Classification of busses and lorries in an automatic road toll system

Examensarbete utfört i bildkodning
vid Linköpings tekniska högskola
av
Adam Jarl
LiTH-ISY-EX-3329-2003

Handledare: Niclas Fungbrandt
Examinator: Robert Forchheimer

Linköping 2003-04-16
Klassificering av bussar och lastbilar i ett automatiskt vägtulls -system

Classification of busses and lorries in an automatic road toll system

Nyckelord
Keyword
image processing, algorithm, mathematics
ABSTRACT

This work was done in order to improve an automatic road toll system developed by Combitech Traffic Systems AB\(^1\) in Jönköping, Sweden. Every vehicle passing the toll is categorised and divided into one of several classes in order to charge the right fee. Similar vehicles should belong to the same class. At the moment one class is being used for busses and lorries (trucks) of a weight more than 3.5 tons. There is a desire to introduce separate classes for these types of vehicles. The goal of this work was to design methods for a classifier that will divide each of these vehicles into the right class with as great certainty as possible. The work started 2002-09-02 and finished 2003-02-26. Most likely the classifier will be implemented, probably slightly modified, in the existing classification system.

\(^1\) After a recent name change the company is called Kapsch TrafficCom AB.
# TABLE OF CONTENTS

1 INTRODUCTION ............................................................................................................................... 3  
1.1 GOAL OF THIS WORK ......................................................................................................................... 3  
1.2 MOTIVATION OF THE SOLUTION METHOD ..................................................................................... 3  
1.3 SYSTEM DESCRIPTION ...................................................................................................................... 3  
2 WHAT INFORMATION TO USE AND THE TEST PROGRAM ........................................................ 5  
2.1 HEIGHT IMAGE .................................................................................................................................... 5  
2.2 TEST PROGRAM ................................................................................................................................... 6  
3 STRUCTURE OF THE CLASSIFIER ................................................................................................... 8  
3.1 THE PRINCIPLES OF CLASSIFICATION ............................................................................................. 8  
3.2 GENERAL APPROACH OF THE HEIGHT IMAGE ............................................................................... 9  
3.3 FROM RESULTS TO POINTS ............................................................................................................... 9  
4 THE IMAGE AND THE PROFILES ................................................................................................... 12  
4.1 IMAGE FORMAT ................................................................................................................................... 12  
4.2 PROFILE CALCULATIONS .................................................................................................................. 12  
4.3 OUTER PROFILE CALCULATIONS .................................................................................................... 13  
5 CLASSIFICATION METHODS....................................................................................................... 15  
5.1 BEFORE CLASSIFICATION .................................................................................................................. 15  
5.2 EDGE ACROSS .................................................................................................................................... 17  
5.3 COUNT HEIGHT VALUES .................................................................................................................... 18  
5.4 OUTER PROFILE HEIGHT DIFFERENCES ........................................................................................... 19  
5.5 GAP IN PROFILE .............................................................................................................................. 19  
5.6 PARALLEL SIDES .............................................................................................................................. 20  
5.7 DIFFERENCES IN WIDTH ................................................................................................................... 21  
5.8 HEIGHT DIFFERENCE BETWEEN FLAT AREAS ............................................................................... 22  
5.9 CLASSIFICATION OF SMALL AND LARGE VEHICLES ................................................................... 24  
6 SETTING OF POINTS AND CLASSIFICATION ............................................................................... 25  
6.1 WHY POINTS ARE SET THE WAY THEY ARE .................................................................................. 25  
6.2 SET POINTS ...................................................................................................................................... 25  
6.3 Classify .......................................................................................................................................... 26  
7 CLASSIFICATION ALGORITHMS ................................................................................................... 27  
7.1 INFORMATION ABOUT THE PRESENTATION .................................................................................. 27  
7.2 EDGE ACROSS ................................................................................................................................... 29  
7.3 COUNT HEIGHT VALUES ................................................................................................................... 30  
7.4 OUTER PROFILE HEIGHT DIFFERENCES ........................................................................................... 31  
7.5 GAP IN PROFILE .............................................................................................................................. 31  
7.6 PARALLEL SIDES .............................................................................................................................. 32
7.7 DIFFERENCE IN WIDTH ...................................................................................................................... 34
7.8 HEIGHT DIFFERENCE BETWEEN FLAT AREAS....................................................................................... 36
8 RESULTS ............................................................................................................................................ 39
9 PROBLEMS AND FURTHER DEVELOPMENT ..................................................................................... 41
  9.1 GENERAL PROBLEMS ................................................................................................................... 41
  9.2 DIFFICULT TYPES OF VEHICLES.................................................................................................... 41
  9.3 IMPROVEMENTS............................................................................................................................ 42
A. APPENDIX - SETTINGS FOR PARAMETERS FOR SMALL VEHICLES ........................................ 44
B. APPENDIX - SETTINGS FOR THE POINT SETTING FUNCTION ............................................ 45
C. APPENDIX – LEAST SQUARE METHOD FOR 2 DIMENSIONS....................................................... 48
REFERENCES.................................................................................................................................... 49
1 INTRODUCTION

1.1 Goal of this work

The purpose of this work was to develop methods that will be used in the future for classification of busses and lorries of a weight more than 3.5 tons. Existing classification only use length, width and height of a vehicle together with information about a possible trailer attached to the vehicle. To improve the classification and increase the number of classes it was decided to develop a classifier specialised in the problem of busses and lorries. No certain degree of correctness was determined, but the work was carried out in the sense of making the classifier as good as possible. A vague lower limit was though 95 % correctly classified vehicles.

1.2 Motivation of the solution method

I considered two ways of solving the problem. The first way was to use some form of complex mathematical algorithm for recognition, which were to be trained with information from many vehicles. This would be rather complex and it would be hard to interpret the result in terms of right or wrong from a human point of view.

The second way was to use several methods each testing for specific features of a vehicle and the combined results from all methods would then yield the final class. All methods would be rather simple and the outcome easily interpreted.

I chose the latter way because I thought it would be easier to implement, yet achieving the necessary degree of correctness.

1.3 System description

This kind of automatic toll system differences from an ordinary toll system. Usually vehicles have to travel in separate lanes and stop when the payment is done. The automatic system allows vehicles to move freely between the lanes and there is no need to stop as the paying is done automatically. Vehicles can travel with a speed of up to 160 km/h. Consequently, the system has very little impact on the traffic flow and it is almost as the toll was not there.

Every vehicle has a little box, called a transponder, placed on the inside of the windscreen. The transponder contains information about the vehicle and the relevant account. A unit, called transceiver, reads information of the transponder, such as vehicle class and payment method. Another sensor, called VDC (Vehicle Detection and Classification), tracks the vehicles and determines the class for each vehicle. It is the information from this sensor that will be used in this work. If there is no transponder in a vehicle, the license plate is photographed by the VR-sensor (Vehicle Registration) with the help of an infrared flash. In figure 1.1 a schematic picture of a road mounted system can bee seen.
Figure 1.1. An automatic toll system mounted above the highway on a so-called gantry. A vehicle is about to enter the tolling area. The transponder is located inside the windscreen, 1. A transceiver extracts the information of the transponder, 2. The VDC sensor tracks and classifies the vehicle, 3. If a transponder is missing, the VR unit photographs the rear license plate when the vehicle has passed, 4.
2 WHAT INFORMATION TO USE AND THE TEST PROGRAM

2.1 Height image

Every VDC-sensor (there is one for every lane) has a pair of video cameras that take images with an interval of 80 milliseconds. The two cameras have the same distance to the ground but are separated in the horizontal plane by a distance of one meter approximately. The direction of the separation is diagonally – equally in the road direction and the direction across, see figure 2.1.

Figure 2.1. The VDC unit seen from underneath. The two cameras (circles) are separated by a distance of approximately one meter.

The two cameras of a VDC sensor take images of the same view but from different positions. Two such images are referred to as a stereo pair. By using a pair of images from two cameras information about the depth in the view can be obtained. From just one photograph only different intensity can be seen. It is true that a human being often can tell the relative distance in a single photograph but here a computer will do the judgement, so the depth must be calculated. An algorithm calculates the depth from every pair of photographs. The result is a height image and it will be this kind of image that will be used as information in this work.

The height from the ground is used rather than the distance from the cameras. All areas of the ground level will then have the height zero, while a point one meter closer to the cameras will have the height one meter. There are 255 levels for a height value and 1 step corresponds to 4 cm. A height of 1 m is then 25 steps. It is important to mention that there must be contrasts in the photographs in order to achieve the correct height. Without any differences between the two photographs no information can be obtained and the height will be zero. On flat areas, like a roof, there are not too many differences when comparing a stereo pair, and there will consequently not be many height values there. Other areas, like the edges of a vehicle, will be very easy to detect because the sudden increases of height. The height image will consist of a value in each point. In order to clarify those values when looking at a height image it is good to assign a colour for each value. The correct value of each height is then easier to see, than if only a grey scale was to be used. In this work a colour palette has been used, but only of about 20 colours, so the height values have been divided into intervals where the same colour has been used. Most images in this report are though in grey scale. In most cases the understanding does not demand a colour image.

The company is unique in the world of detecting and classifying vehicles using height images. Figure 2.2 contains a stereo pair and the corresponding height image.
Figure 2.2. The two grey scale photographs, marked SU1 and SU2 (Sensor Unit), to the left are a stereo pair taken by a VDC unit. The vehicle, 6, is a lorry and the driving cabin can be seen to the right in the two images. Behind it is a large loading space with constant height. Up to the right is the height image (HI) and here have colours been added to make it easier to comprehend. Because of the flat loading space no height values can be obtained there, besides from the surrounding edges. The yellow rectangle surrounding every vehicle is supposed to have the same area and position that a vehicle has on the ground plane.

2.2 Test program

It was determined from the beginning that the classifier only should use the information that could be extracted from the VDC height image. Other information such as photographs taken by the VDC sensors were not to be used.

From start there was an existing program for experimenting with single height images. With single height images I mean that one image at the time is used and not a sequence of a passing vehicle. This program was used as a base to load single height images. Functions for classification could then be added. In order to test an image in this program, height images had to be loaded manually one at a time. The region of interest, i.e. the borders of a vehicle, was also manually set before a calculation could be done. This tended to get irritating when you wanted to test a lot of images.
Therefore another solution was to use an existing simulator of the VDC display. The simulator needs recorded photographs from the VDC stereo cameras. The height image of every stereo pair is then calculated in real time. Classification functions could be added to the code and thereby obtain an automatic classification. There were still two problems. First, there was lack of recorded material, but recordings from the E4 highway could not be used because the file format was different then the required. Second, the computer where the program was installed were to slow (my computer) – vehicles were moving approximately 10 times slower than the computer could handle.

