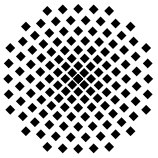


University of Stuttgart
Institute for Signal Processing and System Theory
Professor Dr.-Ing. B. Yang

Bachelorthesis S1132

Extended Human Upper Body Detection

Erweiterte Oberkörpererkennung

Author: Larissa T. Triess

Date of work begin: 04/14/2015
Date of submission: 08/03/2015

Supervisor: Roman Streubel, M.Sc.

Keywords: Matlab Cascade Object Detector, Features, Training, Parameters, Simulated Images, Real Images, Ground Truth Data

Pedestrian detection has gained a lot of importance in the last years. Due to a great number of different postures and clothing the detection of humans is not trivial. The **Matlab Cascade Object Detector** possesses a pretrained detector for upper body detection, which does not provide reliable detection of persons at side view. Therefore, we decided to train four detectors, one for each plane of rotation (back, front, right, and left), for each parameter setting. With the default parameters we achieved an improvement of recall for persons at back view of 79.27%, from 13.17% to 92.44% and for left side view an improvement of 73.25%, from 7% to 80.25%. As the training function has six parameters with a wide value range, a lot of parameter settings are possible. The most significant ones are presented in this work.

Contents

1. Introduction

Object recognition is gaining more and more importance in many fields of application and has been improved in the last years. The detection of humans is crucial for surveillance or automated driving. It has to be very accurate and reliable. Due to various postures, clothing, hair style and body structure the recognition of humans is complex and immature. The aim of this work is to train a detector in order to obtain reliable detection of humans in all planes of rotation. The detection of an upper body is sufficient for detecting humans, because every human has an upper body. Additionally the upper body detection is more robust against occlusion, which mostly occurs on the lower body.

We decided on the **Matlab Cascade Object Detector** which already has some pretrained detection models, among others, a detector for upper bodies. In order to create a custom detector Matlab provides a training function for this detector. With the several parameters many different evaluation results can be achieved, defining the performance of the detector. In order to find out about the influence of one single parameter, our first step was to select values for the parameters, train the detectors and then evaluate the results as a function of the single parameters. Afterwards we trained detectors with the parameter settings, that achieved best performance. The resulting detectors were not able to reach good performance, due to the coherence of the parameters. Additionally we want to know whether there is a difference between training and evaluation with RGB and gray-scale images.

The pretrained upper body detector does not provide satisfactory detection of persons that are not at front view (reached recall under 10%). Therefore we decided to train four detectors for each parameter combination. The detectors are for front, back, right, and left side view. With this training we can reach an improvement of 50% precision (from 45% to 95%) and 70% recall (from 7% to 77%) for right and left side view, back view has an improvement of 20% precision (from 74% to 94%) and 70% recall (from 14% to 84%). Well adjusted parameter combinations can cause higher improvement, even for front view. In practice the four side detectors are combined and applied to an image, therefore we want to know if the performance changes with this combination.

The following shortly outlines the structure of this work. Chapter 2 explains the basics of image processing, the feature types we used in this work and the evaluation foundation. Chapter 3 presents the **Matlab Cascade Object Detector** and the evaluation metric which states whether two bounding boxes are a match or not. Moreover the chapter explains the training and detector parameters. Evaluation results are presented in Chapter 4. The chapter is subdivided into the four steps of investigation and a summary section. The first Section 4.1 presents the evaluation results of the pretrained detector, referred to as starting conditions in the following. Based on those results we are able to determine the rate of improvement for the detectors we trained ourselves. The other three sections present the evaluation results of the investigations mentioned above.

2. Fundamentals

This chapter shortly explains the basic techniques of object recognition and the features used in this work. For more detailed information read [1].

Figure 2.1 gives a simple overview on the consecutive steps of object recognition. From left to right the images are sampled and quantized in order to gain a digital signal for further processing. In the first processing stage undesired noise or geometric distortion are removed with a low-pass filter. Because of its ideal properties Gaussian filter is used most commonly. The image is filtered and transformed in order to highlight or reinforce special parts. Besides the objects of interest, the image also contains the background of the scene. Therefore these objects and the background are separated from another within a segmentation process. Features quantify the objects which allows classification in different classes and types. An analysis then offers the requested statements about the examined objects.

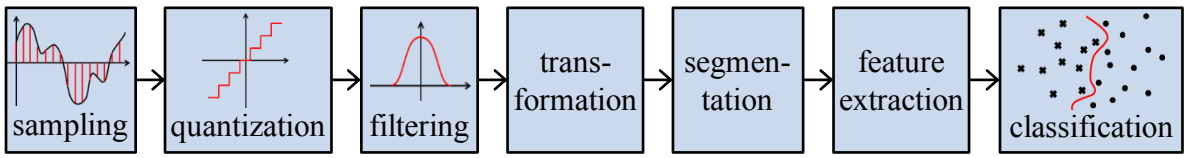


Figure 2.1.: The seven boxes show the consecutive steps of object recognition. The first four steps (sampling, quantization, filtering, and transformation) are used in every digital image processing technique. Segmentation, feature extraction and classification are used in object recognition.

The description of an object is complex and the method depends on application field and the object of interest. In the following Section 2.1 a selection of features that are used in object recognition are presented. Section 2.2 describes the classification process and Section 2.3 shows some measures how to evaluate such a classifier.

2.1. Features

A feature describes an interesting part of an image which is then used for further processing, e.g. training a detector. The resulting detector's performance depends on how good the features can describe an object. Furthermore several different feature descriptors have been developed to describe various kinds of objects.

Table 2.1 shows four categories of features. There are edges, corners, blobs and ridges. Features based on edges compute gradients in an image and are able form complete descriptions of boundaries between two image regions, called edges. Those features are most commonly

used to detect more overall objects, like cars or people and are not so good used on fine scale textures. One example of edge based features are Histograms of Oriented Gradients (HOG) which will be explained more detailed in the following section. Two famous edge detectors are Canny and Sobel, developed by John Canny in 1986 [2] and Irwin Sobel in 1968, respectively.

Corner features developed out of edge detection in order to find rapid changes in direction. They are point-like features with local two dimensional structure, also called interest points. By looking for high levels of curvature in the image gradient, no explicit edge detection is longer required. The field of application is similar to those of edges. An example for a corner detector is Smallest Univalue Segment Assimilating Nucleus (SUSAN) [3] which compares pixels to a circular mask (nucleus). Because of its high-speed performance, Features from Accelerated Segment Test (FAST) corner detector is used in real time video processing applications [4].

A blob is an image region in which some properties, like brightness or color are approximately constant. Therefore a blob detector can detect areas which are too smooth to be detected by corner detectors. Nevertheless blobs contain a preferred point, like a local maximum or center of gravity. Examples for blob detectors are Maximally Stable Extremal Regions (MSER), proposed by Matas et al. [5] and Principal Curvature-Based Region (PCBR), introduced by Deng et al. in 2007 [6].

The fourth feature category are ridges. They were first mentioned by Haralick in 1983 [7] and Crowley in 1984 [8] and are best used on elongated objects like roads or blood vessels. A ridge is a one-dimensional curve representing an axis of symmetry and has an attribute of local ridge width for each ridge point. Ridges are harder to extract than the other feature categories mentioned above and are therefore rarely used.

Table 2.1.: The table shows four feature categories and some corresponding detector examples. The third column shows the field of application in which the feature type is used best.

Types	Description	Field of Application	Detectors
edges	points within a boundary of two image parts	overall objects	Canny, Sobel
corners	interest points referring to point-like features; rapid changes in direction	objects	SUSAN, FAST
blobs	regions with similar properties	fine scale textures	MSER, PCBR
ridges	connected set of curves called the function's relative critical set	elongated objects	

After the feature detection a local image patch around the feature can be extracted. It is then called a feature descriptor or feature vector. The following sections introduce the three feature

types that are available in the training function of the **Matlab Cascade Object Detector**.

2.1.1. Histograms of Oriented Gradients

Histograms of oriented Gradients (HOG) are often used for object detection, like cars and people, because they describe the overall shape of an object and no fine structures, needed for e.g. face detection. Dalal and Triggs [9] first described HOG descriptors and focused on pedestrian detection in static images. Later they expanded their tests to include human detection in videos. HOG is very similar to Scale-Invariant Feature Transform (SIFT), the algorithm was published by David Lowe in 1999 [10].

Figure 2.2 and Figure 2.3 show steps of the generation of a HOG descriptor. The HOG method divides the image into small connected regions called cells. They form a dense grid of uniformly spaced cells over the whole image. Histograms of gradient directions are compiled for the pixels of each cell and called local histograms. The descriptor is formed by the concatenation of these histograms and shown in Figure 2.3. To improve accuracy the local histograms are contrast-normalized by calculating a measure of the intensity across a block and then use the value to normalize all cells within the block. This leads to a better invariance to changes in shadowing and illumination.

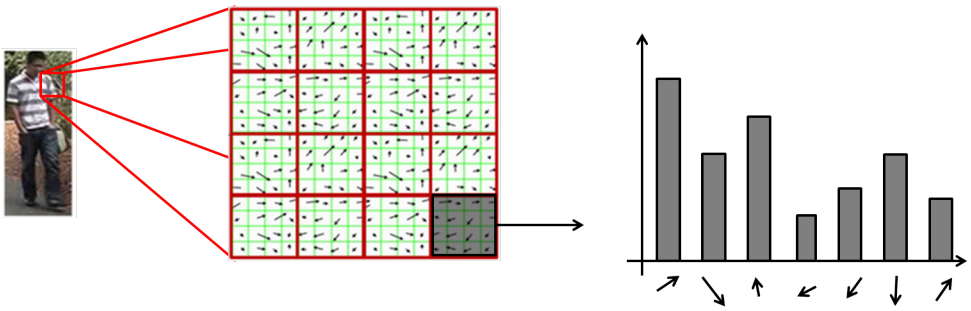


Figure 2.2.: The HOG method divides the image into cells and for each cell the histograms are calculated. [11]

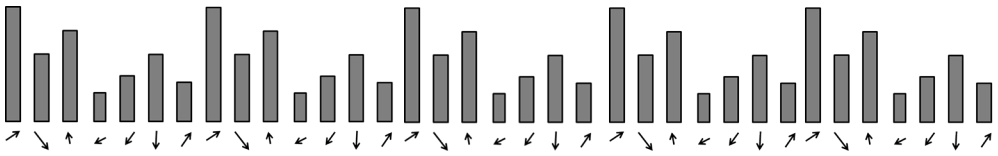


Figure 2.3.: The concatenation of the histograms of each cell forms the feature descriptor.

2.1.2. Local Binary Pattern

The simplest Local Binary Pattern (LBP) feature vector is created by concatenation of histograms, similar to HOG descriptors.

Figure 2.4 shows the first steps for gaining a LBP feature vector. At first the image or examined window is divided into cells, e.g. 16x16 pixels for each cell. For each pixel in a cell, the algorithm compares the pixel to each of its 8 neighbors and follows the pixels along a

circle (red arrow in Fig. 2.4). If the neighbor pixel's value is greater than the center pixel's value it writes "0" otherwise "1", resulting in a binary number with 8-digits. Afterwards the histograms over the cell are computed and optionally normalized. The feature vector for the window is the concatenation of the histograms of all cells, similar as in Fig. 2.3.

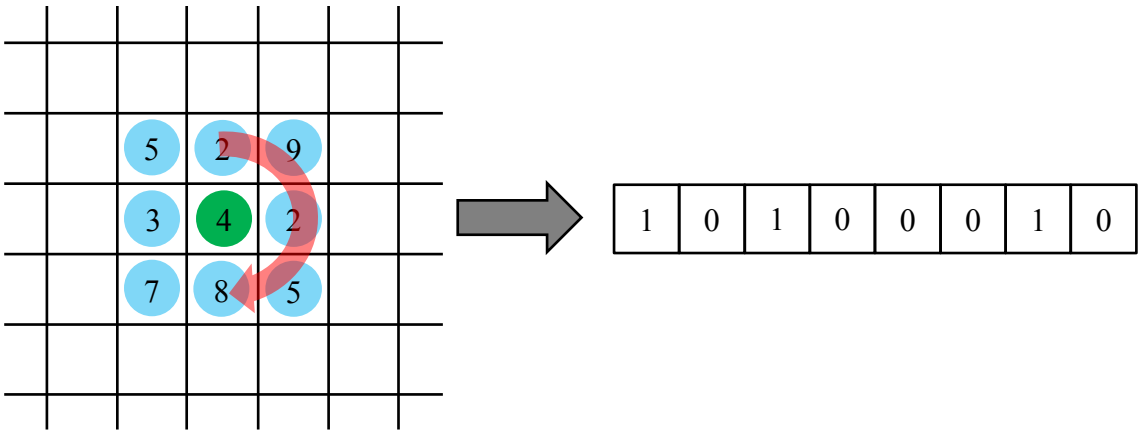


Figure 2.4.: The image is divided into cells. The comparison of a pixel with its 8 neighbors results in a binary LBP 8-digit number.

LBP was first described by Ojala et al. in 1994 [12] and was originally developed to supplement existing texture classification techniques. In 2009, Wang et al. combined LBP with HOG descriptor and achieved an improved detection performance described in [13].

2.1.3. Haar-like Features

These features owe their name to their similarity with Haar-wavelets. The Haar-wavelet is a sequence of rescaled "square-shaped" functions and were first proposed by Alfred Haar in 1909.

Figure 2.5 shows such a wavelet, for more information read [14].

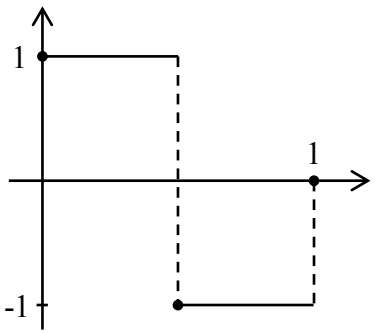
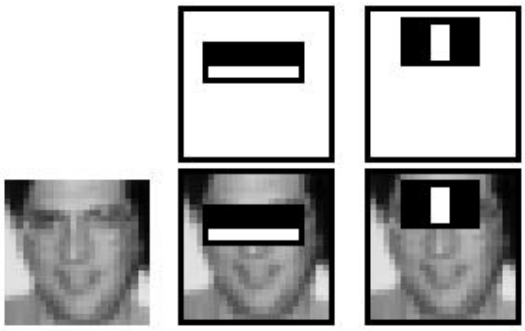


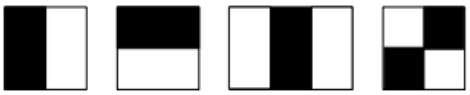
Figure 2.5.: The Haar-Wavelet is the first and also simplest known wavelet. Haar features owe their name to their similarity with Haar-wavelets.

Figure 2.6 shows some Haar-like features. Viola and Jones published an object recognition method [15] for gray-scale images in real time applications. Four object description features

based on differences in brightness are used. For instance the area around the eyes are darker than on the cheeks or nose, see Figure 2.6a. The Haar-like features used by Viola and Jones are shown in Figure 2.6b. The algorithm was originally developed for face detection and was used in the first real time face detector. Certainly it was found that it is suitable on detection of other objects as well. The position and scale of each feature is arbitrary thus a variety of possible features per image exists.



(a) The example shows two of the Haar features introduced by Viola and Jones applied to a face. In most faces the eyes are darker than the cheeks (pictures in the middle) and the nose appears brighter than the eyes (right pictures).



(b) These four Haar-like features were introduced by Viola and Jones in 2001. There are two edge, one line and one diagonal feature.

Figure 2.6.: In (a) two of the Haar-like features by Viola and Jones of (b) are applied to a face [15].

Figure 2.7 shows an extended feature set. Lienhart and Maydt extended the amount of used Haar features in 2002 [16]. Regard that one feature (at the bottom) has been removed, it can be replaced by the remaining.

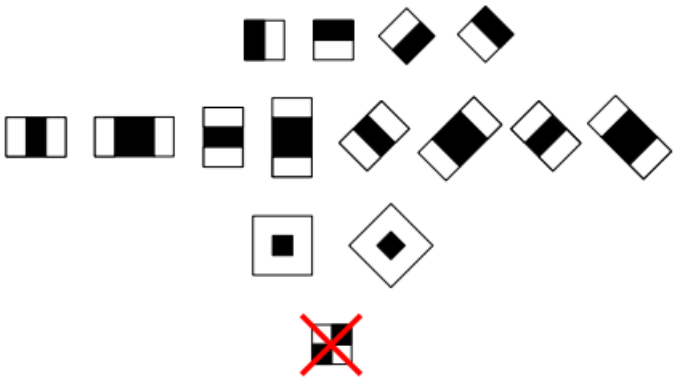


Figure 2.7.: Lienhart and Maydt extended the feature set of Viola and Jones. They removed the feature shown in the last row, because it can be described by the others [16].

2.2. Classification

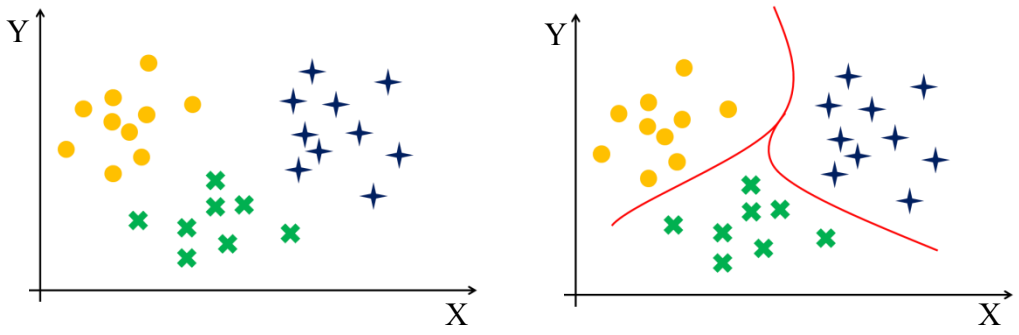
Humans classify all the time. Classification is a necessary first step before basic forms of decision-making and hardly needs elaboration. In image processing its goal is to identify

characteristic features, patterns or structures within an image to match them to a particular class. Classification and pattern recognition are closely related.

There are two classification techniques, namely supervised and unsupervised. Supervised classification uses example patterns or feature vectors, already assigned to a defined class. This is the training data which designs a specific classifier. In this work we use supervised classification. In order to train a classification model we need the ground truth data of the training images. The goal of unsupervised classification is to directly identify groups and distinguish them from another. The examples are not labeled.

