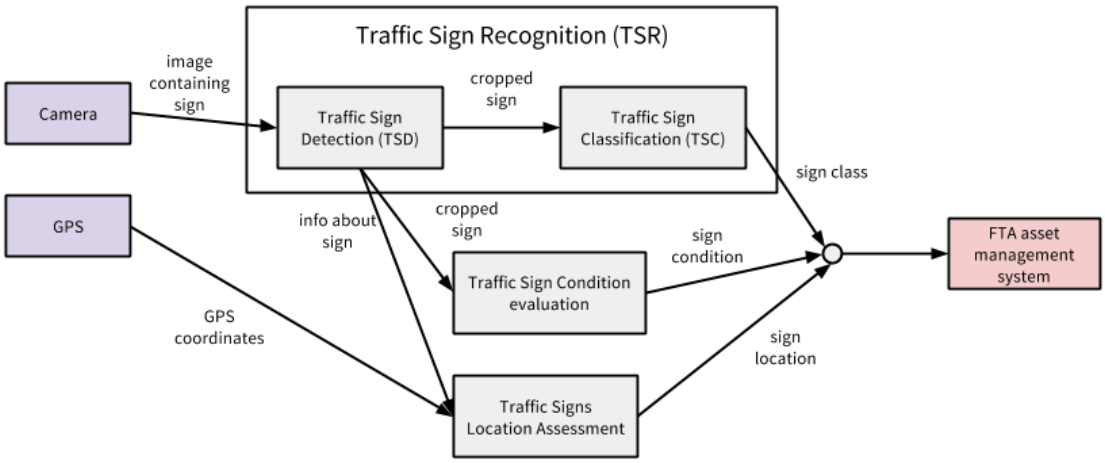


Figure 2. Simple TSI system overview including condition analysis.

the task of maintaining and updating road and traffic signs automatically. The task consists of detecting, classifying and analyzing one or more traffic signs from a complex scene when imaged by a camera mounted on a vehicle. A core idea is to present an inexpensive option without the need for a complex installation or an expensive system with high maintenance costs. One possibility to meet the above goals is to make a mobile application for assessing traffic sign conditions automatically. This is taken into account in the system design. This thesis is part of the FTA funded TrafficVision project and provides the documentation displays some of the results of the project.

The objective of this thesis is to survey, test, and design methods that can be used for TSI including the analysis of machine-vision-based traffic sign condition. Special care is taken to select methods that can be used in real-time applications using a mobile phone as the platform. Traffic sign posts are excluded from the scope of the research, although almost the same methods could be used for sign posts. Traffic sign images and models are limited to signs specific to the Nordic countries. The problem is approached as a generic vision problem with few assumptions pertaining to road signs, the road as an environment, and a camera mounted on a moving vehicle.

The specific objectives of the research are the following:

1. Evaluate the robustness TSD and TSC during road maintenance.
2. Study the automatic assessment of traffic sign location.
3. Evaluate the possibilities for condition analysis of traffic signs during TSI.
4. To specify the requirements for the equipment needed for such a system.

1.3 Structure of the thesis

The rest of the thesis is structured as follows. Section 2 outlines the task and clarifies the practical requirements this thesis is set to solve. It introduces the reader to the terminology and provides a literature review of the research subject. Section 3 discusses in general terms the machine vision tasks that are needed to solve the tasks defined in the previous section. The justification for selecting the methods used in the thesis are presented. Section 4 presents in detail the algorithms based on the previous selection. Section 5 contains the data collection, experiments, and results. Section 6 discusses the methods used, practical problems and the future directions of the research. Finally, Section 7 summarizes the thesis.

2 ROAD MAINTENANCE AND INVENTORY

This section discusses further the background, definitions, and requirements for the system. A system overview is provided to justify the topics discussed in Section 3. The purpose of this section is to give an idea about what the TSI and condition analysis is, what are the open problems, and how the problems have been solved before by machine vision.

2.1 Automatic traffic sign recognition

The purpose of traffic signs is to warn, control, guide traffic, and give information to the road users. Machine vision based TSR is an actively researched [3, 4] machine vision application area [5]. Majority of the research has been driven by the automobile industry to create support systems, autonomous vehicles, and road sign inventories for mapping services. When the TSR is coupled with condition analysis and location assessment, it can be used for semi-automatic assets management systems. The survey by Mongelmose et al. [5] provides a detailed analysis of the recent developments, datasets, and terminology. The localization of traffic signs has no published research available. The only traffic sign condition analysis research [6] uses the reflectance of special infrared light as measurements; this thesis has a different approach to the problem. TSR can be used for the following purposes [5]:

1. TSI: Collect and catalogue traffic signs with machine vision.
2. Highway maintenance: Check the presence and condition of signs along the main roads.
3. Driver assistance systems: Assist the driver by informing about the current restrictions and warnings.
4. Intelligent autonomous vehicles: An autonomous vehicle must obtain knowledge of current traffic regulations from the traffic signs.

An up-to-date inventory of traffic signs is ideally needed to help ensure adequate updating and maintenance of traffic signs. An automated process of TSI could also help developing the inventory accurately and consistently. Automatic condition analysis ensures that the condition of the signs on the road are known, and it is easier to locate the signs in the worst condition and replace them. TSI algorithms have to cope with a natural and complex dynamic environment, high accuracy demands, and real-time operations. These demands are usual in generic machine vision and do not differ from the methods in generic machine vision uses. The task of this

thesis is the application of general machine vision object detection, classification and analysis methods for the specific task of using traffic signs as objects. To make the task easier, installation locations of traffic signs with respect to the road and the traffic signs themselves are strictly defined in Finland by the FTA [7].

TSR approaches in literature make use of two prominent features: colour and shape information. Due to diverse natural lighting conditions the treatment of colour is difficult and many heuristics have been proposed and applied [8, 9]. Regarding shape, two paradigms are currently pursued: model (such as circles) based and methods arising from the Viola-Jones [10] detector. TSD is usually performed with a computationally complex sliding window [3] approach or computationally inexpensive colour thresholding [11, 12]. There are several approaches to TSC [13].

The survey by Mongelmose et. al [5] highlighted that a direct comparison of the methods and results of different algorithms is difficult. Studies usually use different data, either consider the complete task chain of detection [14], classification and tracking or consider only part of the chain. Commonly the researches focus on the classification or detection [15] only, and use different comparison metrics. A major part of the published research concentrates on a certain subclass of signs, for example, speed limit signs. Three large open traffic sign datasets for detection and classification have been recently released: Belgian 2011 [16], Swedish 2011 [17], and two German(2012, 2013) datasets [13, 4]. There exists no datasets related to traffic sign condition analysis. The German datasets have been used in two benchmarking competitions in 2012 and 2013. The competition results and papers published based on them were used as the starting point in developing the system presented in this thesis.

2.2 Traffic signs as objects

In Europe traffic signs were standardized at the United Nations Vienna convention on Road Signs and Signals in 1969 [18]. Shapes are used to categorize different types of signs: circular signs are prohibitions (such as speed limits), triangular signs are warnings, and rectangular signs are used for recommendations and as sub-signs in combination with other signs. Additionally, an octagonal sign is used for full stop, and a downward-pointing triangle is to signal yield responsibility. There are several signs that do not strictly follow the conventions.

The United Nations Vienna convention designates white as the second colour of prohibitory signs. In Finland and Sweden, white is replaced by yellow [19, 7] for better visibility in the snowy landscape. The pictograms and the font used differs from country to country. Signs in Sweden and Finland are very similar. The traffic signs in Finland come in three standard sizes [7]: small (400 mm), medium (640 mm) and large (900 mm). The normal size for a traffic sign is medium, and other sizes are rare. Traffic signs are placed consistently along the road. Traffic signs can be located on both sides of the road, on the middle line of road, and above of the road. There is a defined maximum of three traffic signs on each pole. Installation locations of traffic signs and the signs themselves are defined in Finland by the FTA [7]. Using this information in TSI would require knowledge of the road location in the image. Detecting the road is a difficult task [20], especially in winter road maintenance conditions.

Traffic signs are designed with the following features to make them easily recognisable and informative to humans with respect to the environment [7]:

1. Road signs are designed, manufactured, and installed according to strict regulations.
2. Each sign has a certain defined 2D shape such as triangle, circle, octagon, or rectangle.
3. The colour of the sign is chosen to contrast with the surroundings, to make it easily recognisable by the driver.
4. The colours are regulated mostly by the category of the sign.
5. The information on the signs is in one colour and rest of the sign is in a different colour.
6. The sign is located at well-defined locations with respect to the road so that the driver can anticipate the location of the signs.
7. The signs can contain a pictogram, a string, or both.
8. Traffic signs (and sign posts) use fixed text fonts and character heights.

Unfortunately for machine vision, traffic signs are not designed in an exactly standardized way. The traffic signs can be divided into five categories. Figure 3 shows example sign models for each category. The categories are designed as follows:

(a) **Mandatory**: round, blue inner, white symbols and such.
(b) **Danger**: triangular (corner up), white (yellow in Sweden and Finland) inner, red rim. Newer warning signs have a thin yellow edge.
(c) **Prohibitory**: round, white inner (yellow), red rim.
(d) **Priority**: signs that do not belong to any of the previous and govern who

should drive first.

(e) **Other**: signs not belonging to any of the above.

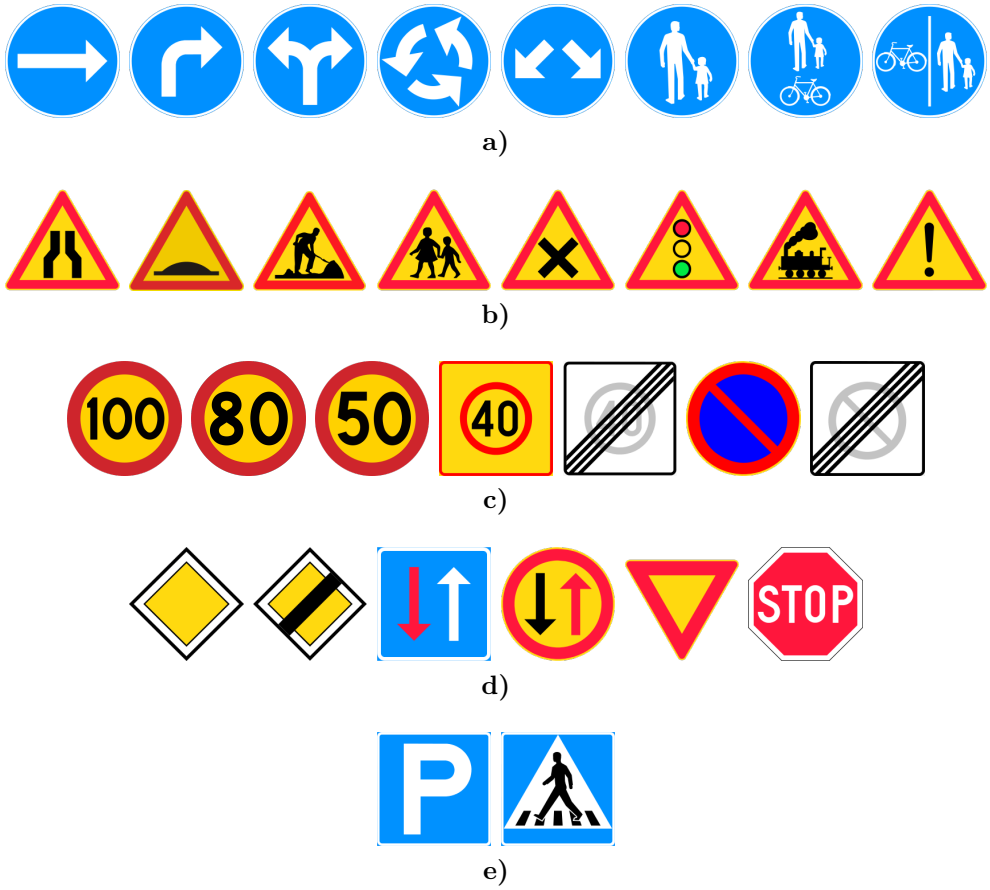


Figure 3. Traffic sign model examples from different categories: a) Mandatory signs; b) Warning signs; c) Prohibitory signs; d) Priority signs; e) Other signs [19].

2.3 Traffic sign condition analysis

Traffic sign condition analysis is used to define a proper time for the replacement and repairing of traffic signs. The conditions of traffic signs are collected during arduous work taking road maintenance inventories. The known condition of the traffic signs is used in different maintenance task and when evaluating the maintenance contracts and calculating costs for these contracts.

Traffic signs condition analysis is done to all constant traffic signs on roads and pedestrian traffic paths. This includes traffic signs, traffic signposts, and other equipment used to guide the traffic. The condition analysis includes mechanically rotatable signs, but not Light Emiting Diode (LED) based signs. Traffic sign con-

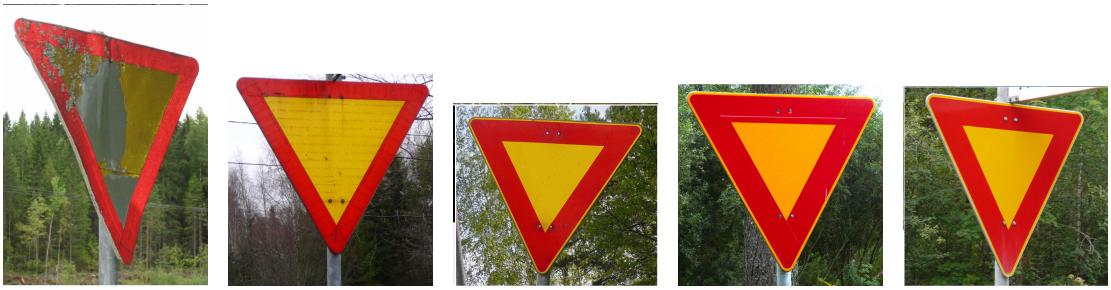
dition analysis is performed just face side of the traffic sign, excluding the pole and the feet of the sign. The declination of the pole is not evaluated during analysis. In principle, the daily condition such as snow, dirt, and vegetation is dismissed in the condition analysis. The traffic signs surface is either painted (older signs) or made of reflective tape (newer signs). The material and the environmental condition determine how the signs are affected by corrosive effects.

The condition of traffic signs in Finland is evaluated according to guidance from the FTA [1]. In the current process, the signs are analysed visually and a verbal analysis is added with explanatory pictures. The reflectance of traffic signs is evaluated based only on visual cues, such as the amount of damage a sign has suffered. The overall condition of a traffic sign is a categorical value between 1 (worst) and 5 (best) based on to the bottom value of three subcategories. Table 1 summarizes the evaluation guidelines for the verbal visual condition category. Figure 4 shows examples of different sign condition categories. If there are multiple signs in one sign pole, the signs are evaluated separately. The condition analysis of traffic signs is based on the following three parameters [1]:

- **Structural condition**: The phase of technical life cycle. The evaluation value is decreased by weariness, distortions, surface membrane detachment, cracks, and tears.
- **Appearance condition**: Visually detectable by discolouring, darkening, accumulated dirt that cannot be removed, and smudges. Also, colour differences of the panels should be considered.
- **External damage**: Correlates to the condition decrease caused by external force and mechanical damage.

Table 1. The three fuzzy traffic signs condition category parameters [1], used by the FTA's subcontractors.

Class	Structural	Appearance	Damage
5	As new	Flawless	No damage
4	Little weariness	Good	Little damage
3	Weariness	Does not affect recognition	Noticeable damage
2	Clear deficiencies	Covering errors	Clear damage
1	Bad deficiencies	Affects the readability	Bad damage



a) b) c) d) e)

Figure 4. Traffic signs in different phases of their technical life cycle. Corresponding conditions categories are: a) 1; b) 2; c) 3; d) 4; and e) 5. The images are provided and annotated by the FTA.

2.4 Operating environment

Roads are complex environments. The colour of traffic sign fades with time as a result of long exposure to sunlight and the reactions of the paint with the air. The presence of objects of a similar colour to traffic signs, such as buildings and vehicles increases the difficulty for machine vision task. There might be illegal advertisements resembling traffic signs along the sides of the roads. The legal advertisement is regulated, but only based on location and direct resemble to traffic signs. Colour information is also strongly related to the type of camera, illumination, and age of the sign. The visibility of signs is affected by weather conditions such as fog, rain, clouds, and snow. Appearance of the signs is sensitive to variations in the lighting conditions, such as shadows, clouds, and the sun. Colour is also affected by the illumination colour (daylight), illumination geometry, and viewing geometry (angle, distance). Signs can also be damaged, disoriented, or occluded.

It is possible to use road maintenance vehicles as a platform for the camera. This would provide several benefits in addition to lowering costs. The vehicle provides the lighting, no separate lighting is needed. Road maintenance vehicles traverse same roads several times a week. Therefore, the system could get multiple shots of the traffic signs for TSI and evaluation. Road maintenance vehicles operate throughout the year, but winter would be preferable for the system because denser maintenance period of the roads. A possible problem for machine vision is that in the winter the maintenance vehicles move in difficult conditions and in the dark. The system should be tested especially under these conditions. In the data collection of the TrafficVision project the camera is installed inside the vehicles cabin. Because the image is acquired from a moving car, it often suffers from motion blur and car vibration.

2.5 Camera and the geometry

An important part of the TSI and condition analysis system is the camera and the set up camera is installed on. Approaches in the literature for TSI use either a single camera, a dual camera [21, 16], or specialized equipment such as infrared cameras [6]. The camera and the lenses used asserts the spatial resolution of images, the field of vision, colour accuracy of images, and the lighting conditions required to capture images. There are also other variables effecting the imaging and camera, such as the amount of motion blur, and the amount of vibration, optical stabilization, that the selection of camera effects. An important factor for TSI is how far the camera can be from the signs to capture shots accurately enough for the condition analysis. The amount of information contained inside each patch relative to the distance is illustrated in Figure 5. Estimating visually from the image the size of the patch extracted around the traffic sign from the image has to be around 100×100 pixels to distinguish features related to the sign's condition.

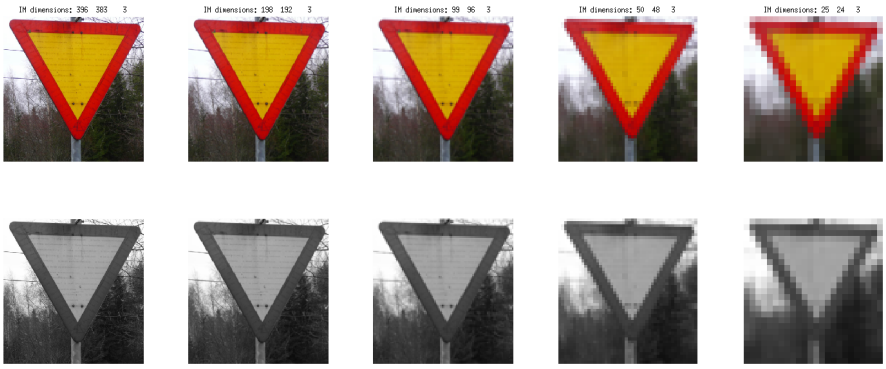


Figure 5. Simulated effect of distance to image quality and spatial resolution with colour and greyscale images. The image resolutions from left to right are 396×383, 190×192, 99×96, 50×48, and 25×24. With the camera (Garmin VIRB Elite Black) used in the experiments the pictures should to be taken at distances of 2.18 m, 4.35 m, 8.70 m, 17.41 m, and 34.81 m respectively. The amount of details disappears as the distance increases.

Figure 6a) illustrates the localization and location assessment situation. The observer moving forward detects traffic signs in relative motion coming towards the observer. The signs are detected, classified, and localized using the observer's known GPS coordinates. Visualization of the camera angles needed for accurate localization is shown in Figure 6b). In the localization of this thesis, the third dimension is also considered, but to simplify the illustration the method is described in 2D. The camera and GPS are positioned at the observer's location relative to the road (angle

β). The observer is moving along the movement vector V. As can be seen from Figure 6b), the camera is not necessarily aligned to point towards the movement vector. The angle α is the angle from the sign positioned at the side of the road to the centre of the field of vision.

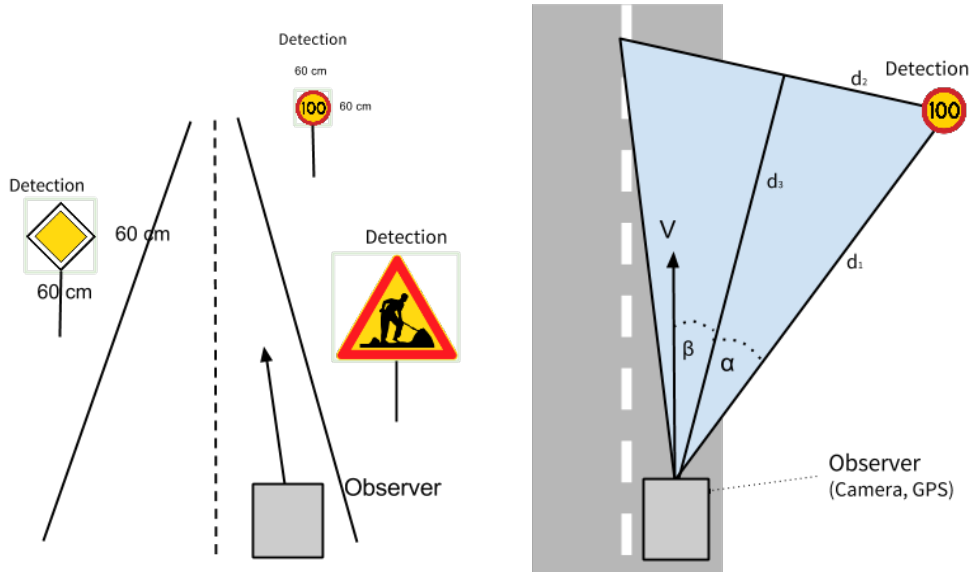


Figure 6. Geometry in the road environment: a) Perspective projection; b) Camera angles, distance and the relation to observer roads, and sign.