The solution was to make a new application. The technical expert of the VDC unit did this, see [1]. The new program called BusLorry only used pre-calculated height images. BusLorry is the program where the main work has been done. Also an application to record height images (in the right file format) from the VDC unit installed over the E4 highway, was made by [1]. Only one height image per vehicle is saved and the front position, in the travel direction, is always within a short length (±1 m) straight below the VDC sensor. This is necessary in order to get a height image best suited to the classification functions. Figure 2.3 shows the interface of the BusLorry application.

![Figure 2.3. The interface of the test application called BusLorry. This is a lorry and the travelling direction is from left to right. Height images are loaded one at a time or a series of images of different vehicles can be run automatically. The yellow rectangle defines the area the vehicle has on the ground.](image)
3 STRUCTURE OF THE CLASSIFIER

3.1 The principles of classification

Before presenting the methods of classification in detail it could be good to view the classification from another level. From the beginning the idea was to treat all vehicles in the same way, but it has shown to be necessary to use different methods and settings depending on the size of the vehicles. Let us call vehicles with smaller dimensions based on certain criteria on the length, width and height for small. There can be one or more sets of length, width and height. Important is that the total set that defines small vehicles is formed by the union of all sets, see figure 3.1. The set for all other vehicles, the large ones is the inverted set, see figure 3.1.

![Figure 3.1. Two sets of dimensions for small vehicles. The union is defined by the total area and the outer surroundings define the set for large vehicles.](image)

The set of dimensions for small vehicles are defined as the dimensions that are minimal for a bus specified in the actual project\(^2\): this means that all small vehicles should either be classed lorry or something else, but not bus. The dimensions of a small vehicle would then give information whether the class should be some form of lorry, personal car or some other vehicle. Therefore no use of the classifier is needed for small vehicles, but it is interesting to see how the classifier works if the definition of the set would change. Observe that the values of the dimensions are the instant values and not the values used by the ordinary classifier. Those values depend on the whole series of images for every vehicle.

A large vehicle could either be a lorry, bus, or some other vehicle, an agriculture machine, for example. The present classification handles all these vehicles as one class. Let us call this class ordinary large. One can say that the wanted classes lorry and bus are subsets of this class. Except the main vehicle, there can also be a trailer attached. Occurrence of a trailer would yield the final class lorry with trailer or bus, see table 3-1. A bus can consequently transport a trailer without any extra charge.

<table>
<thead>
<tr>
<th>Type of large vehicle</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bus</td>
<td>Bus</td>
</tr>
<tr>
<td>Bus with trailer</td>
<td>Bus</td>
</tr>
<tr>
<td>Lorry</td>
<td>Lorry</td>
</tr>
<tr>
<td>Lorry with trailer</td>
<td>Lorry with trailer</td>
</tr>
</tbody>
</table>

Table 3-1. The types of vehicles and their corresponding classes.

---

\(^2\) At present in this project vehicles within the following sets are NOT considered to be busses: \( \{ \text{length} < 8.5 \ \text{AND width} < 2.3 \ \text{AND height} < 3.5 \} \) or \( \{ \text{length} < 6.5 \} [m] \).
The detection of trailers is already functional and the new classifier works independent of it. If a vehicle is to be considered for classifying at all, the height and width must be greater than certain minimum values. In order to clarify all this, let us look at the flow chart in figure 3.2.

![Flow chart](image)

**Figure 3.2. The influence of the ordinary classification.**

### 3.2 General approach of the height image

All methods are designed to detect those kinds of attributes that can be seen in a height image with the human eye. There are things like the shape, both from the side and from above, or certain details specific for a vehicle type. Considering a normal bus, it has a flat roof with constant height and flat walls on the sides without any great variations (the width is constant) - one can say that it looks like a box.

On the other hand, an ordinary lorry usually has either a height variation or a width variation. In other words, it doesn't look like a box. The methods of problem solving are rather straightforward in the sense of the natural structure of them. Theoretically, all results from the classifier can be verified by a human being by just looking at an image. A person would do just fine as a classifier, but the concentration wouldn’t be on top all the time and many and fast going vehicles would make it even more difficult. Other methods, like principal component analysis, are more complex to understand.

The methods of classification all use the height image in some way. Either directly by operating on the pixel values, or indirectly by using a height profile taken alongside the vehicle.

### 3.3 From results to points

The classifier consists of a number of methods. All methods deliver some form of results whether it is a single value, several different types of values, or a vector of values.
The outcome of each method must be translated so that they can be compared with each other. Independent of type of result delivered from a method the translation consists of one new value called point. This is an integer and the value should reflect the character of each result. Great probability for lorry characteristics (equals small probability for bus characteristics) results in a great positive point. In the case of busses, the situation is the same, but with a change of sign. If the point is zero, the meaning is that no obvious information has been obtained, so the method has not detected anything of use.

All methods operate independently of each other, and the point of each method is also set independent of other results, with some exceptions which will be shown later. The final class depends on the total sum of all points, with some special cases. A sum of points greater than or equal to zero means lorry class and a sum less than zero means bus class. The certainty of the class increases with the absolute sum of points.

The points are integers as mentioned before. For all methods the points are stepwise discrete functions of the results, which do not need to be integers. Maybe it would be more correct with linear discrete functions, but having them stepwise discrete introduces a form of robustness in the system, see figure 3.3.

![Figure 3.3. The principle of the relationship between an answer from a method (level) and the corresponding point.](image)

The points, certainty and class appear in a command window together with information about the position of the vehicle, in the test application, see figure 3.4.
Figure 3.4. The results of a classification in the test application. The information in the upper region is data about the position of the vehicle (from figure 2.2). Then there are points from the classification methods and the total sum (LORRY 4). The name of each point is the name (or similar) used for the corresponding classification method in the source code, see section 7 for the explanation of each name.
4 THE IMAGE AND THE PROFILES

4.1 Image format

The direction of the road is from left to right in the height image. For a straight going vehicle the orientation in the height image is as follows: left side up, right side down, rear end left and front right.

The VDC sensor delivers two greyscale photographs every 80\textsuperscript{th} millisecond. After some transformations of the photographs, one height image is calculated. A vehicle travelling with a speed of 100 km/h moves 2 m between every photo shot. The visible area is approximately 20 m in the road direction and 16.5 m in the across going direction. Therefore a lot of height images is received for each vehicle.

In order to use the new classification functions, the quality of the height image must be good and the vehicle must be in the right position. Also great variations from one image to the next 80 milliseconds later may occur. Therefore, and for simplicity, only one height image per vehicle has been used in the development, as mentioned in section 2.2. The ordinary classification uses all images of each vehicle for the length, width and height measurements. The class decision from ordinary classification is therefore determined when the vehicle has passed, while the new classifier operates when the vehicle is fully visible.

The specific height image format of the recorded material has been 224 times 183 pixels and the resolution has been 0.09 m/pixel. This is equivalent to 20.16 times 16.47 m. The height values range between 0 and 255 units, where one unit equals 0.04 m. One meter has the value 25 units, for example. See figure 4.1 for the image format.

![Figure 4.1. The height image dimensions and orientation.](image)

4.2 Profile calculations

First we give the definition of a height profile:

**Definition 4.1. Height profile**

*For each position, x, in a direction, d, take the greatest height value, h, in the direction perpendicular to d and let the profile value in position x be h.*
The ordinary classification uses only length, width and height of a vehicle to place the vehicle in a class. In order to get the height, the side height profile is calculated, then the greatest value in the profile in chosen to represent the height. Before placing a value in the height profile a filtering process is carried out. Let \( x \) represent the direction along the vehicle – from rear to front, and let \( y \) represent the direction across the vehicle – from left to right, see figure 4.2. Then proceed according to algorithm 4.1.

**Algorithm 4.1. The principle of height profile calculation**

1. Set \( x \) to the rear most \( x \)
2. Run through all height values in the \( y \)-direction and save the 5 greatest values
3. Depending on the width and the number of values greater than zero, remove 1 or more of the saved values in decreasing order
4. Place the greatest remaining value in the profile at position \( x \)
5. If \( x \) is smaller than the front most \( x \), increase \( x \) with 1 and go to step 2
6. End

This profile calculation will also be used in the new classifier, with some modification. Another type of profile to be used is a **count profile**, which has the following definition:

**Definition 4.2. Count profile**

*For each position, \( x \), in a direction, \( d \), count all height values greater than zero, \( c \), in the direction perpendicular to \( d \) and let the profile value in position \( x \) be \( c \).*

In the new classifier a height profile along the long side of a vehicle will be used and a count profile will be taken across the vehicle, in different positions, which will be described in detail later on.

The height profile will also be filtered to reduce noise and to make the profile more adjusted to the functions.

### 4.3 Outer profile calculations

By studying the profile the shape of the vehicle can be seen. If there are obstacles corrupting the shape, like a top mounted fan box for example, it can be difficult to recognise the vehicle just from the profile. Instead just the outer areas of the vehicle can be used. By the outer areas I mean the left and right sides of the vehicle. The middle part from the front to the rear is not taken into account. This often gives a more correct understanding of the *shape* of the vehicle, but the height
could in some cases differ substantially from the ordinary profile. Figure 4.3 displays a vehicle where its characteristic shape is restrained by obstacles on the top of the driver cabin, producing an ordinary profile different from the expected shape - the outer profile.

The outer profile will more often than the ordinary profile describe the shape of a vehicle in a better sense. Besides the more obvious shape of the height, also the differences in width can be detected, because of the narrow bands of the left and right sides. Even if the outer profile may differ from the ordinary profile, it is more suitable for detecting differences between busses and lorries. The outer profile is calculated accordingly to algorithm 4.1, with the difference that the inner area of the height image isn’t considered. A parameter indicates how wide the left and right side bands should be respectively.
5 CLASSIFICATION METHODS

5.1 Before classification

The classification methods can not be used directly, without preparations. The VDC unit tracks each vehicle and delivers the middle position of the front. Together with that position and the length and the width, the four corner points can be calculated. All values originally have the unit meters and the coordinate system does not have the correct position. Transformation from meters to pixel units and translation of the coordinate system are done before delivery to the test program. The corner points form a rectangular area, the region of interest, surrounding the vehicle. Figure 5.1 defines the direction of the coordinate system and the borders of the region of interest that will be used from now on.

![Diagram of region of interest](image)

Figure 5.1. The four corner points of the region of interest and the point delivered by the VDC unit.