Figure 2.8 shows an example for classification. In Fig. 2.8a a 2-D feature vector, acquired in the steps of feature extraction, is plotted as a point in a 2-D feature space. The three classes form distinct clusters which means that the two chosen features X and Y are sufficient to distinguish between the three classes. Now, the actual classification process is to draw the boundaries between classes, this is shown in Figure 2.8b. In this case the boundaries are obvious, in other cases more features are needed for definite classification. This leads to higher dimensions and more complex calculations, but results in more precise classification. Now, the separation is no longer a line but a hyperplane and the diagram would show a lot more axes.

In the application of human detection a binary classifier is used, defining whether it is a human or not. This means that contrary to Fig. 2.8, only two clusters exist. One for humans and the other for non humans. Because of the complexity of "human features" the two cluster will not be as easy to distinguish as the objects in the figure. In order to improve the classification many features are needed and the parameters that are used to train the classification model have to be adjusted to gain the maximum differentiation between the two clusters. Either way, the classifier will not be able to make no mistakes, therefore the next section introduces the evaluation of such a classifier.



(a) Feature space with three classes defined by two features X and Y.

(b) The red line represents the classifier boundaries, which separate the three classes.

Figure 2.8.: Example of a 2-D feature space

2.3. Evaluation of a Classifier

In a classification process, a classifier sorts objects to sundry classes due to certain features. Usually the classifier makes mistakes in the process, meaning it allocates some of the objects

to false classes. Due to the frequency of these errors, an evaluation of the classifier is possible. Often classification is binary, meaning there are only two possible classes. In our case, either the area contains a person's upper body or not. For this kind of classification two types of mistakes occur, an object is classified as a class one object, though it belongs to class two and vice versa.

Table 2.2 shows the error types and correct inference. The top row gives the ground truth information and the first column shows the assigned classes, classified by the detector. An image contains a person (positive image) and is classified as positive, means the classification process proceeded flawless. This is called a True Positive (TP). If the classifier labels the image as negative, then a type II error occurred, also called False Negative (FN). A type I error, also referred to as False Positive (FP), occurs when a negative image is classified as positive. The last case the classification of a negative image as negative, meaning no error. This case is called True Negative (TN).

Table 2.2.: In binary classification two types of errors occur. Type I errors are FP and type II errors are FN. Precision, recall and other rates can be calculated with these values.

	Positive Image	Negative Image
Positive Detection	True Positive (TP)	False Positive (FP)
Negative Detection	False Negative (FN)	True Negative (TN)

Figure 2.9 visualizes Table 2.2. On the left relevant elements are shown. These are TPs and FNs, meaning originally all positive items. The right side shows the irrelevant elements, meaning FPs and TNs, originally all negative items. The circle in the rectangle shows the selected items, those are all items the classifier labeled as positive.

Figure 2.9 shows on the right side how precision and recall are calculated with the information of the four occurring cases. Precision, also called Positive Predictive Value

$$\text{PPV} = \frac{|\{\text{relevant items}\} \cap \{\text{retrieved items}\}|}{|\{\text{retrieved items}\}|} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{2.1}$$

is the fraction of retrieved instances that are relevant. Recall, also known as sensitivity and True Positive Rate

$$\text{TPR} = \frac{|\{\text{relevant items}\} \cap \{\text{retrieved items}\}|}{|\{\text{relevant items}\}|} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{2.2}$$

is the fraction of relevant instances that are retrieved. High precision means that an algorithm returned substantially more relevant results than irrelevant, while high recall means that an algorithm returned most of the relevant results.

Besides recall and precision there are other data that are important to describe a detector's performance.
One of them is the False Positive Rate

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}, \tag{2.3}$$

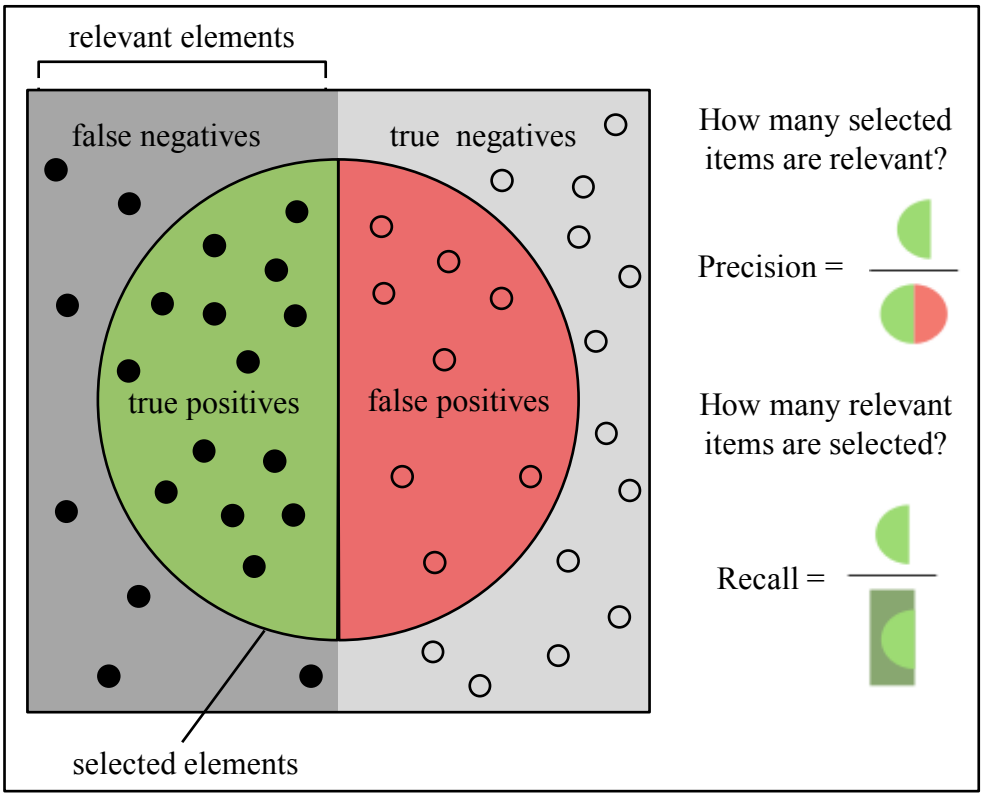


Figure 2.9.: Recall and Precision are important to measure relevance in pattern recognition. Precision is also known as positive predictive value and recall is the sensitivity. [17]

which is also called fall-out. By plotting the True Positive Rate (TPR) against the False Positive Rate (FPR), a Receiver Operating Characteristic (ROC) curve is created. It is a graphical plot that illustrates the performance of a binary classifier system as its discrimination threshold is varied. ROC is also known as a relative operating characteristic curve, because it is a comparison of two operating characteristics (TPR and FPR) as the criterion changes. For further information see [18].

Figure 2.10 shows an example of such a ROC curve. The four curves represent four different algorithms or classification models. The best performance has the classifier represented by the red curve (top curve) and the worst the one by the green curve (lowest curve). The closer the curves are to the dashed line, the more random the classification is. The dashed line represents a random process. Usually the FPR increases as the TPR is increased and the TPR decreases as the FPR is decreased. The optimal threshold (green area) is reached when the TPR is as high as possible and the FPR is as low as possible. The tow red areas show classifier systems with strict and lenient threshold, both categories do not represent good classifiers.

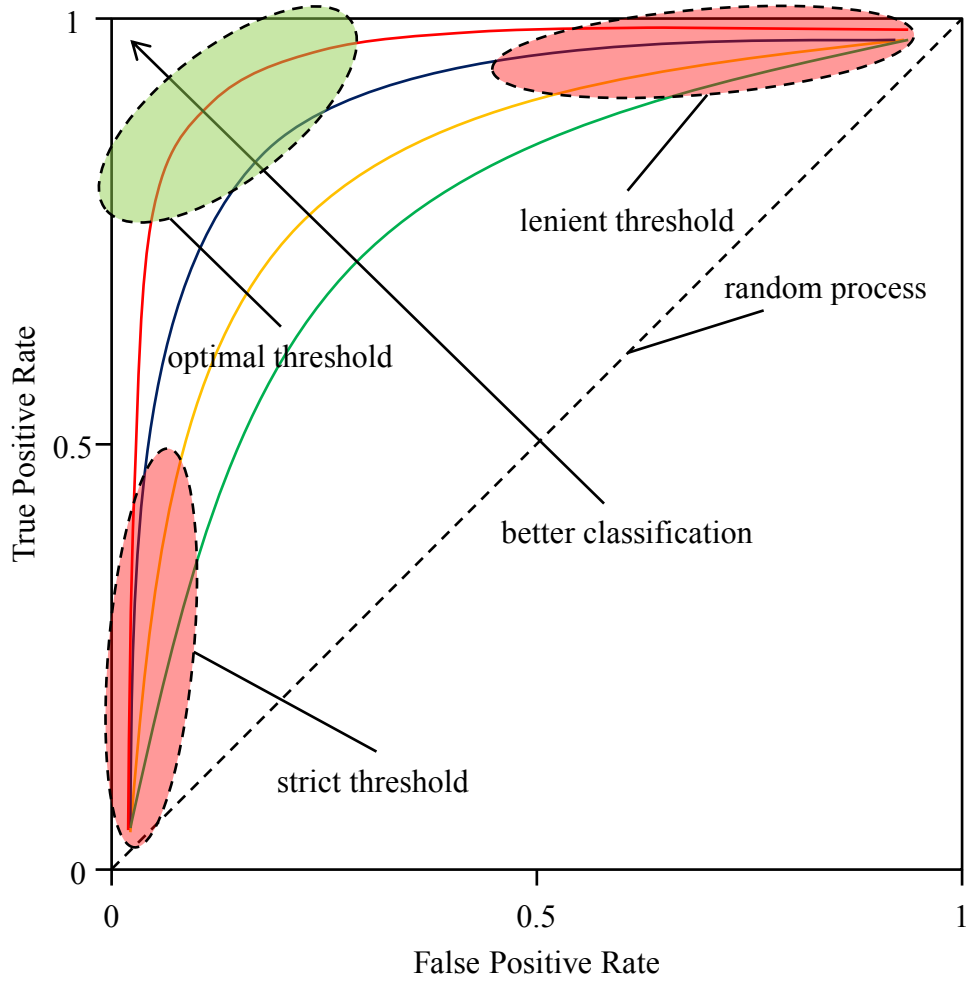


Figure 2.10.: The ROC curve is created by plotting the TPR against the FPR at various threshold settings. If the classifier's performance is located in the green area, then the threshold has its optimal value. The dashed line represents a random process [18].

3. Detection Approach and Evaluation Metric

This chapter explains the training (`trainCascadeObjectDetector`) and detector function (`step`) of the **Matlab Cascade Object Detector** and the evaluation metric, used to evaluate the performance of the detectors.

The **Matlab Cascade Object Detector** and the corresponding training function are described in the MathWorks® Documentation [19, 20]. The detector uses the Viola-Jones algorithm [21] to detect a couple of people's body parts, like faces, eyes, mouth, nose, or upper body. The function `trainCascadeObjectDetector` can be used to create a custom detector, in the following we refer to this function as the training function. In this work we focused on detecting people's upper bodies. Thus we compared the pretrained upper body detector (evaluation results referred as starting conditions in the further text) to the ones we trained ourself.

The following Section 3.1 gives an overview on how we proceeded in this work and how the training, detection and evaluation interact. Section 3.2 describes the training function, its parameters and gives an overview of the used training images. Section 3.3 presents the parameters of the detector. Thereafter Section 3.4 presents the evaluation metric, this includes the formula used to determine a match of bounding boxes and the processing of the detector output.

3.1. Overview

Figure 3.1 shows the approach of object recognition and training of the **Matlab Cascade Object Detector**. The horizontally oriented boxes represent the usual object recognition process (seen in Fig. 2.1) with subsequent evaluation and discussion of results. The vertically oriented boxes represent the training process, which affects and defines the classification model. The input are images or videos taken by a mono camera, showing people's upper bodies (positive images) or not (negative images). For feature extraction three feature types (HOG, LBP and Haar) are available when using the Matlab Detector. For the following classification a classification model has to be defined, it can either be the pretrained upper body model or a custom one, created by the training function. For training, a lot of training images are needed (the more, the better). Due to the fact that we want to improve the detection of people at side view, images of all four rotations are needed (front, back, right side, and left side view). After training the function outputs a xml file, which specifies the trained classification model and can be used the same way, as the already existing upper body model. To evaluate the performance and present the improvements, a consistent evaluation metric is needed. Therefore we oriented ourselves to the performance evaluation presented by Dollár et al. in [22]. Another

important component of the evaluation is the ground truth data. They display where there is the object of interest (upper body) in an image. There are two options, one is called a binary mask, the other are Bounding Boxes (BBs), shown in cyan and green in the ground truth box of Figure 3.1. The advantage of binary masks is, that they explicit depict the object and no background. Since the handling of bounding boxes is simpler, Matlab uses them in the form of vectors with four columns [x, y, width, height]. They surround the object of interest in an image. The evaluation and the required data, like ground truth data, are described in Section 3.4.

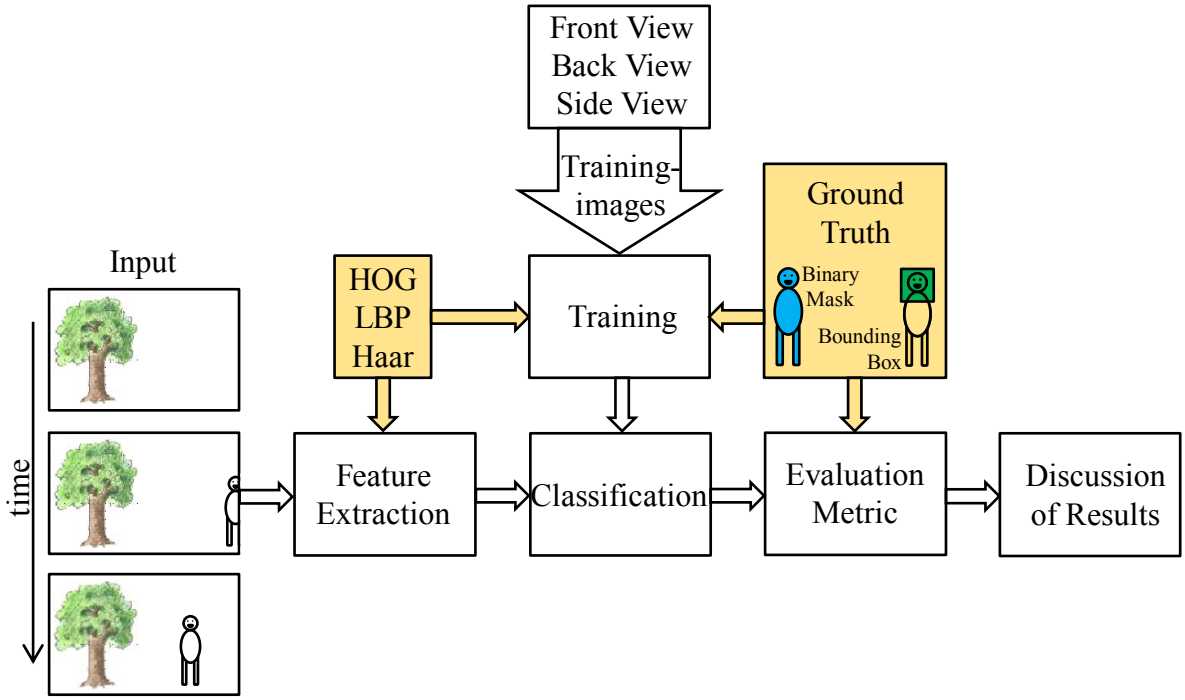


Figure 3.1.: The figure shows the overview of this thesis. Both the evaluation and the training path require images, the features and the ground truth data. In the end a discussion of results is possible because of the evaluation metric.

3.2. Training

The purpose of the training function is to create a custom classifier. The pretrained classifiers may not be sufficient in all fields of application. In our case the upper body classifier does not provide satisfactory results for the detection of upper bodies in all planes of rotation. This is because the **Matlab Cascade Object Detector** can only detect object categories with a steady aspect ratio, meaning objects like humans, road signs or cars viewed from one side. According to this consideration multiple detectors have to be trained to ensure proper detection of upright upper bodies in all planes of rotation. This explains why the pretrained upper body detector is used best at front and also partly back view, but is impractical at side view. It is a single detection model, trained with training images showing humans only from the front and the back. The MIT Database was used for this [11].

Figure 3.2 shows the arrangement of cameras at top view to obtain images of persons in different planes of rotation. In this work we used four of these cameras (red), showing persons at front, back, right side, and left side view. Additionally it would be interesting to know whether the detection rate increases if the amount of cameras increases. An advantage of having four or more individual detectors is, that it is not only possible to detect the person, but also determine the probability of direction, in which the person is oriented.

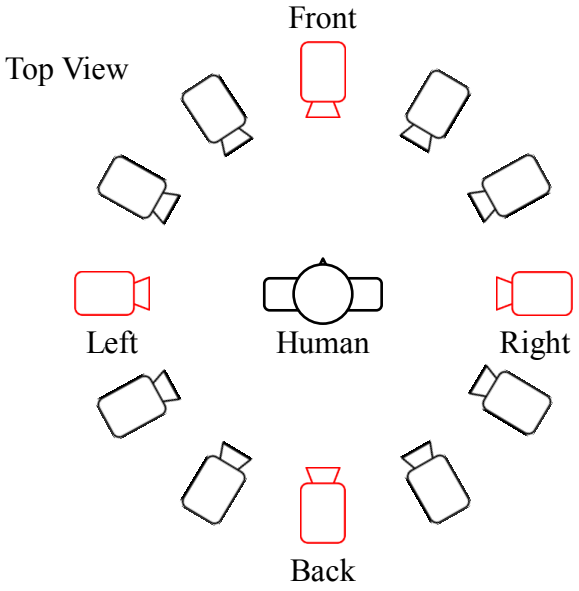


Figure 3.2.: Top view of a camera arrangement around a person. The more cameras, the more detailed are the transitions from one camera position to another. In this work we used back, front, right, and left side view, represented by the red cameras.

The training function needs a set of positive samples and a set of negative images. The positive samples are positive images with regions of interest specified, also referred to as ground truth data. Either the **Training Image Labeler** can be used to specify the BBs or the ground truth data provided by some of the databases. The training function can generate negative samples automatically from the negative images provided. The more training images are used to train the detection model the more accurate and reliable the detector will be. The number of training stages and the other training parameters have a great effect on these properties too. They are described in the following two sections.

3.2.1. Cascade Classifier

The **Matlab Cascade Object Detector** uses a cascade classifier that consists of multiple stages. Each stage is a combination of weak learners. They are simple classifiers called decision stumps. Boosting is used to train each stage and provides the ability to train a highly accurate classifier with the decisions of weak learners.