2.6 System overview

The combined TSI and condition analysis system is presented in Figure 7. The modules of the system (marked as grey) work together to perform the condition analysis and TSI task. Object detection, object classification, and condition analysis all contain feature extraction, feature post-processing, and classification submodules. The modules and their purposes are as follows:

1. **Camera and GPS**: A camera captures video material and corresponding GPS locations are stored. The camera can be the camera in a mobile phone with build-in GPS, for example.
2. **Image pre-processing**: A phase where the images are processed to be more easily processable later.
3. **Object detection**: The main task of the detection module is to detect traffic signs in the 2D image plane. The detection outputs the location of a possible

signs in the image and the reliability of the detection.

4. **Object classification**: The located signs (objects) are classified to know which of the signs they are.
5. **Localization** When the detection is combined with known camera parameters it enables the estimation of the distance to the detected signs. The distance can be further refined using known angles. The refined locations can be projected to a 3D space and the possible positions in the next and corresponding positions in the preceding frames are determined (assignment problem).
6. **Trajectory prediction**: Information about the localized signs is further refined by predicting the space-time trajectories for the signs. This information is used as a prior for the next detection round. The relationship between trajectories and the detections is asymmetric, new detections can occur while old ones vanish.
7. **Global location assessment**: The sign positions have to be accurately mapped to the world coordinate system using the interpolated/extrapolated GPS coordinates and the 3D localized signs.
8. **Condition evaluation**: The condition of the found signs is analyzed. The sign is first segmented, then sign condition features are extracted, and the condition category is determined.

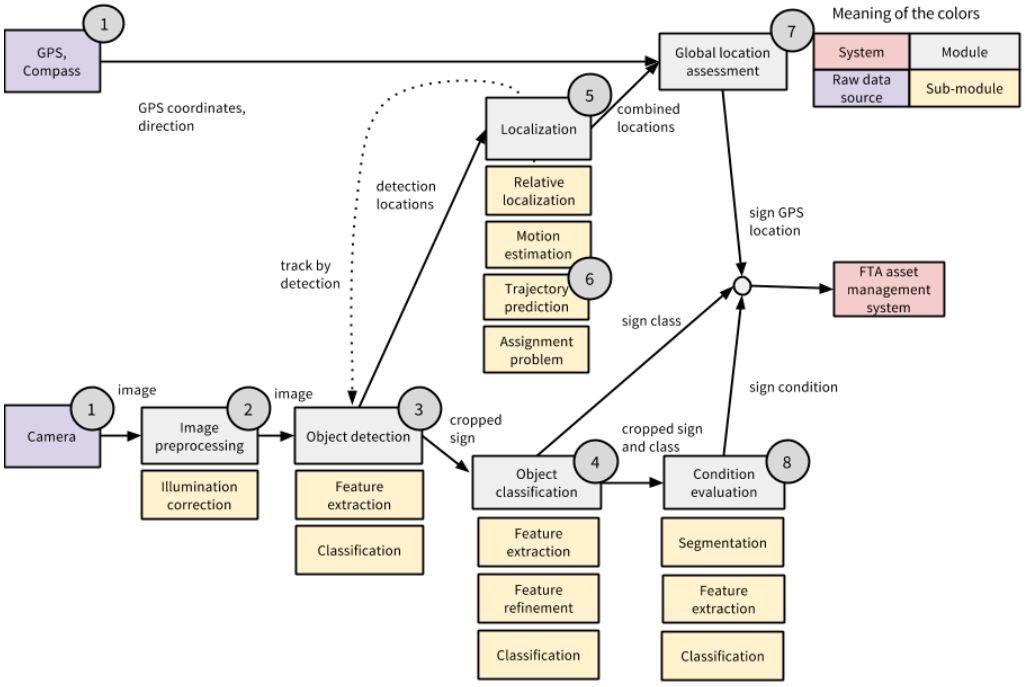


Figure 7. Modules of TSI and condition analysis system.

3 METHODS FOR TRAFFIC SIGNS

This section describes machine vision tools and methods needed for TSI and traffic sign condition analysis. Section 2 presented modules going to be solved in this section with specific machine vision methods. The possible methods are first analyzed using a general literature review and afterwards a method is going to be chosen for using requirements of the system. The classification is presented before detection because the detection is a special case of classification with few specific methods.

3.1 Image pre-processing

The purpose of pre-processing images before any other operation is to normalize and transform the images to be more suitable for machine vision. For example, a commonly used operation in pre-processing is colour and lighting effect normalization. Selection of low-level transformation/normalization varies amongst methods and the requirements of the application. Low level details have an important impact on the final results. The choice is between no normalization, local normalization [22], and global normalization [3].

3.1.1 Colour constancy

An image is formed usually from three colour channels [23]. When combined, these channels form a colour space. The simplest way to remove a light's effect on the image is to move from the normally used Red, Green, Blue (RGB) colour space to one that defines the colour channels differently. Common alternative representations are Lightness and chromaticity coordinates U and V (CIE LUV) and Hue, Saturation, and Value (HSV) colour spaces. RGB is commonly used in images because it reflects the way camera sensors and display matrices are constructed. The CIE LUV is used in machine vision because it normalizes the L2 norm, corresponding to euclidean distance (L2) between different colours. HSV colour space is intuitive for humans because it is divided into hue, saturation and value (brightness) channels.

Colour constancy is an important step in many problems and it is a prerequisite to ensure the perceived colour of the surfaces in the scene does not change under varying illumination conditions. The observed colour of the surfaces in the scene is

a combination of the actual colour of the surface, i.e., the surface reflection function as well as illumination and sensor. Estimation of illumination is the main goal of the colour constancy task. The colour constancy aims to correct the effect of the illumination by computing invariant features or by transforming the image to remove the effects of the colour of the light.

Several surveys [24, 25] have been conducted to compare the performance of colour constancy algorithms. For the method selection for this thesis, only colour constancy algorithms for single light source are evaluated, though there are also algorithms for several light sources [24]. The white patch and max-RGB methods estimates the maximum response from different channels. Another well-known method is based on the Grey World hypothesis [26] assuming the average reflectance in the scene is achromatic. Grey Edge [27] is a version which assumes that the average reflectance in the scene is achromatic. Shades of grey [25] is another grey-based method using Minkowski p-norm instead of regular average averaging. These methods deal with the image as a bag of pixels and the spatial relationship is not considered.

An example of previous colour constancy algorithms applied to a frame is shown in Figure 8. It has been shown that global normalization [28, 3] can have a medium impact on TSD and TSC performance. Despite this, the improvements are marginal and are not really worth the computation time. The colour constancy is thought to be useful in condition analysis, when the colour correctness really matter. Grey World algorithm is chosen for the condition analysis systems colour constancy method because it provides stable results and is fast to compute.

3.2 Feature selection and extraction

In machine learning feature selection is the process of selecting a subset of relevant features x to form feature vectors $\vec{x}$ and to combine them into feature sets $\vec{x}_1, ..., \vec{x}_N$. The feature vector sets are used to create statistical model M using mathematical object called classifier. The purpose of the feature vectors is to describe the object abstractly. The problem is difficult because objects usually vary greatly in appearance. Variations are created by changes in illumination, different viewpoints, non-rigid deformations, intraclass variability in shape, and other visual properties. Image data contains many redundant and irrelevant parts. Redundant parts provide no more discriminative information than the previously selected features, and irrelevant features provide no useful information in any context. In the case of traffic

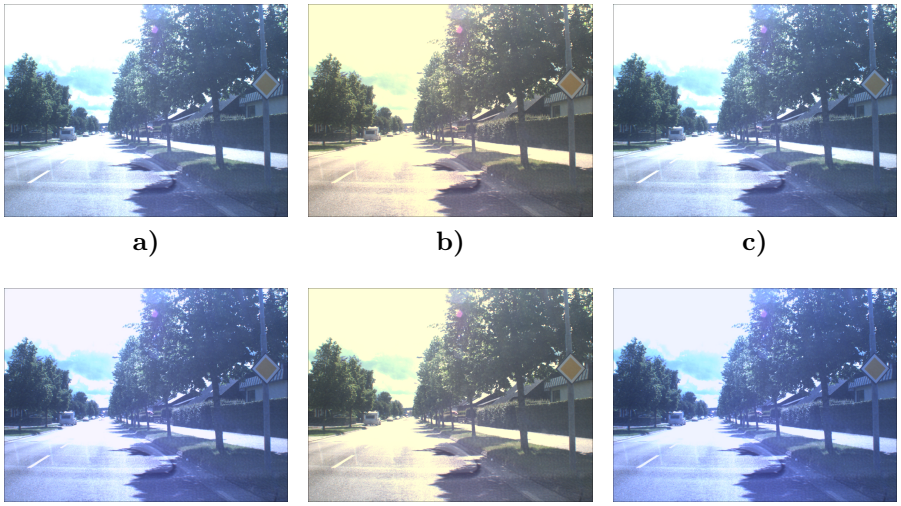


Figure 8. Different colour constancy algorithms: a) the original image; b) Grey World; c) Max-RGB; d) Grey Edge; e) Shades of Grey; f) Weighted Grey Edge.

signs, the relevant information, that the features should contain, is the information defining traffic signs and separating them from the background and from each other. Feature selection is a key design choice during the TSR.

In machine vision, a feature vector set $\vec{x}_1, ..., \vec{x}_N$ is an array whose feature entries are multi-dimensional feature vectors computed from a dense grid of locations in an image. Intuitively feature vector $\vec{x}$ describes an object inside a local image patch. The model M can be used to compare the similarity of new feature vectors $\vec{x}_{new}$ to the feature vector set $\vec{x}_1, ..., \vec{x}_N$ used to create the model. Image features x are divided into two categories: low- and high-level features. Figures 9b) and 9c) show two pixel level features where individual pixel values are used as features. The individual pixel values are concatenated to form feature vectors. In higher-level features the feature is a combination of pixel information from a larger area. An example of this is presented in Figure 9d).

Edges [29] are low-level features describing edges around an object or on a surface of an object. Modern edge features and the edge localization accuracy is compared by Bansal et al. [30]. One possibility for edge detection are Gabor filters, that have been shown to have many invariant properties [31]. Another low-level feature is colour either as a pixel-wise feature or an area feature such as average colour. A group of increasingly popular low-level features are automatically optimized convolution filters [32]. These features can combine several filters together to form a filter-bank that is used to extract a feature vector from the image. The filters in a filter-bank

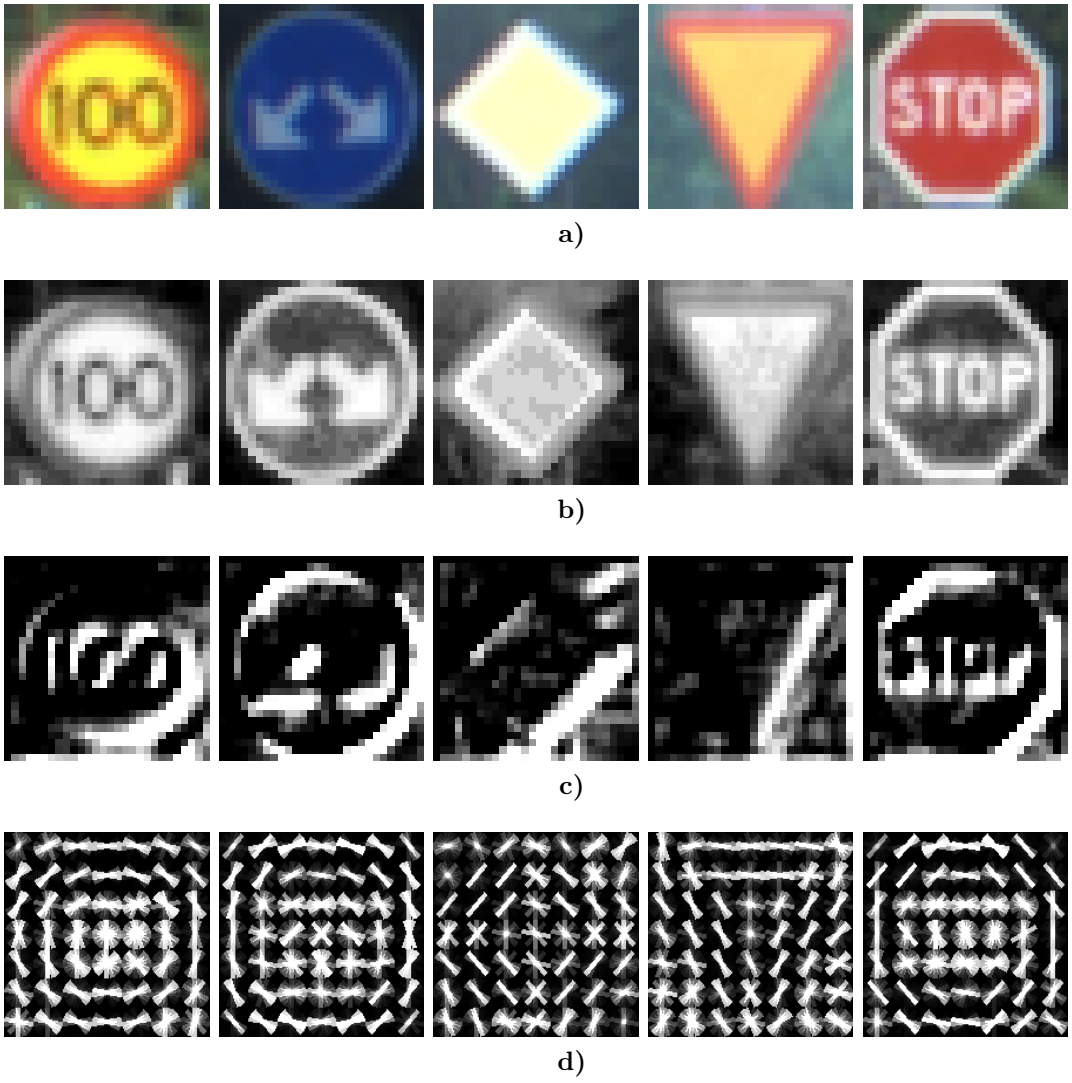


Figure 9. Illustration of different features: a) Cropped signs; b) Cropped converted to grey-scale; c) Edge features; d) Histogram of Oriented Gradients (HOG) features.

can be optimized automatically [33].

Common high-level feature extractors, also known as descriptors, used in machine vision are Scale-Invariant Feature Transform (SIFT) [34] and Histogram of Oriented Gradients (HOG) [22]. Many modern object detection and semantic segmentation systems are built on top of one or both of these features. HOG is a good method to capture dense shape features of rigid objects and SIFT sparse features of non-rigid objects. The features can be either single-scale, or multi-scale features where the original image is resized and features are computed several times at different scales [13]. State-of-the-art methods use multi-layer filters, so that the features of the first layer are fed to a second layer to get high-level features [35].

Traffic signs are constructed to be easily detectable by humans. There are well-defined cues (such as shape and colour) that can be utilized for the use of feature extraction algorithms. TSD is a classic instance of rigid object detection, and HOG features have been used on several occasions as features for traffic sign [13, 4, 36] related problems. The research [3] conducted by Mathias et al. has a comparison of HOG feature parameters, different scales, and their performance as features for traffic signs.

In this thesis, colour channel and HOG features are going to be used for TSD and for TSC HOG features are used. The choice is based on the literature [3, 4, 13]. The systems condition analysis uses edge and colour variance inside regions to form feature vector $\vec{x}$. The edges were chosen because signs in bad condition begin to deteriorate and form ridges that can be detected on the surface of the signs. A Canny [29] edge filter was chosen for the system. Colour variance was chosen because colours in signs should be constant across the same colour in surface. There are two different ways that could have been taken, individual feature detection from a surface (e.g. rust marks or vegetation) or the statistical approach. For this research the latter was chosen based on the simplification it provides. Vegetation could have specifically engineered features to extract it from the surface, but for the condition analysis, it would be enough to tell if there is something wrong with the surface of the sign.

3.3 Feature post-precessing

There are two commonly used methods in feature post-processing: feature set scaling and dimensionality reduction. The methods are applied after the features are concatenated to feature vector sets $\vec{x}_1, ..., \vec{x}_N$. The right method depends on the circumstances, but the idea is that transformation needs to make the extracted feature vector set $\vec{x}_1, ..., \vec{x}_N$ more easily processable for machine learning methods. For example, the feature vectors often contain outliers, datapoints that are distant from other observations often because of errors in measurement. The feature post-processing is a good place to remove those outliers if needed.

3.3.1 Feature scaling

There are currently two simple methods for feature scaling: normalization and standardization. The methods are straightforward, common knowledge. The basic use case is to apply them when using multiple feature vectors that are in different units of measure, in order to make the features comparable to each others. In normalization, the range of feature vector values $\vec{x}$ is normalized to be between 0 to 1. The lowest value $\vec{x}_{min}$ is set to 0 and the highest value $\vec{x}_{max}$ is set to 1. This is useful when all features need to have the same positive scale. In normalization the outliers are lost because they are often the minimum or maximum values. In this case, all the other data will be scaled according to outlier producing a negative effect on the data. Normalization is defined as

$$\widehat{\vec{x}} = \frac{(\vec{x} - \vec{x}_{min})}{(\vec{x}_{max} - \vec{x}_{min})} \tag{1}$$

where $\vec{x}_{max}$ is the maximum value of the feature vector and $\vec{x}_{min}$ is the minimum value of feature vector. Standardization rescales data to have a mean of 0 and a standard deviation of 1 (unit variance). For the most applications' standardization is recommended as it makes outlier spotting easy and makes the different features easily comparable with each other. Standardization is defined as

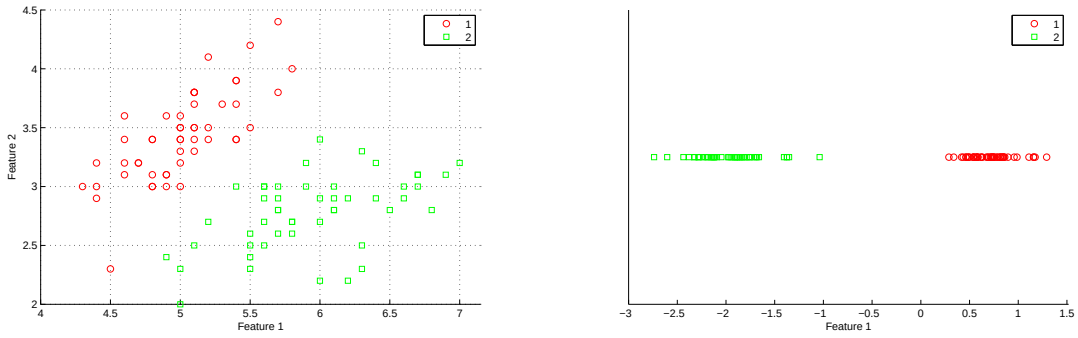
$$\widehat{\vec{x}} = \frac{(\vec{x} - \text{mean}(\vec{x}))}{\text{std}(\vec{x})} \tag{2}$$

where mean corresponds to the mean of feature vector and std denotes the standard deviation of the feature vector. Both of the methods are applied in the experiments of the thesis. The normalization is used when dealing with image data and the standardization to process condition analysis data.

3.3.2 Dimensionality reduction

The idea of dimensionality reduction is to refine the feature vector set $\vec{x}_1, ..., \vec{x}_N$ by removing unneeded features or feature dimensions while maintaining most of the descriptive power of original feature vector set [37]. Using too big feature space requires lots of memory and processing time for machine learning algorithms. Using too small feature space impoverishes the capacity of the machine learning, and lead to a bad results. A common way to deal with a big feature space is to use dimensionality reduction techniques. Dimension reduction is used to project the data from

higher feature dimensions to lower, removing unneeded features x. Figure 10 illustrates projecting data from two dimensions to one dimension, making two example classes more easily separable. In the example illustration Figure 10a) both classes contain a two dimensional feature vector. After the Linear Discriminant Analysis (LDA) dimension reduction, the feature vector (as shown in Figure 10b)), is reduced to one dimension still containing the same discriminative information.



a) **b)**

Figure 10. LDA projection of two features and classes: a) Two dimensions; b) One dimension.