The x-axes reaches from 0 to 223 and the y-axes reaches from 0 to 182, as can be seen in figure 4.1 and these are pixel units. One can also see what limits the region of interest. The VDC information about the length, width and front position corresponds to the ground level. Because the roof is closer to the VDC sensor it will appear larger than if it were on the ground. The region of interest will then be a bit smaller than the vehicle in the height image. In order to magnify the region of interest each line forming the rectangle must be moved a bit out from the centre. Depending on the actual position of the VDC unit in relation to the road, each line need to be moved a different amount of pixels. In all testing only the upper and lower line has been moved. The upper has been moved 1 pixel and the lower 2 pixels. Front and rear lines have remained the same because it has not been necessary to move them, from a classification point of view. The minimal and maximal values are defined by:

**Definition 5.1. Limits of the region of interest**

If the orientation of the vehicle is used as reference and the middle point of the front is \((F_x, F_y)\) and the length and width are known, the region of interest is limited by

\[
(x_0, y_0) \leq (x, y) \leq (x_1, y_1)
\]
\[ x_0 = F_x - \text{length} \quad \text{the rear x-position}, \]
\[ y_0 = F_y - \frac{\text{width}}{2} - 1 \quad \text{the left y-position}, \]
\[ x_1 = F_x \quad \text{the front x-position}, \]
\[ y_1 = F_y + \frac{\text{width}}{2} + 2 \quad \text{the right y-position} \]

There has not yet been decided which width and length values that will be used. These values can be obtained in two ways that in reality would give the same results. The first method is the one that is used in the ordinary classification. All measured values of the width and length - two values for each height image - of one vehicle are considered to estimate the next values (with help of a so called Kalman filter). In the other method the value of the width and length only depend of the actual height image. At present the second method is used.

From now on there will be frequent use of the height value belonging to position \((x, y)\) in a height image and the height value in position \(x\), in a height profile – ordinary and outer. To simplify the further description, definitions for these values are needed:

**Definition 5.2. Height, profile and outer profile**

If the usual coordinate system is used,

\[
\text{height}(x, y) \quad \text{the height image value in position } (x, y),
\]
\[
\text{profile}(x) \quad \text{the height profile value in position } x,
\]
\[
\text{outer profile}(x) \quad \text{the outer height profile value in position } x
\]

As mentioned before, the height profile calculated according to algorithm 4.1, needs to be filtered to reduce noise. This is accomplished by using a median filter of length 5. By doing this, the original profile is transformed into a median profile, see algorithm 5.1.

**Algorithm 5.1. Median filtering of height profile**

1. Set \(x\) to \(x_0 + 2\)
2. Find the 3 greatest values among \(\text{profile}(x - 2)\) to \(\text{profile}(x + 2)\)
3. Sort the 3 values from above by increasing order
4. Assign the smallest value from step 3 to median profile\((x)\)
5. If \(x\) is smaller than \(x_1 - 2\), increase \(x\) with 1 and go to step 2
6. Copy \(\text{profile}(x_0), \text{profile}(x_0 + 1), \text{profile}(x_1 - 1)\) and \(\text{profile}(x_1)\) to corresponding positions in the median profile

7. End

As can be seen in the algorithm, the two outermost profile values in each end, are unaffected by the filtering. This is due to that no information outside the interval \(x_0\) to \(x_1\) is available. The median profile will from now on be called the profile, see statement 5.1.

**Statement 5.1**

The result from the filtering - the median profile - will from now on also be referred to as the profile since the ordinary profile will not be used anymore. This is true for both ordinary profile and outer profile.
The remaining sections of this chapter will in short describe the different classification methods. Chapter 7 contains the detailed descriptions and algorithms.

5.2 Edge across

It is true that the VDC-unit delivers a height image, or distance image, but it is only possible to calculate the height where there are contrasts in the image. For example, edges, like the ones surrounding a vehicle, become very marked in the height image. A vehicle of constant height and no sharp differences in the colours at the roof would appear like a perfect frame in the height image.

The most apparent feature of a height image of a lorry is that it has one or more distinct lines, or edges, across the vehicle. This is due to that a lorry almost always has a driver cabin that in some way are separated from the rest of the lorry. Therefore a more or less clear line occurs. A bus, on the other hand, doesn’t usually have this kind of lines, unless the front of the roof has a different angle than the rest of the roof, or the windscreen isn’t entirely vertical. There could also be one or more roof hatches forming lines.

There are three methods of finding lines. All methods are very much the same and one function instead of three could be used. This function must then be called three times with three different sets of parameters. However, making one function and calling it three times is only cosmetic change and will only have the benefit of some less lines in the code. This will of course be done later on, but having three different functions, keeps the presentation more easily comprehensible.

5.2.1 Edge across – single line

The first method just looks for a single line with pixels in the y-direction. For each position in the x-direction, a count for values greater than a minimum level is done in the y-direction. For every x-position where the count value is relatively high compared with the width of the vehicle, an edge is assumed to be found and it is noted in a vector. Depending on where possible edges have been found, this method can give 0 or 1 point.

5.2.2 Edge across – wider edge followed by empty area

In order to make the edge detection more robust, an interval instead of single lines, can be checked. The requirement for this kind of line is that the line must cover an area. Behind the line there must be an area that is relatively empty. This is to ensure that the line is relatively straight, in the rear any way. This method also gives 0 or 1 point. See figure 5.2.

Figure 5.2. A vehicle with a wide edge with an empty area behind.

5.2.3 Edge across – wider edge surrounded by empty areas

This is slightly the same as the previous method, but the edge must not be too wide (in the x-direction), which wasn’t the case in the previous method. There must be one empty area to the rear of an edge area and one empty area in front. This method would be the best method of finding an edge, but the conditions are that two empty areas with a line in the middle occurs, which isn’t too common. The method can give 0 or 1 point. For a vehicle receiving 1 point, see figure 5.3.
5.3 Count height values

A surface with variations and contrasts gives rise to height values. By counting the relative number of height values in an area, an understanding about the form of the surface could be received. A bus usually has a flat roof with few height variations. The possible height values except those from the surrounding edges, originate from roof hatches or the fact that the length of rise from the front to full height are relatively long. Figure 5.4 displays one bus with few height values and one bus with many height values.

The counting is done in an area of the front that is a few meters long and the width is somewhat smaller than the width of the vehicle, so that the edges originating from the sides of the vehicle are excluded. These edges are always present for every type of vehicle. It is the front that is most interesting and where the differences can be seen. Counting over the whole vehicle could result in relatively few height values in spite of many values in the front. Figure 5.5 displays the count area.

If there are relatively few height values in the front, there is great probability that the vehicle is a bus and the other way around for a lorry. The result from this method can produce points in the range from –3 to 2.
5.4 Outer profile height differences

As mentioned in section 3.3 the outer profile will be more effective in finding differences between the shapes of a bus and a lorry. Because of the somewhat noisy character – width variations - of the surrounding edges of a vehicle in a height image, the outer profile often gets fluctuated and this increases with poor image quality and great width differences. To reduce possible noise in the outer profile a sliding averaging is applied. For each value the average of the value and it’s nearest neighbours are calculated. When all points have been averaged the greatest height difference is found. This method can give from –1 to 2 points. The averaging length is approximately one meter or shorter. In figure 5.6 you can see an averaged outer profile together with the other profiles.

![Figure 5.6](image)

*Figure 5.6. Normal height profile, left, outer height profile, middle, and averaged outer height profile, right. The averaging length has been 1 m (11 pixels).*

5.5 Gap in profile

Some vehicles have a space between different parts of the vehicle. Most common is a gap between a driving cabin and a loading space or an attached trailer. The area behind the driving cabin could also be very low compared to it – an open loading area. Busses can have a great height difference between the front and some part far rear, if the bus have the motor mounted in front of the driver for example. It is therefore suitable only to look for height gaps where the major height has a position in front of the smaller. To avoid low height in the rear of a vehicle, the search is conducted from the middle to the front. If there are no gaps in that region, there will probably not be any gaps behind it either. If there are, they are most likely to be referred to as missing values in the height image. In figure 5.7 two different types of gaps can be seen in the profiles.

![Figure 5.7](image)

*Figure 5.7. Two different types of gaps in height profiles. The gap in the left profile originates from the space between the driver cabin and the loading space. The gap in the right profile depends on a low loading area.*

The outer profile is too risky to use, because it is not based on information of the whole vehicle. Instead the profile is used, but it is averaged first. The profile is averaged with two different length – one length to locate the major value and another, shorter, to find the minor one. The reason is that the gaps could have short extension and a too long averaging length could lead to an increase
of the height in the gap. A longer length is used to find the maximum, in order to decrease the impact of unrealistic high values that may occur as disturbances. There are not so many vehicles with this kind of gaps, so the detection rate will be pretty low. In the other hand, when a gap is detected, it is rather sure that the vehicle is a lorry. Depending on maximal height difference the method yields 0, 1 or 2 points.

5.6 Parallel sides

A bus has a distinct form in the front section. This form consists of two parallel sides - the left and right – each forming a 90-degree angle with the front. The widths of the left and right sides are usually pretty thin for a bus. If the front is also very steep this results in a characteristic form, see figure 5.8.

![Figure 5.8. A typical bus with a front consisting of straight parallel sides, left, and the ideal form, right.](image)

The aim for this method is to investigate whether the left and right sides are
- parallel
- straight
- equal in height

In order to accomplish the goals three points on the left and right sides respectively are selected. To each set of three points a line is fitted in the best possible way. The position of the points in the x-direction is constant relative to the position of the front and the distance between them is now a bit more than one meter. In the y-direction the desired position is in the middle of each side at every x-position, see figure 5.9.

![Figure 5.9. The positions of the three points in the left and right sides respectively, used for line estimation. The wanted y-positions should be in the middle of each side for every x-position. Note that all values are black. This does not mean that the height is the same, only that the height is not zero.](image)
When all points have been found the least square method, see appendix C or [2], is used to fit a line in the best possible way to each set of points respectively. Then the following things are calculated

1. The angular difference between the left and right lines
2. The maximum distance from a point to the corresponding line
3. The maximum width difference between a left point and the corresponding right
4. The maximum height difference between the points of each side

Besides these four criteria the count value can not be to high if the method is going to give a negative point. The method is designed to detect a bus front, but it can also give positive points. The latter may happen if the lines are not too straight and the angle between them is considerable. This is not likely to occur very often. The range of points is −2 to 2 and the desired thing is to get −2 or −1 for a bus and no negative points otherwise. In other words:
It is easier to detect the front of a bus than the front of something else with this method.

5.7 Differences in width

An apparent characteristic to look for is difference in width in one way or the other. There are two methods for doing this. One gives only negative points while the other only gives positive points.