Boosting is a machine learning algorithm which converts weak learners to strong ones. A weak learner is a classifier that is only slightly correlated with the actual classification. A classifier that is highly correlated with the true classification is called a strong learner. In

1990 Robert Schapire proved that it is possible to combine weak learners and obtain a strong one [23], leading to an advancement in machine learning and resulting in the development of boosting.

Figure 3.3 shows the process of classification. Stages use a sliding window (red) for the image processing. The current location of the window is either labeled as positive or negative by the classifier. Negative means, that no objects were found and the classification for this region is complete, the window then slides to the next location. A positive label indicates that an object was found and therefore, the classifier passes the region to the next stage. When the final stage classifies a region as positive, the detector reports an object found at this location.

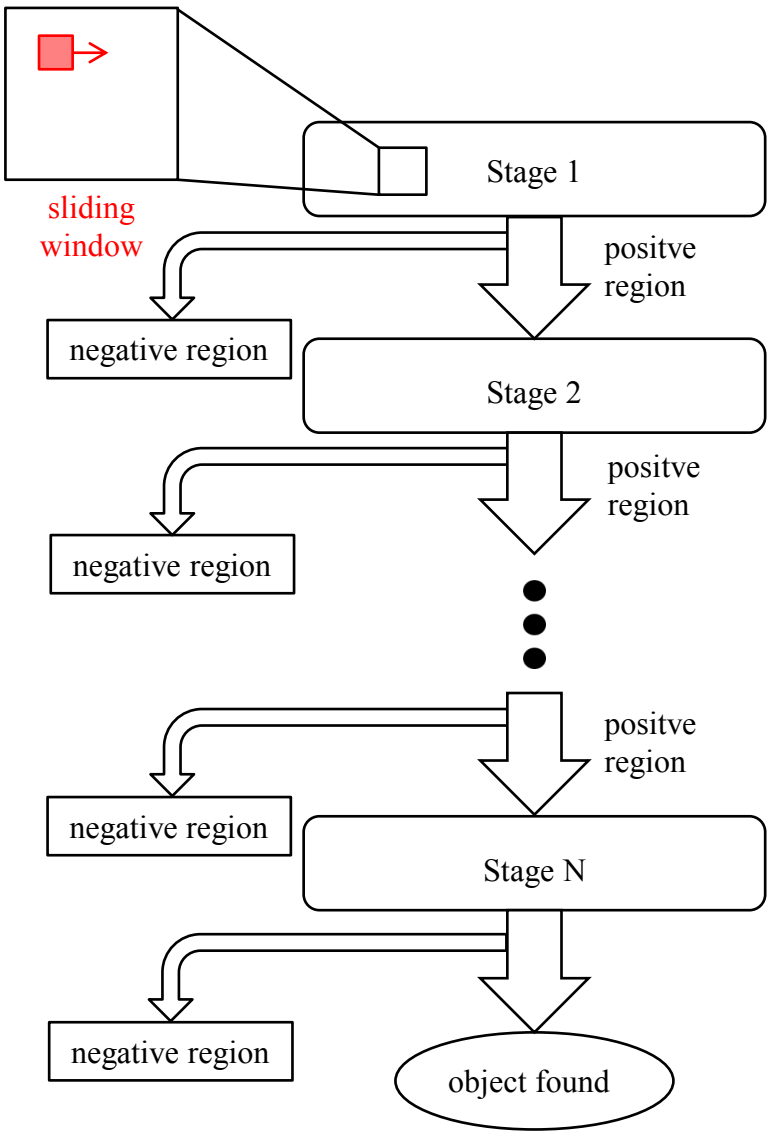


Figure 3.3.: The detector is trained with stages of weak learners. In each stage the classification labels a location of the sliding window as either positive or negative. According to this decision the region is either passed to the next stage or the classification for this region is completed [20].

Due to the assumption that the majority of windows do not contain the object of interest (upper body), the stages are designed to reject negative samples as fast as possible. Nevertheless it is very important that the False Negative Rate (FNR) stays low at each stage. If an object of interest is incorrectly labeled as negative, the classification stops and the window slides to the next region, hence a correction is not possible later on. For the other case, an incorrect labeling as positive, it is possible and also crucial that mistakes are corrected in subsequent stages. This happens because each stage can have a high TPR. Even if the detector mistakenly labels a nonobject as positive, the mistake can be corrected in the following stages.

The overall FPR

$$\mathrm{FPR} = \mathrm{FPR'}^{S} \tag{3.1}$$

and the overall TPR

$$\mathrm{TPR} = \mathrm{TPR'}^{S} \tag{3.2}$$

depend on the number of stages. FPR' and TPR' $\in$ (0 1] each is the rate per stage and S is the number of stages. Increasing the number of stages reduces the overall FPR and overall TPR.

3.2.2. Training Parameters

Table 3.1 shows the function parameters and its value range. All of the parameters can either be set manually by the user or the function automatically uses default values.

The first parameter is called *ObjectTrainingSize*. Before starting the training, the training function resizes the positive and negative samples to this size. The size must either be defined by a vector with [height, width] or by the string 'Auto', which is the default value. The function then determines the size automatically using the median width-to-height ratio of the positive instances (BBs). For optimal detection accuracy the *ObjectTrainingSize* has to be specified close to the expected size of the object in the image. For faster training and detection the value can be set smaller than the expected size.

With the next parameter the training function can calculate the number of negative samples per stage. The parameter is called *NegativeSamplesFactor* and its values have to be real-valued scalars. The default value is 2, meaning that in each stage there are twice as many negative samples than positive ones.

As mentioned previously, the number of stages has a wide influence on the performance and accuracy, it is defined with the parameter *NumCascadeStages*. Increasing the number of stages requires a longer training time and more training images, but results in a more accurate detector. For the number of stages all positive integers can be chosen, 20 stages is the default value.

The parameter *FalseAlarmRate* sets the acceptable FPR at each stage, which is the fraction of negative training samples incorrectly classified as positive samples. The value must be in the range of (0 1]. Lower values increase the complexity of each stage and can achieve fewer false detections, on the other hand the training and detection takes more time. The default value is 0.5, meaning that it is suitable to classify half of the negative samples as positive at each stage.

TruePositiveRate defines the minimum TPR required at each stage. Equal to the *FalseAlarmRate*, it is specified with a value in the range of (0 1]. The default value is 0.995. The TPR

is the fraction of correctly classified positive training samples. Higher values of *TruePositiveRate* increase the complexity of each stage. Therefore the detector can achieve a greater number of correct detections but requires more time for training and detection.

The last parameter, *FeatureType* may have the most impact on the detection result. According to the object of interest, one of the three feature types (HOG, Haar or LBP), has to be selected. If none is selected, the training runs with HOG features. The pretrained upper body detector uses Haar-like features. It encodes details of the head and shoulder region, but uses more features around the head, to make it more robust against pose changes (e.g. head rotations). The computation of Haar-like features take much more computation time than HOG or LBP, therefore it is recommended to first find the optimal parameter settings with HOG or LBP and then later test it on Haar-like features in order to save computation time [20].

Table 3.1.: The table shows the description of all training parameters with their default values and the value range.

Parameter name	Description	Value range	Default value
ObjectTrainingSize	before training, the function resizes the positive and negative samples to *ObjectTrainingSize*	[height, width] or 'Auto'	'Auto'
NegativeSamplesFactor	used to specify the number of negative samples for each stage	real-valued scalar	2
NumCascadeStages	sets the number of stages to train; accuracy, training time and number of samples depend on the number of stages	positive integer	20
FalseAlarmRate	acceptable FPR (fraction of negative training samples incorrectly classified as positive samples); lower values increase complexity of each stage	$\in$ (0 1]	0.5
TruePositiveRate	minimum TPR required per stage (fraction of correctly classified positive training samples); higher values increase complexity of each stage	$\in$ (0 1]	0.995
FeatureType	sets the feature type	'Haar', 'LBP', 'HOG'	'HOG'

The previous text already mentioned some of the interactions between the parameters. Be-

cause of the structure with weak learners, the *FalseAlarmRate* and the *TruePositiveRate* closely correlate with *NumCascadeStages* and the accuracy, training time and performance. Higher values for *FalseAlarmRate* require a greater amount of cascade stages to achieve reasonable detection accuracy. This results in longer training times and the training requires more training images. At each stage some of the positive and negative samples are eliminated. If the amount of training images is not enough, the function might run out of either positive or negative samples. The training then stops at the current stage.

For the beginning we wanted to examine the impact of a single parameter (results in Chapter 4.2). Therefore we decided which parameters are the most important for the performance and chose values that might be interesting to investigate. Except the investigated parameter all other parameters retained their default values. Additionally the impact for all three feature types was calculated.

Table 3.2 shows the investigated parameters and the chosen values. All value vectors of the parameters also contain the default values. Therefore a comparison between the default values and the starting conditions, as well as a comparison of the other values of a parameter and the default value can be useful. In the table there are five of the six parameters listed, *ObjectTrainingSize* always retained its default settings.

Table 3.2.: The table is a list of the training parameters, that were used to examine the influence of a single parameter. The various values chosen for the calculations are listed in the second column.

Parameter name	Value Vector
NegativeSamplesFactor	[1 2 3 4]
NumCascadeStages	[1 2 5 10 15 20 25 30 35 40 45 50]
FalseAlarmRate	[0.1 0.3 0.5 0.7 0.9]
TruePositiveRate	[0.99 0.995 0.999 0.9999]
FeatureType	['HOG' 'LBP' 'Haar']

Though *NegativeSamplesFactor* and *TruePositiveRate* might not have a great influence on the result, it is interesting to know whether the training is able to pass all stages or if it runs out of negative training images and if recall increases with higher *TruePositiveRate*.
NumCascadeStages and *FalseAlarmRate* have major impact on the performance, therefore, we decided to evaluate more values for those parameters. A *FalseAlarmRate* of 0.9 or only 1 or 2 cascade stages clearly seem to not increase the performance, but the evaluation results are crucial anyway.
The last row of the Table 3.2 shows the three feature types. Actually it is not necessary to explicitly list them in the table, because all calculations were performed with all three feature types. This allows a direct comparison of those three.
The results of the training and detection with these parameters are presented in Chapter 4.

3.2.3. Training Images

One of the most important components of training a detector, is the selection of training images. The training images have to meet some requirements.

First of all there is the amount of training images. In the MathWorks®Documentation [20] it says that their number has to go into the thousands. The more training images the higher the likelihood that the detector is able to always detect human upper bodies. Certainly it increases the training time. Furthermore the scenarios of the training images shall resemble the scenarios of the images in which the object of interest is supposed to be detected. The negative images must resemble the positive ones as well. Moreover the images must either be gray-scale or true color (RGB).

For the training and evaluation images we used two databases. The MIT pedestrian Database [11] and the Graz-02 Database [24], some more information on these databases are in the appendix Chapter A.1. Unfortunately the amount of images is not evenly distributed within the four planes of rotation (front, back, left, and right).

Figure 3.4 shows the distribution of the images within the four planes of rotation and the amount of negative images. The MIT and the Graz-02 Database do not provide enough images. Moreover the MIT Database only provides images for front and back view (about 450 each) resulting in an imbalance of the four sides. The Graz-02 Database provided only about 50 images for each left and right side view and about 110 for front and back view, each. To fix the problem we decided to use simulated images, shown in green and yellow. For the simulation of the images we used the tool MakeHuman™[25] and Blender™[26] to render such images.

To get more variance in body height, stature and hair style, the simulated images are divided into "John Doe" (green) and "Erika Muster" (yellow). Additionally the background, color of the hair and clothes change in every image. In the simulation the people walk through a room, therefore their posture varies too. To obtain equal detectors the amount of images is filled up with simulated images to 2000 images per rotation. Most of the negative images are provided by the Graz-02 Database. Those images have VGA resolution (640x480 pixels) so the training function can generate several negative samples from one image. In purple there are the background images of the simulated ones, providing additional negative images.

The major difference between the Graz-02 Database and the other images is, that the others are scaled to the size of 64x128 pixels and aligned so that the person's body is in the center of the image. The Graz-02 Database also contains images with more than one person per image.

Figure 3.5 shows some examples of the four image categories. Fig. 3.5a proves that the persons in the images of the Graz-02 Database have different sizes and are not centered. The images have VGA resolution. The images of the MIT dataset (Fig. 3.5b) and the simulated ones (Fig. 3.5c, Fig. 3.5d) have all the same size of 64x128 pixels and persons have the same scale.

To make the results comparable, always the same images were used. The ratio of training images and evaluation images is 80% to 20%. So 80% of the images are used for training and 20% of the images are used for detection and evaluation. The proportion of the four datasets stays the same for every rotation. It is not possible, in our evaluation, that an image with

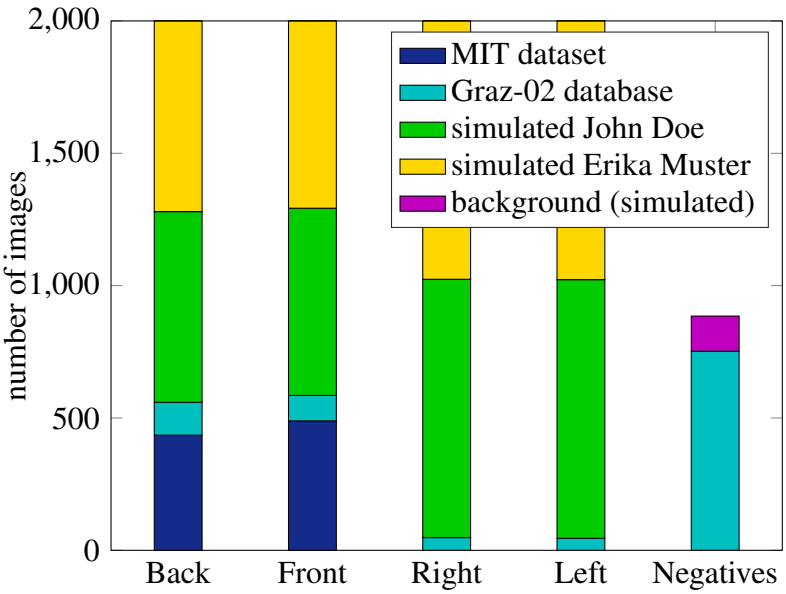


Figure 3.4.: The Figure shows the amount of images used for training and evaluation et each plane of rotation. It is divided into the two databases and the two simulated image sets. The bar on the right shows the amount of negative images.

which the detector was trained, also occurs in the image set on which the detector is tested and evaluated.

Beside training and evaluation images, the positive samples data is needed as well. The training function generates these samples data from the ground truth data. Therefore the region of interest (ROI) has to be labeled in the images. As described previously, either the **Matlab Training Image Labeler** [27] or, if provided, the ground truth data of the datasets can be used. To label the images of the Graz-02 Database we used the image labeler. Creating the BBs for the MIT and simulated images was simple because of the centering and same size of all people in the images. That's why the BBs per dataset and per plane of rotation are the same for all images.

Figure 3.6a shows a binary mask and the BB of a simulated image, Fig. 3.6b shows the same for an image from the Graz-02 Database. The right picture of Fig. 3.6b shows that the binary masks of the two persons border, they are therefore interpreted as a single bounding box by the computer. In order to prevent this behavior, the Graz-02 Database was labeled with the Image Labeler, resulting in BBs like in the middle picture of Fig. 3.6b.

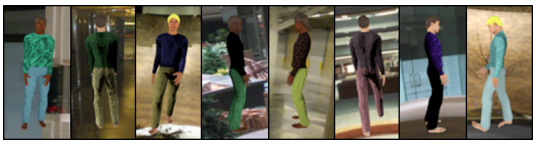
3.3. Detector

Similar to the training function, the detector possesses several parameters as well. Unlike the training parameters, the detector parameters remained constant for all measurements. The modification of the detector parameters has influence on the detection time, but not necessarily the accuracy or other properties. For these first measurements the detection time was not the most important factor for us.

Table 3.3 shows the detector parameters and its values. The parameters *MinSize* and *MaxSize*

(a) Graz-02 Database [24]

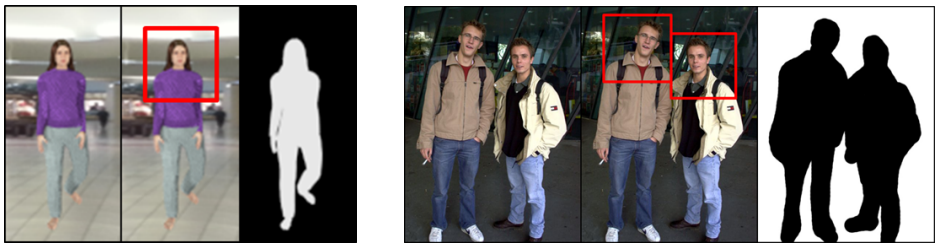
(b) MIT pedestrian dataset [11]



(c) simulated "John Doe"

(d) simulated "Erika Muster"

Figure 3.5.: The figure shows example images of the two databases we used and the two simulated image sets. The simulated images are divided into a man we named "John Doe" and a woman called "Erika Muster". More information about the databases is provided in the appendix A.1.



(a) (b)

Figure 3.6.: In (a) an example of the ground truth data of a simulated image is shown, in (b) it is an image of the Graz-02 Database [24]. The middle pictures show the BB_{gt} in red and the right pictures show the binary masks provided by the database and the simulation tool, respectively.

define the smallest and the largest detectable object. They are both defined with a two-element [height width] vector, defining the minimum or maximum size region containing an object in pixels. *MinSize* must be greater or equal to the image size used to train the classification model. In order to reduce computation time the value can be set closely to the smallest image size. We do not specify a value for this property, so the detector sets it to the size of the image used to train the classification model, meaning the value of *ObjectTrainingSize*

(training parameter). *MaxSize* must be smaller or equal to the image size. The computation time can be reduced by knowing the maximum object size prior to processing the image. The detector sets this value to the size of the image if it is not specified.

The *ScaleFactor* is specified with a value greater than 1.0001. It is used for multi-scale object detection. The scale factor incrementally scales the detection resolution between *MinSize* and *MaxSize*, it calculates the ideal value

$$\text{ScaleFactor}_{\text{ideal}} = \frac{\text{size(I)}}{\text{size(I)} - 0.5} \tag{3.3}$$

with the size of the current image (I). The detector scales the search region round((*TrainingSize*) × (*ScaleFactor*N)) at steps between *MinSize* and *MaxSize*. N is the current step and *TrainingSize* is the image size used to train the classification model. The default value for *ScaleFactor* is 1.1.