When using appearance-based features (such as traffic signs), image $m \times n$ is usually represented by a feature vector $\vec{x}$ in an $m \times n$ dimensional space. In practice these spaces are too large to allow robust and fast object classification. A common way to attempt to resolve this problem is to use dimensionality techniques. Two of the basic methods are: Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) [37]. LDA directly deals with the classes, and PCA just tries to find from the entire data the principal components without taking into account the class structure. Sparse representation based graph embedding has been found useful in [38] when using traffic sign features as inputs. LDA is a linear projection technique and non-linearities $\vec{x}_1, ..., \vec{x}_N$ might be lost in the process. There are non-linear dimension projection methods, but they are outside the scope of this thesis. In the classification experiments of this thesis, LDA and PCA are compared.

3.4 Classification

In machine learning and statistics, classification is used to decide a class for a feature $\vec{x}_{new}$ with unknown class, based on the previous features of known classes using a

model M. The model M is a combination of set of features $\vec{x}_1, ..., \vec{x}_N$, known class labels corresponding to feature vectors in the $\vec{x}_1, ..., \vec{x}_N$, and a statistical method. Training is the method of creating the model and testing is the testing of a sample against the model. Classifier is an abstract term for applying a certain model to the observations.

Classification has been approached with a number of different classification methods such as K-Nearest Neighbours (KNN) [39], Support Vector Machine (SVM) [40], different kinds of tree classifiers such as AdaBoost [41], and Random Forests [42]. There are numerous classifiers from which different ones work better for different kinds of data. The factors to base of the classifier selection are: the number of feature vectors, number of different classes or categories, the dimensionality of data, and the distribution of features among dimensions (linear or non-linear). Some classifiers (such as Gaussian mixture models [43]) return a probability predicting how probable the correct classification is. This probability (commonly known as posterior probability) is useful, but not all classifiers can produce this information.

In the problem context of this thesis the purpose of the classifier is to model possible variations of the environment can have on the traffic sign. TSI and condition analysis together contain three separate classifications tasks. During the TSD (first task) a classifier is used to discriminate a set of traffic signs from the background (also known as detection). The second task uses a classifier on the image patch found in the previous step (TSD) to determine the class of the sign (TSC). TSC is a multi class categorization problem with thousands of dimensions to distinguish among different classes. The third classification task, the condition analysis, is similar to the second task, but there are only five condition categories (classes) to make the classification decision.

The first classification task, TSD, is a special case of classification, and will be discussed in more detail later. TSC has been approached in the literature with KNN [3] Random Forests [13, 36], Neural Networks (NN) [13, 33], and different variations of SVMs [3]. In TSC the difference in results caused by the different selection of a classifier is usually small when dimension reduction techniques are used and features are reasonable [3]. The biggest differences appear in training and testing times. In the results, SVM appears to be slow in both testing and training. Random Forest are slow to train, but fast to test. KNN does not require training, and the testing time is the fastest of the compared methods. In TSC and condition analysis experiments, KNN is used as the base-line method. The more complex Random Forests classifier is also tested.

3.5 Detection

Detection is a task where different object classes are searched and localized from images. In detection, the classification (also known as the search for the object in image) has to be performed on the whole image, not just for a patch of image such as in the basic classification. There are two ways to perform this search; sliding window or segmentation based selective search method. The process is similar in both. First a patch of image is extracted, the patch is pre-processed, a set of features is calculated for the patch, and lastly the result is compared to a model (classified) to find out if the patch contained an object being searched. Because of the large amount of model comparisons the detection method has to be very fast, the classification accuracy is less important.

The sliding window approach [22] for object detection is currently popular and provides good results [41, 44, 14, 8]. In the approach, the detection is done by defining a score from a classification model at different positions and scales in an image. The highest scores compared to a threshold are considered as detections. The sliding window detector can be thought of as a classifier that takes as input an image, a position within image, and a scale. The classification model is usually simple to make the classification as fast as possible. The model in the sliding window can also consist of a set of models trained on discovered sub classes (so-called components) [44]. An alternative to sliding window is recognition using regions [45, 35]. The core idea is to generate category independent region proposals from the input image, and to classify them. It has been shown that, recognition using regions method processes two orders of magnitude fewer image windows compared to sliding window approach [35].

To reach good performance on a sliding window detector, multiple scales' can be used improve the quality [46] of detection results. In the multi-scale method, low- and high-resolution models are used to evaluate a single candidate window. This increases computational cost- and is a problem. The process can be sped up by using computational tricks (such as a feature pyramid) which specifies a feature map for a finite number of scales in a fixed range. In practice this is done [41, 47, 48] by computing the feature pyramid via repeated smoothing and sub-sampling and then computing a feature map from each level of the image pyramid. This way the detection is fast to compute [47]. The problem with the selective search is the highly demanding region proposal, also known as segmentation, process.

In the TSD phase, the background has to be distinguished from the object model that has only two classes (background and traffic sign). The AdaBoost [42] classifier can usually handle such situations well, is fast to train and performs especially well when the feature set is large. The methods chosen for this research uses the concept of the sliding window search and AdaBoost model based approach. The choice was made based on the benchmarks [4, 47]. The regions-based approaches were discarded because successful segmenting an image usually relies on an advanced edge detection methods (such as the Berkley Boundary Detector (gPB) [49]) later becoming object candidates. gPB takes several seconds per image and makes the regions based approach unfeasible despite progress in faster edges [50]. Another possibility for would be colour thresholding [51, 9]. The colour segmentations seem to work only in limited lighting conditions, and is discarded.

3.6 Localization

This subsection presents three different uses and needs for movement analysis and using prior information to improve the detection times and performance. There are several ways to improve the methods using priori knowledge, for example using the car's trajectory. One example of the use of priori knowledge would be to use previously appeared signs to predict an area the sign is going to appear in the next time step. Figure 11 illustrates this by showing the Bounding Boxs (BBs) locations of 600 annotated sign in images. There is no need to use the sliding window search to a whole image, when searching small part of image should be enough. The movement vector is also needed for more accurate object localization, camera angle calculations, and motion blur removal.

3.6.1 Camera model and orientation

A camera can be approximated by a projective model, often called a pinhole projection model. The simplest representation of the camera is a light sensitive surface (sensor): an image plane and a lens (projective projection) at a given position and orientation in space. It has an infinitesimally small hole through which light enters before forming an inverted image on the camera surface facing the hole. Usually, simpler pinhole camera model is used by placing the image plane between the focal point of the camera and the object so the image is not inverted. This mapping of three dimensions onto two is called a perspective projection, shown in Figure 12.

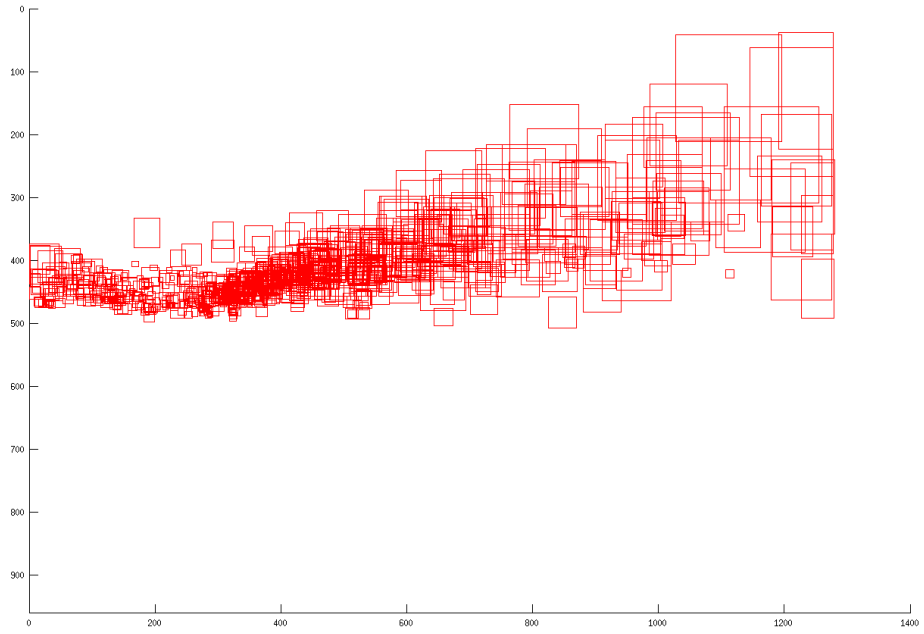


Figure 11. Locations of 600 traffic sign BBs. Figure illustrates that the whole image does not need not be searched.

Perspective geometry [52] is fundamental to mapping points from 2D to 3D.

A perspective projection is the projection of a three-dimensional object onto a two dimensional surface by straight lines passing through a single point. Let f be the distance of image plane to the centre of projection. Then the image coordinates (u_i, v_i) are related to the object coordinates (x_0, y_0, z_0) as follows:

$$u_i = \frac{f_l}{z_0} x_0 \tag{3}$$

$$v_i = \frac{f_l}{z_0} y_0 \tag{4}$$

Equations 3 and 4 are non-linear. They can be made linear by introducing homogeneous transformations, which is effectively just a matter of placing Euclidean geometry into the perspective system. The pinhole camera geometry models the projective camera with two sub-parameterizations, intrinsic and extrinsic parameters. Intrinsic parameters model the optic component (without distortion), and extrinsic parameters model the camera position and orientation in space. This projection of

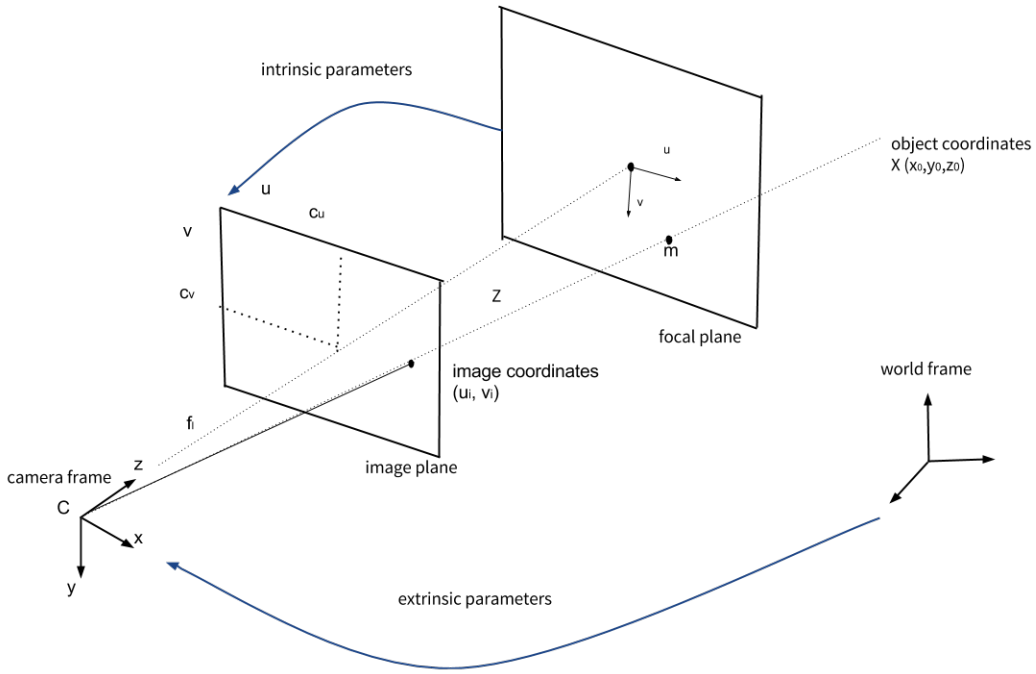


Figure 12. Perspective projection in the pinhole camera model.

the camera is described as follows:

$$P_{3\times4} = \begin{bmatrix} f*k_u & 0 & c_u & 0 \\ 0 & f*k_v & c_v & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5}$$

The equation consists of intrinsic (k_u, k_v, f, c_u, c_v) and extrinsic parameters ($R_{3\times3}$, $t_{3\times1}$). k_u and k_v determine the scale factor relating pixels to distance (usually 1), the focal length f determines the distance between focal and image plane, and c_u, c_v is used to denote the principal point that ideally is at the centre of the image. Extrinsic parameters are the rotation parameters $R_{3\times3}$ and the translation of the camera $t_{3\times1}$. The translation of the camera is the origin of the world coordinate system expressed in coordinates of the camera centred coordinate system. The position of the camera,C_{pos}, expressed in world coordinates is $C = -{R_{3\times3}}^{-1}t_{3\times1} = -{R_{3\times3}}^{T}t_{3\times1}$. A 3D point X_i is projected in an image using homogenous coordinates as follows:

$$x_i = PX_i = K[R_{3\times3}|t_{3\times1}]X_i \tag{6}$$

The estimation of distance from the car to the traffic sign is necessary for accurate traffic sign location estimation. Two different high level schemes for traffic sign

localization can be derived. The first uses the detected traffic sign's height with camera parameters to estimate projection using the geometric method called triangle similarity. Another, more constrained (and maybe more accurate) way to do the localization would be to wait until the sign reaches the camera's edge and from that information map the location.

Equation 5 can be used directly to derive the computations needed for the distance estimation with the triangle similarity. Basically an opposite operation of projecting a point to a plane has to be computed. Now the point on a plane is projected on a 3D world using the known size of the traffic sign as a constraint. The simple camera model in Figure 12 illustrates the problem being solved. The point, X, is then changed to a surface (line) denoting a traffic sign. The side of medium-sized traffic sign S_{sign} is known to be 640 mm. When it is placed known distance Z in front of the camera and its apparent width in pixels is measured to get d. Focal length of the camera is $f = \frac{d \times Z}{S_{sign}}$. When a traffic sign is seen again with this camera with a width of d' pixels, then by triangle similarity it is known that $\frac{f}{d'} = \frac{Z'}{S_{sign}}$ and the distance Z' can then be calculated as:

$$Z' = S_{sign} \times \frac{f}{d'} \tag{7}$$

After the triangle similarity to get the distance, there is still a need to evaluate the corresponding transforms to get the relative position of the sign compared to the car. This can be computed by simple geometric transformation because the angles are known or can be calculated in respect with the image plane.

3.6.2 Motion estimation

Motion estimation is one essential component in video processing. It is often used for motion-compensated temporal interpolation to reduce motion blur artefacts. The motion vectors can be obtained by using a predictive block based motion estimator. To avoid mismatches, additional metadata (such as knowledge of forward movement) can be used to support the motion estimator. Motion vectors can also be used to estimate the camera angle. When the observer is moving forward, the motion vectors seem to be coming from the vanishing point. When the deviation between the vanishing point and the camera centre point is known, the angle of the camera's deviation with respect to the vehicles' movement direction can be computed.

The perceived motion field of the camera image plane is the sum of translational

and rotational components. Several methods (such as Lucas-Kanade [53] and Horn-Schunck [54]) have been proposed to perform the recovery of three-dimensional motion from image flow fields by applying the model of the pinhole camera and perspective projection. The use of optical flow has been adapted to road navigation [55]. In the system implementation, optical flow is used to estimate the camera angle by estimating the vanishing point from a moving vehicle. Figure 13 illustrates the optical flow magnitude calculated from frames taken in moving vehicle, 0.3 s apart in time. The blue corresponds to a low value and red to a high value.

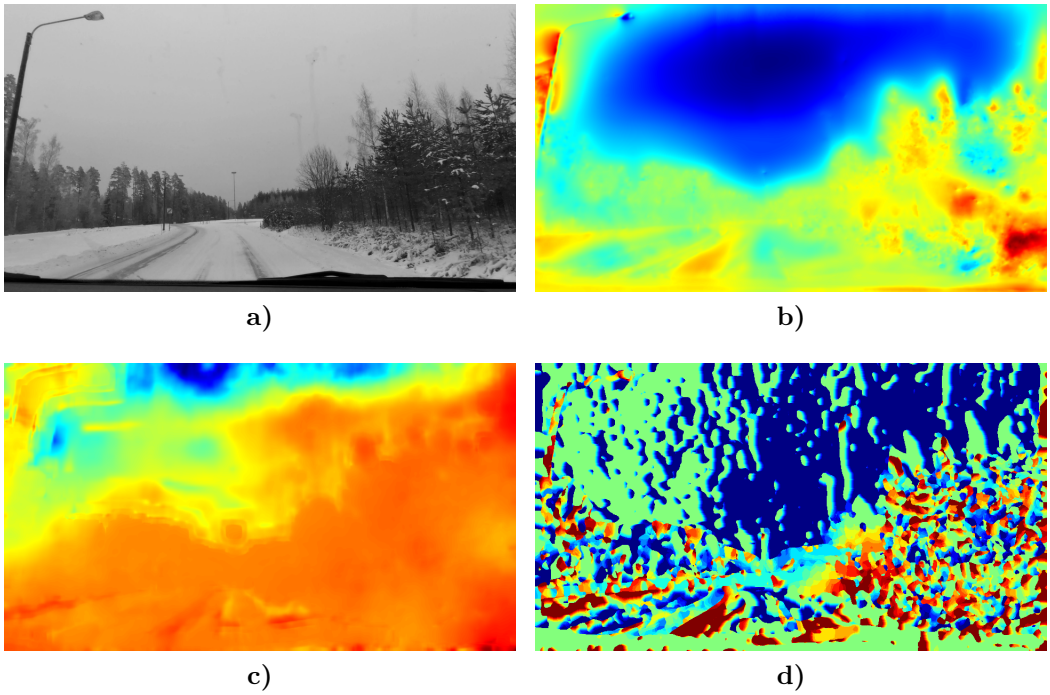


Figure 13. The optical flow computed from succeeding frames imaged from a forward moving vehicle, taken 0.3s apart: a) original image; b) Horn-Schunk; c) Lucas-Kanade; d) Sum of Squared Differences. Only the magnitude information is shown.

3.6.3 Trajectory prediction and assignment

In the standard multi target tracking problem the targets move continuously in a given region, typically independently according to a known, Markovian process. Targets arise at random in space and time, persist for a random length of time and then cease to exist. The sequence of states a target follows is called a track. Positions of moving targets are measured typically in a periodic scan measuring the positions of all targets simultaneously. The position measurements are noisy and

occur with detection probability of less than one. In this scenario there are three sub-problems: tracking, prediction, and assignment problems.

Recent approaches to tracking pursue tracking by detection strategy [56] where the targets are detected in a preprocessing step, usually either by background subtraction or using a discriminative classifier from which trajectories are later estimated. In the TSI system, the detector can be used directly for this task. The benefits are improved robustness against drifting and the possibility of recovering from tracking failure. In the relatively simple single-target setting, where only one target is present in the scene, tracking can be approached by searching for the object of interest within the expected area and forming a plausible trajectory by connecting object's locations over time. When a higher, often unknown number of targets are observed simultaneously, the problem becomes much more complicated because it is no longer obvious which object corresponds to detections. This task of correctly identifying different objects over time is often referred to as data association. Motion, appearance (known class of sign), and visibility of objects are affected by mutual dependencies that have to be taken into account. From a probabilistic point of view this entails inference, often Maximum A Posteriori Estimation (MAP), in a posterior distribution over several not independent variables.

Many tracking algorithms utilize recursive methods where the current state is predicted using information from previous frames. Kalman filter approaches [57] are a prominent example. Particle filtering (also known as sequential Monte Carlo) was introduced later. In particle filtering, a set of weighted particles sampled from a proposal distribution is maintained to represent the current (unknown) state [58]. This allows handling non-linear multi-modal distributions. As the number of targets grows, a reliable representation of the posterior requires an ever-increasing number of samples and is hard to handle in practice. The assignment/data association problem can be solved using Joint Probabilistic Data Association Filter (JPDAF) [59], Markov Chain Monte Carlo (MCMC) [60] based models, or Hungarian algorithm [61].

This thesis utilizes tracking by detection approach for the traffic signs. Previously introduced detector can be used as the detector for the tracker. Kalman filter is used as the predictor for the detector and the Hungarian algorithm is used to solve the assignment problem.

3.7 Global location assessment

After the object has been detected and classified it has to be localized and finally the location has to be converted to world frame defined by GPS coordinate system. The task requires an understanding of 3D computer vision [52] and geodesic on an ellipsoid of revolution [62]. These problems have mathematically proved solutions; the possible error comes from measurement inaccuracy.

The shortest path between two points on Earth, customarily treated as an ellipsoid, is called a geodesic. The direct problem is to find the end point of a geodesic, given the starting point, initial azimuth and length. The inverse problem is to find the shortest path between two given points. Every geodesic problem is equivalent to solving the geodesic triangle, given two sides and their included angle (the azimuth at the first point) in the case of a direct problem, and the longitude difference in the case of an inverse problem. The mathematical foundation was laid in the beginning of the 19th century. The modern counterpart algorithms [62] can be computed fast and accurately. For the problem of this thesis Karney's implementation is used [62].

3.8 Condition evaluation

The surface condition evaluation can be divided into three steps: defining exactly where the surface is, extracting the features, and then estimating the condition of the exact surface. The region proposition has lots of research behind it, but the requirement of exactness is difficult. An example of a same sign in condition 1 and condition 5 is presented in Figure 14.