5.7.1 Difference in width – maximum difference

The first method finds for every x-position the position in the y-direction, where the left and right sides begin respectively. The y-position is depending on the surrounding area in order to filter noise and to deal with missing parts of the height image. For every x a width is found and the search is conducted in a surrounding of x in order to be able to filter. For the finding of the left side, the search starts at \(y_0\) and progresses inward the region of interest until the side has been found and this is done for each x-position. Finding of the right side is performed in a corresponding way. Figure 5.10 displays the principle of finding the sides.

![Figure 5.10. The principle of finding the sides (y-values) for a given x. The desired y-positions are marked in the figure together with the width.](image)

The width for every x-position is then calculated and the maximum and minimum width values are saved and the difference is used to set the point, which lies in the interval −3 to 0.
5.7.2 Difference in width – lorry type

Just investigating the width difference is not always the best method. The former method gives zero points if the width difference is big enough. Great width differences do not tell whether there really is a great difference or if there is an incorrect height image or some other reason. Looking at a height image of a typical lorry, see figure 5.11, you can see from the front that

- the smaller width is in front of the bigger
- both y-positions of the smaller width are farther away from $y_0$ and $y_1$ respectively than what is the case for the great width

![Figure 5.11](image)

*Figure 5.11. A vehicle where a width difference has been found where the smaller value is in front of the greater and where the sides at the smaller value both are positioned closer to the middle than the sides at the greater value.*

These observations could be used in order to refine the search for the greatest width difference. The method described in the former section is used to find small differences in the width and therefore often bus characteristics. This method will in contrast to that be used to find lorry characteristics and therefore give 0, 1, or 2 points. It can be necessary to say that the resulting points from each method can not be separated from zero at the same time.

5.8 Height difference between flat areas

A rather obvious thing about both the profile and the height image is the difference in height, which have not been dealt with yet. Just measuring single height values in the profile soon proved to be too insecure. Noise and missing parts of the height image requires more sophisticated methods. An area of certain minimum length must be relatively flat if the height value there should be taken into account. Flatness needs to be defined:

**Definition 5.3. A flat area in a profile**

For every point, $x$, in the profile, the greatest number, $N$, is found for which holds

$$|\text{profile}(x) - \text{profile}(x - i)| \leq \varepsilon,$$

where $i = 0, 1, 2, \ldots, N$ and $\varepsilon$ is the maximum deviation from $\text{height}(x)$. The length, $L$, of the flat area is then $L = N + 1$.

There are two different methods for detecting height difference between flat areas. Both methods could use either the profile or the outer profile. Using the profile would make the results more reliable, but on the other hand, using the outer profile, would make them more easily detectable. So either profile could be used. Figure 5.12 displays a profile where the flat areas have been marked.
5.8.1 Height difference between flat areas – maximum difference

Here, the greatest difference between two flat areas is detected. There are one minimum length of the flat area when finding the maximum height value and another when finding the minimum. This is due to the fact that erroneous height values more often are relatively high than low, so the length for finding the maximum value should be greater than the one for finding the minimum value, see figure 5.13. Currently, the outer profile is used and the points range from 0 to 2.

5.8.2 Height difference between flat areas - step

Instead of just finding the height difference between flat areas, a sudden height increase can be detected. A step in the front of the profile is rather common among lorries and this method is designed to detect that feature, see figure 5.14. The former method would also detect a step if there were any, but it wouldn’t know about it. If a step do exist, this method detects it and the former method just amplifies this. Currently, the profile is used and the points range from 0 to 2.

---

3 This is depending on the settings of both methods. Equal parameter values would always make the two methods work together, but otherwise the points could differ between the two methods.
5.9 Classification of small and large vehicles

All methods are now described and more detailed description will come in later chapters. As mentioned in section 3.1, the vehicles regarded as small will be less interesting than the large ones, mostly because of the definitions for the classes for small vehicles have not yet been set up. There is an interest (personal) of testing the methods for small vehicles too. All methods are not used when classifying these vehicles and the parameters for the remaining are different. A small vehicle with no differences in the side form and relatively constant height has been regarded as a bus and otherwise as a lorry. The methods that have been used are:

- gap in profile (5.5)
- parallel sides (5.6)
- difference in width, both (5.7.1 and 5.7.2)
- height difference between flat areas, both (5.8.1 and 5.8.2)

Methods left out:

- edge across, all three (5.2.1 – 5.2.3):
- count height values (5.3)
- outer profile height differences (5.4)

The reasons for leaving them out are the same for all these methods. Because of the greater slope of the front - longer rise length from front to roof - giving rise to many height values, these methods would be rather hazardous.
6 SETTING OF POINTS AND CLASSIFICATION

6.1 Why points are set the way they are

It has been said that negative points should reflect a bus feature and the opposite for lorries. If the result from a method is hard to categorise, the point will be zero. Some methods will not give rise to any negative points and some will not give rise to any positive points. That’s because only features for one type of vehicle are assumed to be detectable. Let’s take the method called “edge across”, described in section 5.2, as an example. If there are any edges - in the right places - they probably belong to a lorry and the point will be positive, but if there are no edges, nothing really can be said. That’s why no negative points will be produced from the result of this method.

Besides the fact that some methods only will produce points strictly positive or negative (including zero), there are other differences for other methods. Some methods have an interval of possible points were the positive interval is less or greater than the negative. This is the case for the method called “outer profile height differences”, described in section 5.4. Here, the interval of points is –1 to 2. If there are great height differences in the outer profile there should be 2 points, but if there are small differences, it is harder to tell because there are both buses and lorries with this characteristic. There is though higher probability to be a bus, but to reduce the damage if it in spite is a lorry, only –1 is the minimum point.

The maximum positive or minimum negative points are often 2 or –2 respectively, but in some cases there could be more than 2 or less than –2. The interval of points for the method called “count height values”, described in section 5.3, is –3 to 2. An almost empty area results in –3 points and is considered to be very sure of the bus feature. It is also a way to compensate for the fact that there are more methods that can result in positive points than there are methods that can result in negative points, so the maximum sum of points is greater than the minimum. By having -3 points instead of -2, the relationship between the maximum sum of points and the minimum is somewhat more equal.

6.2 Set points

All methods operate independently of each other. Most of the points are also set independently of each other i.e. a method only affects the corresponding point, but there are some exceptions.

6.2.1 Set points - parallel sides

If the result from “parallel sides” (5.6) should give –1 or –2 points, the point from “count height values” must be less than or equal to zero in order to receive the desired point for “parallel sides”. Otherwise the point is set to zero. From this statement one can draw the conclusion that it is necessary that the point for “count height values” is set before the one for “parallel sides”.

6.2.2 Set points - difference in width

The point from “difference in width - maximum difference” (5.7.1) depends on a variable used by the other method of the same type, “difference in width – lorry type” (5.7.2). If the width difference that should give a negative point in the first method is the same that should give a point
greater than zero in the second, we have a conflict, because one method is telling us that the width
difference is small enough and another says the opposite.
The solution is to give one method a higher priority than the other one. I have chosen to let the
“lorry type” method have a higher priority. A variable, which we here can call *lorry*, denotes with
true or false whether any real width differences have been found by the “lorry type” method. If
not, the point for “lorry type” gets zero and the point for “maximum difference” can be set to
reflect the width difference. If lorry is true the point for “maximum difference” becomes zero and
the “lorry type” point can be set accordingly to the result.

6.2.3 Set points – edge across

The two edge methods described in 5.2.2 and 5.2.3 (wider edges) also depends on one of the width
difference methods (lorry type, 5.7.2). In order to receive any points for 5.2.2 and 5.2.3, a width
difference must have been found, see 6.2.2 for further description.

6.3 Classify

Earlier in this report it has been said that the final class depends on the total sum of all points and
that all methods operates independently. This is not entirely true. The classifier (for large vehicles)
looks like:

1. *Calculate the sum of all points, sum*
2. *If* \(-2 \leq \text{sum} \leq 0 \text{ AND points for gap in profile} = 2*
   \[ \Rightarrow \text{sum} := 1 \]
3. *Else if* \(\text{sum} = 0 \text{ AND points for parallel sides} = -2\)
   \[ \Rightarrow \text{sum} := -1 \]

Step 2 says that for a small negative sum of points, the point for “gap in profile” should decide the
outcome, because it is a pretty secure method as mentioned before. If the sum of points is zero, the
class is lorry according to step 1, but a sum equals zero means lorry with no certainty. A total sum
of zero point combined with 2 points for “gap in profile” gives the same class, lorry, but increases
the certainty by increasing the sum of points.

If the sum of points is zero and the point for “gap in profile” was less than 2, there is in step 3
controlled if the points for “parallel sides” equals –2. If it does, the sum should be changed to –1.
The class *and* the certainty have then been changed.

It is not common that step 2 or 3 is used. Even if the sum is zero or close, it is not sure that the
points that have been mentioned have the right values. The two methods “gap in profile” and
“parallel sides” are used because of their high reliability as mentioned before.

The classifier for small vehicles looks like:

1. *Calculate the sum of all points, sum*
2. *If* \(-1 \leq \text{sum} \leq 0 \text{ AND points for gap in profile} = 2*
   \[ \Rightarrow \text{sum} := 1 \]
3. *Else if* \(0 \leq \text{sum} \leq 2 \text{ AND points for parallel sides} = -2\)
   \[ \Rightarrow \text{sum} := -1 \]

---

4 Assuming that the settings for these methods are equal or slightly the same, which they are at the present moment.
7 CLASSIFICATION METHODS - THE ALGORITHMS

7.1 Information about the presentation

This chapter contains the algorithms that the methods from chapter 5 consist of. It could be skipped without affecting the understanding of the rest of the report. Chapter 5 contained a short description of the principles of every method, but no explanation was given of how the results are obtained. This chapter will in a detailed way describe each function with help of algorithms in step by step form, also referred to as pseudo code. To every algorithm there is a figure making the explanation easier. All algorithms use many parameters. These parameters are the same and have the same names as the parameters in the source code. This is for convenience. The source code, written in C++, see [3], is left out in this report because this chapter gives all necessary information and could also be understood by those who are not experienced with C++ code. Every index of a section correspond with the index in chapter 5 where the methods first where described. Section 5.2 and 7.2 deals with the same method and so on.