Figure 3.7 shows the scaling of the image. In order to locate the target object, the detector incrementally scales the input image. At each scale increment, a sliding window (same size as the training image size) scans the scaled image in order to locate objects. The *ScaleFactor* property determines the amount of scaling between consecutive steps. The search window traverses the image for each scaled increment.

Figure 3.8 illustrates the behavior of the detector with several values for *MergeThreshold*. This parameter defines the criteria needed to declare a final detection in an area where there are multiple detections around an object. The default value is 4, which means that a final detection occurs if a minimum of four bounding boxes detect an object. The final bounding box is then formed around the target object by merging the others. Increasing the threshold may suppress false detection but requires that the object is detected multiple times during the detection phase. Figure 3.8 shows which bounding boxes are returned at which *MergeThreshold* value. At 0 all detection are returned without performing thresholding or merging operation.

The last detector parameter is called *UseROI* and is either true or false. It defines whether to detect an object within a rectangular region of interest (ROI) within the input image or not.

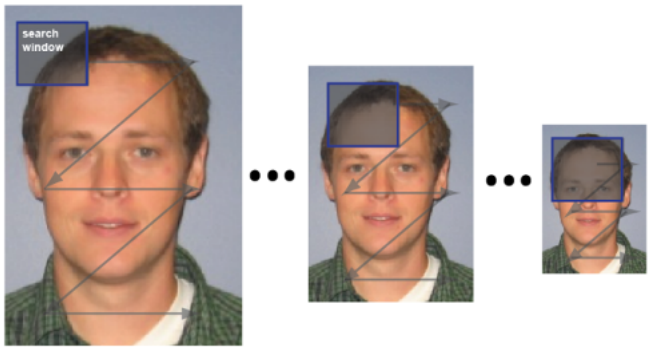


Figure 3.7.: In multiscale object detection a search window traverses over the image, which is then scaled to the next stage. With this method objects of interest at various sizes are able to be found [19].

Table 3.3.: The table shows the description of the detector parameters and their values.

Parameter name	description	values	default value
MinSize	size of the smallest detectable object	[height width]	[]
MaxSize	size of the largest detectable object	[height width]	[]
ScaleFactor	scaling for multiscale object detection	≥ 1.0001	1.1
MergeThreshold	detection threshold	positive scalar integer	4
UseROI	use region of interest	boolean	false

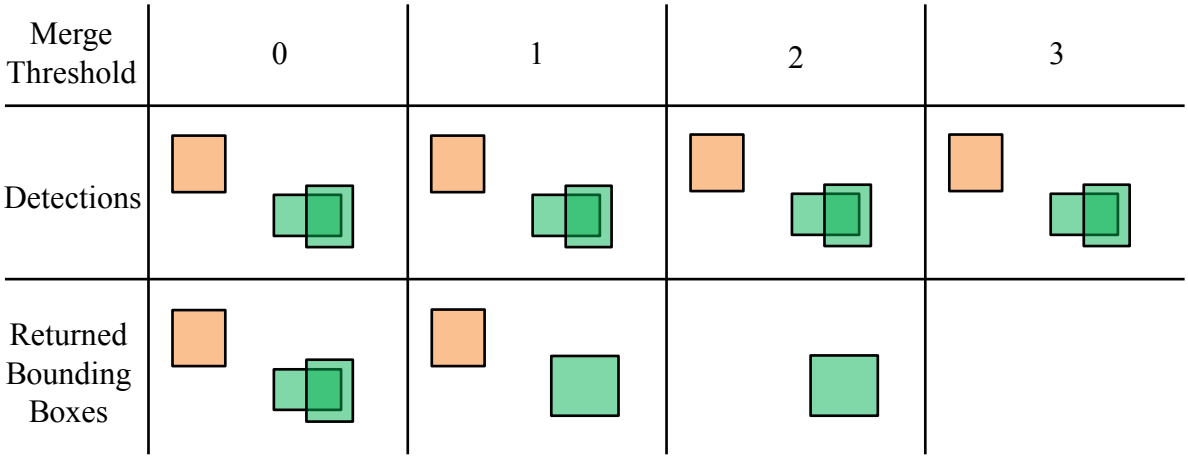


Figure 3.8.: The merging of BBs depends on the value of *MergeThreshold*. The column of returned bounding boxes show the final BB (green and orange) for each value of *MergeThreshold*. [19]

3.4. Evaluation Metric

This chapter explains which input data is required for the evaluation, which data types are provided by the training and detector and how the evaluation is implemented. The formula of calculating a match of bounding boxes is the heart of the evaluation.

Our aim is the reliable detection of a person in an image. As mentioned above we started training of four separate detectors to cover the four basic planes of rotation. In actual fact it is not vital which detector eventually detects the person but that the person is detected generally. Nevertheless we examined both situations, the results are presented in Chapter 4. The detection of an upper body automatically indicates the detection of an entire human but is more robust against occlusion. The occlusion statistics of Dollár et al. [22] states that most occluded body parts belong to the lower body.

The training function requires the positive samples data, the positive images and the negative images. The positive samples data is a structure array containing the image filename and the BBs. The **Training Image Labeler** directly creates such a struct. The function **`trainCascadeObjectDetector`** has additional input arguments which specify the training parameters and the name of the output xml file. In this work we used the planes of rotation and the parameters as filenames to directly assign them.

Figure 3.9 shows the detection and evaluation step after receiving the xml file from the training. Here the process is shown with one image and one xml file, meaning only one detector to evaluate. In the actual evaluation we used more than 400 images per rotation, three feature types and five parameters with 4 to 11 different values. This means we had to evaluate over 300 trained detectors. The **`step`** function needs two input arguments, the detector object specified by the xml file and the image. It outputs the bounding boxes as a matrix, where each line specifies one bounding box with [x y width height]. Figure 3.9 also shows the output image with the detected bounding boxes in yellow. The output image is not used in the evaluation. Furthermore an array of structs (similar to the one of the positive samples data) is needed to save the matrices when evaluating more than one image. In the case of absolutely perfect detection this struct and the ground truth struct have matching BBs. Such a perfect detection never occurs therefore the following text explains how to evaluate the performance.

The substance of the evaluation metric is the formula for the calculation of matches. A match is called if a BB_{gt} and a detected Bounding Box (BB_{dt}) overlap sufficiently. We used the PASCAL measure by Everingham et al. [28] to determine such sufficient overlaps. Dollár et al. used this method to compare several data sets and evaluation protocols for pedestrian detection [22].

The Matlab detector meets the conditions to apply the PASCAL measure. The detection system should perform multiscale detection and any necessary nonmaximal suppression (NMS) for merging nearby detections. In the Matlab detector these values are tunable with the *ScaleFactor* and the *MergeThreshold* (See Chapter 3.3).

The PASCAL measure states that the area of overlap

$$a_0 = \frac{\text{area}(BB_{dt} \cap BB_{gt})}{\text{area}(BB_{dt} \cup BB_{gt})} > 0.5 \, . \tag{3.4}$$

of BB_{dt} and BB_{gt} must exceed 50%. To gain the numbers of TP, FP and FN we simply count the bounding boxes. For each image we compare each BB_{dt} with the BB_{gt}. If the formula states a match, the number of True Positives (numTP) is incremented, because the detector correctly classified an upper body as positive. The detector sometimes labels BBs where there is no object, this means that a BB_{dt} has no corresponding BB_{gt}. In this case the evaluation increments number of False Positives (numFP), when an object was incorrectly classified as an upper body (FP). The other case is, if the detector was not able to detect the object where there is the BB_{gt}. Now there is a remaining BB_{gt} and the number of False Negatives (numFN) is increased. FN means, that the detector incorrectly classified an object as a non upper body. In order to double-check the result we count the total number of BB_{dt} and BB_{gt} of all evaluated images. If BB_{dt} matches multiple BB_{gt}, than it is matches with the BB with highest overlap.

This evaluation method combined with these evaluation images has one disadvantage. We are not able to count TNs. All parts of the images where there are no BB_{gt} and no BB_{dt} would

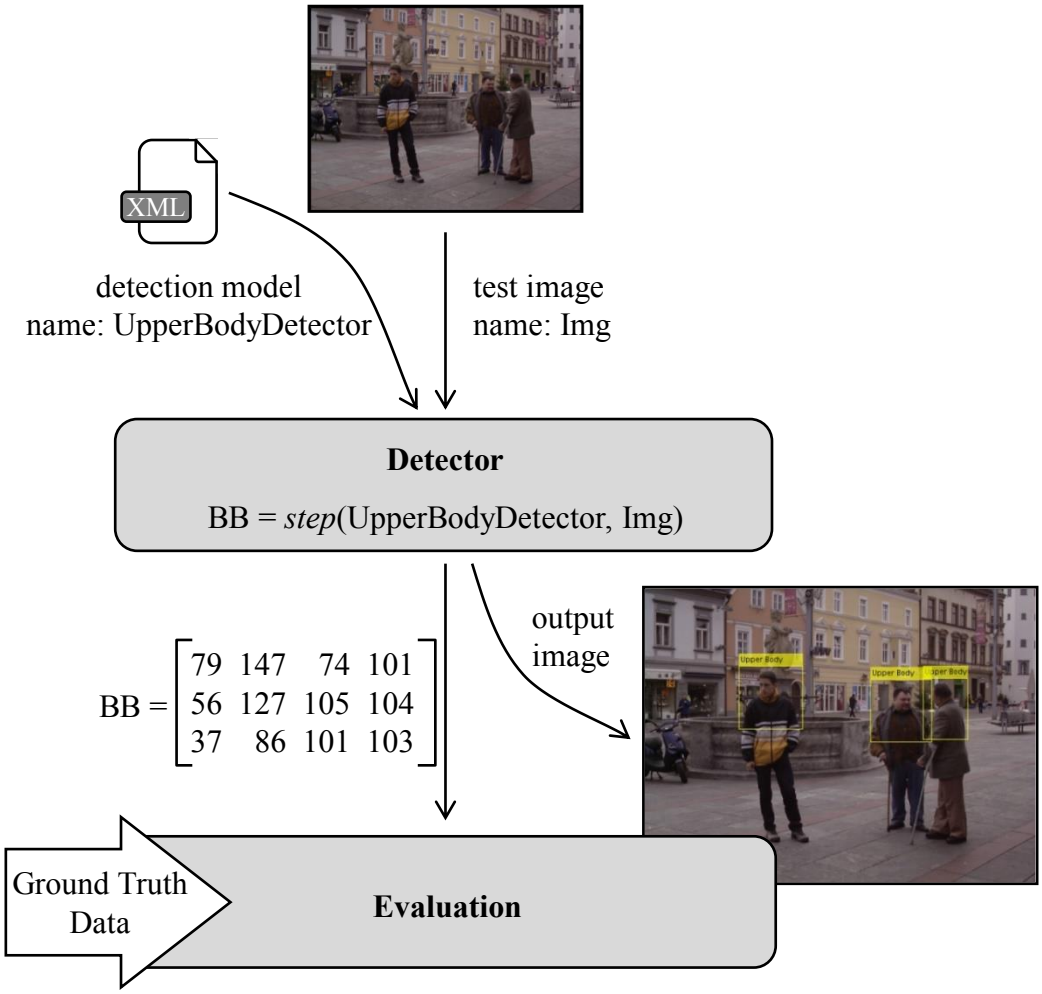


Figure 3.9.: With a trained detection model, specified by the xml file and an evaluation image the detector function outputs the detected bounding boxes as a matrix. These bounding boxes are shown in yellow in the output image. The matrix is later needed for the evaluation. It is compared to the ground truth data.

be TNs, but areas cannot be randomly be denoted as TN or not. The calculation of the FPR is not possible because of this constraint as the number of True Negativess (numTNs) are missing (See Chapter 2.3). Therefore this work does not show any ROC curves.

4. Presentation of Results

This section presents the evaluation results of the different training parameter settings. How the evaluation works is explained in the previous chapter in Section 3.4.

Figure 4.1 shows the structure of this chapter. It is subdivided into five sections, where the last one, Section 4.5, is the summary. It outlines the most important results of the evaluation. The first Section 4.1 presents the starting conditions, meaning the performance of the pretrained upper body detector. Section 4.2 is divided into four subsections. The first three show the evaluation results of trained detectors, where only one training parameter has varying values. The last Subsection 4.2.4 shows how the precision and recall are affected, when combining the parameter values that proofed best performance in the three previous subsections. The third Section 4.3 of this chapter compares the performance of detectors trained with RGB and gray-scale images. For this measurement we used training and detection with default parameters. Section 4.4 explains the effects of running all four rotation detectors on one image category. The procedure is explained with an example image and afterwards presented for the default detectors. Section 4.5 summarizes the results mentioned above.

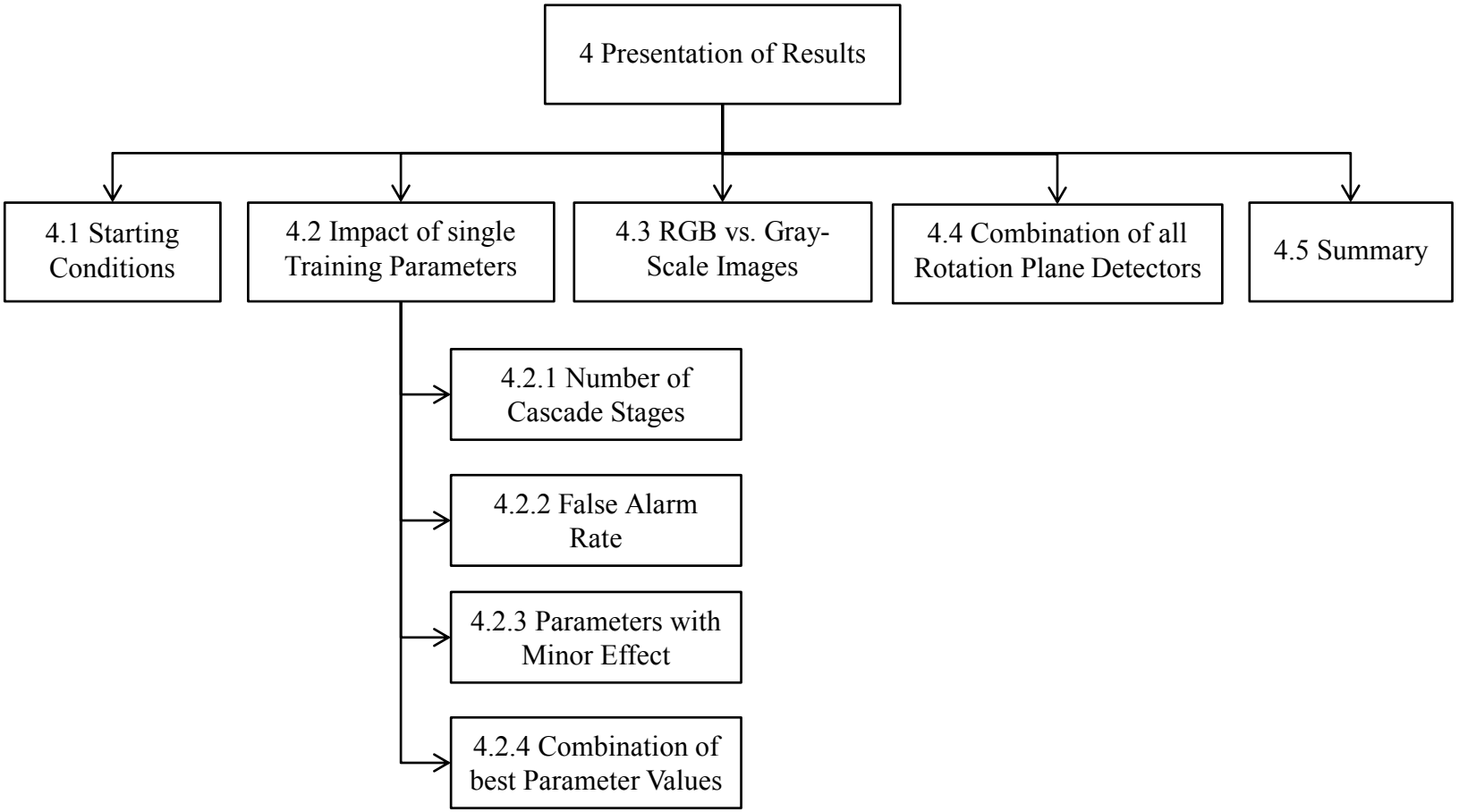


Figure 4.1.: This is the structure of the results chapter. Section 4.5 summarizes the chapter and shortly explains the most important effects.

4.1. Starting Conditions

As starting conditions we call the performance of the pretrained Matlab upper body detector applied on our evaluation images. We wanted to know how good the already existing detector can detect upper bodies in general. We evaluated the four rotations separately, so we were able to determine on which rotations the detector works best and which rotations need further improvement.

Figure 4.2 and Table 4.1 show the evaluation results of the starting conditions. We evaluated on RGB and gray-scale images, but they resulted in the exact same results for precision and recall. That was to be expected because the Haar-like features (used to train the pretrained detector) do not use colors for feature extraction. Figure 4.2 shows both precision (dark blue) and recall (cyan). Recall is also referred to as TPR and precision as 1-FDR (false discovery rate). For more information on the evaluation terms, read Section 2.3.

Figure 4.2 clearly shows that the recall is poor on back (13.17%), right (6.16%) and left (7.00%) view. Even at front view it only reaches 52.43%. The precision shows better values on all four sides. At front view it reaches an acceptable value of 91.53%, back view only has a precision of 73.97%. Right and left side have a precision under 50%. This results in a total precision of 76.90% and recall of 19.84%. These values will occur as black horizontal lines in subsequent diagrams in order to provide a direct comparison of the newly trained detectors and the starting conditions. They are only for illustration purposes and are not constant with changing parameters (as it is shown in the diagrams), but refer to the default parameters 3.1.