Segmentation is a well-researched subject [63]. For the segmentation of intensity images, there are four main approaches: thresholding techniques, boundary-based methods, region-based methods, and hybrid techniques combining boundary and region criteria. Thresholding techniques are based on a postulate that all pixels whose value (grey level, colour value, or other) lies within a certain range belong to one class. Such methods neglect all the spatial information of the image and do not cope well with noise or blurring at boundaries.

Boundary-based methods use a postulate that the pixel values change rapidly at the boundary between regions of the image. The basic method is to apply a gradient edge operator such as a $[1, 2, 1]^T \times [-1, 0, 1]$ filter. High response value to this filter

a) b)

Figure 14. Signs with different surface conditions: a) condition 1; b) condition 5.

provide a candidate for region boundaries, which must then be modified to produce closed curves representing the boundaries of the regions. Converting the edge pixel candidates to boundaries of regions of interest is a difficult task, but there are good solutions [49, 50].

The regions-based methods rely on a postulate that neighbouring pixels within one region have a similar pixel value. The general procedure is to compare one pixel to neighbouring pixels. If a criterion of homogeneity is satisfied, the pixel is said to belong to the same class as one or more of its neighbours. The choice of homogeneity criteria is critical for success, and the results are easily distorted by noise. The methods include superpixels [63] and region-growing [64].

The fourth type is the hybrid techniques combining boundary and region criteria. An example of this kind of method is the watershed algorithm. The watershed algorithm is usually applied to the gradients of the image. The gradient image can be viewed as a topography with boundaries between the regions as ridges. Segmentation is equivalent to flooding the topography from the seed points [65].

For the segmentation of traffic signs a region-based method was chosen. Seeded region growing [64] has been shown to work in traffic sign segmentation [12]. The algorithm is simple, but has several parameters to be tuned.

4 ALGORITHMS FOR TRAFFIC SIGNS

This section introduces algorithms chosen in Section 3. Figure 15 illustrates the algorithms going to be presented (blue boxes) in this section and their relation to each others.

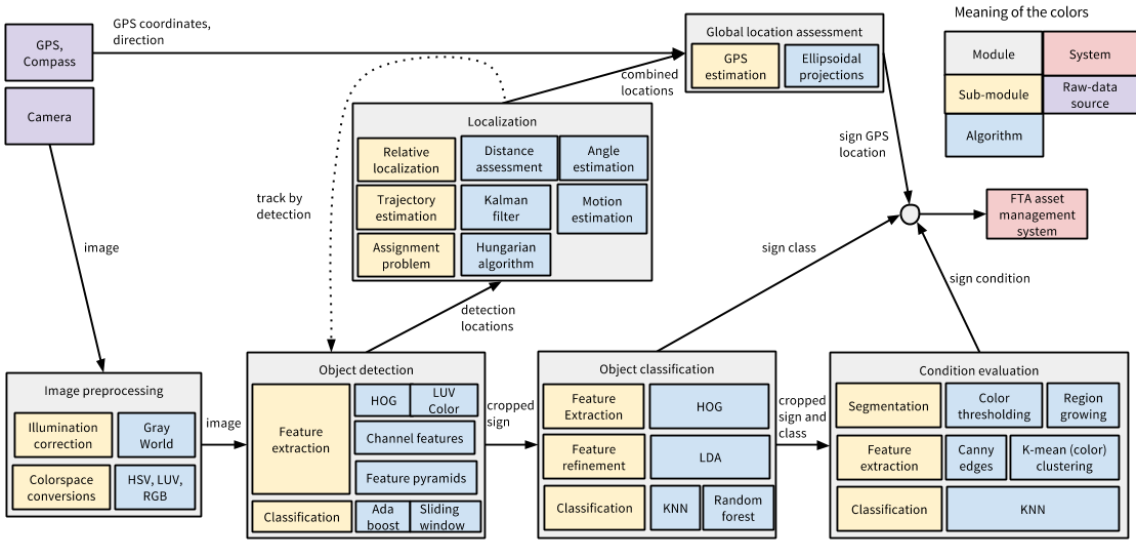


Figure 15. Algorithms of the TSI and condition analysis system.

4.1 colour space and colour constancy

The HSV model has been widely used in colour segmentation. A RGB image is converted into the HSV colour space with the following three pixel-wise equations (each channel is formed separately) [23]:

$$H = \cos^{-1}\{\frac{(R-G)+(R-B)}{2\sqrt{(R-G)^2+(R-B)(G-B)}}\}\ , R \neq G \text{ and } R \neq B \tag{8}$$

$$S = \frac{\max(R,G,B) - \min(R,G,B)}{V} \tag{9}$$

$$V = \max(RGB) \tag{10}$$

In general, the goal of computational colour constancy is to estimate the chromaticity of the light source and then to correct the image to a canonical illumination using a diagonal model. The grey based methods have been formulated into a unifying framework [66, 24]. The process consists of three steps: reflection model,

illumination estimation, and diagonal correction model.

Surface reflection model

Image colour $I = (I_R, I_G, I_B)^T$ for a Lambertian surface at location x can be modelled as:

$$I_C(x) = \int_w E(\lambda, x) S(\lambda, x) \rho_c(\lambda) d\lambda \tag{11}$$

where $C \in \{R, G, B\}$, $E(\lambda, x)$, $S(\lambda, x)$ and $\rho_c(\lambda)$ are the illuminant spectrum distribution, surface reflectance, and camera sensitivity, respectively. For a given location of x, the colour of the light source $L(x)$ can be computed as follows:

$$L(x) = \begin{pmatrix} L_R(x) \\ L_G(x) \\ L_B(x) \end{pmatrix} = \int_\omega E(\lambda, x) \rho(\lambda) d\lambda \tag{12}$$

Normally colour constancy is involved with estimating the chromaticity of the light source. Estimating this chromaticity from a single image is an under-constrained problem (underdetermined system), as both $E(\lambda, x)$ and $\rho(\lambda) = (\rho_R, \rho_G, \rho_B)^T$ are unknown. Therefore, assumptions are needed to impose on the imaging conditions. Typically, assumptions are made from the statistical properties of the illuminants or surface reflection properties. Most colour constancy algorithms are based on the assumption that illumination is uniform across the scene $E(\lambda, x) = E(\lambda)$.

Illumination estimation

Illumination estimation methods can categorized into two groups: (1) static methods trying to estimate the illuminant for each image based on the statistical properties, and (2) learning-based methods trying to estimate the illuminant learned from training images. For example, the white-patch algorithm is based on an assumption that the maximum response in a scene is white, and the grey world algorithm is based on an assumption that average colour in the scene is achromatic. These assumptions are used to make a global estimate of the light source and to correspondingly correct the images. The grey-based methods have been formalized into a single framework:

$$\left(\int \left| \left| \frac{\partial^n I_{C,\sigma}(x)}{\partial x^n} \right| \right|^p dx \right)^{\frac{1}{p}} = k L_C^{n,p,\sigma} \tag{13}$$

where $L^{n,p,\sigma}$ is used to denote different instantiations of the framework, $||\cdot||$ denotes Frobenius norm, $C = \{R, G, B\}$, n is the order of the derivative, p is the Minkowski norm, and $I_{C,\sigma} = I_C \otimes G_\sigma$ is the convolution of the image with a Gaussian filter with smoothing parameter σ. According to the characteristics of the Gaussian filter the derivative can be described by

$$\frac{\partial^{a+b} I_{c,\sigma}}{\partial x^a y^b} = I_C * \frac{\partial^{a+b} G_\sigma}{\partial x^a \partial y^b} \tag{14}$$

where $*$ denotes the convolution and $a + b = n$. Using Equation 13, many colour constancy algorithms can be derived by varying one or more parameters (i.e., n, p, σ). Pixel based colour constancy algorithms ($n = 0$) can be created by varying Minkowski norm p and smoothing parameter σ. The Grey World algorithm $n = 0, p = 1, \sigma = 0$, i.e., $L^{0,1,0}$ and the white-patch algorithm $p = \text{inf}$, i.e., $L^{0,\text{inf},0}$ are simple first order colour constancy algorithms. Using higher order colour constancy methods (i.e., $n = 1$) and (i.e., $n = 2$) results in the first-order grey-edge ($L^{1,1,1}$) and the second order grey edge ($L^{2,1,1}$).

Diagonal colour correction model

After the colour of the light source is estimated, the aim is to transform the input images, taken under an unknown light source, into colours as if they appear under a canonical light source (a theoretical equal energy radiator where equal weights is given to all wavelengths), into colours as if they appear under a canonical light source. This is done using a diagonal model described as:

$$I^C = \Lambda^{u,C} I^u \tag{15}$$

where I^u is the image under an unknown light source while I^C is the image transformed, appearing as if taken under canonical illuminant. $\Lambda^{u,C}$ is the mapping diagonal matrix described as:

$$\Lambda^{u,C} = \begin{pmatrix} \alpha & 0 & 0 \\ 0 & \beta & 0 \\ 0 & 0 & \gamma \end{pmatrix} = \begin{pmatrix} \frac{L_R^C}{L_R^u} & 0 & 0 \\ 0 & \frac{L_G^C}{L_G^u} & 0 \\ 0 & 0 & \frac{L_V^C}{L_B^u} \end{pmatrix} \tag{16}$$

where L^u is the unknown light source and L^C is the canonical light source.

4.2 Histogram of oriented gradients

Algorithm 1 describes the HOG algorithm generally. Let $\theta(x, y)$ and $r(x, y)$ be the orientation and magnitude of the intensity gradient at a pixel (x, y) in an image. The gradients are computed using simple $S_x = [-1, 0, 1]$ and $S_y = [-1, 0, 1]^T$ filters. The gradient orientation at each pixel is discretized into one of the orientation p values with contrast insensitive definition:

$$B(x, y) = \text{round}(\frac{p\theta(x, y)}{\pi}) \mod p \tag{17}$$

where round means rounding and mod is the modulus. The pixel level feature map specifying a sparse histogram of gradient magnitudes is then defined. Let $b \in 0, \ldots, p-1$ range over orientation bins. The feature vector at pixel (x, y) is then defined as:

$$F(x, y)_b = \begin{cases} r(x, y) & \text{if } b = B(x, y) \\ 0 & \text{otherwise} \end{cases} \tag{18}$$

F can be though of as an oriented edge map with p orientation channels. For each pixel a channel is selected by discretizing the gradient orientation. The gradient magnitude can be seen as a measure of edge strength.

Let F be a pixel level feature map for $w \times h$ image. Let $k > 0$ be a parameter specifying the side length of a square of a square image region. A dense grid of rectangular cells is then defined and pixel level features aggregated to obtain a cell-based feature map C, with feature vectors $C(i, j)$ for $0 \leq i \leq [(w-1)/k]$ and $0 \leq j \leq [(h-1)/k]$. This aggregation provides invariance to small deformations and reduces the size of the feature map. The simplest approach for aggregation is to map each pixel (x, y) into a cell and to define the feature vector $\vec{x}$ at the cell to be the sum of the pixel level features in the cell. A more complex way is to use soft binning where each pixel contributes to the feature vectors in the four cells around it using bilinear interpolation.

Gradients are invariant to changes in bias. Invariance to gain can be achieved via normalization. Four different normalization factors have been used the new feature vector $C(i, j)$. These factors can be written as $N_{\delta,\gamma}(i, j)$ with $\delta, \gamma \in \{-1, 1\}$ as:

$$N_{\delta,\gamma}(i, j) = (||C(i, j)||^2 + ||C(i + \delta, j)||^2 + ||C(i, j + \gamma)||^2 + ||C(i + \delta, j + \gamma)||^2)^{\frac{1}{2}} \tag{19}$$

Each factor measures the "gradient energy" in a square block of four cells containing

(i, j). Let $T_\alpha(v)$ denote the component wise truncation of a vector v by α. The HOG feature map is obtained by concatenating the result of normalizing the cell based feature map C with respect to each normalization factor followed by truncation defined as:

$$\vec{x}(i,j) = \begin{pmatrix} T_\alpha(C(i,j)/N_{-1,-1}(i,j)) \\ T_\alpha(C(i,j)/N_{+1,-1}(i,j)) \\ T_\alpha(C(i,j)/N_{+1,+1}(i,j)) \\ T_\alpha(C(i,j)/N_{-1,+1}(i,j)) \end{pmatrix} \tag{20}$$

Commonly used HOG features use $p = 9$ contrast insensitive gradient orientations, a cell size of $k = 8$ and truncation of $\alpha = 0.2$. This leads to a 36 dimensional feature vector.

Algorithm 1: HOG feature extraction.

input : I - greyscale image
k - cell size
p - number of orientation angles
α - truncation factor

output: $\vec{x}$ - feature vector

Create edge filters $S_x = [-1, 0, 1]$ and $S_y = S_x^T$
Get the edges $G_x \leftarrow I \otimes S_x$ and $G_y \leftarrow I \otimes S_y$
Compute the gradient magnitude $G \leftarrow \sqrt{{G_x}^2 + {G_y}^2}$
Compute the gradient direction $\theta \leftarrow \arctan(\frac{G_y}{G_x})$, and round to closest $\frac{\pi}{p}$ angle
Create edge based feature map F
Divide F into sub images C using k
Normalize the subimages C
Truncate orientation channel using α to get the feature vector $\vec{x}$

4.3 Aggregated Channel Features detector

The starting point for the detection is the Integrated Channel Features (ICF) detector. In a sense is can be seen as a successor to the classic Viola and Jones work [10]. ICF [41] is a precursor to the Aggregated Channel Features (ACF) detection framework introduced in [48]. Both ICF and ACF use the same features and boosted classifiers. The key difference is that ACF uses pixel lookups in aggregated channels

as features while ICF uses sums over rectangular channel regions. ICF has been used to detect the traffic signs [3], and the results are good.

The detection framework and algorithms related are straight-forward. Given input image I, several feature channels $C = \Omega(I)$ are computed, every block of pixels is summed in C, and the resulting lower resolution channels are then smoothed. Features are single pixel lookups in the aggregated channels. Boosting is used to train and combine decision trees over these features (pixels) to distinguish object from background and a multi scale sliding window approach is employed.

In general the detection approach in this thesis can be divided into 3 steps:

1. **Channel computation**: The detectors feature channels are chosen based on the literature review [41]. The features are formed from 10 different channels; 6 different orientation HOG features, CIE LUV colour channels, and gradient magnitude channel.
2. **Pyramid construction**: The pyramid is constructed using 8 scales.
3. **Detector training**: For traffic sign detection, AdaBoost is used to train and combine decision trees over the candidate features (channels pixel lookups) in each window.

4.3.1 Channel Features

The channel features are a name given to a combination of separate features to form a single feature vector. The used channels are normalized gradient magnitude, histogram of oriented gradients in 6 directions (6 channels) and CIE LUV colour channels separately. Prior to computing the 10 channels, I is smoothed with a Gaussian filter. The channels are divided into 4×4 blocks and pixels in each block are summed. Finally, the channels are smoothed, again with a Gaussian filter. Figure 16 illustrates the feature channels used to train the ACF detector.

Rectangular regions are then selected and assembled in a set of weak classifiers using boosting. Final strong classifier is a linear combination of the weak classifiers. The ACF framework enables the use of multiple kinds of ”channels” (low level pixel wise features). The ICF and ACF frameworks have been show to perform at the state-of-the-art level using HOG and CIE LUV features [41, 48].

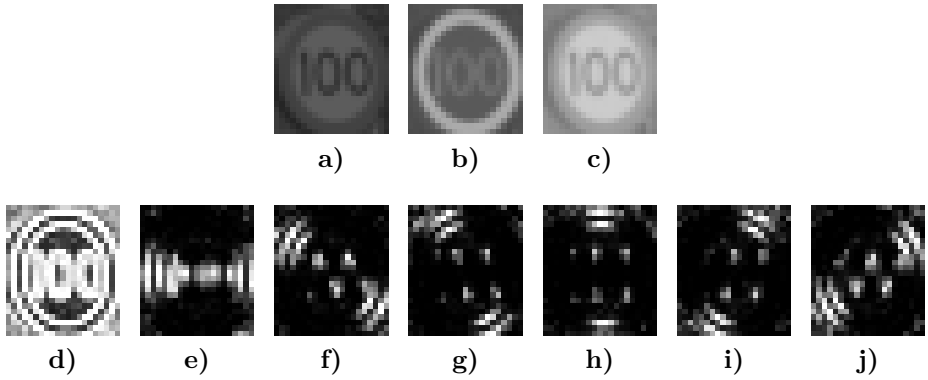


Figure 16. Feature channels to detect traffic signs. a) CIE LUV L, b) CIE LUV U, c) CIE LUV V, and d) gradient magnitude. The remaining are gradient channels with different orientations.

4.3.2 Fast feature pyramids

Estimating features at a dense set of scales is computationally expensive for sliding window search. By decreasing the redundancy in the representation, the computational costs can be decreased. This can be done by extrapolating computations carried out expensively, but infrequently, at a coarse sampled set of scales. This insight can reduce the computational cost considerably [48]. The speed-up in done in pre-computing image features for feature pyramids.

Let I_s denote I captured as scale s and $R(I, s)$ denote image I re-sampled by scale s. The channel image is denoted by Ω and the sampling factor λ, defined at [48]. Suppose $C = \Omega(I)$ has been computed. It is possible to predict channel image $C_s = \Omega(I)$ at a new scale s using only C. The standard approach is to compute $C_s = \Omega(R(I, s))$, ignoring the information contained in already computed $C = \Omega(I)$. The following approximation is proposed:

$$C_s \approx R(C, s) \cdot s^{-\lambda\Omega} \tag{21}$$

On per pixel basis, the approximation of C_s in Equation 21 is noisy. The accuracy of the approximation for C_s improves if information is aggregate over multiple pixels of C_s. A simple strategy for aggregating over multiple pixels and thus improving the robustness is to downsample and/or smooth C_s relative to I_s (each pixel in the resulting channels will be weighted sum of pixels in the original full resolution channel). Downsampling C_s also allows for faster pyramid construction. For object detection the channels are downsampled by a factor of 4 to 8 (e.g., HOG uses 8×8 bins).

A feature pyramid is a multi-scale representation of an image I where channels $C_s = \Omega(I_s)$ are computed at every scale s. Scales are sampled evenly in a log-space, starting at $s = 1$, with typically 4 to 12 scales per octave. An octave is the interval between one scale and another with half or double of the value. The standard approach to constructing a feature pyramid is to compute $C_s = \Omega(R(I, s))$ for every scale s.

The approximation in Equation 21 suggest straightforward method for efficiently computing feature pyramid. Computing $C_s = \Omega(R(I, s))$ at one scale per octave and estimating the rest provides a good trade off between speed and accuracy. The cost of evaluating Ω within 33% of computing $\Omega(I)$ at the original scale and channels do not need to be approximated beyond half an octave. While the number of R is constant (evaluations of $R(I, s)$ are replaced by $R(C, s)$), if each C_s is downsampled relative to I_s, evaluating $R(C, s)$ is faster than $R(I, s)$. The computational saving of computing approximate feature pyramids is significant. The total cost is $\frac{4}{3}n^2$ of computing single up/downsample scale is only 33% more than the cost if computing single scale features. Typically, detector is evaluated on 8 to 12 scales per octave, and so an order of magnitude saving over computing Ω densely, and intermediate C_s are computed efficiently through resampling afterwards.

4.3.3 AdaBoost

AdaBoost (Adaptive Boosting) [67] is a non-linear binary classification algorithm that uses several weak learners to learn a strong classifier. In AdaBoost each tree contains three stumps. Stumps are the simplest weak classifiers. It has been shown that level 2 decision tree perform well using AdaBoost [28]. Since the introduction of AdaBoost, the boosting principle has been used in numerous algorithms, each of them claiming to be superior to others. In the context of image classification it is unclear method is best in practice. Experiments comparing AdaBoost, Realboost, and LogitBoost show insignificant performance differences [41].

Algorithm 2 presents AdaBoost training phase. Let $\vec{x}_1, ..., \vec{x}_N$ be a set of training samples and $y_1 ... y_N,\ y \in \{-1, 1\}$ desired outputs. AdaBoost algorithm learns a strong classifier $F_T = \sum_{t=1}^{T} f_t(x_i)$, where $f_t = \alpha_t y_i h_t$ is a weak learner returning a real valued confidence in the classification of the sample x_i. Each weak learner t produces hypothesis $h(x_i)$ in combination with a weight α_t. T layer classifier will be positive if the sample is positive and negative otherwise.