Only parameters and variables needed for the explanation are considered. Each function takes many variables and parameters as input, but only those who will be needed for the understanding will be used here. Global variables like the height image (height), width, profile, outer profile, or the boundaries of the region of interest (x₀, y₀, x₁, and y₁) are used directly without being defined in these descriptions, but in the real code they must be, see table 7-1.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>‘height (x, y)’</td>
<td>The height image</td>
</tr>
<tr>
<td>‘width’</td>
<td>The maximum width of a vehicle</td>
</tr>
<tr>
<td>‘profile (x)’</td>
<td>The height profile</td>
</tr>
<tr>
<td>‘outer profile (x)’</td>
<td>The outer profile</td>
</tr>
<tr>
<td>‘x₀, y₀, x₁, y₁’</td>
<td>The limits of the region of interest</td>
</tr>
</tbody>
</table>

Table 7-1. Variables that not will be defined before use in the pseudo code.

Also variables or parameters that are needed for the functionality of the code will be left out here. Parameters in the methods are written in bold characters whenever they are used in the pseudo code, except in the parameter listing in the beginning. Internal variables defined inside the pseudo code will be written in italic the first time they are used, then in ordinary style. These variables are used in order to shorten the text and their names will consist of an explanation of their use. Global variables like height and profile, mentioned above, will be written with ordinary text, see table 7-2 for a summary.

<table>
<thead>
<tr>
<th>Type of data</th>
<th>Notation</th>
<th>Example (first…then)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter related to a method</td>
<td>Plain when defined, then bold</td>
<td>‘minHeight’ … ‘minHeight’</td>
</tr>
<tr>
<td>Internal variables</td>
<td>Italic when defined, then plain</td>
<td>‘yC’ … ‘yC’</td>
</tr>
<tr>
<td>Global variables</td>
<td>Plain</td>
<td>‘profile’ … ‘profile’</td>
</tr>
</tbody>
</table>

Table 7-2. The different pseudo code data types and their corresponding notations.
All parameters to a method are listed in the beginning of each description. After every parameter follows the latest value for large vehicles in brackets. The parameters concerning length, width or height does not have the dimension meter. Length and width are given in pixels and height has another unit. For conversions, see table 7-3.

<table>
<thead>
<tr>
<th>Dimension</th>
<th>pixel (p) or height unit (hu)</th>
<th>meter (m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Length</td>
<td>( 1 \text{ p} = 0.04 \text{ m} )</td>
<td>( 1 \text{ m} = 25 \text{ p} )</td>
</tr>
<tr>
<td>Width</td>
<td>( 1 \text{ p} = 0.04 \text{ m} )</td>
<td>( 1 \text{ m} = 25 \text{ p} )</td>
</tr>
<tr>
<td>Height</td>
<td>( 1 \text{ hu} = 0.09 \text{ m} )</td>
<td>( 1 \text{ m} = 11.1 \text{ hu} )</td>
</tr>
</tbody>
</table>

Table 7-3. Conversion table for length, width and height into parameter values.

In the algorithms there will often be a need of answering the question – for a given \( x \), what is the y-position where the left (right) side first is detected? The wanted positions are marked in figure 5.10. In order to filter noise and to detect the y-position for a given \( x \), the following strategy is used:

**Algorithm 7.1. Finding the y-position of the outer side for a given \( x \).**
1. Choose a search length, \( s \), and a minimum count number, \( n \)
2. Start with \( y = y_0 \) in the left case (or \( y_1 \) in the right case)
3. Count the number of values greater than zero (or a minimum height) in the row consisting of \( [x-s+1, y] \) to \( [x, y] \)
4. If the count from step 3 are at least \( n \), then the wanted position is \( y \), and end of search
5. If not, \( y := y + 1 \) (or \( y := y - 1 \) in the right case)
6. Continue searching (go to step 3) until a position is found or till the other side (\( y = y_1 \)) is reached
7. END

Figure 7.1 displays the principle and where it can be seen where the correct y-position for the left side (or right) should be.

![Correct y-position](image)

**Figure 7.1. The principle of finding the y-position for a side. This is an enlarged part of a side of a vehicle. The correct position is marked with an arrow. Here the search length is 13 and the minimum count number is 7.**
7.2 Edge across

7.2.1 Edge across – single line

Source code function name: `edgePosition_3 (..)`

Purpose:
Find single lines of pixels in the y-direction. The minimum number, \( n \), of pixels within each line, in order to be defined as an edge, is given as a parameter, \( \text{quote} \), that states \( n \) relative to the width, \( \text{quote} = n / \text{width} \).

Parameters of interest:
\( \text{quote} (0.80) \)

A: For all \( x \) of \([x_0, x_1]\):
1. Count all \( \text{height}(x,y) \geq \text{minHeight} \).
2. If the count \( \geq \text{quote} * \text{width} \), then an edge has been found for \( x \).

7.2.2 Edge across - wider edge followed by empty area

Source code function name: `edgePositionGreat2 (..)`

Purpose:
Find edge in front of a relatively empty area and save each position, \( x \), where this occurs.

Parameters of interest:
\( \text{bFull} (3), \text{nFull} (2), \text{qFull} (0.80), \text{minHeightFull} (45), \)
\( \text{bEmpty} (22), \text{nEmpty} (4), \text{qEmpty} (0.20), \text{minHeightEmpty} (30), \text{yEmptyOff} (7) \)

\[ y_0 + \text{yEmptyOff} \]
\[ y_1 - \text{yEmptyOff} \]
\[ \ldots \]
\[ \ldots \]
\[ \ldots \]
\[ \ldots \]
\[ \text{bEmpty} \]
\[ \text{bFull} \]
\[ y_0 \]
\[ y_1 \]
\[ x \]

Figure 7.2. The sliding search area with the edge area in front of the empty

Do for all \( x \) of \([x_0, x_1]\):

A. For the edge area:
1. Count the number of rows of length \( \text{bFull} \) (x-direction) which have \( \text{nFull} \) or more height values greater than or equal to \( \text{minHeightFull} \).
2. The total number of such rows must be equal or greater than \( \text{qFull} * \text{width} \).

B. For the empty area:
1. Count the number of rows of length \( \text{bEmpty} \) (x-direction) which have \( \text{nEmpty} \) or less height values greater than or equal to \( \text{minHeightEmpty} \).
2. The total number of such rows must be less than or equal to \( \text{qEmpty} * (\text{width} - 2 * \text{yEmptyOff}) \).

If A and B are fulfilled => an edge has been found for \( x \).

7.2.3 Edge across - wider edge surrounded by empty areas

Source code function name: `edgeBEF (..)`
Purpose:
Find edge that has empty areas in front of it and behind and save each position, $x$, where this occurs.

Parameters of interest:
- $bb$ (7), $nb$ (2), $qb$ (0.20), $yb_{off}$ (7), $minH_b$ (12),
- $be$ (4), $ne$ (2), $qe$ (0.85), $minH_e$ (12),
- $bf$ (5), $nf$ (1), $qf$ (0.20), $yf_{off}$ (7), $minH_f$ (12)

Figure 7.3. The sliding search area width the edge area surrounded by the empty ones.

Do for all $x$ of $[x_0, x_1]$:

A: For the front (and rear) empty area:
1. Count the number of rows of length $bf$ ($bb$) (x-direction) that have $nf$ ($nb$) or less height values greater than or equal to $minH_f$ ($minH_b$).
2. The total number of such rows must be equal or less than $qf*(width-2*yf_{off})$ (and equivalent for $qb$, $yb_{off}$).

B: For the edge area:
1. Count the number of rows of length $be$ (x-direction) that have $ne$ or more height values greater than or equal to $minH_e$.
2. The total number of such rows must be equal or greater than $qe*width$.

If A (both front and rear) and B are fulfilled => an edge has been found for $x$.
The relative number of height values greater than or equal to $minH_e$ in the edge area is associated with $x$.

7.3 Count height values

Source code function name: count_rel (...) 

Purpose:
Count the relative number of height values in a specific area and thereby estimate the degree of variation of the surface.

Parameters of interest:
- $x_{Low}$ ($x_1-30$), $y_{Low}$ ($y_0+5$), $x_{High}$ ($x_1-10$), $y_{High}$ ($y_1-5$), $MIN_H$ (45)

Figure 7.4. The counting area, small rectangle, in the region of interest.
A: For all values in \([x_{\text{Low}}, x_{\text{High}}] \times [y_{\text{Low}}, y_{\text{High}}]::
1. Count, c, all height values greater than or equal to \(\text{MIN}_H\) in the area.
Return \(c / ( (x_{\text{High}}-x_{\text{Low}}+1) \times (y_{\text{High}}-y_{\text{Low}}+1) )\), i.e. the relative count.

### 7.4 Outer profile height differences

Source code function name: `maxAverageDiff2 (...)`
Purpose:
Find height differences in the outer profile.
Parameters of interest:
- `length` (11), `x_{\text{Low}}` ((\(x_0+x_1\))/2), `x_{\text{High}}` (\(x_1-10\))

![Diagram](image)

Figure 7.5. The averaging area and length in the outer profile.

A: For all \(x\) of \([x_{\text{Low}}, x_{\text{High}}]::
1. For each point, \(x\), in the outer profile, take the average from outer profile(\(x\)) down to and outer profile(\(x-\text{length}+1\)).
2. Save the smallest, \(s\), and greatest, \(g\), average values.

Values smaller than a minimum constant, \(a\), will be disregarded in the averaging and will have no effect what so ever. If all values in an averaging interval are smaller than \(a\), this interval (the averaged value) will be disregarded.

Return \(g-s\), if they were found, else return -1

### 7.5 Gap in profile

Source code function name: `gap_detection2 (...)`
Purpose:
Find a gap in the profile, which is defined, as a low area behind a higher.
Parameters of interest:
- `x_{\text{Lower}}` ((\(x_0+x_1\))/2), `x_{\text{Upper}}` (\(x_1\)), `length_r` (7), `length_f` (11), `minHeight` (12)

![Diagram](image)
A: For all \(x\) of \([x_{\text{Lower}}, x_{\text{Upper}}]\):
1. Average from \(\text{profile}(x-0.5*\text{length}_r)\) up to \(\text{profile}(x+0.5*\text{length}_r)\). This result is saved in a vector, \(\text{minAver}[x]\). Repeat the averaging for all \(x\), but use \(\text{length}_f\) instead and place the result in \(\text{maxAver}[x]\). Averaging is done accordingly to the method in section 7.4.

B: For all \(x\) of \([x_{\text{Lower}}, x_{\text{Upper}}]\):
1. Find the greatest value, \(\max = \max\text{Aver}[x']\), where \(x'\) has a position in front of \(x\).
2. If \(\min = \min\text{Aver}[x] > \min\text{Height}\), set the difference, \(\text{diff} = \max - \min\). If \(\text{diff}\) is the greatest value so far, then set the maximum difference, \(\max\text{Diff} = \text{diff}\).