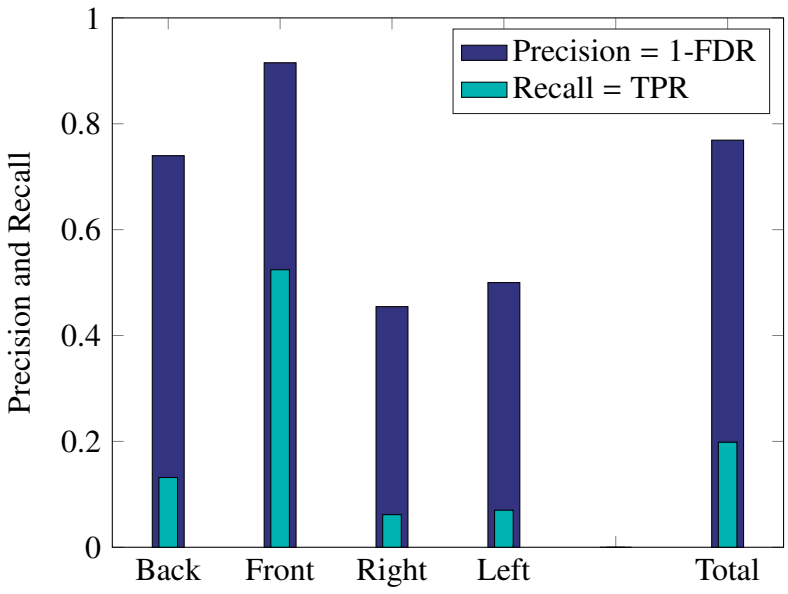


Figure 4.2.: The pretrained Matlab detector best works on front view (Precision = 91.53%, Recall = 52.43%). At back view the precision is still high (73.97%), but recall is quite as low as for right or left side view (below 14%). The precision for the side views is below 50%. In total the precision is at 76.90% and recall at 18.84%.

The meaning of higher precision than recall is, that there is less occurrence of the detector incorrectly labeling negative objects as positive than the incorrect labeling of positive objects as negative. So we have quite few FP detections, but rather a lot of FN. This means that the

Table 4.1.: The table shows the evaluation results of the starting conditions, that are illustrated in Figure 4.2

	Back	Front	Right	Left	Total
Precision	73.97%	91.53%	45.45%	50%	76.90%
Recall	13.17%	52.43%	6.16%	7.00%	19.84%

detector is more precise in distinguishing between upper bodies and non-upper bodies but therefore misses more upper bodies in its detection process.

Now that we have these results we can determine the degree of improvement depending on the parameter settings for the following measurements. The aim is to get approximately same detection accuracy for all four rotations and values for precision and recall that are as high as possible (best case: ≈ 1).

4.2. Impact of single Training Parameters

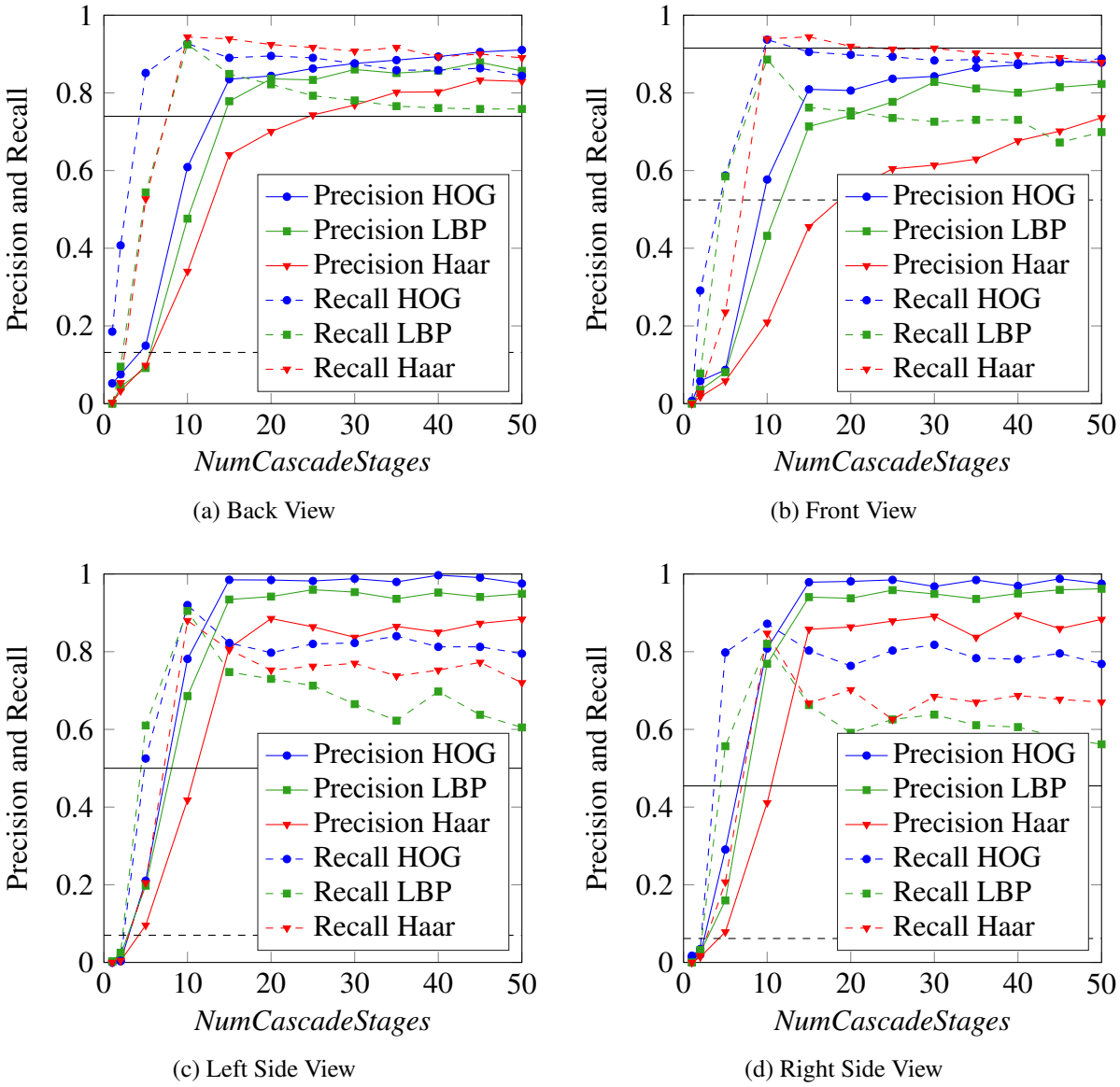
This Section presents the evaluation results of several parameter settings. The idea was to find out about performance changes as a function of a single parameter. The chosen parameters are introduced in Chapter 3.2.2, they are shortly mentioned again in the related chapters below. In the end we chose the value of each parameter that resembled the best results and combined them to a single detector, the outcome is presented in Section 4.2.4.

4.2.1. Number of Cascade Stages

The number of stages used to train a detector is an important value for the training function. The more stages, the higher is the chance that a sample is incorrectly classified as negative (cf. Figure 3.3). Therefore it is more challenging to pass many stages, than passing only a few. On the other hand, if a sample was labeled as positive at all stages, the probability that it is actually a positive sample is higher with more stages than with less. So we can make the presumption, that precision increases and recall decreases with increasing number of stages.

Figure 4.3 shows precision and recall of the four planes of rotation dependent on the number of stages. The three colors represent the feature types HOG, LBP and Haar. The values are discrete at 1, 2, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50 stages and therefore the depiction as a line graph is strictly speaking not correct. For better illustration we chose this type of diagram and no bar graph.

Generally the four diagrams look alike. At 1, 2 and 5 stages precision and recall are poor, then it rises rapidly between 5 and 15 stages. Afterwards increasing the number of stages does not result in major improvement, values stay approximately the same. The lookup tables of precision A.1 and recall A.2 are contained in the appendix.



(a) Back View

(b) Front View

(c) Left Side View

(d) Right Side View

Figure 4.3.: The figure shows precision and recall for various number of stages for back (a), front (b), left (c) and right (d). Precision and recall reach very low values for 1, 2 and 5 stages at all four planes of rotation. After rapid growth, precision and recall do not undergo major changes for increasing number of stages. 20 stages is the default value. The two black horizontal lines show precision (solid line) and recall (dashed line) of the starting conditions.

The Figures show that HOG features achieve the best precision and recall, except on front 4.3b and back 4.3a view, where Haar features reach slightly higher values for recall. Subsequent measurements show that on most parameter combinations HOG features work best. Recall is lower than precision on back and front view, but higher on right and left side view (values over 90%), this effect needs further investigation.

Recall increases faster than precision between 2 and 15 stages. This is because at these numbers of stages it is more easy to label a sample as positive (not so many stages, that might

classify the sample as negative) than classifying as negative. With higher number of stages numFP decreases (more stages, that can classify the sample as negative). Coincidentally the value of recall decreases a little (at 10 stages peak) because the increasing number of stages does not only increase the correct labeling of samples as negative, but also labeling some positives as negatives.

Whether precision or recall will be exposed to rapid changes with more than 50 stages needs further investigation. The more stages, the higher is the probability that the training is canceled due to insufficient number of images. Therefore, training with considerably more stages, needs more training images. Furthermore, the proportionality is doubtful, as the computation time increases with every additional stage, but the performance does not change significantly.

Figure 4.4 shows the comparison of the starting conditions with the detector performance of detectors with various number of stages. It is not subdivided into the four planes of rotation. For all three feature types the number of stages must be 15 or higher in order to achieve an improvement of both precision and recall. For Haar features it needs more stages to also improve precision, this is because the pretrained detector is trained with Haar features and therefore, the values are already quite high. For recall we obtain major improvement for all detectors with more than 2 stages. The highest improvement is achieved with 10 stages, but here precision is lower than with starting conditions.

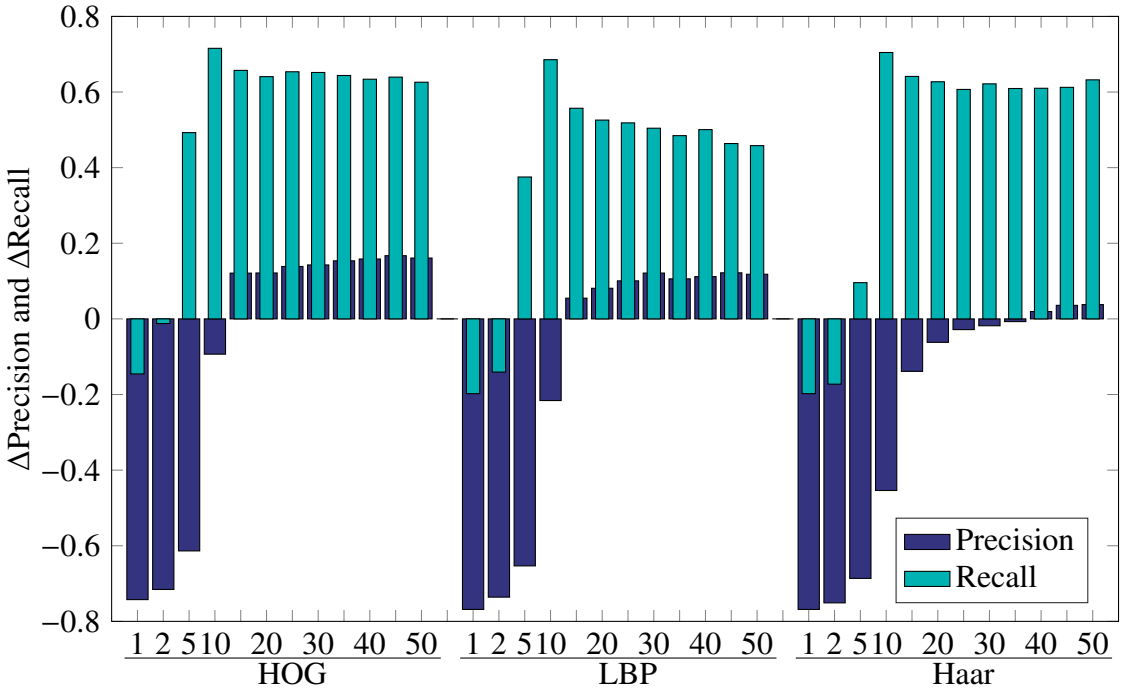


Figure 4.4.: The figure shows Δprecision and Δrecall for various number of stages compared to the starting conditions. A bar in the positive area states an improvement, a bar in the negative area is a deterioration. The graph easily states that a training with less than 15 stages is not recommended.

4.2.2. False Alarm Rate

The *FalseAlarmRate* parameter sets the acceptable FPR per stage. The overall FPR ($FAR^{NumCascadeStages}$) is calculated with the number of stages and the *FalseAlarmRate*. In this section recall and precision for five values (0.1, 0.3, 0.5, 0.7, 0.9) of *FalseAlarmRate* are presented. All other parameters maintain their default values (See Section 3.2.2). As in the previous Section 4.2.1, a bar graph would be the appropriate depiction, but with the same reasoning we decided on a line graph.
The aim is to have few FP detections in order to have a detector with a high precision. This means that precision is high for low values of *FalseAlarmRate* and decreases with higher *FalseAlarmRate*.

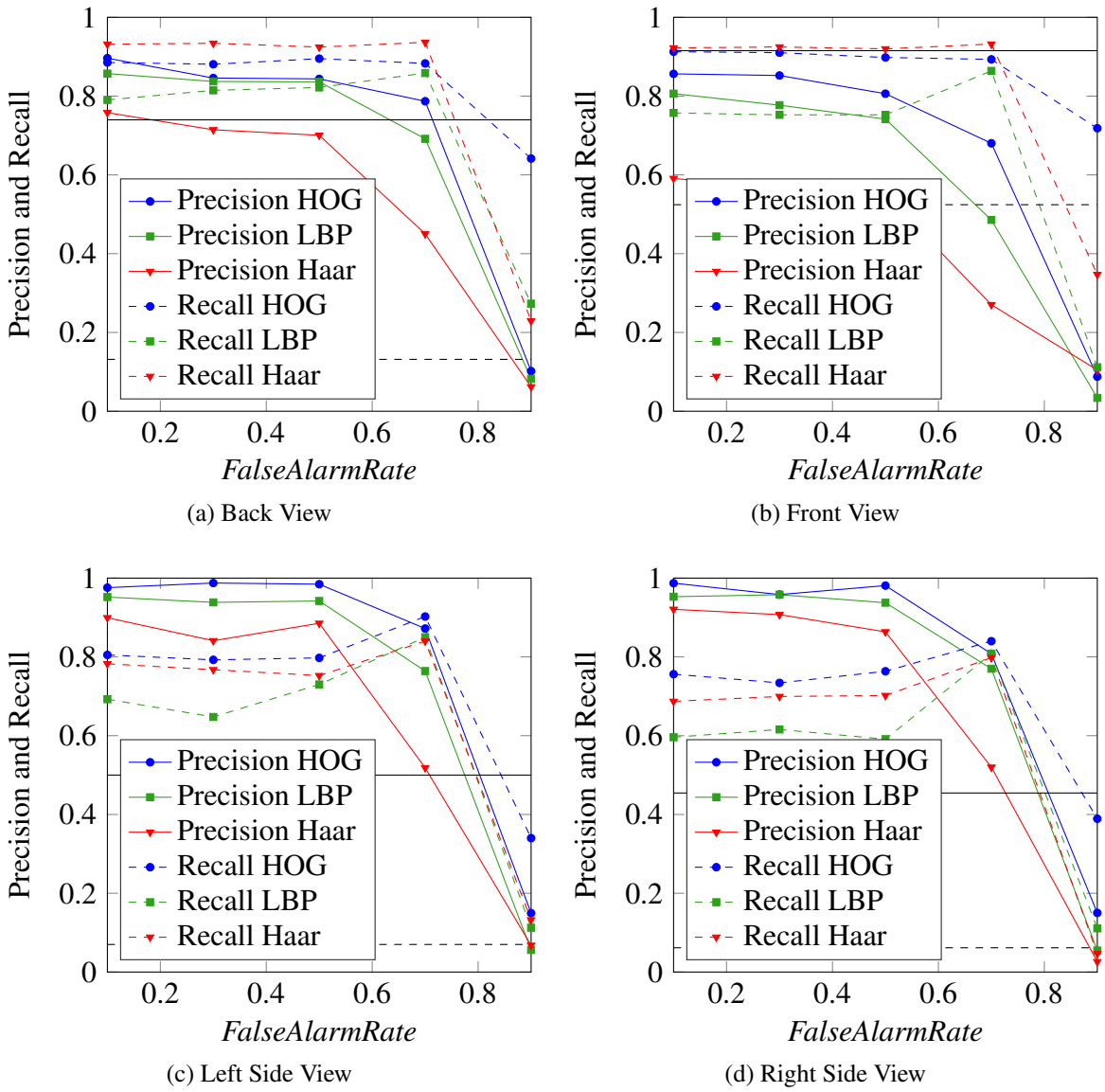
Figure 4.5 shows precision (solid line) and recall (dashed line) for back, front, right side and left side view dependent on the *FalseAlarmRate*. The two additional black horizontal lines represent the starting conditions. The corresponding lookup Tables A.3 and A.4 are contained in the appendix.

All four Figures show a decrease of precision for *FalseAlarmRate* higher than 0.5. At 0.9 precision is very low (under 20%), as expected before. Recall has approximately same values for 0.1, 0.3 and 0.5, for 0.7 it reaches higher results and then drop rapidly, like precision. The higher values for 0.7 are caused due to more positive labeling, resulting in more FP (lower precision) and more TP (higher recall). The expectation would be, that recall increases for a *FalseAlarmRate* of 0.9 as well, but with this value the training function was not able to ensure proper detection. Usually the detector spend under one minute to label the 400 evaluation images, but processing with *FalseAlarmRate* of 0.9 it took about 1-3 hours.

Figure 4.6 shows the comparison of the results for precision and recall for various *FalseAlarmRate* with the starting conditions. Here it is not subdivided into the four views, the values for total are in the Tables A.3 and A.4 in the appendix. In Figure 4.5 precision and recall showed very low values for 0.9 and partly 0.7, this Figure shows that precision has major deterioration compared to the starting conditions at these *FalseAlarmRate* values. For the other values at HOG and LBP it shows a slight increase, at Haar precision for the various *FalseAlarmRate* is always lower than the starting conditions. The pretrained detector (starting conditions) was trained with Haar features and is optimal adjusted to its field of application, our Haar detectors only vary their *FalseAlarmRate* and the other values are not adjusted, but maintain their default values. Therefore the pretrained detector reaches better values. Recall shows major improvement on all features, except at 0.9 with LBP and Haar features. The major improvement of recall are the two additional side detectors, which are able to detect people at side views and the back detector. The pretrained detector showed very low recall for these three sides (Figure 4.2) because it was almost entirely trained on front view images. With the training we achieved an improvement of recall between 50% to 70% (*FalseAlarmRate* of 0.9 excluded). The default value of 0.5 is the best to choose, because both recall and precision are high.