Algorithm 2: Discrete AdaBoost training algorithm

input : $\vec{x}_1, ..., \vec{x}_N$ - samples
$y_1 ... y_N,\ y \in \{-1, 1\}$ - desired outputs
$E(f(x), y, i) = e^{-y_i f(x_i)}$ - error function
output: $H(x) = \sum_{t=1}^{T} \alpha_t h_t(x_i)$ - strong classifier.
Initialise weights $w_{1,0} ... w_{n,0}$ to $\frac{1}{N}$
for $t = 1, ..., T$ **do**
Choose weak learner $f_t(x)$:
Find weak learner $h_t(x)$ minimizing $E_t = \sum_i w_{i,t} e^{-y_i h(x_i)}$
Choose $\alpha_t = F_{t-1}(x) + \alpha_t h_t(x)$
Add to ensemble:
$F_t(x) = F_{t-1}(x) + \alpha_t h_t(x)$
Update weights:
$w_{i,t+1} = w_{i,t} e^{-y_i \alpha_t h + t(x_i)}$ for all i
Re-normalize $w_{i,t+1}$ so that $\sum_i w_{i,t+1} = 1$

The AdaBoost algorithm minimizes the total error $\sum_i e^{-\sum_{t=1}^{T} \alpha_t y_i h_t(x_i))}$, by sequential selecting h_t and computing α_t greedily. At each step the goal is to minimize:

$$E_t = \sum_i D_t(i) e^{-\alpha_t y_i h_t(x_i)} \tag{22}$$

with coordinate descent using two stage algorithm:

1. Select the best weak classifier from the candidate pool minimizing E_t.
2. Compute the α_t by taking $\frac{dE_t}{d\alpha_t} = 0$.

An important property of AdaBoost is that after certain number of rounds, the test error still descends even the training error is not improving. This makes AdaBoost less prone to overfitting problem than many other classifiers. This is because for any data (x, y),

$$\text{margin}(x, y) = \frac{y \sum \alpha_t h_t(x)}{\sum_t \alpha_t} \tag{23}$$

$\text{margin}(x, y)$ essentially gives the confidence of the estimation y to x. The margin is directly tied to the discriminative probability. There are three direction to reduce the test error:

1. increase the margin (related to training error).
2. reduce the complexity of the weak classifier.
3. increase the size of training data.

AdaBoost and the different variations of AdaBoost approach asymptotically the posterior distribution [67]:

$$p(y|x) = \frac{e^2 y \sum_t a_t h_t(x)}{1 + e^{2y \sum_t \alpha_t h_t(x)}} \tag{24}$$

4.4 Classification

After the detection the areas found from the image, a set of new features are computed and features are classified using a classifier. When using dense image features, the dimensions are commonly reduced before classification using dimension reduction techniques (such as PCA or LDA). Finally, the data is classified.

4.4.1 Linear discriminant analysis

It is beneficial to reduce the dimensionality of the features to improve performance. LDA [37] was chosen for projecting the original feature representation to lower manifolds. LDA is an embedding technique, maximizing inter class variance while minimizing the intra class variance. The LDA projection thus tries the best discriminate among classes. The solution can be obtained by solving an eigenvector problem. By construction LDA can lead to an embedding space with a number of dimensions less than the number of classes. For example, in the case of traffic signs there are less than 100 classes and number of HOG descriptors dimensions is 1568. By doing LDA on the HOG features, the dimensionality of HOG features can be reduced to the number of classes without significant loss of discriminative information.

In other words, LDA [68, 37] searches for vectors $\vec{x}$ in the underlying space that best discriminate between the classes $y_1 ... y_N$ corresponding to vectors. Given a number of independent features relative to which the data is described, LDA creates a linear combination of these yielding the largest mean differences between the desired classes. For all the samples of all classes, two measures are defined:

1. Within-class scatter matrix, defined as:

$$S_w = \sum_{j=1}^{c} \sum_{i=1}^{N_j} (x_i^j - \mu_j)(x_i^j - \mu_j)^T \tag{25}$$

where x_i^j is the sample i of class j, μ_j is the mean of class j, c is the number of classes, and N_j the number of samples in class j.

2. Between class scatter matrix:

$$S_b = \sum_{j=1}^{c} (\mu_j - \mu)(\mu_j - \mu)^T \tag{26}$$

where μ is the mean of all classes.

The goal is to maximize the between-class measure while minimizing the within-class measure. One way to do this is to maximize the ratio

$$\frac{\det(S_b)}{\det(S_w)}. \tag{27}$$

where det is the determinant function applied to matrix.

If S_w is a nonsingular matrix then this ratio is maximized then the column vectors of the projection matrix W, are the eigenvectors of $S_w^{-1}S_b$. There are at most $c-1$ nonzero generalized eigenvectors and therefore an upper bound on f is $c-1$, and that at least $t+c$ samples are required to guarantee that S_w does not become singular. To solve the problem intermediate space is often used. A common intermediate space is PCA space [37]. This, the original t-dimensional space is projected onto an intermediate g-dimensional space using PCA and then onto a final f-dimensional space using LDA. LDA is based on a MAP of the class membership.

Algorithm 3: LDA dimension reduction.

input : $\vec{x}_1, ..., \vec{x}_N$ - feature vectors

$y_1 ... y_N$ - corresponding classes

output: $\widehat{\vec{x}}$ - transformed feature vector

Compute class means $\mu_i = \text{mean}\, \vec{x}_{y_i}$

Compute $w = S_w^{-1}(\mu_i - \mu_j)$

Project data $\widehat{\vec{x}} = w^T \vec{x}$

4.4.2 K-nearest neighbour classifier

The nearest neighbor decision rule [69] assigns an un-classified sample $\vec{x}$ to a class that is the nearest compared to previously classified $y_1 ... y_N$ samples $\vec{x}_1, ..., \vec{x}_N$. The nearest neighbour rule is interesting classification algorithm in the sense that it

does not require training phase and the results are proven to be good. Algorithm 4 describes the decision (classification) rule.

Algorithm 4: KNN Classification

input : $\vec{x}_1, ..., \vec{x}_N$ - feature vector set.

$y_1 ... y_N$ - vector of classes corresponding to feature set.

k - number of the nearest neighbours to take into account.

$\vec{x}$ - unknown feature vector to be classified.

output: $y_{class} \in y_1 ... y_N$ - class of the sample vector

Calculate the distances $\vec{d}$ from $\vec{x}$ to each vector in $\vec{x}_1, ..., \vec{x}_N$

Sort the $\vec{d}$ and order $y_1 ... y_N$ indices correspondingly to produce $\vec{C}$.

Pick k first labels in $\vec{C}$ and calculate mode to get the y_{class}

Various distance measures can be used, for example L1 norm, corresponding to absolute distance (L1) or L2 distances. In the experiments of this thesis the L2 is used. When using KNN the best practice for the k is to use uneven number. This ensures there is only a single mode value in the k set. The easiest way for selection k is cross validation maximizing the performance. As the number of samples grows the KNN starts to approach Bayes decision rule and is bounded above by twice the Bayes probability error [69].

4.4.3 Random forest classifier

Trees, a set of simple decision rules, are popular learning and classifying approaches because they perform well when using unbalanced data sets. They are fast to build and update. Random forests is an extension of tree classifier introduced by Breinman and Cutler [42] in 2001, where multiple trees are used in voting scheme. To classify a sample, the classification of each random tree in the forest is taken into account. The class label of the sample is the one with majority of votes.

There are various different algorithms used to create decision trees. Hunt's Algorithm [70] is one of the earliest and serves as a basis for more complex algorithms. The decision tree is constructed recursively until each path ends in a pure subset (each path taken must end with a class chosen). There are three steps repeated until the tree is fully grown:

1. Examine the record data and find the best attribute for the first node.

2. Split the record data based on this attribute
3. Recurse on each corresponding child node choosing other attributes

Algorithm 5 describes the tree construction at high level.

Algorithm 5: Top down tree construction

input : t - node of the tree.
$\vec{x}_1, ..., \vec{x}_N$ - feature vector set.
S - split decision method (Gini, Entropy or classification error).

output: t - trained node of the tree

Apply S to $\vec{x}_1, ..., \vec{x}_N$ to find splitting criterion

if $t \neg$ *leaf node* **then**
Create children nodes of t
Partition $\vec{x}_1, ..., \vec{x}_N$ into children partitions
Recurse on each partition

How the tree is split is based on what kind of attributes the tree deals with. If the trees are dealing with binary attributes binary splits are used and when dealing with nominal, ordinal and continuous attributes either binary or multi-way splits are used. Certain attribute is selected based on how successful the split will be. The measure how pure each split will be can be calculated and then chosen. How successful these splits is measured using the GINI, Classification Error and Entropy. Another choice is how deep the tree will be. There is no certain measures to end splitting, the correct value must be set with cross validation or by pruning the tree after it has been fully grown. The pruning is the inverse of three growing and means removing the splits from the tree contributing the least to the validation error.

4.5 Multi-object tracking and trajectory estimation

The object tracking can be used to reduce the computation time of the detection algorithm. By combining the Hungarian algorithm [61] with Kalman filter [57] it is possible to estimate the trajectory of the detections and predict next appearance position, reducing the search area of next detection phase. In this thesis implementation the objects are tracked in 3D with Kalman filter, to make the filter linear.

The camera used in the experiments uses a sampling rate of 30 Frames Per Second (FPS). For the performance reasons every 10th frame is used. Therefore, the change

of location and between frames is relatively large in two adjacent frames. The size of the window and the centre position of moving target is considered. After the moving objects have been detected, process preparations for subsequent moving object tracking is needed. The detection search window is made slightly larger than the object size reducing noise interference and increase processing time. The Kalman motion tracking model can be divided into three submodules: motion model, feature matching, and model update.

4.5.1 Kalman filters for trajectory estimation

Mathematically Kalman filter [57] is an estimator that predicts and corrects the states of wide range of linear processes. It is not only efficient practically but also theoretically. The optimal state is found with the smallest possible variance error, recursively. However, an accurate model is essential requirement.

In Kalman filter x_k is state vector representing the dynamic behaviour of the object, subscript k indicates the discrete time. The objective is to estimate x_k from the measurement z_k. F is the state transition matrix transforming the last estimation to the current one and H is the measurement matrix. Both F and H have the size of $[n \times n]$ where n is the dimension of the state space, and w_k is the process noise with covariance Q, and e_k is the measurement noise with noise covariance R. Mathematical description of Kalman filter can be divided into four phases:

1. Process equation:
$$x_k = Fx_{k-1} + w_{k-1} \tag{28}$$
where A represents the transition matrix and x_k the state at time $k-1$ to k. Vector w_{k-1} is the Gaussian process noise $N(\cdot)$ with following normal probability distribution $p(w)$.
2. Measurements equation:
$$z_k = Hx_k + e_k \tag{29}$$
where H is the measurement matrix and z_k is the measurements observed at time $k-1$ to k respectively. v_k is the Gaussian measurement noise $N(\cdot)$ with normal probability distribution $p(v)$.
3. Time update equations:

   Equations 28 and 33 describe a linear model at time k. As x_k is not measured directly, therefore the information provided by measured z_k is used to update the unknown states x_k. Apriori estimate of state $\hat{x}_k^-$ and covariance error P_k^-

$(e_k = x_k - \vec{x_k})$ estimate is obtained for next time step k by computing:

$$\hat{x}_k^- = F\hat{x}_{k-1} + w_k \tag{30}$$

$$P_k^- = FP_{k-1}F^T + Q \tag{31}$$

4. Measurement update equations:

   These equations are associated with the feedback of the system. The objective is to estimate aposteriori estimating $\hat{x}_k$, a linear combination of the apriori estimate and the new measurement and the new measurement z_k

$$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1} \tag{32}$$

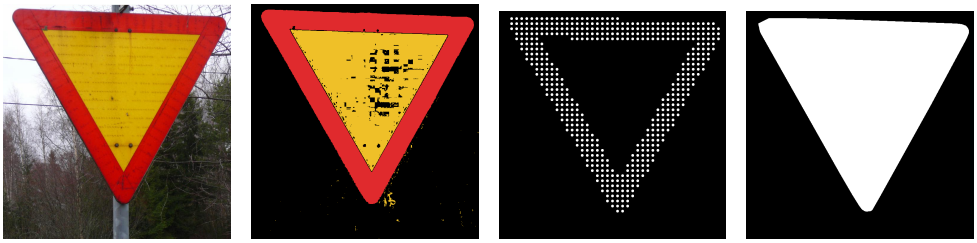
$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-) \tag{33}$$

$$P_k = (1 - K_k H)P_k^- \tag{34}$$

   K_k is the Kalman gain computed by above the measurements update equation. After the aposterior state estimate $\hat{x}_k$ and aposterior error estimate P_k is computed by the measurement z_k. The time and measurement equations are calculated recursively with previous aposterior estimates to predict new aprior estimate. This recursive behaviour of estimating the states is one of the highlights of the Kalman filter.

4.6 Segmentation

Segmentation is a necessary step in evaluation the condition of the traffic signs. Segmentation algorithm consist of three parts: Thresholding the colour channels [11], creating a seed image based on the thresholding results [12], and using the seed image as a set of starting points for the region growing algorithm [64]. The approach is not very efficient in sense of time and robustness. The seed image helps to overcome problems arising from the incomplete thresholding. Algorithms 6 and 7 describes the algorithms used for segmentation. Figure 17 illustrates the steps in the segmentation using the algorithms.



a) b) c) d)

Figure 17. Illustration of the segmentation process images step by step: a) original; b) HSV thresholded; c) seed points from the red segmentation; d) the final mask where both red and yellow seeds are used with region growing.

4.6.1 Colour thresholding

The colour thresholding starts by converting RGB images to HSV colour space. HSV values are normalized to a range between $0 \leq HSV \leq 255$. The HSV colour space is chosen because Hue colour channel is invariant to shadows and highlights, excluding specular reflections. The normalization of values is done to improve the speed and to make the values easier to handle.

The hue threshold value is considered priori. Saturation and value channels are used to specify and avoid the achromatic subspaces in HSV colour space. When the hue value of colour of the pixel in the input image with the specific colour range and the value is not in the achromatic area then the corresponding value in the output image is excluded from evaluation.

4.6.2 Seeded region growing

The output image of HSV segmentation is then divided into subimages to be used as a seed image for the region growing algorithm [64]. A seed is initiated if the amount of white pixels in the output image is above certain threshold. Region growing algorithm removes noise and other small objects.

Seeded region growing [64] performs a final segmentation of an image with respect to a set of points, known as seeds. The seeds are first grouped into n sets. It is in the choice of seeds that the decision of what is a feature of interest and what is irrelevant or noise is embedded. Given the seeds, seeded region growing then finds tessalation of image into regions with the property connecting each component of a

Algorithm 6: Seed image creation algorithm

input : I_{rgb} - RGB image
H_{min}, H_{max} - Hue thresholds
S_{min} - Saturation threshold
V_{min}, V_{max} - Value thresholds
S_s - Seed image area size
T_s - Seed threshold

output: Seed image I_{seed}

$I_{out} = 0, I_{seed} = 0$
Convert RGB image I_{rgb} into HSV colour space image I_{hsv}
Normalize all I_{hsv} channels between $0 \leq HSV \leq 255$
if $(H_{min} \leq H(i,j) \leq 255) \vee 0 \leq H(i,j) \leq H_{max}$ **then**
  $I_{out}(i,j) = 1$
if $S(i,j) < S_{min}$ **then**
  $I_{out}(i,j) = 0$
if $\neg(V_{min} \leq V(i,j) \leq V_{max})$ **then**
  $I_{out}(i,j) = 0$
Divide I_{out} into $S_s \times S_s$ sub-images I_{sub}
for $\forall I_{sub}$ **do**
  Calculate number of white pixels n_{white}
  if $n_{white} \geq T_s$ **then**
    Add white pixel into seed image I_{seed}

region to exactly one seed group. The region growing is presented as in Algorithm 7.

The process evolves inductively from seeds, namely, the initial state of the sets $A_1, A_2, \ldots, A_n$ [64]. Each step of the algorithm involves the addition of one pixel to one of the above sets at step m.

Let T be the set of all as-yet unallocated pixels bordering at least one of the regions

$$T = \{x \notin \bigcup_{i=1}^{n} A_i | N(x) \cap \bigcup_{i=1}^{n} A_i \neq \emptyset\} \tag{35}$$

where $N(x)$ is the set of immediate neighbours of the pixel x. It is common to use rectangular eight connected grids. If $x \in T$ and $N(x)$ meets just one of the A_i, then it can be defined $i(x) \in \{1, 2, \ldots, n\}$ to be index so that $N(x)\hat{A}_{i(x)} \neq \emptyset$ and to define $\phi(x)$ to be measure of how different x is from the region it adjoins. The simplest definition of $\phi(x)$ is

$$\phi(x) = |g(x) - \underset{x \in T}{\text{mean}}[g(x)]| \tag{36}$$

where $g(x)$ is the grey value of the image point x. If $N(x)$ meets two or more of the A_i is taken $i(x)$ to be value of i so that $N(x)$ meets A_i and $\phi(x)$ is minimized. Alternatively in this circumstance the pixel x can be classified as a border pixel and append it to the set B of already found border pixels.

Then $x \in T$ is taken as

$$\phi(x) = \min_{x \in T}\{\phi(x)\} \tag{37}$$

This completes step $m + 1$. The process is repeated until all pixels have been allocated. The process commences with each A_i being just one of the seed sets. Definitions in Equations 36 and 37 ensuring the final segmentation is into regions as homogenous as possible given the connectivity constraint.

Algorithm 7: Seeded region growing

input : I - Greyscale image
n - seed point groups $A_1, A_2, \ldots, A_n$ (or I_{seed} with labels)
ϕ - threshold for acceptance

output: I_{label} - Label image

Initialise label image $I_{label} = 0$

Put neighbours of initial seed points into P_{seed}

while $P_{seed} \neq \emptyset$ **do**
  $y = P_{seed}$
  Extract the neighbours y_{nhood} of y
  if y_{nhood} *higher than* ϕ **then**
    Add y_{nhood} to y_{grow}
  if $y_{grow} \notin I_l$ **then**
    Add y_{grow} to I_{label} with label of y
    Update the mean of corresponding region
    Add the new points to P_{seed}

4.7 Condition assessment and features

The purpose of condition analysis is to find out condition category the sign belongs to. The condition is analysed using two features, colour variance via k-means [49] and amount of edges inside the sign. Algorithm 8 describes the process on high level. The condition assessment algorithm combines many parts introduced before,

including colour spaces, colour normalization, standardization, and segmentation.

Algorithm 8: Condition assessment algorithm overview

input : I_{rgb} - Image containing traffic sign
BB_s - Bounding Box of traffic sign
S_r - Resize size
M - Classification model

output: S_{cond} - Sign condition (1-5)

Apply the Grey world colour constancy algorithm to image I_{rgb}
Crop the traffic sign I_{sign} from image using BB_s
Apply segmentation Algorithm 6 to I_{sign} to extract I_{seed}
Apply region growing Algorithm 7 to I_{sign} to extract I_{mask}
Resize image I_{sign} to size S_r
Convert I_{sign} image to HSV colourspace
Use HSV colour space for edge detection to get feature f_1
Create colour vector from signs 3 dimensional RGB pixels
Cluster colour vector using k-means, k is the number of colours
Calculate means of clusters to get f_2
Standardize the feature f_1 and f_2
Classify using KNN model M to get S_{cond}

4.7.1 K-means clustering

K-Means is a simple cluster analysis [71] algorithm. The goal of K-Means algorithm is to find the best division of n features into k groups, so that the distance between the group's members and the corresponding centroids, representatives of the group, is minimized. The difference between classification and clustering is that in clustering the classes of the features is unknown. Formally the goal is to partition the n entities into k sets in order to minimize the within-cluster sum of squares. K-means clustering is presented in Algorithm 9. The stopping criteria is normally that if

during one iteration no feature vector changes group, the execution stops.

Algorithm 9: K-means clustering

input : $\vec{x}_1, ..., \vec{x}_N$ - set of feature vectors

k - number of clusters the features are divided

output: $\vec{\omega_k}$ - label vector with k labels

Select k random seed $\vec{s_k}$ out of $\vec{x_1}, \dots, \vec{x_N}$

Set the cluster mean $\vec{\mu_k} = \vec{s_k}$

Set the label corresponding to $\vec{x}_1, ..., \vec{x}_N$ as empty $\omega_k = \emptyset$

while *Cluster means* $\vec{\mu_k}$ *change* **do**

  for $n = 1 \dots N$ **do**

    $j = \arg\min_{j'} |\vec{\mu_{j'}} - \vec{x_n}|$

    Reassign vector to $\omega_j = \omega_j \cup \vec{x_n}$

  Recompute centroids $\vec{\mu_k} = \frac{1}{|\omega_k|} \sum_{x \in \omega_k} \vec{x}$

4.7.2 Canny edge detection

Canny edge detector is a simple edge detector used for edge detection from grey-scale images [29]. The algorithm is more complex than just simple filtering. It contains hysterisis threshold where upper and lower thresholds are compared for uncertain edges. Canny recommended a upper:lower ratio between 2:1 and 3:1. The algorithm

itself consists of 5 high level steps as shown in Algorithm 10.