The difference, \(\max\text{Diff}\), is the greatest height difference where the higher averaged value is positioned in front of the smaller when \(x\) is in \([x_{\text{Lower}}, x_{\text{Upper}}]\).

7.6 Parallel sides

Source code function name: \texttt{bus\_front (...)}

Purpose:
Estimate in which extent the front left and right sides are parallel and straight.

Parameters of interest:
\texttt{frontOffset (8), sideWidth (7), countLength (12), minCount (7), minHeight (20)}

This function is much more complicated than the previous ones. The major steps are:

- place 3 points with equal spacing in the middle of each front side
- adjust a straight line to the points for each side
- determine how parallel and straight the sides are

A: The \(x\)-positions (same for left and right side):
\(xa = x1 - \text{frontOffset}\),
\(xb = xa - \text{countLength}\),
\(xc = xb - \text{countLength}\)
Before finding the y-values - which should be positioned in the middle of each side (y-direction) for the three x-values - the beginning and ending of each side at the actual x, in the y-direction, must be found.

B: Calculation of a count profile:
1. Calculate a count profile, \( \text{count}(y) \), of the area \([x-\text{countLength}+1, x]*[y0, y1] \). Count in the x-direction and project the values on the y-axes. Only values \( \geq \text{minHeight} \) are counted.

C: For all 3 x, find the outer y-values, \( y\text{Outer}[x] \), for the left side:
1. Set \( y = y0 \).
2. If \( \text{count}(y) \geq \text{minCount} \), outer edge is found \( \Rightarrow y\text{Outer}[x] = y \).
3. Else if \( \text{count}(y) < \text{minCount} \), set \( y = y+1 \) and go to 2.

D: For all 3 x, find the inner y-values, \( y\text{Inner}[x] \), for the left side:
1. If a corresponding outer y-value, \( y\text{Outer}[y] \), were found in C, set \( y = y\text{Outer}[x] \), else end.
2. If \( \text{count}(y+1) < \text{minCount} \), end of left side is found \( \Rightarrow y\text{Inner}[x] = y \).
3. Else if \( \text{count}(y) \geq \text{minCount} \), set \( y = y+1 \) and go to 2.

Steps C and D are repeated for the right side in an analog fashion

E: Controlling the values \( y\text{Outer}[x] \) and \( y\text{Inner}[x] \):
1. If \( \text{abs} ( y\text{Outer}[x] - y\text{Inner}[x] ) < \text{sideWidth} \), then the spacing between the values is not too great (the width of a side is not too great) and the values are OK.

F: If valid \( y\text{Outer}[x] \) and \( y\text{Inner}[x] \) were found in C, D and E, then the correct y-values, \( y\text{C}[x] \), can be calculated, i.e. positioned in the middle of each side. Do for all 3 x, left and right sides:
1. \( y\text{C}[x] = \text{sum}(y*\text{count}[y]) / \text{sum}(\text{count}[y]) \). Sum from \( y = y\text{Outer}[x] \) to \( y = y\text{Inner}[x] \).

The positions obtained from F are a kind of weighted positions rather than just the middle between \( y\text{Outer}[x] \) and \( y\text{Inner}[x] \). All 6 points achieved in F (3 from each side) must be OK according to C, D, and E if we should continue. We then have all 6 points \((x, y)\) and can go on.

The next step is to fit a line to the left and right sets of points respectively.
G: Adjust a line to 3 points in the best possible way:
By using the least square method, see appendix C, the values $k$ and $m$ in the equation

$$y' = kx + m,$$

can be obtained. If $y' = y$ then the point is on the line.

RESULTS - OBTAINING THE ANGLE DIFFERENCE
From the $k$-value the angle, $\nu$, can be obtained by

$$\nu = \arctan(k)$$

The angular difference between the right and left sides is given as result.

RESULTS - MAXIMUM LINE TO POINT DEVIATION
The shortest distance, $h$, from a point $(x, y)$ to the corresponding line is

$$h = |(kx + m - y) \cos(\nu)|,$$

where $\nu$ is the corresponding angle, from above.

The maximum $h$ from the left side is compared with the maximum $h$ from the right side and the greatest value of these two is given as result.

RESULTS - MAXIMUM WIDTH DIFFERENCE
In step F, the corresponding $y$-values (the weighted values) to the corresponding $x$-values are calculated. The distances from the left side to the right side for points with the same $x$-value are calculated, i.e. the distances in the $y$-direction. We then have 3 values of the width. The greatest difference between these 3 values is calculated and given as result.

RESULTS - MAXIMUM HEIGHT DIFFERENCE
In each side there are 3 areas and from every area the greatest height value is found. Then the greatest height difference, $dL$, between the values from the left side is found and the same is done for the right side, $dR$. The biggest of $dL$ and $dR$ is then given as result.

An area is defined by $[x-searchLength+1, x] \times [yOuter\{x\}, yInner\{x\}]$, where $yOuter\{x\}$ and $yInner\{x\}$ are the values used in step C, D, and E and $x$ is one out of 3 $x$-positions from step A.

Summary

The results from this method are
- angle difference between the two lines
- maximum point to line deviation
- maximum width difference when constant $x$
- maximum height difference within left or right side

7.7 Difference in width

7.7.1 Difference in width – maximum difference

Source code function name: side_width_1 (...)  
Purpose: Measure the width difference and return the greatest value. 
Parameters of interest:
**Figure 7.9. The borders that limits the search for width differences in areas of \( b \) pixels.**

A: For all \( x \) of \([x_{\text{LowerLimit}} + b - 1, x_{\text{UpperLimit}}]\):
1. Find the \( y \)-positions, \( y_{\text{Left}} \) and \( y_{\text{Right}} \), of the left and right sides
   according to algorithm 7.1. Use count length, \( s = b \) and the minimum count number, \( n = \text{minNr} \). Height values smaller than \( \text{minH} \) should not be counted.
2. The positions must not differ more than \( y_{\text{Width}} \) from \( y_0 \) in the left case or \( y_1 \) in the right case.
3. If \( y_{\text{Left}} \) and \( y_{\text{Right}} \) were found (step 2 true), calculate the width, \( w = y_{\text{Right}} - y_{\text{Left}} + 1 \).
4. If \( w > \text{max} \) (greatest width so far), then set \( \text{max} = w \).
5. If \( w < \text{min} \) (smallest width so far), then set \( \text{min} = w \).

Return \( \text{max-min} \).

### 7.7.2 Difference in width – lorry type

Source code function name: `side_width_3 (..)`

Purpose:
- Find the greatest width difference where the smaller value is positioned in front of the larger
- The offset in the \( y \)-direction, from the corresponding boundary of the region of interest, to the
  side of the vehicle is smaller, on both left and right sides, for the greater width.

Parameters of interest:
- \( x_{\text{Lower}} \) (x1-44), \( x_{\text{Upper}} \) (x1-6), \( \text{boxLength} \) (11), \( \text{nr} \) (6), \( \text{L\_off} \) (1), \( \text{R\_off} \) (1),
- \( \text{minHeight} \) (12), \( y_{\text{Width}} \) (8)

**Figure 7.10. The borders that limits the search for width differences in areas of \( \text{boxLength} \) pixels.**
A: For all x of \([x_{\text{Lower}}, x_{\text{Upper}}]\):
1. Find the y-positions, \(y_{\text{Left}}\) and \(y_{\text{Right}}\), of the left and right sides according to algorithm 7.1. Use count length, \(s = \text{boxLength}\) and the minimum count number, \(n = nr\). Height values smaller than \(\text{minHeight}\) should not be counted.
2. The positions must not differ more than \(y_{\text{Width}}\) from \(y_{0}\) in the left case or \(y_{1}\) in the right case.
3. If step 2 is true, search among \([x_{\text{Lower}}, x-1]\) and find \(y_{\text{LeftRear}}\) and \(y_{\text{RightRear}}\) in the same sense as in steps 1 and 2 (\(y_{\text{Left}}\) and \(y_{\text{Right}}\)).
4. If \(y_{\text{LeftRear}} \leq y_{\text{Left}}-L_{\text{off}}\) AND \(y_{\text{RightRear}} \geq y_{\text{Right}}+R_{\text{off}}\), calculate the two widths, \(w = y_{\text{Right}}-y_{\text{Left}}+1\) and \(w_{\text{Rear}} = y_{\text{RightRear}}-y_{\text{LeftRear}}+1\).
5. If \(w_{\text{Rear}}-w > \text{max}\) (greatest such difference so far), \(\text{max} = w_{\text{Rear}}-w\).
Return \(w_{\text{Rear}}-w\).

7.8 Height difference between flat areas

7.8.1 Height difference between flat areas – maximum difference

Source code function name: flat_height2 (...)  
Purpose:  
Find the greatest height difference between flat areas in a height profile.  
Parameters of interest:  
\(x_{\text{LowerMax}} ((x_{0}+x_{1})/2-11), x_{\text{UpperMax}} (x_{1}), x_{\text{LowerMin}} (x_{1}-30), x_{\text{UpperMin}} (x_{1}),\)  
\(\text{minLengthMax} (15), \text{minLengthMin} (6), \text{maxDiff} (1), \text{minHeight} (12), \text{maxHeight} (110)\)

Figure 7.11. A profile where a search is conducted to find flat areas of minimum length \(\text{minLengthMax}\) (\(\text{minLengthMin}\)) with a maximum deviation of \(\text{maxDiff}\) from profile(x).

The first thing to do is to calculate a profile, which will contain the lengths of the flat areas. The definition of a flat area, with certain length, starting in the position x is (definition 5.3):

For every point, \(x\), in the profile, the greatest number, \(N\), is found for which holds  
\[|\text{profile}(x) - \text{profile}(x-i)| \leq \epsilon, \quad \text{where } i = 0, 1, 2, ..., N \text{ and } \epsilon \text{ is the maximum allowed deviation from } h(x).\]  
The length, \(L\), of the flat area is then \(L = N + 1\).

A: For all x of \([x_{0}, x_{1}]\):
1. Calculate a profile, \(\text{flatProfile}[x]\), containing the lengths of the flat areas. Use \(\text{maxDiff}\) as the greatest allowed deviation.
B: Find the maximum height value of a flat area for x of \([x_{\text{LowerMax}}, x_{\text{UpperMax}}]\):
1. If flatProfile[x] ≥ minLengthMax AND outer profile[x] > max (greatest such value so far), then max = outer profile[x].

C: Find the minimum height value of a flat area for x of \([x_{\text{LowerMin}}, x_{\text{UpperMin}}]\):
1. If flatProfile[x] ≥ minLengthMin AND outer profile[x] < min (smallest such value so far), then min = outer profile[x].