4.2.3. Parameters with Minor Effects

This section presents the evaluation results of the remaining parameters. They are consolidated in this section due to their unimportance concerning the effects. Shown are four values



(a) Back View

(b) Front View

(c) Left Side View

(d) Right Side View

Figure 4.5.: The values for *FalseAlarmRate* of 0.1 to 0.7 almost stay the same, but for 0.9 both precision and recall drop rapidly. The two black horizontal lines shows precision (solid line) and recall (dashed line) of the starting conditions.

for *NegSamplesFactor* and *TruePositiveRate*, each. More information about the parameters is given in Section 3.2.2. Here the evaluation results are not subdivided into the four planes of rotation, the detailed diagrams (Fig. A.1 and Fig. A.2) and the corresponding Tables (A.5, A.6, A.7 and A.8) are contained in the appendix.

Figure 4.7 shows the precision and recall for *NegativeSamplesFactor* of 1, 2, 3 and 4. The *NegativeSamplesFactor* determines how many negative samples are generated by the training function, by multiplying the amount of positive samples with the *NegativeSamplesFactor*. This factor does not severely affect the outcome of the evaluation of the trained detector. A proper parameter value can ensure, that there are enough negative samples provided in order to avoid premature canceling of the training, but not evoke major modification of the

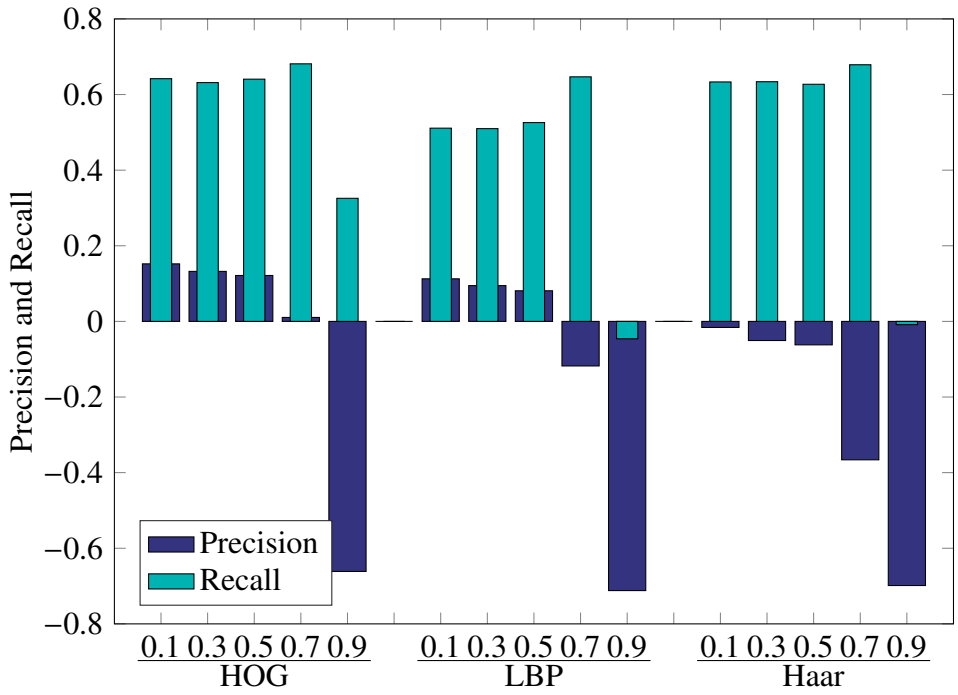


Figure 4.6.: The figure shows the precision and recall with various *FalseAlarmRate* compared to the starting conditions (Section 4.1). For recall we get a major improvement. For Haar features and *FalseAlarmRate* of 0.7 and 0.9 precision decreased. For the other values of HOG and LBP improvement is minor.

performance. The figure shows, that the recall slightly increases with rising negative samples factor and precision slightly decreases. They behave similar on all feature types.

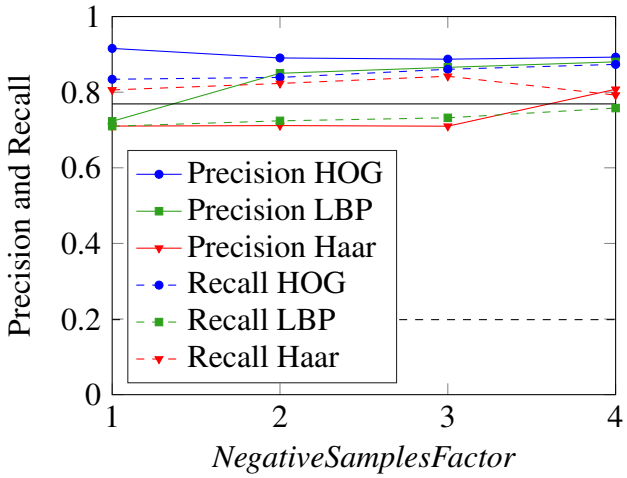


Figure 4.7.: Illustrated are the recall and precision for four values of *NegativeSamplesFactor* and the three feature types. The two black horizontal lines shows precision (solid line) and recall (dashed line) of the starting conditions. As the results stay approximately the same, this parameter has no big influence on the performance.

Figure 4.8 shows precision and recall for various *TruePositiveRate* values. With this param-

eter the desired TPR per stage is selected. It can be assumed, that recall (equivalent to TPR) depends on this parameter. The default value for *TruePositiveRate* is 0.995. As we want recall to be as high as possible, we examined the impact of slight changes of high *TruePositiveRate* values. As expected recall rises with increasing values for *TruePositiveRate*. The overall TPR is calculated by multiplying the TPRs of every stage. Not always a stage is able to reach the desired TPR modulated by the parameter value. Due to the multiplication of these varying values, recall never equals the value of *TruePositiveRate*. The Figure also shows no or only slight changes of precision with increasing *TruePositiveRate*.

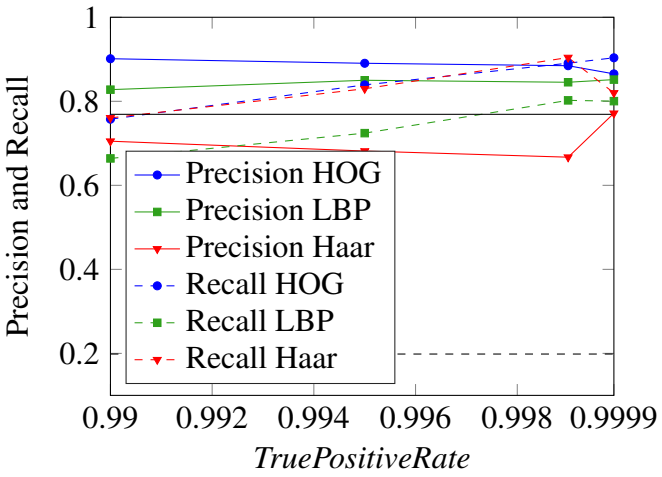


Figure 4.8.: The figure shows recall and precision for four values of *TruePositiveRate* and the three feature types. Precision stays constant and recall increases slightly with rising *TruePositiveRate*. The two black horizontal lines shows precision (solid line) and recall (dashed line) of the starting conditions.

To sum it up, we can say that these two parameters are more useful in doing the fine tuning after selecting the best combination of the other parameters. *NumCascadeStages* and *FalseAlarmRate* are the two most important parameters of the training function, along with the feature types. Anyway, in the next section we combined the values which achieved the best outcome to one detector.

4.2.4. Combination of best Parameter Values

This section presents the evaluation results of detectors that are trained with the parameter values that achieved best outcome in the evaluation of the single parameters. At first glance the combination should resemble a better detector, but in fact it is more difficult due to the correlation of the parameters.

First we evaluated the different parameter settings and then selected the ones with the best results for precision and recall. Table 4.2 shows the parameter values that reached highest results for all parameters. It is subdivided into the best results for precision and the best ones for recall. After that, we trained the detector with these parameter settings.

Table 4.2 shows that a good precision needs more stages and a lower *FalseAlarmRate* and *TruePositiveRate*, HOG is the best feature type here. For a high recall it is the other way

around, fewer stages and higher *FalseAlarmRate* and *TruePositiveRate* is needed. For back view and front view Haar features worked best, at the side views it is HOG. With these parameter combinations we can make first assumptions what outcome we are expecting.

Table 4.2.: The table shows the single parameter values that reached best results for precision and recall.

Parameter name	Precision				Recall			
	Back	Front	Right	Left	Back	Front	Right	Left
NumCascadeStages	50	45	45	40	10	15	10	10
FalseAlarmRate	0.1	0.1	0.1	0.3	0.7	0.7	0.7	0.7
NegativeSamplesFactor	1	1	1	1	1	1	4	4
TruePositiveRate	0.99	0.99	0.99	0.995	0.9999	0.999	0.9999	0.9999
FeatureType	'HOG'	'HOG'	'HOG'	'HOG'	'Haar'	'Haar'	'HOG'	'HOG'

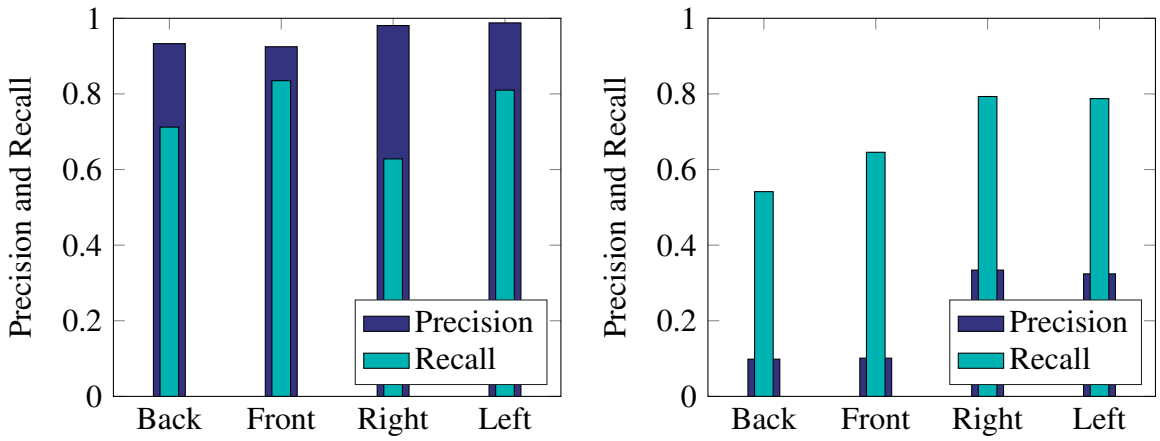
A high value for *NumCascadeStages* needs higher *FalseAlarmRate*, otherwise too many positives might be classified as negatives. For the precision combination we have many stages and a low *FalseAlarmRate*, this might result in a low recall. For the recall combination we have few stages and a high *FalseAlarmRate*. This means that after these few stages still a lot of negatives are classified as positive, so we expect to get a very low precision

Figure 4.9 shows two diagrams with the values of Table 4.2. Fig. 4.9a shows the parameter combination that achieved best precision on the individual parameters, the second Fig. 4.9b shows the best combinations for recall. Obviously the parameter combination for recall does not attain good results for both precision and recall, the results for the precision combination are better.

The expectation of a very low precision for the best recall combination fulfilled (see Figure 4.9b). The recall is also low for this combination, especially on front and back view. The lower values for back and front are because of the feature type, here Haar features were used. At the side views, where HOG was used, recall almost has 80%. This parameter combination does not result in a reliable detector.

Previously, we also expected a low recall for the best precision combination. Figure 4.9a in fact shows a lower recall, but still values are higher than 60%. So the detector with this parameter combination achieves better results than the one for the best recall combination. The precision here is high, over 92%. In order to improve this detector, the impact of the different parameters needs further investigation.

Shortly summarized, the parameters highly depend on each other and therefore, it does not suffice to find out the best value for every single parameter and to combine them. Especially the modification of *NumCascadeStages* reacts to the *FalseAlarmRate* and the *TruePositiveRate*. In order to find out the perfect parameter setting, a lot more time and training images are needed. In this work we focused on the effect of a single parameter, the next step is to examine the impact of two combined parameters.



(a) Combination for best Precision

(b) Combination for best Recall

Figure 4.9.: In (a) the results of a detector trained with parameter values that reached best outcome on precision are shown. In (b) it is the combination of single parameter values that reached best recall.

4.3. RGB vs. Gray-Scale Images

This section presents the evaluation results of the default training for both gray-scale and RGB images. The training parameters of the training function all obtain their default values, except *FeatureType*.

Figure 4.10 and Figure 4.11 show precision and recall of these parameter settings. The results for RGB images are shown in dark blue and for gray-scale images in cyan. The bars are grouped in three, they represent the three feature types: HOG, LBP and Haar (in this order). Table A.11 shows the corresponding values and is included in the appendix.

Figure 4.10 states that RGB images achieve higher values for precision than gray-scale images. The difference of the two image types is greater on front and back view. On the side views RGB and gray-scale reach approximately same values. *FeatureType* seems to cause outcome independent from the image type. For both RGB and gray-scale HOG features reach the highest precision on all four planes of rotation. For RGB images, precision values are between 97% and 99% and for gray-scale images between 80% and 98%. The difference between RGB and gray-scale are due to the better color discrimination with RGB images and therefore sharper edges.

Figure 4.11 shows better recall values for gray-scale images. Generally the recall values are lower than the precision values. Especially on the side views recall is quite poor, here it only reaches values between 58% and 80%. For front and back view recall values are at least higher than 70% and the difference between RGB and gray-scale images is larger than for the side views. With HOG features RGB images even have a slightly higher recall than gray-scale images and achieve the best performance for left and right views. At front and back views Haar features worked best on both image types.

Figure 4.12 summarizes precision and recall of the Figure 4.10 and Figure 4.11. Here the comparison of the three feature types is clearer. The image type and the feature type do not influence each other, they react independently. The ratio of precision and recall for the

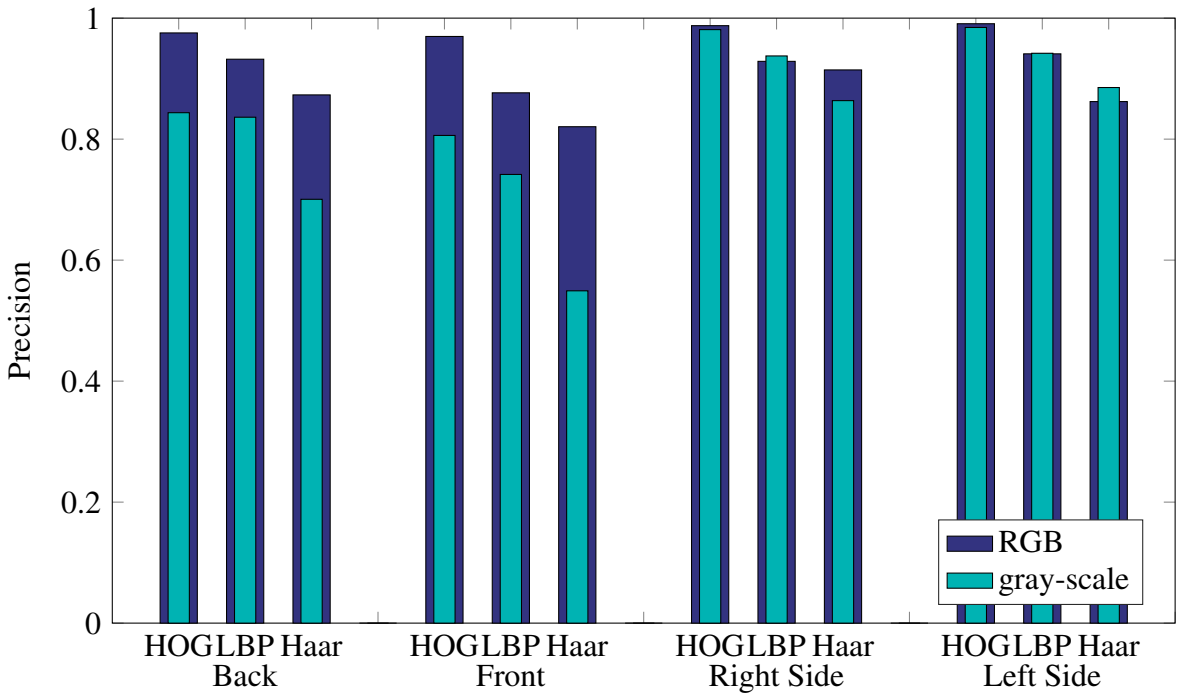


Figure 4.10.: The Figure shows the precision of a detector trained with default parameters and all three feature types for RGB and gray-scale images. The results for RGB exceed 80% for all feature types and rotations. For gray-scale the precision is considerably lower at back and front view.

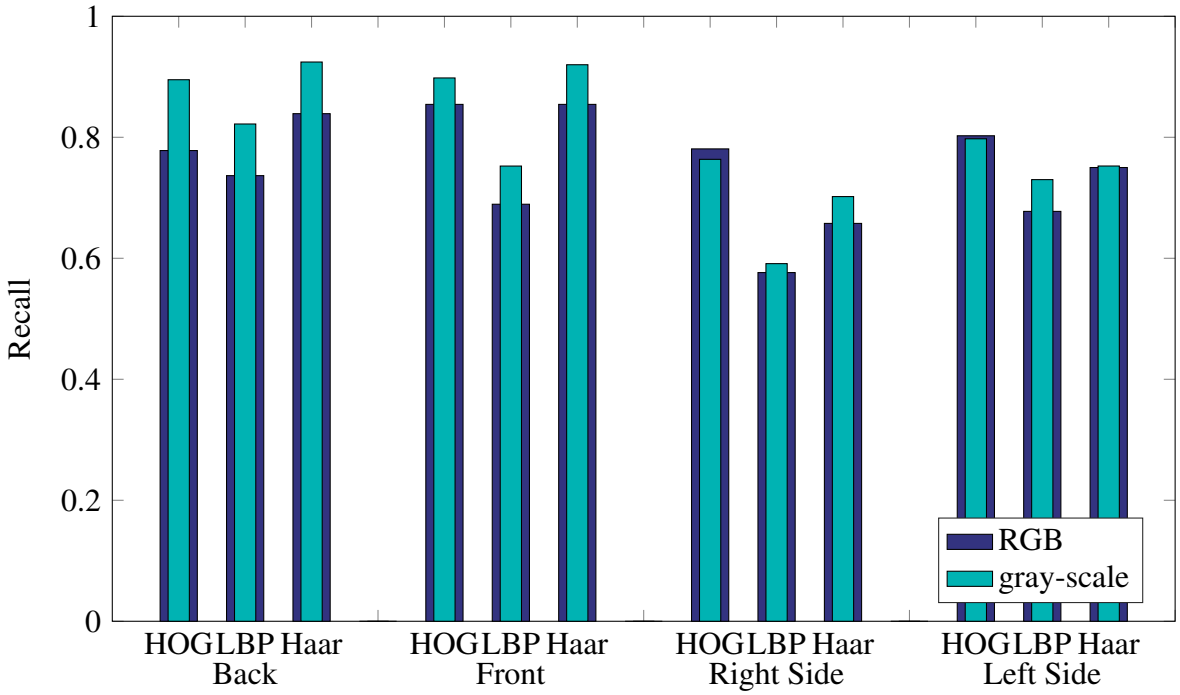


Figure 4.11.: Shown are the results of recall for a detector trained with default parameters and all three feature types. Here training with gray-scale images show slightly better results than with RGB images. LBP has the lowest recall of the feature types. At right side view it only reaches 57.64% for RGB images.

two image types stays the same for all three feature types. For this parameter setting HOG features are the best to choose, generally they show the highest precision and recall for both RGB and gray-scale. For LBP and Haar either precision or recall are too low to provide a reliable detection of upper bodies.