Algorithm 10: Canny edge detection algorithm

input : I - image

t_{min} - lower Canny threshold

t_{max} - upper Canny threshold

σ - variance for the Gaussian filter

output: E - black and white edge image

Create a Gaussian smoothing filter $G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$

Create edge filters $S_x = [1, 2, 1]^T \times [-1, 0, 1]$ and $S_y = S_x^T$

Smooth the image $I \leftarrow I \otimes G$

Get the edges $G_x \leftarrow I \otimes S_x$ and $G_y \leftarrow I \otimes S_y$

Compute the gradient magnitude $G \leftarrow \sqrt{{G_x}^2 + {G_y}^2}$

Compute the gradient direction $\theta \leftarrow \arctan(\frac{G_y}{G_x})$, and round to closest $\frac{\pi}{4}$ angle

Compute non maximal suppression $E \leftarrow$ nonmaximal_supression(E)

Apply hysteris threshold $E \leftarrow E > t_{max} \cup E > t_{max}(t_{min} < E < t_{max})$

5 EXPERIMENTS AND RESULTS

In this Section the used datasets, experiments and results are presented. Result presented for traffic sign distance evaluation, but not for full TSI process, because of the Ground Truth (GT) data available. The colour segmentation results are not presented separately, but in combination with condition analysis results.

5.1 Datasets and evaluations

Different kinds of data was used for experiments in TSD, TSC and sign condition analysis. Three datasets were used: the Swedish summer dataset for sign detection and classification, the Finnish winter roads dataset collected in the TrafficVision project for TSR performance assessment, and the Lappeenranta road sign's dataset collected also in the project for traffic sign condition analysis.

All performance evaluation are done using a laptop with Intel i5-4200 processor, with performance intensive code programmed using Streaming Single instruction stream multiple data Streams Extensions 2 (SSE2) extensions. Algorithms run on a single thread if not otherwise notified. The testing computer has 8 gigabytes of memory. The code for the project was written in Matlab and C.

5.1.1 Cross validation

The goal of the classification algorithm is to build a model which makes accurate predictions on the training set. Because of this, classifiers tend to perform very well on the data they were trained on, provided they have the power to fit the data. Training set accuracy is not a good indication how well the classification process will perform when classifying new data outside of the training set. Other measures are needed to give an idea of how accurate the classification will be when it will be deployed.

The cross-validation process provides a much more accurate picture of systems true accuracy. In cross-validation, the data is divided into a larger training set and a smaller validation set, then train on the training set and use the validation set to measure the accuracy. The training set is usually made larger if there is not enough data available to present all possible feature variations. To be a good measure of

accuracy, the validation data has to represent the range of inputs the classifier is likely to encounter. This has two important implications:

1. It is better to randomly select the validation examples from the existing collection of data. The validation set has to be diverse.
2. Accuracy and usefulness of the cross-validation process depends on having a data set representing the full range of possible inputs. For example, if work is being done on a machine vision application and samples have been gathered only under a very specific set of lighting conditions, cross-validation will not help to determine how well the system will perform under different lighting conditions.

There are different approaches to selecting the training and validation sets. One simple approach is to randomly select, e.g., 80% of the existing data is used for training and 20% to use for validation. There is a risk that by 'unlucky' coincidence the validation points contain a disproportionate number of difficult or obscure examples. To counter this, in the evaluation of the results k-fold cross validation is performed.

In k-fold cross validation the data is first randomly sorted and then divided into k folds, for example $k = 10$ would mean dividing the data into 10 parts. Then k rounds of cross validation is run. In each round, one of the folds is used for validation and the remaining for training. After training the classifier its accuracy is measured on the validation data. Mean accuracy over the k rounds is the cross validation accuracy.

5.1.2 Data format for experiments

The experiments were conducted using same data format, extended BB, for all the datasets. BB is the rectangle drawn around a single traffic sign in image by four corner points. Each traffic sign bounding box object has the following fields:

- lbl: a string label describing object type (eg: "mandatory").
- bb: [l t w h] BB indicating predicted object extent.
- occ: 0/1 value indicating if BB is occluded.
- bbv: [l t w h] BB indicating visible region (can be [0 0 0 0]).
- ign: 0/1 value indicating BB was marked as ignore.
- ang: [0-360] orientation of BB in degrees.
- class: [n] value indicating the signs class.

- condition: [g s c d] sign condition general, structural, condition, damage.

One text file holds the information of traffic signs of one image. The file ”001.txt” corresponding to the image ”001.jpg” containing two traffic signs would then contain two lines after storing:

- special 1630 845 580 559 0 0 0 0 0 0 0 511 3 5 3 5
- mandatory 1613 112 589 532 0 0 0 0 0 0 0 423 2 4 2 4

5.1.3 Dataset 1: Swedish summer dataset

Several public datasets are available for traffic signs detection and classification including Belgian [16], German [13] and Swedish [17] datasets. In the experiments the Swedish Summer [17] dataset is used, because of the similarities with Finnish traffic signs, including similarity in pictograms and colouring of the signs.

The Swedish Summer dataset provides 20 000 images from video sequences of which 20% have been annotated [17]. The dataset consists of continuous video sequences, recorded on a single tour and at the day. Accordingly, the same traffic sign appear repeatedly several times in the dataset. Lighting conditions and driving scenarios (rural, urban highway) have little variance. In the experiments the Swedish summer dataset is used for both TSD and TSC.

TSD and TSC use traffic sign identification numbers corresponding to the Finnish traffic signs. The identification 361 is a general identification for every speed limit, the corresponding speed is added after the number 361. Different identifiers are presented in Table 2.

For experiments the data is divided into two sets, the train set and the test set. The division is based on the division introduced by the original authors of the data. For the evaluation the annotations are converted into BB GT object format for processing. After removing ”undefined”, ”blurred misc signs”, ”urdbl”, ”other” and ”n/a ” signs statuses and ”urdbl” types there are total 6639 annotations left. Out of those ”blurred”, ”sideroad”, ”other” and ”occluded” are marked to be ignored because of the average low quality (the training with these is not meaningful). This removes 3447 from total signs.

Traffic sign annotations are not necessary squared or frontal. Since the correct rectification is not available, the annotations are aligned by stretching them into

Table 2. Different sign classes available in the Swedish summer dataset.

ID	Name	Sign category	Signs in category
211	Priority road	Priority	443
231	Give way	Priority	134
232	Stop and give way	Priority	4
371	No standing or parking	Prohibitory	76
372	No parking	Prohibitory	93
415	Allowed direction. Turn left or right	Mandatory	24
417	Pass this side. Left of right	Mandatory	309
36130	Speed limit 30	Prohibitory	20
36150	Speed limit 50	Prohibitory	145
36160	Speed limit 60	Prohibitory	17
36170	Speed limit 70	Prohibitory	196
36180	Speed limit 80	Prohibitory	103
36190	Speed limit 90	Prohibitory	27
361100	Speed limit 100	Prohibitory	91
361110	Speed limit 110	Prohibitory	25
361120	Speed limit 120	Prohibitory	28
Total			1725

squared model window in both TSD and TSC. A big problem with Dataset 1 is a large amount of the signs in are very small and the number of annotated signs in classes and overall number of classes is small. For TSD the size of the traffic signs is limited to 28×28 pixels or larger. The relative number of small annotation is illustrated by the histogram in Figure 18.

5.1.4 Dataset 2: Finnish winter dataset

Approximately 20 hours of video material was collected for system testing in difficult condition. The dataset contains videos in urban environment recorded from a van and from main roads collected from a road maintenance vehicle. The camera was chosen to be Garmin Virb Elite based on its specs and build in GPS. The camera Charge-Coupled Device (CCD) had a pixel resolution of 1980×1080 and framerate of 30 FPS. The camera has width of view of $151\,^{\circ}$. The exposure time for the video is controlled automatically based on average lighting of the view. The camera has also optical image stabilization. In both vehicles the camera is placed inside the drivers cabin. The lighting conditions are not selected but for the purpose of testing the system in different environments including low light conditions. The video material

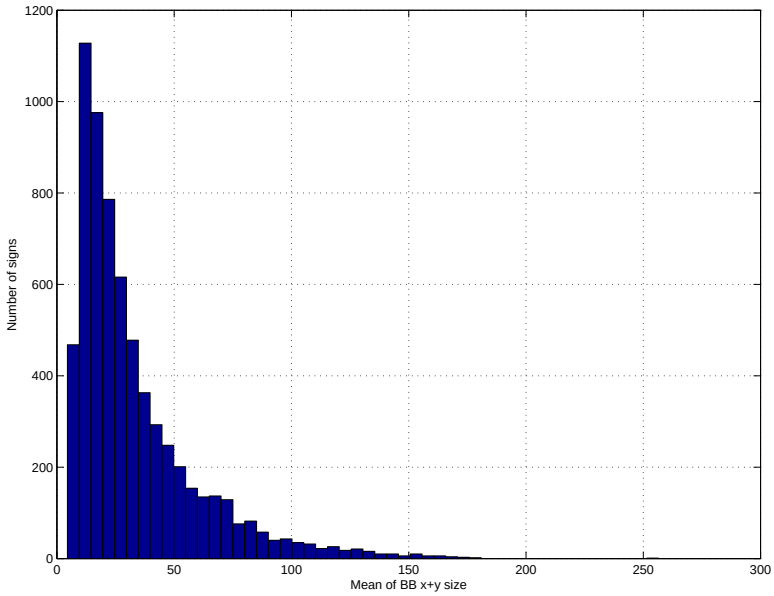


Figure 18. Histogram of BB size distribution of the 6639 traffic signs in the Swedish summer dataset. The size value used is mean of x + y dimensions.

includes GPS location data stored in GPS Exchange Format (GPX) for localization testing.

5.1.5 Dataset 3: Lappeenranta road signs dataset

The dataset 3, Lappeenranta road signs dataset consists of 325 still images for traffic sign condition analysis. The dataset contains 399 condition and class annotated traffic signs. This dataset is collected using an ordinary Canon digital camera from urban environment in good lighting conditions. The camera used has a CCD with spatial resolution of 3264×2448 pixels. The images are taken by foot and the goal was to collect traffic signs in as bad condition as can be found. There are also several excellent quality signs for reference. The condition dataset has been annotated by an independent expert who has annotated more than 100 000 traffic signs during his career.

The annotations contain BB around the traffic sign, the condition information based on three parameters, and sign class information. A single image can contain several traffic signs. In total the dataset contains 397 annotated traffic signs in different conditions. Cropped signs of this dataset is presented in Figure 19.

Figure 19. Traffic signs condition dataset shown cropped. Figure demonstrates the amount of variance in lighting conditions and colours as well as reflections.

5.2 Detection tests

To assess TSD performance two different tests are performed:

1. Dataset 1 is split into three categories: mandatory, priority and prohibitory. One detector was trained on each category. This approach is influenced by the literature[3].
2. Whole Dataset 1 is used to train one detector.

All four detector models are trained using similar setup and algorithms presented in Section 4. The training was performed in four rounds, with increasing numbers of weak learners (32, 128, 512 and 2048). Each round adds 3640 hard negative windows (images of the background). It is expected that the traffic signs are frontal in relation to the camera, no other aspect ratios were used. The parameter for HOG and feature pyramid can be found in Tables 3 and 4 shows the AdaBoost model parameters. The parameters are chosen based on the literature [3].

Table 3. HOG, colour features, and feature pyramid parameters for TSD.

Name	Explanation	Value
modelDs	model height+width without padding	50×50
modelDsPad	model height+width with padding	56×56
nPerOct	number of scales per octave	8
nOctUp	number of upsampled octaves to compute	0
smooth	radius for channel smoothing	1
concat	if true concatenate channels	true
cEnabled	if true enable CIE LUV colour channels	true
gradMag	if true enable gradient magnitude channel	true
normRad	normalization radius for gradient m	5
normConst	normalization constant for gradient m	0.005
HOGEnabled	if true enable gradient histogram channels	true
nOrients	number of orientation channels	6
clipHog	value at which to clip hog histogram bins	0.2

Table 4. AdaBoost parameters for the TSD.

Name	Explanation	Value
X0	negative feature vectors	[N0xF]
X1	positive feature vectors	[N1xF]
nBins	maximum number of quantization bins	256
maxDepth	maximum depth of trees	1
minWeight	minimum sample weight to allow split	0.01
fracFtrs	fraction of features to sample for each node split	1
nWeak	number of trees to learn	32, 128, 512, 2048
discrete	train Discrete-AdaBoost	true

Assessing the performance of traffic signs detection algorithm is not simple. The detectors are limited by near real time constraints needed by the common applications. Other commonly evaluation criteria include the amount of true positives(successful detections), number of false negatives (signs not detected at all), false positives (detections not really signs), and the overlap between the GT BB and detection BB (also known as Area Under Curve (AUC)). The performance of the detection is measured using two main indicators False Positives Per Image (FPPI) and miss rate. FPPI is the amount of detections that are not traffic signs compared to the total number of images. Miss rate describes what percentage of the real signs are missed (true negatives) during the detection and is calculated simply by 1 - the true positive rate. The evaluation of the experiments are performed per image (not per window). In the tests the detection is considered to be successful when the overlap

of detection BB is more than 0.8 of the GT BB area. This is standard practice in detection and 0.8 is considered a high overlap rate [3].

Because there are very few false negatives, as presented in Table 5 the simple detection performance is not enough. Figure 20 shows Receiver Operating Characteristic (ROC) curve where false positives are compared to the amount of false negatives. Because the results are too good to compared on normal scale the Figure 21 shows ROC curves comparing FPPI to miss rate on a logarithmic scale. The mean miss-rates reported in the legend of Figure 21 is based on 200 samples of curve (in the range 0 to 1 FPPI, evenly spaced log space). The false positive and 50 strongest negative windows for the three category detectors are presented in Figures 22, 23, and 24.

Table 5. The false negatives in detection using dataset 1 and three separate detectors.

Type	Mandatory	Priority	Prohibitory	Total
Train	379	666	1075	2120
Test	515	830	936	2281
False negative	3	12	4	19
Relative error	0.006	0.014	0.0042	

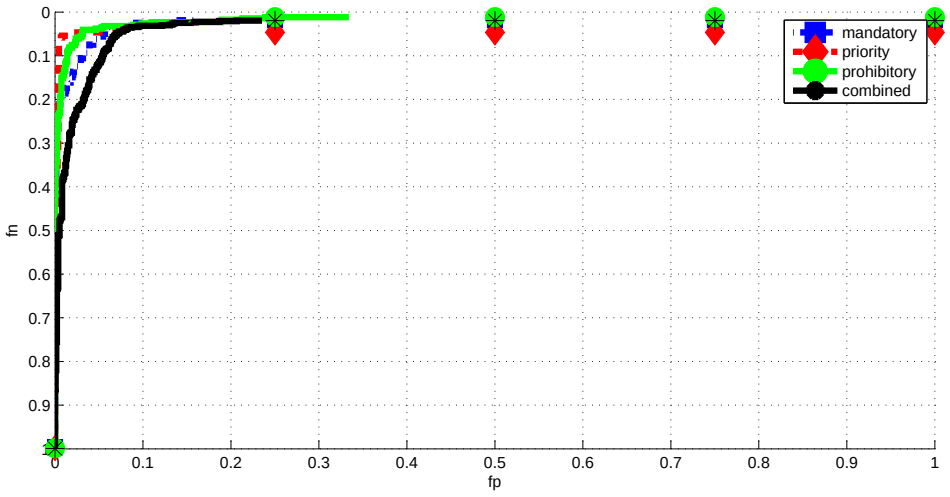


Figure 20. Traditional ROC curve for category detectors and combined detector. Number of false positives against number of false negatives.

The detector training time for mandatory category was 416 seconds, priority category 446 seconds and for prohibitory 416 category seconds. The detector described

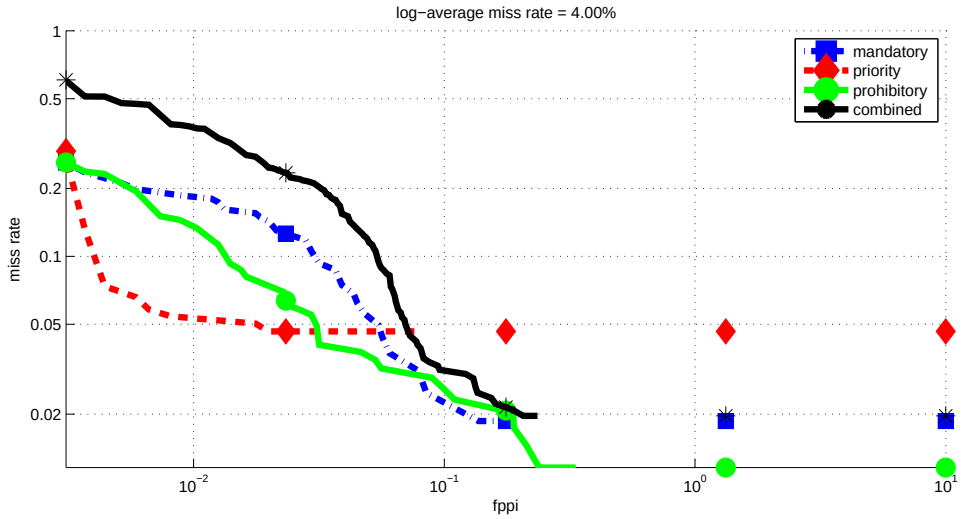
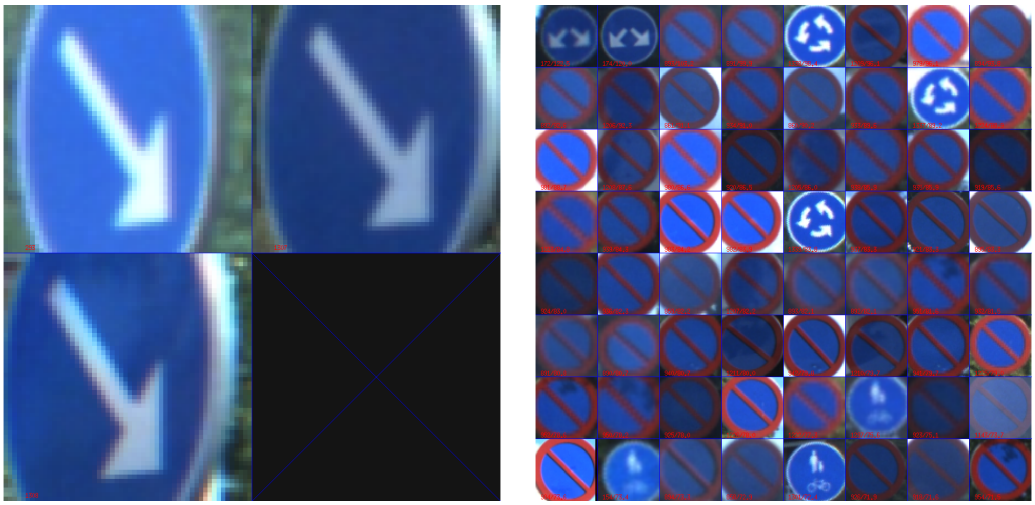


Figure 21. ROC curves for category detectors and combined detector. Logarithmic miss rate compared to FPPI value computed over category detectors.

above runs on average 15 FPS on whole images from Swedish Summer dataset with resolution of 1260×960 pixels (0.06 seconds per image).

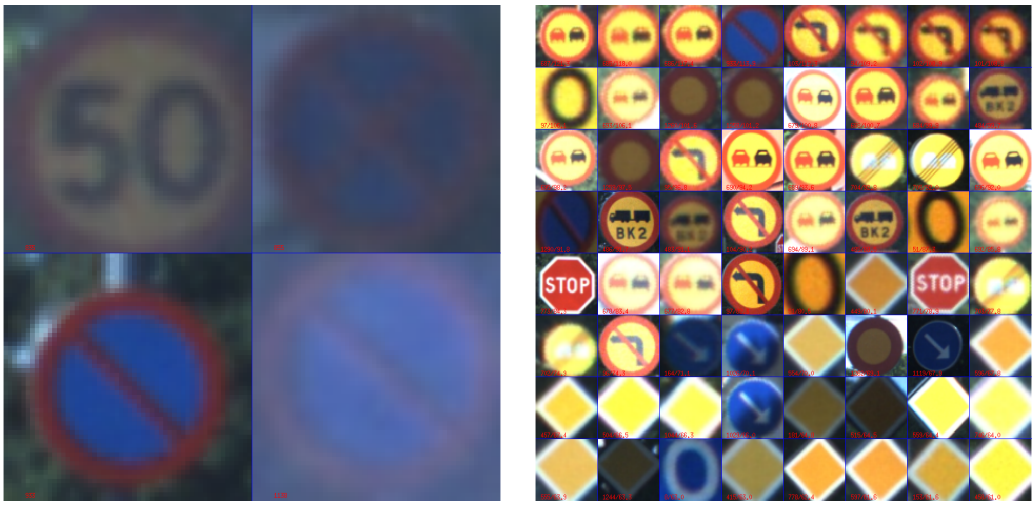


a) **b)**

Figure 22. Mandatory sign category detector results: a) True negatives; b) False positives.

a) b)

Figure 23. Priority sign category detector results: a) True negatives; b) False positives.



a) b)

Figure 24. Prohibitory sign category detector results: a) True negatives; b) False positives.