It could be noted that the outer profile(x) must be in the interval \([\text{minHeight}, \text{maxHeight}]\) in steps B and C.

Return max-min.

The reason of having two different minimum lengths of a flat area is that it is more probable that the long flat areas have a higher height. Short high areas may be disturbances. You also want another search area because it is easy to find minimum in the front, but it may originate from the foremost front and this area is preferred to avoid when searching for small values because it is often not trustworthy.

7.8.2 Height difference between flat areas – step

Source code function name: \texttt{step (...)}

Purpose:
Find the greatest height difference in a height profile where the demand is that a flat area should be followed (to the rear) by a sudden increase and where the higher area also should be flat, a step that is.

Parameters of interest:
xUpper \(x_{1-6}\), searchLength \(33\), stepLength \(5\), minFlatLengthFront \(6\), minFlatLengthRear \(11\), maxFlatDiff \(1\), minHeight \(40\)

\[\begin{array}{c}
\text{minFlatLengthRear} \\
\text{minFlatLengthFront} \\
\text{stepLength}
\end{array}\]

Figure 7.12. A profile with a step in the front. The rise length of the height increase must not be greater than stepLength.

A: For all x of \([x_0, x_1]\):
1. Calculate a profile, \texttt{flatProfile}[x], containing the lengths of the flat areas. Use \texttt{maxFlatDiff} as the greatest allowed deviation. See algorithm 7.1 for the definition of flat.

B: Find the greatest step difference for all x of \([x_{\text{Upper}}-\text{searchLength}, x_{\text{Upper}}]\):
1. Set $flatLengthFront = flatProfile[x]$ (length of front flat area) and
   $heightFront = profile[x] > minHeight$ (heightFront: height of front flat
   area).
2. If $flatLengthFront > minFlatLengthFront$, go on, else end (next x).
3. The rear flat area must start within $stepLength$ pixels from the end of the
   front flat area ($x-flatLengthFront+1$), i.e.
   $flatProfile[xRear] > minFlatLengthRear$,
   where $xRear$ is in $[x-flatLengthFront-stepLength, x-flatLengthFront]$, see
   figure 7.12.
4. Find the greatest height value, $heightRear = profile[xRear]$, that fulfils
   the requirements in step 3.
5. If $heightRear-heightFront > max$ (greatest such value so far), then
   max = $heightRear-heightFront$ (profile difference).

Return max.
8 RESULTS

There has been limited access to the unit that receives images from the E4 highway. For best possible evaluation of the classifier, probably more images would be needed. In this result study 436 images have been used. These images consist of large vehicles and trailers have been removed in order to make the evaluation easier. Trailers will not effect the classification anyway, because there already is a functional detector for them. Small vehicles, as mentioned in section 3.1, will be left without any results, because the classes for these vehicles have not yet been decided and therefore no classification results can be obtained. Consequently, only results from large vehicles are regarded. Not all images have been used when developing the methods and setting the parameters.

Out of 436 vehicles there were 385 lorries, of which 4 were classed as busses and there were 51 busses, all classified correctly. The relative error is then $4 / 436 \approx 0,009 = 0,9 \%$.

The erroneous lorries can be seen in figure 8.1.

Table 8-1 shows how often the result from each method corresponds to the final class or not. In the columns marked “Right”, “Wrong” and “Nothing” are the number of times when each point, $p$, is related to the sum of points, $sum$, in the following way:

Right: $sum \geq 0$ AND $p > 0$ or $sum < 0$ AND $p < 0$
Wrong: $sum \geq 0$ AND $p < 0$ or $sum < 0$ AND $p > 0$
Nothing: $p = 0$

<table>
<thead>
<tr>
<th>Method</th>
<th>Right</th>
<th>Wrong</th>
<th>Nothing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edge across 1 (5.2.1)</td>
<td>342</td>
<td>4</td>
<td>90</td>
</tr>
<tr>
<td>Edge across 2 (5.2.2)</td>
<td>185</td>
<td>0</td>
<td>251</td>
</tr>
<tr>
<td>Edge across 3 (5.2.3)</td>
<td>43</td>
<td>0</td>
<td>393</td>
</tr>
<tr>
<td>Count values (5.3)</td>
<td>383</td>
<td>24</td>
<td>29</td>
</tr>
<tr>
<td>Outer profile (5.4)</td>
<td>362</td>
<td>17</td>
<td>57</td>
</tr>
<tr>
<td>Gap in profile (5.5)</td>
<td>79</td>
<td>0</td>
<td>357</td>
</tr>
<tr>
<td>Parallel sides (5.6)</td>
<td>42</td>
<td>0</td>
<td>394</td>
</tr>
<tr>
<td>Diff in width 1 (5.7.1)</td>
<td>46</td>
<td>23</td>
<td>367</td>
</tr>
<tr>
<td>Diff in width 2 (5.7.2)</td>
<td>271</td>
<td>0</td>
<td>165</td>
</tr>
<tr>
<td>Diff in height 1 (5.8.1)</td>
<td>161</td>
<td>0</td>
<td>275</td>
</tr>
<tr>
<td>Diff in height 2 (5.8.2)</td>
<td>16</td>
<td>0</td>
<td>420</td>
</tr>
</tbody>
</table>

Table 8-1. The number of times each point corresponds with the final class for 436 vehicles.

It can be seen that the major part of the methods gives satisfying results. The worst problem is “Diff in width 1” which seems to take a guess every time. The 23 vehicles in the “Wrong” column are most certainly originated from lorries with small difference in width, which are rather common. One can also see that “Count values” and “Outer profile” have rather much erroneous detection. On the other hand are the relative occurrences pretty low.

Probably, the methods “Gap in profile” and “Parallel sides” are the ones with most reliability. The problem for the first one is that there are not so many lorries with the characteristic “gap” or similar height difference needed for the detection. Most busses got detected by “Parallel sides” and there where no lorries receiving a positive point. You can also see that “Diff in width 2” has a very high certainty. The reasons are that the method seems to be a good method and that many lorries have this feature, in contrast to “Gap in profile”.

39
Figure 8.1. The four erroneous vehicles were all of the same type, snowploughs. The points (not zero) for the vehicle above were: edge across 1 1, count values –3, outer profile 2 and diff in width 1 –1, giving a total sum of –1. The other three vehicles had similar points.
9 PROBLEMS AND FURTHER DEVELOPMENT

9.1 General problems

A major problem has been that a lorry can look like a typical bus with constant height and width, but it is rather unusual with a bus having the characteristics of an ordinary lorry, with height and width differences. If a vehicle is shaped in the first way, like a box, it is much harder to classify, because the vehicle could be either of the two types. So, great differences in height and width are preferable.

Another problem associated with the classification methods is that they of course are dependent of the parameters, both the ones directly connected to the methods and those who are involved in the point setting function.

9.2 Difficult types of vehicles

One type of vehicle causing some problems is a tank lorry – transporting milk and gasoline etc. Such lorry has almost constant width and the height profile does not vary much. The loading space usually is cylinder shaped, though, and this sometimes makes it easier to detect differences in the outer profile. However, constant width and straight sides makes a tank lorry detectable for the method “parallel sides” (5.6) and the risk of obtaining a negative point is rather great. In the study from section 8, there were 3 tank lorries and one of them had a total sum of zero. The others had rather low positive sums of points. See figure 9.1 for two tank lorries and note the similarity with the busses in the same figure.

There are also problems among busses, but not so specific. A bus with great contrasts on the roof will have many height values in the corresponding position in the height image. The shapes of the bus will still be box like, but the methods “edge across” (5.2) and “count height values” (5.3) will not be reliable then. The undesired contrasts may have numerous origins, but the most likely reasons are roof hatches (many), round sides and long slope in the front. The first reason may form an edge across the bus and increase the number of height values giving rise to positive points from two methods. If the left and right areas, connecting the roof with the sides, of a bus are rounded rather than sharp, the result is that the sides in the height image also get wider. This may cause problems for the method “parallel sides” because one of the requirements is that the sides must not be too wide. See figure 9.1 for two problematic bus images.
Figure 9.1. Two tank lorries, above, with great similarities to the busses below. The bus to the left has an across going edge that will be detected as a lorry feature. Because of the rounded transitions from roof to sides, the height image to the right will have wider side areas than an ordinary bus, and this will cause problems for the method “parallel sides”.

The points and sums for the vehicles in figure 9.1 are put together in table 9-1.

<table>
<thead>
<tr>
<th>Method</th>
<th>Left lorry</th>
<th>Right lorry</th>
<th>Left bus</th>
<th>Right bus</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edge across 1 (5.2.1)</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Edge across 2 (5.2.2)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Edge across 3 (5.2.3)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Count values (5.3)</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
</tr>
<tr>
<td>Outer profile (5.4)</td>
<td>0</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>Gap in profile (5.5)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Parallel sides (5.6)</td>
<td>0</td>
<td>0</td>
<td>-1</td>
<td>0</td>
</tr>
<tr>
<td>Diff in width 1 (5.7.1)</td>
<td>-2</td>
<td>0</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>Diff in width 2 (5.7.2)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Diff in height 1 (5.8.1)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Diff in height 2 (5.8.2)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Sum</td>
<td>0</td>
<td>1</td>
<td>-3</td>
<td>-2</td>
</tr>
</tbody>
</table>

Table 9-1. The points and sums of the problematic vehicles in figure 9.1.

9.3 Improvements

Although the error rate is low much can be done to make the classifier better and more robust. The first thing is to investigate whether all methods should be part of the classifier. Maybe some methods are too risky and maybe some new method could be added. The function “difference in width – maximum difference” (5.7.1) is not very reliable according to the result in the last chapter. Not only could the functionality of the methods be improved. There are many parameters and the settings of them could need more time and material. Depending on the location where the classifier is to be used, the parameter values must be adjusted to fit the type of vehicles in the region. Certain methods could be really sensitive to some parameter changes. An example is the “difference in width” method. One parameter denotes the minimal height when estimating the width. All height values with a value smaller than that height should be regarded as zero, i.e. ignored. A rather astonishing difference was made when changing the minimal height between 12 (48 cm) and 13 (52 cm) or more. Larger width differences were detected because the side finding algorithm went deeper in the sides, when the minimum height was increased. The best alternative was to choose 12. Otherwise the width differences got to large and unrealistic.
Vehicles travelling in a direction different from that of the road (fast lane change, for example) would be harder to classify but the most probable is that all heavy vehicles travels almost straight ahead. No vehicles in the test material had a direction considerably different from that of the road.
A. APPENDIX - SETTINGS FOR PARAMETERS FOR SMALL VEHICLES

The parameter values for small vehicles were left out in section 7 in order not to confuse the reader. Because there are no existing classes for small vehicles, the results will not have any real meaning.