The default detectors trained with RGB images reached better precision values, the ones trained with gray-scale images reached higher recall values. RGB with its higher precision and lower recall consequently has fewer FP and more FN than gray-scale. This means with RGB images the detector is more precise in distinguishing between upper bodies and non upper bodies (quality), but has issues in recognizing all positives. The evaluation with gray-scale images resulted in more FP and fewer FN. Meaning that this detector does not miss as many upper bodies as the one with RGB images and consequently classifies more negatives as positive as well (quantity). To sum it up, RGB images are better used in order to be sure an object is an upper body and gray-scale images are better to detect as many upper bodies as possible.

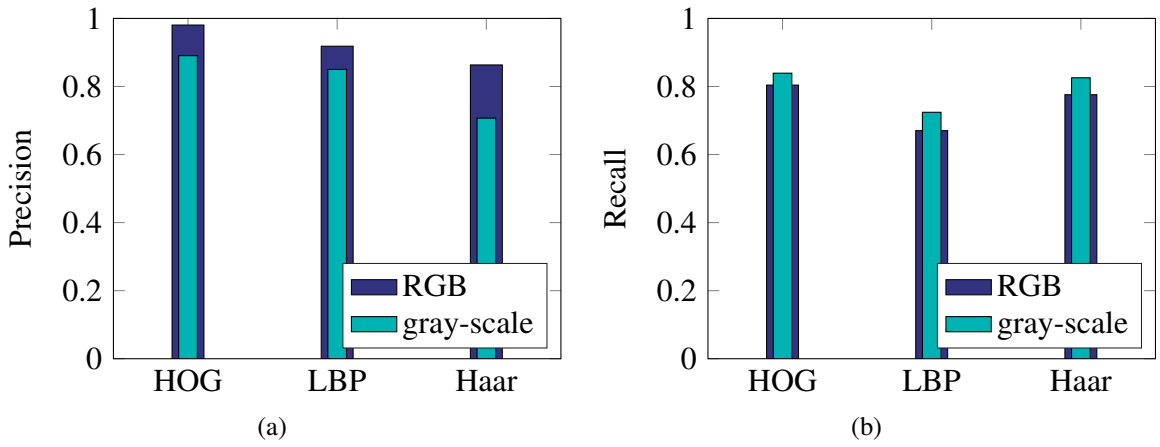


Figure 4.12.: (a) shows precision and (b) recall of the detector with default parameters; not subdivided into the rotation planes as in Figure 4.10 and Figure 4.11. RGB images reach higher precision and gray-scale images reach higher recall.

4.4. Combination of Rotation Plane Detectors

All of the previously presented detectors were considered as independent, meaning we trained a back view detector and applied it to back view images. For the actual application we have a set of images with people in all planes of rotations and the detectors shall detect the upper bodies. That means that the detectors of the four planes of rotations have to be combined. As mentioned before it is not the most important which one of the four detectors detects a person, but we want to guarantee, that approx. 100% of the people are detected by a minimum of one detector, hereinafter known as minimum one match. How we proceeded and evaluated the outcome is explained on the basis of one example image. For this example we chose a RGB image of the Graz-02 Database (filename: person_216.bmp) and RGB default detectors (previous Section 4.3). The ground truth data of the image is shown in red in Figure 4.13. Now we apply the four default detectors (back, front, right, and

left) to the image and count the TP, FP and FN for every detector (as in previous evaluations). Table 4.3 shows the evaluation result.

Table 4.3.: The table shows the amount of TN, FP, FN of the four detectors applied to one image. One BB was missed by all detectors, because the person is highly occluded. The back view detector detected all remaining persons sufficiently.

Detector	numTP	numFP	numFN	Precision [%]	Recall [%]
Back	3	0	1	100	75
Front	2	0	2	100	50
Right	1	1	3	50	25
Left	1	0	3	100	25

Figure 4.13 illustrates the BB_{dt} (yellow) per detector and the BB_{gt} (red). The image contains four persons (4 BB_{gt}), one of them is highly occluded. Both Figure 4.13 and Table 4.3 state, that the occluded person is not detected by any of the detectors. The other three persons are detected by a minimum of one detector. In this case the back view detector would be enough, because it detects the three persons and only misses the occluded one. Except of one, all other BB_{dt} label persons (numFP = 0). The right side view detector has one FP, the stone in the left of Figure 4.13d, this happens if objects show similar features to upper bodies. Indeed the stone could be a shoulder and a head.

Figure 4.13 clarifies that the four detectors are not clearly able to distinguish views, but the combination increases the chance of detecting all upper bodies in an image. Here the combination of the detectors reached a recall of 75%, three of four persons are detected. The detectors still need improvement in detecting occluded persons as well. This part was an example to explain what combination of the four detectors means. In the following we investigated the outcome when applying the default detectors to all of the 1600 evaluation images.

Figure 4.14 shows four diagrams. Each one represents one image category of gray-scale images. On the x axis the four detectors are listed. The Figures show that matching image category and detectors reach highest values for precision and recall. The left side view detector almost reaches 100% precision for left side view images. Moreover the figure shows, that back and front view detectors work better on foreign image categories than the side view detectors. Right side view detector works better on front and back view images, than on left side images. The same it is with left side view detector and right side view images. Front and back view detectors are very similar and as seen in the example before, they are difficult to distinguish.

The effect of very similar back and front view detectors is caused by our training images. As they have no high resolution (128x64 pixels) the head of a person does not feature specific face characteristics. Therefore the front view detector cannot orientate on those features. For the detector people look alike at front and back view without the face. This makes the front

(a) Back view detector

(b) Front view detector

(c) Left side view detector

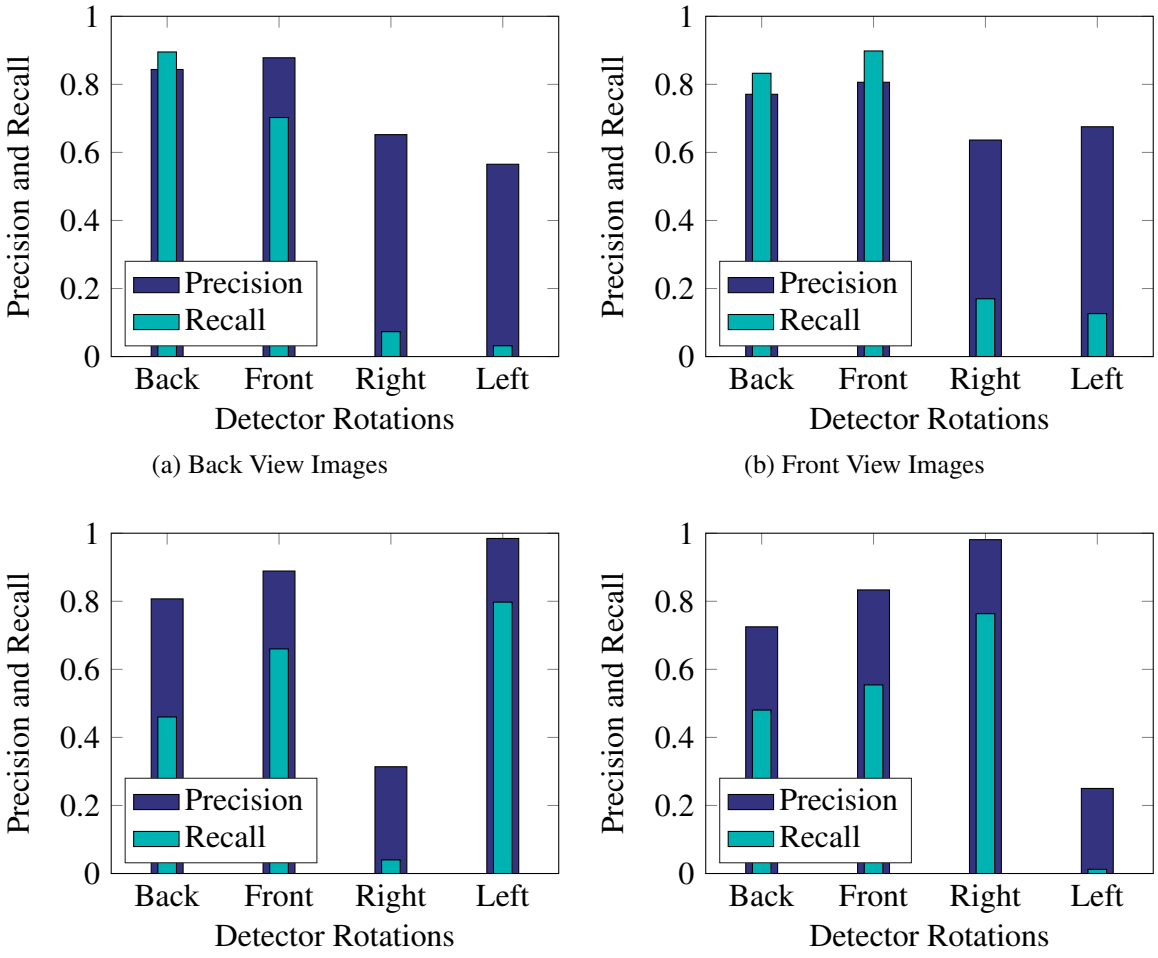
(d) Right side view detector

Figure 4.13.: The four figures (a), (b), (c) and (d) each represent one of the four detectors applied to the example image. The yellow boxes are the BB_{dt} of the corresponding detector. In red the BB_{gt} are shown. The occluded man was not detected by any of the four detectors.

view detector more robust. As mentioned before we want an upper body detector and no face detector. In contrast the pretrained upper body detector used face detection via local context [29] and therefore was not able to properly detect people without the face. In bad light conditions or low resolution the face features were not clear enough, moreover detection at back view did not work, though features are very similar. Here we have two detectors, one specialized on front view and one on back view, but both are able to detect a certain amount of upper bodies of the other side too, so they are able to complement each other.

Obviously there is a higher discrimination between right and left side view, than between those and front and back view. For example it is more likely for a left side view detector to label a front or back view person as positive than to label a person at right side view as positive. This is attributable to the fact that persons in the front and back view images still have a certain degree of turn and therefore also feature some of the side view features, whereas a right side view person with a certain degree of turn can only feature front or back view features, but no left side view features. Table A.10 with the values of Figure 4.14 is provided in the appendix.

Figure 4.15 and Table 4.4 show in dark blue how many of the upper bodies are detected by



(a) Back View Images

(b) Front View Images

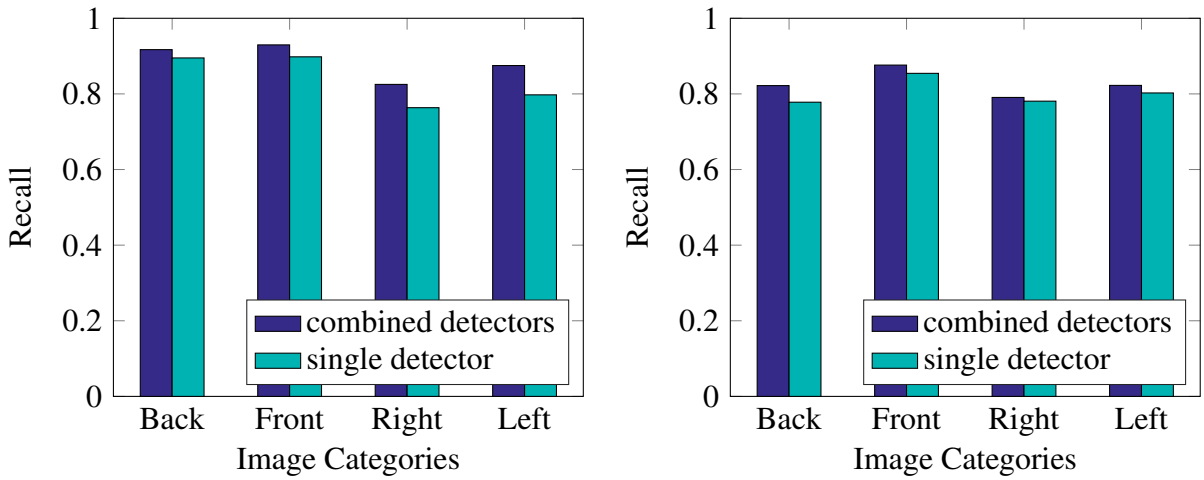
(c) Left Sie View Images

(d) Right Side View Images

Figure 4.14.: The four Figures represent the four evaluation image categories. The four default detectors are applied to all four image categories. The corresponding detector reaches best values for the image category, still back and front view detectors are strong on all image categories.

a minimum of one detector, this can be referred to as recall. In cyan there is the reference of recall when using a single detector. Figure 4.14 shows the default detectors trained and evaluated with gray-scale images, here the overall recall is shown for both RGB and gray-scale. Values are about 80% or higher, which can be further improved. The same evaluation can be done with all the other parameter combinations presented in this work. Due to the amount of combinations and results, only the results with default detectors are presented in this work.

Recall is lower if only the corresponding detector is applied to an image category (cyan), e.g. back view detector on back view images. The combination of the four detectors results in 10-30 more matches (on 400 evaluation images) per image category, resulting in a higher overall recall. For the gray-scale images recall is about 2% (back) to 8% (left) higher for combined detectors.



(a) Gray-Scale Images

(b) RGB Images

Figure 4.15.: The percentage of a minimum of one match represents recall. Due to the combination of the four detectors the overall recall is higher (dark blue) than by only using the corresponding detector. For 400 evaluation images per category, the combination results in 10-30 more matches.

Table 4.4.: Overall recall of combined default detectors in % (Figure 4.15)

	Back	Front	Right	Left
RGB	82.20	87.62	79.06	82.25
Gray-Scale	91.71	92.96	82.51	87.50

4.5. Summary

This section summarizes the presented results and the general improvement. For more detailed information read previous sections of this chapter.

The first step to improve the upper body detection in this work, was to ascertain the impact of a single parameter. In Section 4.2 the evaluation results for *FalseAlarmRate*, *NumCascadeStages*, *TruePositiveRate* and *NegativeSamplesFactor* along with *FeatureType* is presented for all four planes of rotation. In this section the diagrams for the overall results are shown.

Figure 4.16 shows precision and recall for several values of the four parameters and the feature type. The two black horizontal lines (solid line and dashed line) are the values of the starting conditions, all results that lay above these lines show an improvement of performance.

For appropriate detection at least 15 or more stages and a *FalseAlarmRate* below 0.7 (preferably 0.5) is needed. The parameters *NegativeSamplesFactor* and *TruePositiveRate* do not

have wide influence. As the recall of the starting conditions is rather low, almost all parameter combinations cause improvement (on average over 50%). By looking at the precision starting conditions, Haar features are below this line and therefore do not achieve improvement. HOG features work best for all four parameters as the combination of recall and precision reaches best outcome. Recall has an improvement of over 70% on average and precision 10% - 20%. As seen in Section 4.2.4, it is not advisable to take the parameter values with the best outcome in combination. The parameters have to be adjusted to one another in order to obtain a detector with very good results.

After examining the effects of single parameters we wanted to know if there is a difference between training and evaluation with RGB and gray-scale images. All previously evaluated detectors are trained with gray-scale images. For the comparison we chose the default detectors (default parameter settings are presented in Section 3.2.2 in Table 3.1).

Figure 4.17 shows recall and precision of both RGB and gray-scale for all three feature types. Precision is higher for RGB images, but gray-scale images reach a higher recall. As mentioned in Section 4.3, detectors trained with gray-scale images detect more upper bodies (higher TPR) but also label more non-upper bodies as upper bodies. A detector trained with RGB images instead misses more upper bodies but therefore labels less non-upper bodies as upper bodies. Therefore RGB images are better used to be sure an object is an upper body and gray-scale images in order to detect as many upper bodies as possible. For the default detectors HOG features have the best precision and recall, as in most other parameter settings presented before too. The impact of RGB images on other parameter settings needs further training and evaluation.

The last part was the combination of the four rotation detectors: back, front, right, and left. As expected the detector specialized on the image category attained highest results, meaning that e.g. on front view images the front view detector has the best performance. We also discovered that front and back view detectors are very similar, due to the resemblance of persons at front and back view. The front detector is not mainly trained on the face but on the contour of people, which appears the same for the back view. The left and right view detector vary significantly instead. A benefit of the combination is the enhancement of the recall.

Figure 4.18 shows the recall enhancement. In dark blue there are the four combined detectors and in cyan the single detector (e.g. front view detector applied to front view images). The combination cause an improvement of the overall recall from about 2% (back) to almost 8% (left). With 400 evaluation images this makes 10 to 30 more true positive detections. At other parameter settings this gap can be larger or smaller.

In summary, it is not trivial to find the perfect parameter combination due to the dependence and interference of the parameters, especially the feature type. The image itself also has influence on the performance (RGB and gray-scale). In practice the detectors will be combined in order to detect persons at all planes of rotation in an image, resulting in a slight improvement of recall.