5.3 Classification tests

The evaluation of the TSC was done using Dataset 1. The evaluation of the classification is easy, compare the classification results to the GT and see if there is a classification error. The dataset a provides 16 different signs classes for classification. The performance is evaluated using two different features: HOG and grey-scale, two different dimension reduction techniques: LDA and PCA and two different classifiers: KNN and Random Forests. The parameters for methods are presented in Tables 6 and 7. KNN uses $k = 5$ (selected by experimenting) and PCA 300 most

significant principal components explaining over 95% of the variance. The classification performance is evaluated by comparing the classifier predictions to ground truth. When obtaining results the maxDepth parameter of random forest had to be limited below 64 to be able to fit the models into memory.

Table 6. HOG feature parameters for classification.

Name	Explanation	Value
I	greyscale input image	28×28
binSize	spatial bin size	8
nOrients	number of orientation bins	9
clip	value at which to clip histogram bins	0.2
crop	crop boundaries	false

Table 7. Random Forest parameters for classification.

Name	Explanation	Value
M	Trees to train	30
H	Classes amount	based on data
N1	Datapoint for training each tree	5*sample number/M
F1	Features to sample for each split	sqrt(F)
split	Splitting criteria	gini
minCount	Minimum of data points to allow split	1
minChild	Minimum datapoint allowed at child nodes	1
maxDepth	Maximum depth of three	64

Using 10-fold cross validation one training fold contains 1741 samples and one testing fold 165 samples. The classification results with different algorithms are presented in Table 8. Feature column is the feature used, dimension reduction is the dimension reduction technique, "Classifier" is the used classifier and the μ is the mean classification error from 10-fold cross validation. The errors of the highest ranking combination HOG+LDA+KNN are shown in Figure 25 as confusion matrix. The computation times of different algorithms are presented in Table 9.

Table 8. Classification results for different classifier, features and dimension reduction techniques.

Feature	Dimension reduction	Classifier	μ error
HOG	LDA	KNN	0.0146
HOG	PCA	KNN	0.0295
HOG	LDA	Random Forest	0.0493
HOG	PCA	Random forest	0.1215
HOG	none	Random Forest	0.0424
Grey Image	none	Random Forest	0.1387
Grey Image	PCA	KNN	0.1957
Grey Image	LDA	KNN	0.1251

Table 9. Dimension reduction times for different dimension reduction techniques and features. HOG vector contains 1568 dimensions and grey feature contains 784 dimensions. PCA reduces dimensions to 200 long and the LDA to number of classes - 1.

Feature	PCA time	LDA time
HOG	3.7397	1.4178
Grey Image	0.5689	0.1269

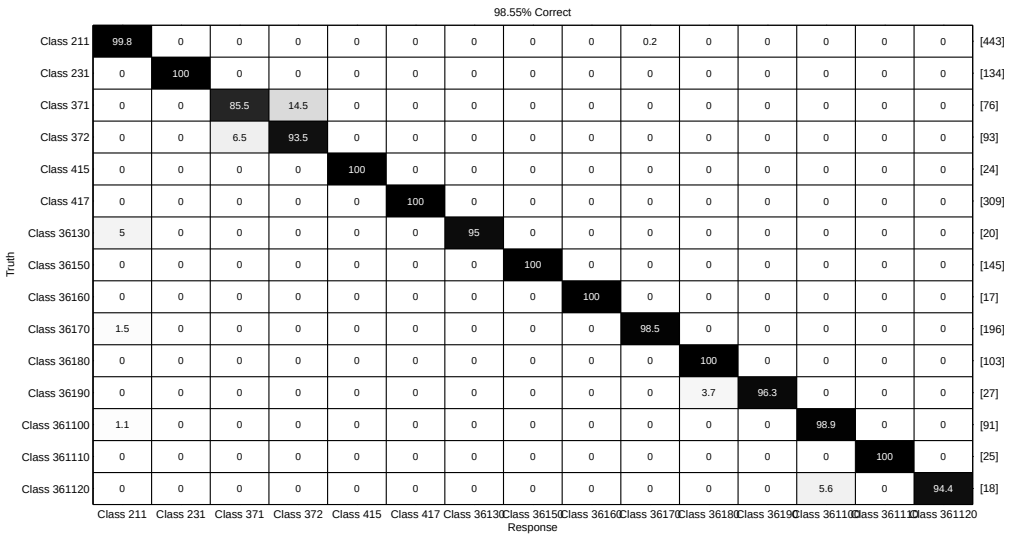


Figure 25. Confusion matrix showing correctly classified percentage. The values are taken from 10-fold cross-validation results over all 10 rounds with the corresponding percentages computed. The right side shows the sample amount in each GT class.

5.4 Distance evaluation

During the collection of data no GT data for the assessment of localization accuracy was collected. The distance estimation to the sign is one of the error prone parts of traffic sign localization, and the GT data can be made easily available. The distance estimation accuracy is evaluated by comparing 6 images with GT measured using laser distance meter and values obtained using Equation 7. The laser distance meter is very accurate, but the place where the image is taken is not exactly the same as lasers. Pictures are taken using the Garmin Virb presented before. The camera producer does not give exact focal length or width of view parameters for camera, and they are approximated using calibration. Camera performs lens correction for images automatically. The BBs for calculating heights are placed by hand and the average of height and width is computed for projection. All the tested signs are 640 mm wide. The results are presented in Table 10.

Table 10. Distance of signs measured with laser distance meter and approximated with machine vision. All values are in meters.

Image	Laser measurement	Machine vision	Difference	error %
1	23.15	23.68	-0.53	-2.2
2	25.63	26.28	-0.65	-2.5
3	14.27	14.46	-0.18	-0.9
4	08.11	07.99	0.12	1.4
5	09.46	10.06	-1.05	-6.2
6	16.26	16.27	-0.01	-0.1
μ	16.15	16.46	-0.31	-1.9

5.5 Condition analysis

Dataset 3 was used for the evaluation of condition analysis. The traffic sign condition was evaluated using the collected GT conditions. The sign GT condition category is thought to be a class dividing the signs into five separate classes based on the condition category. Condition assessment algorithm uses two features: the amount of variance within a sign colour and the amount of edges in the sign.

Figure 26 of the two features against each other and the condition categories. The parameters for segmentation are in Table 11. The parameters for condition analysis

itself are presented in the Table 12. For both phases the parameters were selected by hand, because of there is no GT available for the traffic sign segmentation.

Table 11. Segmentation algorithm parameters.

Name	Explanation	Value
C_{space}	colourspace	HSV
V_{tresh}	Value V threshold	$V > 30$
S_{tresh}	Saturation S threshold	$S > 120$
HR_{tresh}	Red H threshold	$10 > H > 240$
HY_{tresh}	Yellow H threshold	$23 < H < 35$
HB_{tresh}	Blue H threshold	$143 < H < 155$
SCS	Seed image cell size	17
Acc_{tresh}	Seed image acceptance threshold	60
RG_{tresh}	Region growing threshold	40

Table 12. Condition analysis parameters.

Name	Explanation	Value
CCAlg	colour constancy algorithm	Greyworld
I_{size}	Evaluation image size	400×400
$Canny_{tresh}$	Canny threshold	Relative to the highest gradient
$G(\mu, \sigma)$	Canny Gaussian sigma and variance	2, 1
$K_{clusters}$	Number of K-means centers	Relative to signs
$K_{neighbours}$	Number of KNN neighbour	5

The condition evaluation results are evaluated in two ways: correctly classified and amount of error. The amount of error is used because in the reality the traffic signs condition are continuous, not discrete as presented by the five categories. This metrics error is the absolute difference of distances between test and GT vector. After the difference of absolute distances the mean is calculated to tell the error relative to each sign. Using this metrics the classification result is better the closer it is to zero.

The mean error based on 10-fold cross validation based on the mean of absolute distances is 0.583 when using all the data from the condition analysis dataset. The error is not deterministic because of the random C-means clustering initiation, but the results shows consistency. Segmentation computation time for 397 signs was 87.2

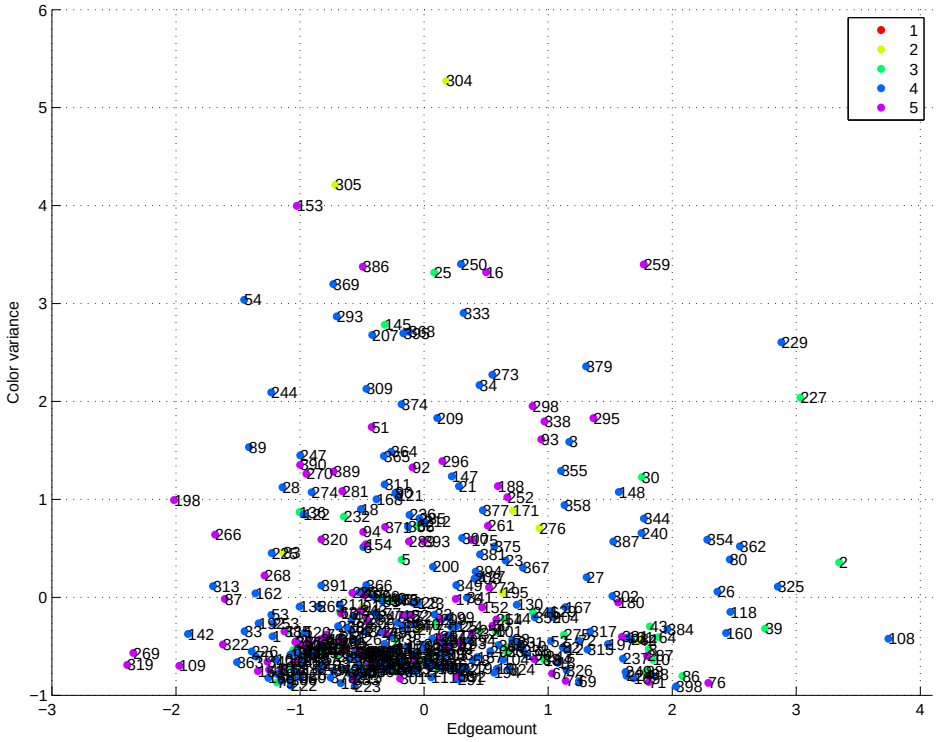


Figure 26. Condition assessment features computed from Dataset 3. Features are standardized to $\mu = 0$ and variance $\sigma = 1$. The numbers correspond to the image numbers in the dataset.

seconds (0.22 second per sign), feature computation time for same signs 98.6 seconds (0.24 seconds per sign), and K-fold cross validation using KNN in 0.12 seconds for all the features and folds. The confusion matrix for direct class comparison is presented in Figure 27. The condition category 1 had to be omitted from the test because there was only one sample of the class.

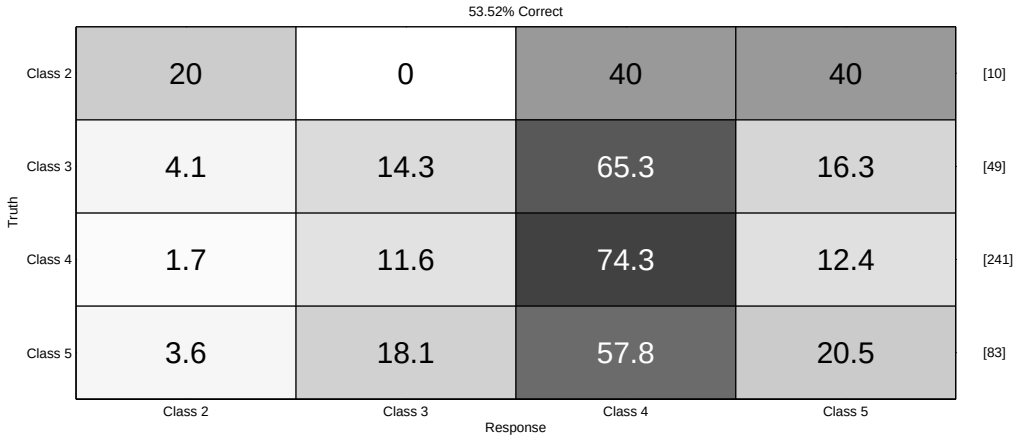


Figure 27. Confusion matrix showing incorrectly classified conditions of traffic signs by class. The values are taken from 10-fold cross-validation results over all 10 rounds with the corresponding percentages computed. The right side shows the amount of samples in each GT class.

6 DISCUSSION

This chapter discusses the obtained results are discussed. The system implementation is presented shortly. The limitation and possible problems with the results and the whole system are discussed. Finally, thoughts for the future research topics and the future of the system in general are presented.

6.1 System implementation

In a complete system, detection, classification, and condition analysis are expected to run jointly. Figure 28 shows the performance of the current system results. The classification, feature extraction times are not shown separately for TSD and TSC because the times are very small. The performance indicators for the Figure are presented in Section 5. The classification experiments expected well aligned detection BB. The detection algorithm localization accuracy is good, compared to Dataset 1. Unfortunately system performance could not be verified on really difficult conditions. To evaluate the performance and test difficult conditions (fog, snowfall, lowlight, and rain) effect on the performance a new dataset is needed.

The system inputs video and GPX files and returns as output the sign location GPS coordinates and class. Every 10th frame in the 30fps video is processed, because of performance limitations. This way, the system runs in real time even when computing detection on the full frames. There were difficulties adding the tracking into the system because the sparsity of the detection results. The similar sign locations are combined trough non-maximal suppression, but more advanced tracking and correspondence problem solving would be better solution. The system implementation returns the inventory results in GeoJSON format, the results can be shown and verified on a map.

6.2 Traffic sign inventory

The low spatial resolution impairs the discrimination of the details such as a single digit on a speed limit signs or the icons on danger signs. The detection is possible even when the signs are far away, but the information is not usable because the class of the sign or sign condition can not be determined. When the work was started,

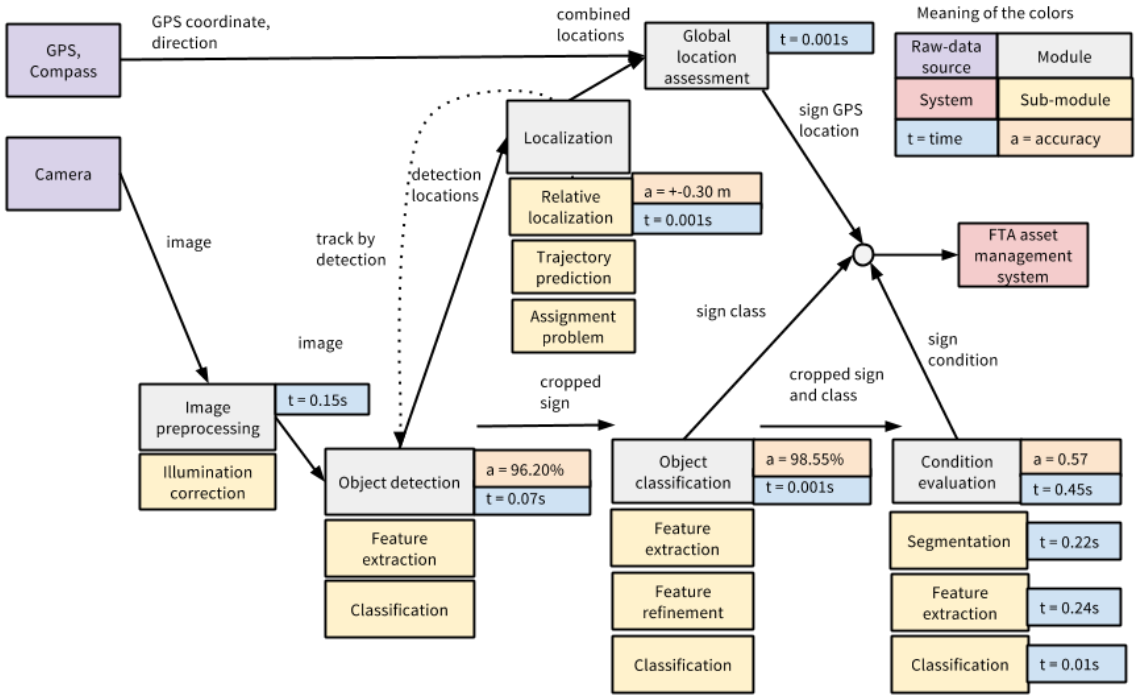


Figure 28. The relevant performance indicators of the best performing methods of TSI system presented in this thesis.

Dataset 1 looked reasonably big. In practice the training consecutive images do not contribute to diversity of Dataset 1 because they are often very similar to each others and often cause an undesired imbalance of dependant images. Another problem is the incorrectly classified samples in the datasets' GT data.

The localization of BBs in TSD determines the success of the accurate localization into GPS coordinates. One of the big contributors to localization error, distance evaluation, was experimented on with a limited dataset. GT data about the real locations of traffic signs would be needed for a more conclusive experimentation. The detector gets several hits for one traffic sign was not taken into account. More accurate localization distance assessment could be derived by combining both methods, the triangle similarity and the point when a sign reaches edge of the image.

6.2.1 Detection

The results from detection shows peculiarities with Dataset 1. Few detections are evaluated as false positives because incorrectly labeled, or missing data. When looking at false negatives, it seems that the signs are localized accurately, though set

contains inaccurately labeled data samples. The priority signs are badly localized, and it can be inferred that the training has failed. In some cases [3], detection is performed using different aspect ratios in finding signs, but in the TSI it is unnecessary. The signs that are not frontal (such as side roads) are not relevant for TSI or driver assistance. The AdaBoost is able to learn wider range of values than just a single sign category. The results are very similar for three separate detectors and the single detector, even with same number of weak learners.

In the traffic sign inventory, false negatives are worse than missing true positives detections. As the vehicle passes multiple shots are obtained from a single traffic sign, and it is not necessary to detect every sign in one image. False positives could be detected again in classification phase, but the simplest solution would be to directly get only true positives from the detector. The threshold for acceptance should be adjusted correctly. In real life situation, as weather and lighting conditions change, the threshold or even the detector and the classifier has to be adjusted to the environment. The performance can be improved using more tightly computed feature pyramid, but in the system of the thesis, the parameters are optimized for both performance and speed.

6.2.2 Classification

The classification results are good. It would be interesting to compare the posterior probabilities of the incorrect classifications to see how certain the classification results are compared to correct classifications. This would require use of a different classification method. From an application point of view, processing time and memory are important aspect then choosing classifier, and the simple KNN was selected. The memory requirements were not experimented on. It is notable how well LDA+KNN combination, a very simple and computationally inexpensive classifier performs, in comparison to more complex approaches like random forest.

LDA is affected by imbalance, the unequal presentation of classes, in Dataset 1. The random forest classifier should not theoretically suffer from this unbalance. Each decision tree in the forest is trained on a different, random sample of the training data. Therefore, the class distribution in this sample can be very different from the overall dataset. Unfortunately the experiments were unable to confirm this, probably due to bad parameters in tree training. The errors are similar using KNN and random forests. The two priority road signs are miss categorized because of the

wrong labels in GT data. The results in both TSD and TSC are in line with results found from literature.

6.3 Condition analysis

Segmentation failed for all the signs in the ”end of speed limit area” class, because of the grey background colour of the sign. The result of vector distance metric seem better than they actually are. In Dataset 3, the condition categories are not balanced evenly and the KNN select the most probable class. The features work as intended, but the evaluation criteria parameters (such as the exclusion of vegetation growth) are problematic. This is demonstrated in Figure 29, where signs get high response from the number of edges, but are categorized as being in good condition. There are few outliers where the segmentation failed (5 in total using Dataset 3), but majority is segmented correctly even without dynamic threshold that could improve thresholding performance. Outliers in the results were studied by hand, and the clear outliers in features are explainable trough failed segmentation. The k-means clustering does not always converge similarly because of the randomized seed, there is small variance in the results.

a) **b)** **c)**

Figure 29. Disparity between the method and condition criteria: a) Image 108, condition category 4; b) Image 2, condition category 3; c) Image 227, condition category 3. Numbers correspond with the numbers in Figure 26.

6.4 Limitations

The system itself would benefit from a more complicated program structure. Parameters and methods would be more easy to optimize trough grid search and clearly defined performance indicators. The current system implementation is not mobile and the data transfer from the camera to a computer is not straightforward. This problem can be solved implementing the system on a mobile device. Then the data can be easily transfered trough network connection.

One of the limiting factors is the size of Dataset 1. Larger, automatically collected and maybe manually labeled dataset would allow more comprehensive testing and also provide material for different environmental conditions. It is not perfectly clear how well the methods could cope with different conditions. Error in distance alone does not give the exact localization error, but gives an estimate of the amount of the error. Sign GPS GT location data would be needed to perform exact evaluation of the localization.

If the sign is in very bad condition, it is harder to detect or classify. The methods for localization rely on the size of the signs being 640 mm wide or tall. There are signs on the road sides that do not belong there and currently the implementation does not contain methods to asses or evaluate them. The sign is always classified into one of the trained classes. This can be fixed by using a classifier producing a posterior probability.

6.5 Future research

The whole inventory and condition analysis process could be improved by incorporating environment specific information. For example, it is possible to add knowledge of the location of the road to improved scene understanding. For the paradigm of selective search for object recognition is an alluring option for TSD. The selection of areas to be searched by the detector could be based on the dynamic colour thresholding. The separability of the problematic blue traffic sign colour could be improved by creating a model of the scene and the environment. Sky, road, and other areas could be segmented into different areas. One of the problems in the condition analysis is the amount of free parameters. Automatic method for parameter validation should be developed.