Gap in profile (7.5)
Source code function name: gap_detection2 (…)
Parameters of interest:
xLower (x0+20), xUpper (x1), length_r (7), length_f (7)

Parallel sides (7.6)
Source code function name: bus_front (…)
Parameters of interest:
frontOffset (8), sideWidth (8), countLength (10), minCount (6), minHeight (30)

Difference in width – maximum difference (7.7.1)
Source code function name: side_width_1 (…)
Parameters of interest:
xLowerLimit ((x0+x1)/2-11), xUpperLimit (x1-11), b (11), minNr (6), yWidth (8), minH (12)

Difference in width – lorry type (7.7.2)
Source code function name: side_width_3 (…)
Parameters of interest:
xLower ((x0+x1)/2-11), xUpper (x1-11), boxLength (11), nr (6), L_off (1), R_off (1), minHeight (12), yWidth (8)

Height difference between flat areas – maximum difference (7.8.1)
Source code function name: flat_height2 (…)
Parameters of interest:
xLowerMax (x1-25), xUpperMax (x1), xLowerMin (x0), xUpperMin (x1-12), minLengthMax (4), minLengthMin (12), maxDiff (1), minHeight (20), maxHeight (110)

Height difference between flat areas – step (7.8.2)
Source code function name: step (…)
Parameters of interest:
xUpper (x1-6), searchLength (25), stepLength (2), minIncrease (10), minFlatLengthFront (4), minFlatLengthBack (10), maxFlatDiff (0), minHeight (40)
B. APPENDIX - SETTINGS FOR THE POINT SETTING FUNCTION

Depending on whether the actual vehicle is small or large there are two different functions. In this chapter the settings for these functions are described method for method. If no alternative is represented the point is set to zero.

Large vehicles

Source code function name: set_pointsL (...)

**Edge across – single line (5.2.1):**
Any edge in \([x1-34, x1-13]\) => +1

**Edge across – wider edge followed by empty area (5.2.2):**
Any edge in \([(x0+x1)/2, x1-13]\) AND width difference found by 5.7.2 => +1

**Edge across – wider edge surrounded by empty areas (5.2.3):**
Any edge in \([(x0+x1)/2, x1-21]\) AND width difference found by 5.7.2 => +1

**Count height values (5.3):**
The count area is \([x1-30, x1-10]\)*\([y0+5, y1-5]\).

\[
0.00 \leq \text{count} \leq 0.15 \Rightarrow -3 \\
0.15 < \text{count} \leq 0.25 \Rightarrow -2 \\
0.25 < \text{count} \leq 0.30 \Rightarrow -1 \\
0.30 < \text{count} \leq 0.35 \Rightarrow 0 \\
0.35 < \text{count} \leq 0.45 \Rightarrow +1 \\
0.45 < \text{count} \leq 1.00 \Rightarrow +2
\]

**Outer profile height differences (5.4):**
The controlled area is \([x1-0.5(x1-x0), x1-10]\).

\[
0 \leq \text{height diff} < 8 \Rightarrow -1 \\
8 \leq \text{height diff} < 15 \Rightarrow 0 \\
15 \leq \text{height diff} < 30 \Rightarrow +1 \\
30 \leq \text{height diff} \Rightarrow +2
\]

**Gap in profile (5.5):**
Definitions:
\(\text{max} = \) the higher height
\(\text{min} = \) the lower height

\[
12 < \text{min} < 0.5*\text{max} \ \text{AND} \ \text{min} < 100 \Rightarrow +2 \\
12 < 0.5*\text{max} \leq \text{min} < 0.75*\text{max} \ \text{AND} \ \text{min} < 100 \Rightarrow +1
\]
**Parallel sides** (5.6)

Definitions:
angle diff = difference of the angles of the lines,
width diff = maximum difference between left and right points for the same x-position (not integer),
line diff = maximum deviation from a point to corresponding line (not integer),
height diff = greatest height difference within each side,
points count = points received from ‘count height values’

0 ≤ angle diff < 0,1 [radians] AND
0 ≤ line diff < 0,5 AND
0 ≤ width diff < 2 AND
0 ≤ height diff < 16 AND
points count ≤ 0 => -2

ELSE
0 ≤ angle diff < 0,15 [radians] AND
0 ≤ line diff < 2 AND
0 ≤ width diff < 2 AND
0 ≤ height diff < 16 AND
points count ≤ 0 => -1

0,3 < angle diff ≤ 0,4 [radians] AND
3 < line diff ≤ 4 AND
4 < width diff ≤ 5 => +1

0,4 < angle diff [radians] AND
3 < line diff ≤ 4 AND
4 < width diff ≤ 5 => +2

**Difference in width – maximum difference** (5.7.1):
If any points are to be set, no valid width from 5.7.2, pointsLT, must not be found.
width diff = 0 AND pointsLT = 0 => -3
width diff = 1 AND pointsLT = 0 => -2
width diff = 2 AND pointsLT = 0 => -1

**Difference in width – lorry type** (5.7.2):
width diff = 4 => +1
4 < width diff => +2

**Height differences between flat areas – maximum difference** (5.8.1):
10 < height diff ≤ 16 => +1
16 < height diff => +2

**Height differences between flat areas – step** (5.8.2):
14 < height diff ≤ 19 => +1
19 < height diff => +2
Small vehicles

Source code function name: set_pointsS (...)

**Gap in profile** (5.5):
Definitions:
\[ \text{max} = \text{the higher height} \]
\[ \text{min} = \text{the lower height} \]

12 < min < 0,5*max AND min < 100 => +2
12 < 0,5*max ≤ min < 0,75*max AND min < 100 => +1

**Parallel sides** (5.6):
Definitions:
\[ \text{angle diff} = \text{difference of the angles of the lines,} \]
\[ \text{width diff} = \text{maximum difference between left and right points for the same x-position (not integer),} \]
\[ \text{line diff} = \text{maximum deviation from a point to corresponding line (not integer),} \]
\[ \text{height diff} = \text{greatest height difference within each side,} \]
\[ \text{points count} = \text{points received from ‘count height values’} \]

0 ≤ angle diff < 0,1 [radians] AND
0 ≤ line diff < 0,5 AND
0 ≤ width diff < 2 AND
0 ≤ height diff < 16 AND
points count ≤ 0 => -2

ELSE
0 ≤ angle diff < 0,15 [radians] AND
0 ≤ line diff < 2 AND
0 ≤ width diff < 2 AND
0 ≤ height diff < 16 AND
points count ≤ 0 => -1

0,3 < angle diff ≤ 0,4 [radians] AND
3 < line diff ≤ 4 AND
4 < width diff ≤ 5 => +1

0,4 < angle diff [radians] AND
3 < line diff ≤ 4 AND
4 < width diff ≤ 5 => +2

**Difference in width - maximum difference** (5.7.1):
*If any points are to be set, the point from 5.7.2, pointsLT, must be zero*
width diff = 0 AND pointsLT = 0 => -3
width diff = 1 AND pointsLT = 0 => -2
width diff = 2 AND pointsLT = 0 => -1

**Difference in width - lorry type** (5.7.2):
1 < width diff ≤ 3 => +1
3 < width diff => +2

**Height differences between flat areas - maximum difference** (5.8.1):
10 < height diff => +1

**Height differences between flat areas - step** (5.8.2):
14 < height diff => +1
C. APPENDIX – LEAST SQUARE METHOD FOR 2 DIMENSIONS

In the method “parallel sides” (7.6) a line is fitted in the best possible way to three points in the left and right sides respectively. By the best possible way it is meant the line that minimises the variance in the y-direction. The x-points are fixed relative to each other. The equation for a line is

\[ y' = kx + m \]

Here \( y' \) is the estimate, \( x \) is a value from the data set and \( k \) and \( m \) are constants. The variance, \( v \), of \( n \) points can be estimated as

\[ v = \frac{1}{n} \sum_{j=0}^{j=n-1} (y_j - y'_j)^2 \]

By using the least square method this expression get as small as possible. The estimations for \( k \) and \( m \) are

\[ k = \frac{n \sum x_j y_j - \sum x_j \sum y_j}{n \sum x_j^2 - (\sum x_j)^2} \]
\[ m = \frac{\sum y_j \sum x_j^2 - \sum x_j \sum x_j y_j}{n \sum x_j^2 - (\sum x_j)^2} \]
REFERENCES

[1] Lars-Gunnar Rinnfors, tel. +46 36 19 43 00, e-mail larsgunnar.rinnfors@kapsch.se

[2] Hackman, Peter, Kossan, Hut, Hyfs och Hållning Productions 1997, -


The company address: Kapsch TrafficCom AB
Box 1063
551 10 JÖNKÖPING
Sweden
På svenska

Detta dokument hålls tillgängligt på Internet – eller dess framtida ersättare – under en längre tid från publiceringsdatum under förutsättning att inga extra-ordinära omständigheter uppstår.

Tillgång till dokumentet innebär tillstånd för var och en att läsa, ladda ner, skriva ut enstaka kopior för enskilt bruk och att använda det oförändrat för ickekommersiell forskning och för undervisning. Överföring av upphovsrätten vid en senare tidpunkt kan inte upphäva detta tillstånd. All annan användning av dokumentet kräver upphovsmannens medgivande. För att garantera äktheten, säkerheten och tillgängligheten finns det lösningar av teknisk och administrativ art.

Upphovsmannens ideella rätt innefattar rätt att bli nämnd som upphovsman i den omfattning som god sed kräver vid användning av dokumentet på ovan beskrivna sätt samt skydda mot att dokumentet ändras eller presenteras i sådan form eller i sådant sammanhang som är kränkande för upphovsmannens litterära eller konstnärliga anseende eller egenart.

För ytterligare information om Linköping University Electronic Press se förlagets hemsida http://www.ep.liu.se/

In English

The publishers will keep this document online on the Internet - or its possible replacement - for a considerable time from the date of publication barring exceptional circumstances.

The online availability of the document implies a permanent permission for anyone to read, to download, to print out single copies for your own use and to use it unchanged for any non-commercial research and educational purpose. Subsequent transfers of copyright cannot revoke this permission. All other uses of the document are conditional on the consent of the copyright owner. The publisher has taken technical and administrative measures to assure authenticity, security and accessibility.

According to intellectual property law the author has the right to be mentioned when his/her work is accessed as described above and to be protected against infringement.

For additional information about the Linköping University Electronic Press and its procedures for publication and for assurance of document integrity, please refer to its WWW home page: http://www.ep.liu.se/

© Adam Jarl