As the signs get closer to the camera, the image becomes more accurate. When the sign is too close, the relative motion of the sign in the camera view port is too fast, and the signs gets blurry. There are several methods to evaluate information content of the image, combine several shots into one higher quality image and reduce motion blur, if the motion between the frames is known. This information is currently unused, but in future it can be used to improve accuracy.

As part of the bigger picture the goal of getting the whole inventory and condition analysis process to work mobile equipment seems oblivious next step. The use of mobile phones as equipment instead of expensive computation equipment is made possible with both respect to mobile computation power and camera performance in the last few years. The use of camera with GPS separate from the computing equipment has caused unnecessary complication such as the requirement to interpolate between GPS coordinates and not be able to control camera exposure time and refresh rate. If mobile environment would be used, practical optimization is easier because increased control over the platform, such as camera aperture and image exposure time.

For road equipment inventory and maintenance, the inventory of the traffic signs and the condition of the surface of the sign is not the only possibility. For example, the angle of sign posts and the inventory of the traffic sign posts are problems that could be studied and have relevance in the context of road maintenance. In addition to the methods presented here, the problem of signs posts requires text recognition and the size of the signs is not constrained by the current 640 mm assumption. Colour segmentation performance for detection would be interesting, and currently untested approach against the bigger datasets and varying conditions.

Many of the important parts of the system are now in place. The data formats have been defined, the possible problems with the algorithms identified, methods for evaluation solidified, and future improvement directions outlined. One of the important gains for the future is that there is now a baseline that new methods can be compared to and if there is improvement, the advances can be included into the system easily. A single currently lacking area is the accurate localization data for signs. Data could be produced by recording stationary GPS coordinates the signs in the roads and collecting video of the same signs from a moving vehicle. The stationary coordinates could then be used as GT for evaluation.

Future research directions can be summarized as follows:

- **Selective search in object detection**: The speed and accuracy can be improved by integrating more case-specific information cues, such as colour, movement, and visual saliency. The detector could intelligently choose how fast and from where it would want to get the frames to maximize the efficiency. This would eventually lead to rejecting the whole feature pyramid idea.
- **Better and faster feature extractors**: The HOG features are not the only possibility. Automatically parametrized Gabor filters are a possibility to improve feature extraction. The feature pyramids could be made faster to create and compute in Fourier frequency space, and if the feature model can be tested in frequency space, the speed improvement would be by an order of magnitude. To distinguish similar classes, better the maximal spread of colour features could be studied. The current grey conversion in classification is not optimal in the sense of class separability.
- **Traffic sign post condition and content evaluation**: Much is shared between traffic signs and sign post, but there are still unanswered questions, such as how to recognize letters and compute the distance to object that are not always the same size. These should be studied to extend the system to include sign posts location, content, and condition.
- **Tighter integration of processes and mobile platform**: The overall process of traffic sign inventory and condition analysis could be tighter coupled for performance gains. For example, features computed for detection can be used in recognition and condition analysis. Streamlining the process would be possible to run the whole process on the fly with mobile phone equipment. This would require rethinking the data flow inside and between the algorithms.
- **Better evaluation metrics and datasets**: The evaluation metrics and datasets are not perfect, and there are several problems with the existing one, such as the diversity of the data. Better and reasonable evaluation metrics should be created to evaluate the results in a standardized way. A good example for traffic signs inventory would be collection of video material with GPS locations, that can be used to assess the inventory performance and total location error.
- **More concise condition analysis**: The future research to condition analysis could focus on statistical versus feature based methods comparison. Parameter values and their effects should be studied for more accurate assessment of traffic sign condition and additional features should be added to give the model more descriptive power. The invariance of the features to different condition changes should also be ensured. The current implementation does not make it possible to assess the bleaching of the colours, but it can be added easily. The condition

estimation could be based on physical measurements. This would reduce the effect of human annotator performance to the results.

- **Synthetic data**: Currently, the TSD and TSC methods rely on large amount of traffic sign images in different environmental conditions to model the dynamic environment. If the different environment can be modeled on the sign model images (as shown in Figure 3), it would remove the need for large training dataset. Therefore, it would be possible to create superior methods that are not limited by the amount and diversity of labeled training data. It might be possible to use the same approach for traffic sign condition analysis.

7 CONCLUSION

The goals of this research was set to evaluate the robustness of TSD and TSC, and to study automatic location information assessment. The results from these three modules form a core of a TSI system. First, the problem was reviewed and a picture of a process with sub-problems were outlined. The sub-problems were studied further trough the literature and possible solution to each of them were proposed. The solution studied further, compared, and algorithms were implemented to solve the problems. Finally, the algorithms were tested against three different datasets. Two of the datasets were collected during the TrafficVision project, one for condition analysis and one for the localization assessment.

This research is the first step in automating and combining traffic sign condition analysis with TSI and in reducing road maintenance costs in Finland. TSD and TSC are actively researched topics. The methods are not usually optimized for traffic signs or roads as the environments. This would make further research into topic interesting. In general, there is no previous research available for the localization of the traffic signs or global location assessment to GPS coordinates. The condition analysis of traffic signs has not been researched before, in exception of automatic reflectance assessment, and in this sense this thesis has novelty value.

The machine vision is ready for implementation a TSI system for automatic asset management. The TSD phase of this thesis uses rigid, HOG+colour feature detector. The detector reaches performance of 96.00% and runs around 15 FPS. The best results for TSC were obtained using HOG+LDA+KNN combination classifying 98.55% of the signs correctly. When the TSD and TSC results are combined with information between multiple frames can be results be further improved. The automatic condition analysis results look good, but still more research is required to estimate the robustness of condition analysis, especially against human performance. The current condition analysis phase per sign mean error of 0.583.

This thesis also evaluated many practical aspects, such as camera, data formats, and the environments' effect on the TSI. The process can be further improved by adding more environment and traffic signs specific information. The proposed system shows promising results, and the implementation of corresponding machine vision solution is feasible.

References

[1] M. Piispanen and H. Lappalainen, Eds., *Liikennemerkkien kuntoluokitus*, Finnish, 978-952-221-256-6 2200060-v-09, Helsinki: Tiehallinto, 2009. [Online]. Available: www.tiehallinto.fi/julkaisut.

[2] F. T. A. (LIVI), *Finnish Transport Agency > contact information*, 2014. [Online]. Available: http://portal.liikennevirasto.fi/sivu/www/e/fta/contact_information.

[3] M. Mathias, R. Timofte, R. Benenson, and L. J. V. Gool, "Traffic sign recognition - how far are we from the solution?" In *IEEE International Joint Conference on Neural Networks (IJCNN)*, 2013, pp. 1–8.

[4] S. Houben, J. Stallkamp, J. Salmen, M. Schlipsing, and C. Igel, "Detection of traffic signs in real-world images: the German traffic sign detection benchmark," in *IEEE International Joint Conference on Neural Networks (IJCNN)*, 2013.

[5] A. Mogelmose, M. Trivedi, and T. Moeslund, "Vision-based traffic sign detection and analysis for intelligent driver assistance systems: perspectives and survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 4, pp. 1484–1497, 2012, ISSN: 1524-9050. DOI: 10.1109/TITS.2012.2209421.

[6] A. Gonzalez, M. Garrido, D. Llorca, M. Gavilan, J. Fernandez, P. Alcantarilla, I. Parra, F. Herranz, L. Bergasa, M. Sotelo, and P. Revenga de Toro, "Automatic traffic signs and panels inspection system using computer vision," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 2, pp. 485–499, 2011, ISSN: 1524-9050. DOI: 10.1109/TITS.2010.2098029.

[7] P. Velhonoja and M. Karhunen, *Yleisohjeet liikennerkkien käytöstä*, Finnish, 951-726-979-X. Helsinki: Tiehallinto, 2003. [Online]. Available: www.tiehallinto.fi/julkaisut.

[8] S. Houben, "A single target voting scheme for traffic sign detection," in *IEEE Intelligent Vehicles Symposium (IV)*, 2011, pp. 124–129. DOI: 10.1109/IVS.2011.5940429.

[9] H. Fleyeh, "Color detection and segmentation for road and traffic signs," in *IEEE Conference on Cybernetics and Intelligent Systems*, vol. 2, 2004, pp. 809–814. DOI: 10.1109/ICCIS.2004.1460692.

[10] P. Viola and M. J. Jones, "Robust real-time face detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, 2004, ISSN: 0920-5691. DOI: 10.1023/B:VISI.0000013087.49260.fb.

[11] A. de la Escalera, L. Moreno, M. Salichs, and J. Armingol, "Road traffic sign detection and classification," *IEEE Transactions on Industrial Electronics*, vol. 44, no. 6, pp. 848–859, 1997, ISSN: 0278-0046. DOI: 10.1109/41.649946.

[12] H. Fleyeh, "Shadow and highlight invariant colour segmentation algorithm for traffic signs," in *IEEE Conference on Cybernetics and Intelligent Systems (CCIS)*, 2006, pp. 1–7. DOI: 10.1109/ICCIS.2006.252225.

[13] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "Man vs. computer: benchmarking machine learning algorithms for traffic sign recognition," *Neural Networks*, 2012, ISSN: 0893-6080. DOI: 10.1016/j.neunet.2012.02.016.

[14] C. Bahlmann, Y. Zhu, V. Ramesh, M. Pellkofer, and T. Koehler, "A system for traffic sign detection, tracking, and recognition using color, shape, and motion information," in *IEEE Intelligent Vehicles Symposium (IV)*, 2005, pp. 255–260. DOI: 10.1109/IVS.2005.1505111.

[15] X. Baro, S. Escalera, J. Vitria, O. Pujol, and P. Radeva, "Traffic sign recognition using evolutionary adaboost detection and forest-ecoc classification," *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 1, pp. 113–126, 2009, ISSN: 1524-9050. DOI: 10.1109/TITS.2008.2011702.

[16] R. Timofte, K. Zimmermann, and L. Van Gool, "Multi-view traffic sign detection, recognition, and 3d localisation," *Machine Vision and Applications*, 2011. DOI: 10.1007/s00138-011-0391-3.

[17] F. Larsson, M. Felsberg, and P.-E. Forssen, "Correlating Fourier descriptors of local patches for road sign recognition," *IET Computer Vision*, vol. 5, no. 4, pp. 244–254, 2011.

[18] G. B. Foreign and C. Office, *Convention on Road Signs and Signals, Vienna, 8 November 1968: Presented to Parliament by the Secretary of State for Foreign and Commonwealth Affairs*, ser. Miscellaneous No.16(1969) Series. Stationery Office, 1969, ISBN: 9780101413909.

[19] Wikipedia, *Road signs in Sweden - Wikipedia, the free encyclopedia*, [Online; accessed 07-Jan-2014], 2014. [Online]. Available: http://en.wikipedia.org/wiki/Road:signs:in:Sweden.

[20] H. Kong, J.-Y. Audibert, and J. Ponce, "General road detection from a single image," *IEEE Transactions on Image Processing*, vol. 19, no. 8, pp. 2211–2220, 2010, ISSN: 1057-7149. DOI: 10.1109/TIP.2010.2045715.

[21] Y. Gu, T. Yendo, M. Tehrani, T. Fujii, and M. Tanimoto, "Traffic sign detection in dual-focal active camera system," in *IEEE Intelligent Vehicles Symposium (IV)*, 2011, pp. 1054–1059. DOI: 10.1109/IVS.2011.5940513.

[22] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, IEEE, 2005, 886–893 vol. 1. DOI: 10.1109/CVPR.2005.177.

[23] E. R. Davies, *Computer and machine vision theory, algorithms, practicalities.* Waltham, Mass.: Elsevier, 2012, ISBN: 9780123869081 0123869080 9780123869913 0123869919.

[24] A. Gijsenij, T. Gevers, and J. van de Weijer, "Computational color constancy: survey and experiments," *IEEE Transactions on Image Processing*, vol. 20, no. 9, pp. 2475–2489, 2011, ISSN: 1057-7149. DOI: 10.1109/TIP.2011.2118224.

[25] S. D. Hordley, "Scene illuminant estimation: past, present, and future," *Color Research and Application*, vol. 31, no. 4, pp. 303–314, 2006, ISSN: 1520-6378. DOI: 10.1002/col.20226.

[26] G. Buchsbaum, "A spatial processor model for object colour perception," *Journal of the Franklin Institute*, vol. 310, no. 1, pp. 1–26, 1980, ISSN: 0016-0032. DOI: http://dx.doi.org/10.1016/0016-0032(80)90058-7.

[27] J. van de Weijer and T. Gevers, "Color constancy based on the grey-edge hypothesis," in *IEEE International Conference on Image Processing (ICIP)*, vol. 2, IEEE, 2005, pp. 722–725. DOI: 10.1109/ICIP.2005.1530157.

[28] R. Benenson, M. Mathias, T. Tuytelaars, and L. J. V. Gool, "Seeking the strongest rigid detector," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2013, pp. 3666–3673.

[29] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679–698, 1986, ISSN: 0162-8828. DOI: 10.1109/TPAMI.1986.4767851.

[30] A. Bansal, A. Kowdle, D. Parikh, A. Gallagher, and C. Zitnick, "Which edges matter?" In *IEEE International Conference on Computer Vision, Workshop on 3D Representation and Recognition.*, 2013.

[31] J. Kamarainen, V. Kyrki, and H. Kälviäinen, "Invariance properties of gabor filter-based features-overview and applications," *IEEE Transactions on Image Processing*, vol. 15, no. 5, pp. 1088–1099, 2006, ISSN: 1057-7149. DOI: 10.1109/TIP.2005.864174.

[32] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: a review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013, ISSN: 0162-8828. DOI: 10.1109/TPAMI.2013.50.

[33] P. Sermanet and Y. LeCun, "Traffic sign recognition with multi-scale convolutional networks," in *IEEE International Joint Conference on Neural Networks (IJCNN)*, 2011, pp. 2809–2813. DOI: 10.1109/IJCNN.2011.6033589.

[34] D. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, vol. 2, 1999, 1150–1157 vol.2. DOI: 10.1109/ICCV.1999.790410.

[35] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," *Computing Research Repository (prerelease)*, vol. abs/1311.2524, 2013.

[36] F. Zaklouta, B. Stanciulescu, and O. Hamdoun, "Traffic sign classification using k-d trees and random forests," in *IEEE International Joint Conference on Neural Networks (IJCNN)*, 2011, pp. 2151–2155. DOI: 10.1109/IJCNN.2011.6033494.

[37] A. M. Martinez and A. C. Kak, "PCA versus LDA," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, pp. 228–233, 2001.

[38] K. Lu, Z. Ding, and S. Ge, "Sparse-representation-based graph embedding for traffic sign recognition," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 4, pp. 1515–1524, 2012, ISSN: 1524-9050. DOI: 10.1109/TITS.2012.2220965.

[39] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft, "When is "nearest neighbor" meaningful?" In *International Conference on Database Theory (ICDT)*, ser. Lecture Notes in Computer Science, C. Beeri and P. Buneman, Eds., vol. 1540, Springer Berlin Heidelberg, 1999, pp. 217–235, ISBN: 978-3-540-65452-0. DOI: 10.1007/3-540-49257-7_15.

[40] J. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Processing Letters*, vol. 9, no. 3, pp. 293–300, 1999, ISSN: 1370-4621. DOI: 10.1023/A:1018628609742.

[41] P. Dollar, Z. Tu, P. Perona, and S. Belongie, "Integral channel features," in *British Machine Vision Conference (BMVC)*, 2009.

[42] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001, ISSN: 0885-6125. DOI: 10.1023/A:1010933404324.

[43] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the royal statistical society, series B*, vol. 39, no. 1, pp. 1–38, 1977.

[44] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1627–1645, 2010, ISSN: 0162-8828. DOI: 10.1109/TPAMI.2009.167.

[45] C. Gu, J. Lim, P. Arbelaez, and J. Malik, "Recognition using regions," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2009, pp. 1030–1037. DOI: 10.1109/CVPR.2009.5206727.

[46] R. Benenson, M. Mathias, R. Timofte, and L. Van Gool, "Pedestrian detection at 100 frames per second," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2012, pp. 2903–2910. DOI: 10.1109/CVPR.2012.6248017.

[47] P. Dollar, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: an evaluation of the state of the art," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 4, pp. 743–761, 2012, ISSN: 0162-8828. DOI: 10.1109/TPAMI.2011.155.

[48] P. Dollar, R. Appel, S. Belongie, and P. Perona, "Fast feature pyramids for object detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 8, pp. 1532–1545, Aug. 2014, ISSN: 0162-8828. DOI: 10.1109/TPAMI.2014.2300479.

[49] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 5, pp. 898–916, 2011, ISSN: 0162-8828. DOI: 10.1109/TPAMI.2010.161.

[50] P. Dollár and C. L. Zitnick, "Structured forests for fast edge detection," in *IEEEE International Conference on Computer Vision (ICCV)*, 2013.

[51] A. Broggi, P. Cerri, P. Medici, P. Porta, and G. Ghisio, "Real time road signs recognition," in *IEEE Intelligent Vehicles Symposium (IV)*, 2007, pp. 981–986. DOI: 10.1109/IVS.2007.4290244.

[52] F. Olivier, "Three-Dimensional Computer Vision – A Geometric Viewpoint," *MIT Press*, 1996.

[53] S. Baker and I. Matthews, "Lucas-Kanade 20 years on: a unifying framework," *International Journal of Computer Vision*, vol. 56, no. 3, pp. 221–255, 2004, ISSN: 0920-5691. DOI: 10.1023/B:VISI.0000011205.11775.fd.

[54] B. K. Horn and B. G. Schunck, "Determining optical flow," *Artificial Intelligence*, vol. 17, no. 1–3, pp. 185–203, 1981, ISSN: 0004-3702. DOI: http://dx.doi.org/10.1016/0004-3702(81)90024-2.

[55] A. Giachetti, M. Campani, and V. Torre, "The use of optical flow for road navigation," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 1, pp. 34–48, 1998, ISSN: 1042-296X. DOI: 10.1109/70.660838.

[56] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 7, pp. 1409–1422, 2012, ISSN: 0162-8828. DOI: 10.1109/TPAMI.2011.239.

[57] D. Reid, "An algorithm for tracking multiple targets," *IEEE Transactions on Automatic Control*, vol. 24, no. 6, pp. 843–854, 1979, ISSN: 0018-9286. DOI: 10.1109/TAC.1979.1102177.

[58] M. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool, "Robust tracking-by-detection using a detector confidence particle filter," in *IEEE International Conference on Computer Vision (ICCV)*, IEEE, 2009, pp. 1515–1522. DOI: 10.1109/ICCV.2009.5459278.

[59] T. Fortmann, Y. Bar-Shalom, and M. Scheffe, "Multi-target tracking using joint probabilistic data association," in *IEEE Conference on Decision and Control including the Symposium on Adaptive Processes, 1980.*, vol. 19, 1980, pp. 807–812. DOI: 10.1109/CDC.1980.271915.

[60] S. Oh, S. Russell, and S. Sastry, "Markov chain monte carlo data association for multi-target tracking," *IEEE Transactions on Automatic Control*, vol. 54, no. 3, pp. 481–497, 2009, ISSN: 0018-9286. DOI: 10.1109/TAC.2009.2012975.

[61] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955, ISSN: 1931-9193. DOI: 10.1002/nav.3800020109.

[62] C. Karney, "Algorithms for geodesics," *Journal of Geodesy*, vol. 87, no. 1, pp. 43–55, 2013, ISSN: 0949-7714. DOI: 10.1007/s00190-012-0578-z.

[63] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "Slic superpixels compared to state-of-the-art superpixel methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2274–2282, 2012, ISSN: 0162-8828. DOI: 10.1109/TPAMI.2012.120.

[64] R. Adams and L. Bischof, "Seeded region growing," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 6, pp. 641–647, 1994, ISSN: 0162-8828. DOI: 10.1109/34.295913.

[65] L. Vincent and P. Soille, "Watersheds in digital spaces: an efficient algorithm based on immersion simulations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 6, pp. 583–598, 1991, ISSN: 0162-8828. DOI: 10.1109/34.87344.

[66] J. van de Weijer, T. Gevers, and A. Gijsenij, "Edge-based color constancy," *IEEE Transactions on Image Processing*, vol. 16, no. 9, pp. 2207–2214, 2007, ISSN: 1057-7149. DOI: 10.1109/TIP.2007.901808.

[67] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, 1997, ISSN: 0022-0000. DOI: http://dx.doi.org/10.1006/jcss.1997.1504.

[68] R. A. Fisher, "The statistical utilization of multiple measurements," *Annals of Eugenics*, vol. 8, no. 4, pp. 376–386, 1938, ISSN: 2050-1439. DOI: 10.1111/j.1469-1809.1938.tb02189.x.

[69] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967, ISSN: 0018-9448. DOI: 10.1109/TIT.1967.1053964.

[70] J. Hunt and D. MacIlroy, *An algorithm for differential file comparison*, ser. Computing science technical report. Bell Laboratories, 1976.

[71] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, Berkeley, California: University of California Press, 1967, pp. 281–297.