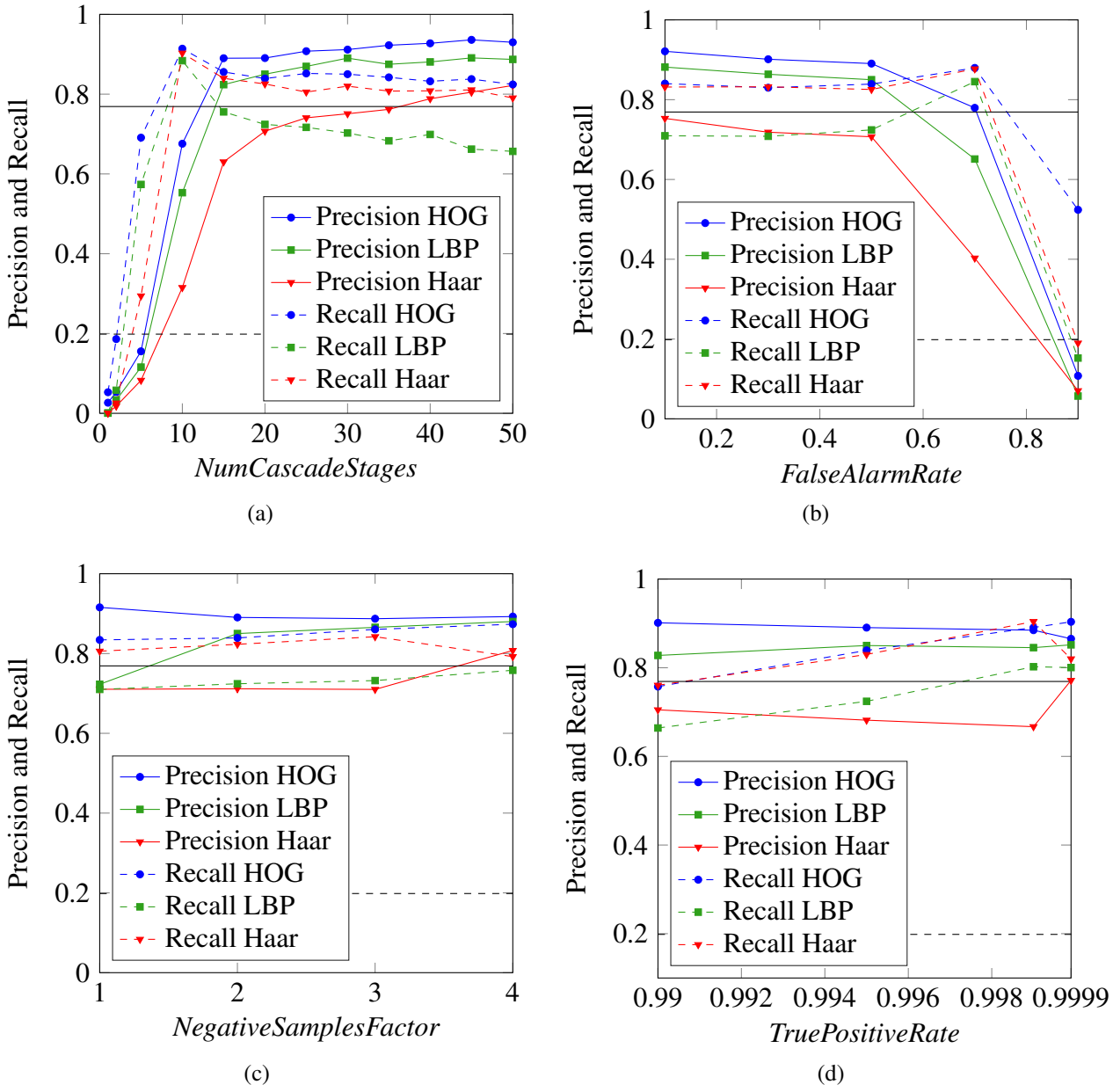


Figure 4.16.: Each marker in these diagrams represents a detector that was trained with the default parameters, a special feature type and the parameter value on the x axis. (a) shows the results for 12 different numbers of stages, (b) presents precision and recall for five values of *FalseAlarmRate*. In (c) and (d) four different values of *NegSamplesFactor* and *TruePositiveRate* are illustrated. The two black lines are the reference lines for precision (solid line) and recall (dashed line), they are the values of the starting conditions.

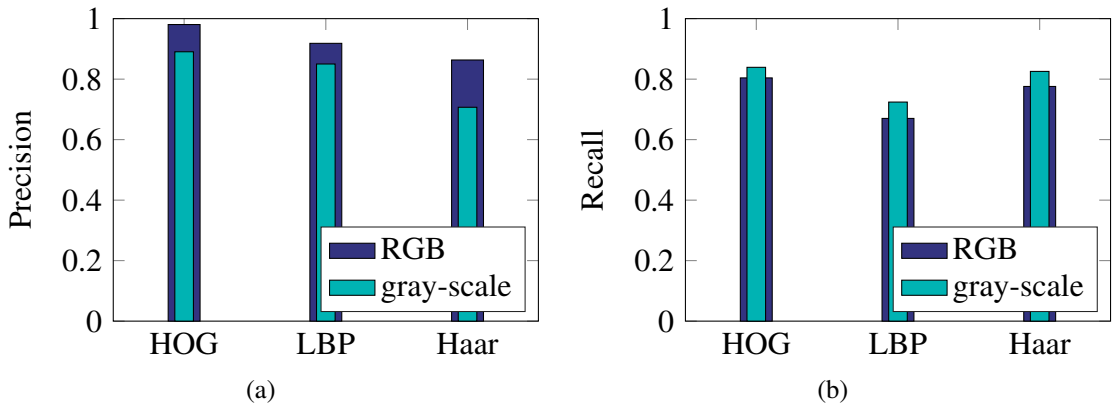


Figure 4.17.: The figure shows the comparison of default detectors trained with RGB and gray-scale images and the three feature types. In (a) there is precision and in (b) recall. RGB images reach higher precision and gray-scale images reach higher recall. HOG features get the best results of the three feature types.

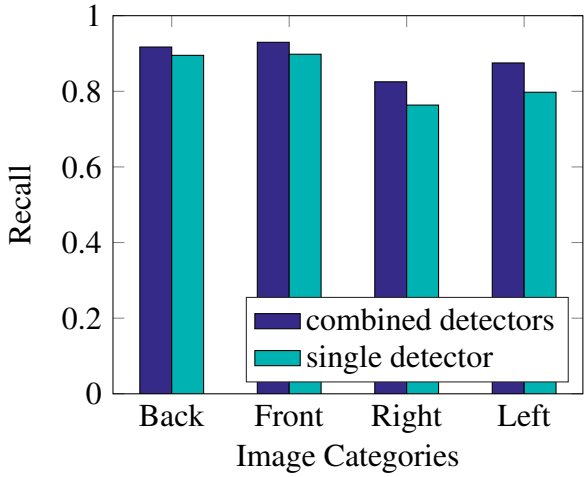


Figure 4.18.: The cyan bars show the recall of a single detector applied to an image category. The recall for combined detectors (in dark blue) is slightly higher.

5. Conclusion and Outlook

A training that is adjusted to the field of application can cause major improvement in terms of the resulting detector performance. For the side views we achieved an improvement of 70% for recall with the default detectors. Other parameter settings can cause higher improvement if they are well-matched, otherwise a worst detector emerge. The investigation of single parameters states, that it takes a lot of time and experience to find out the perfect parameter combination. This is intensified because the parameters are highly dependent on each other. The next step would be to examine the performance as a function of two parameters, then three and so forth. A minimization technique can be used for this in order to find the point of lowest FPR and FNR.

The differences between RGB and gray-scale images can be an advantage according to the field of application. Whether the differences also occur if a detector is trained with e.g. RGB images but is applied to RGB and gray-scale images needs further investigation. Our assumption is that no big changes will occur, as seen with the starting conditions. We took the pretrained detector (which was either trained with RGB or gray-scale images) and applied to both image types, we obtained the exact same results for both types.

In practice we do not apply a single back view detector to an image but all four detectors in order to provide detection of persons in all planes of rotation. This combination does not only ensure the detection of all persons in an image, but also strengthens the overall recall. This means that the back view detector is able to detect 370 of 400 people at back view, with the other three detectors 15 additional back view people can be detected. Depending on the plane of rotation and the parameter settings this can be a recall improvement from 2% to 12%. It would be interesting to know if this effect also occurs when the detectors are more specialized on their rotation. Yet we combined detectors with the same parameter settings but trained on four different image categories. In practice the best parameter settings for every plane of rotation are observed and then the detectors are combined. Due to the better adaption to the particular rotation plane, the detectors are probably not longer able to detect as many upper bodies of a foreign rotation, but therefore are able to detect more of their own upper bodies.

Furthermore we decided on four rotation plane detectors. Whether more additional detectors cause a better robustness against postures, that are between e.g. front and right side, would be interesting to examine. Moreover the detection of persons from various horizontal angles is interesting for e.g. surveillance. Here cameras are mostly attached directly under the ceiling. Therefore, not only detectors at a constant height (this work) should be trained, but also ones with several heights in order to get a variety of horizontal angles. Probably the detection performance will increase.

In this work we always used the same detector parameters. The adaption of those will especially affect the detection time. But the merge threshold parameter can also cause a better detection accuracy. The adaption of this parameter can cause less FP detections (higher

value) or less FN detections (lower value). In order to acquire the best detector, the detector parameters have to be adjusted as well.

What this all amounts to is that we trained over 300 detectors with 75 parameter combinations, evaluated them and either achieved no, low or high improvement but still further improvement of the detectors is necessary. It is easy to either have a high recall or a high precision, but a good detector needs both. This work lays the foundation of training the **Matlab Cascade Object Detector** but there are a lot more parameters and possibilities that can be examined. For example the answering of the question how many detectors are needed in order to provide satisfactory detection of people in all planes of rotation. Anyway it can be said that the training is absolutely reasonable, because it causes, even with default values, a major improvement compared to the starting conditions.

A. Appendix

The appendix contains a short description of the used databases and the tables of the evaluation results. All tables are already mentioned in the chapters, they are only necessary to see the exact evaluation results. An overview is given by the diagrams contained in the chapters.

A.1. Databases

Appropriate data sets are needed to carefully validate any new feature set and show its potential for practical applications. The data sets should be chosen to be representative for the applications under consideration and they should not contain selection biases that will manipulate the results. This appendix section gives a short introduction to the MIT pedestrian dataset and the Graz-02 database.

MIT Pedestrian Dataset The MIT pedestrian dataset is a image database with 924 images [11]. The training database was generated from color images and video sequences. Several different digital cameras and video recorders have been used. The pose of the people in this dataset is limited to frontal and rear views. The images are extracted from raw data and scaled to the size 64x128 pixels. The person's body is in the center of the image and the height is homogeneous; distance between shoulders and feet is approx. 80 pixels.

Graz-02 Database The Graz-02 database [24] is the successor of the Graz-01 database [30]. It is divided into three categories, bikes, persons and cars, in this work we used the person database. Additionally there is a counter-class, these are the images we used to generate the negative samples. The database contains 365 images with bikes, 311 images with persons, 420 images with cars and 380 not containing one of these objects. The database also provides the ground truth data for 300 images of each category. The data is given as binary masks with values between 0 (black) and 255 (white), where pixels with 0 denote the object in the image.

Simulated Images The simulated images contain two image sets, one called John Doe and the other Erika Muster. This is to obtain more variance in body structure, height and hair style. In order to get several postures and backgrounds the persons walk through the room and the background changes during the simulation, the images are gained from this video material. Additionally the color of the clothes, hair and skin changes during the simulation. The size of the images is adapted to the ones of the MIT database, meaning 64x128 pixels, furthermore, the persons are aligned and the distance to the camera stays constant.

A.2. Evaluation Results

In this section there are the tables with the evaluation results of Chapter 4. The tables are to complement the information of the diagrams and to see the exact evaluation results. The two figures contained in this chapter show the more detailed evaluation results for the parameters *NegativeSamplesFactor* and *TruePositiveRate*.

Table A.1.: Precision for various values of *NumCascadeStages* in %

Num Cascade Stages	Back			Front			Right			Left			Total		
	HOG	LBP	Haar	HOG	LBP	Haar	HOG	LBP	Haar	HOG	LBP	Haar	HOG	LBP	Haar
1	5.22	0.00	0.23	0.41	0.00	0.00	1.27	0.00	0.00	0.00	0.24	0.00	2.66	0.06	0.06
2	7.55	4.34	3.27	5.82	3.50	1.75	1.96	2.53	1.40	0.29	1.95	0.37	5.34	3.31	1.79
5	14.91	9.15	9.84	8.68	8.09	5.81	29.06	15.96	7.84	21.04	19.76	9.57	15.54	11.58	8.27
10	60.90	47.61	34.04	57.70	43.20	20.94	80.82	76.91	41.10	78.13	68.56	41.81	67.57	55.30	31.52
15	83.52	77.85	64.06	80.91	71.36	45.55	97.90	94.06	85.76	98.50	93.44	80.90	89.01	82.38	63.02
20	84.37	83.62	70.06	80.61	74.16	54.93	98.10	93.75	86.36	98.46	94.19	88.53	89.05	85.00	70.70
25	86.29	83.33	74.31	83.64	77.69	60.45	98.49	95.85	87.89	98.20	95.96	86.40	90.77	86.96	74.07
30	87.56	86.02	76.86	84.26	82.83	61.40	96.79	94.87	89.10	98.80	95.34	83.70	91.17	89.03	75.08
35	88.44	85.09	80.17	86.49	81.13	62.94	98.45	93.58	83.69	97.96	93.61	86.51	92.26	87.49	76.19
40	89.34	85.71	80.26	87.20	80.05	67.64	96.94	94.98	89.42	99.69	95.22	85.03	92.74	88.08	78.85
45	90.54	87.85	83.30	88.08	81.47	70.17	98.78	95.92	85.94	99.09	94.10	87.29	93.62	89.09	80.49
50	91.05	85.67	82.95	87.77	82.29	73.58	97.50	96.20	88.31	97.55	94.90	88.34	93.00	88.71	82.18

Table A.2.: Recall for various values of *NumCascadeStages* in %

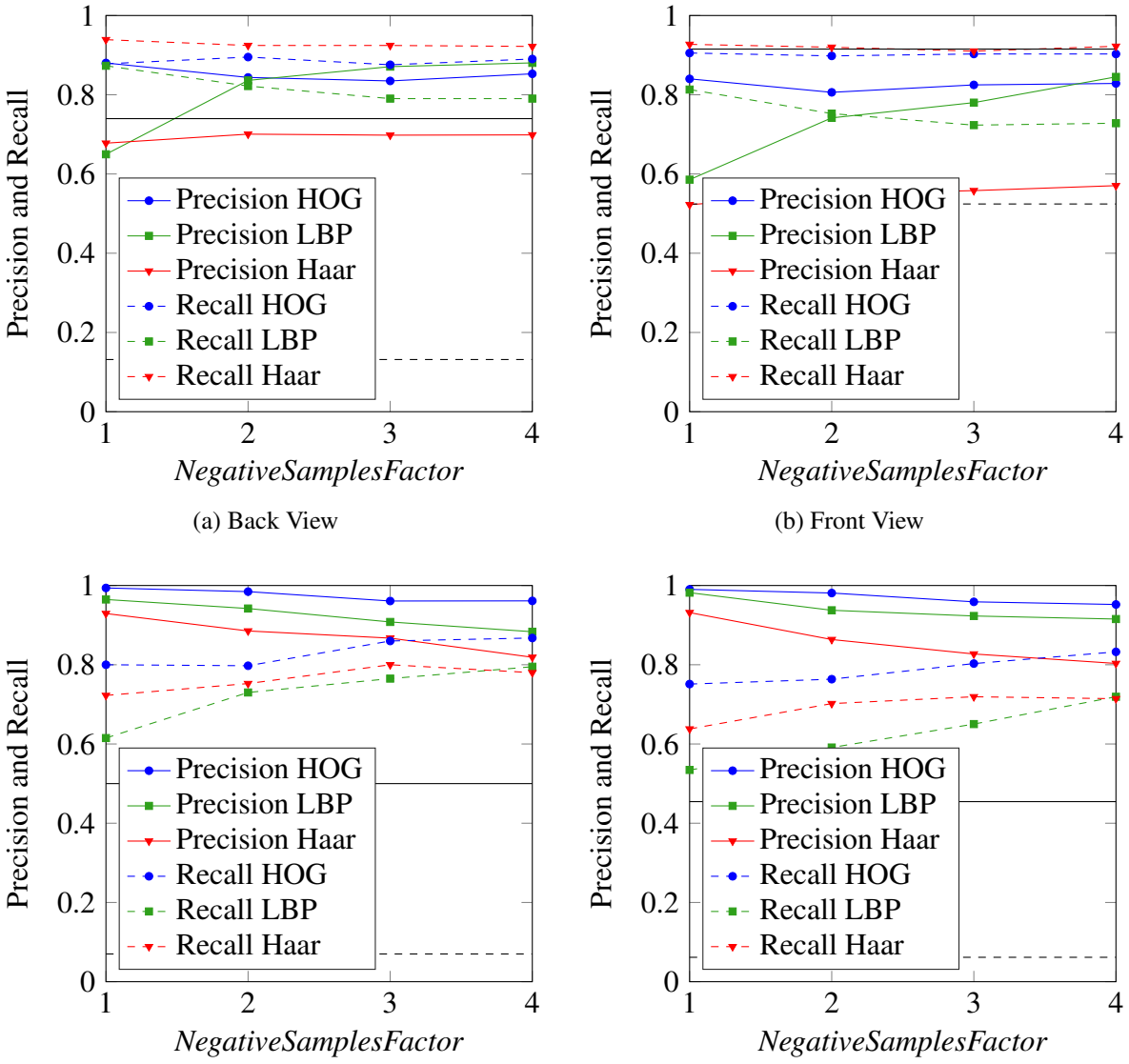
Num Cascade Stages	Back			Front			Right			Left			Total		
	HOG	LBP	Haar	HOG	LBP	Haar	HOG	LBP	Haar	HOG	LBP	Haar	HOG	LBP	Haar
1	18.54	0.00	0.24	0.73	0.00	0.00	1.72	0.00	0.00	0.00	0.25	0.00	5.28	0.06	0.06
2	49.73	9.51	5.37	29.13	7.77	2.67	3.45	3.20	1.72	0.50	2.50	0.50	18.61	5.77	2.58
5	85.12	54.39	52.68	58.74	58.50	23.54	79.80	55.67	20.69	52.50	61.00	20.50	69.10	57.37	29.42
10	92.68	92.44	94.39	93.69	88.59	93.93	87.19	82.02	84.73	92.00	90.50	88.00	91.40	88.39	90.29
15	89.02	84.88	93.90	90.53	76.21	94.42	80.30	66.26	66.75	82.25	74.54	80.50	85.57	75.55	83.97
20	89.51	82.20	92.44	89.81	75.24	91.99	76.35	59.11	70.20	79.75	73.00	75.25	83.91	72.42	82.56
25	89.02	79.27	91.71	89.32	73.54	91.26	80.30	62.56	62.56	82.00	71.25	76.25	85.20	71.68	80.53
30	87.56	78.05	90.73	88.35	72.57	91.50	81.77	63.79	68.47	82.25	66.50	77.00	85.01	70.27	82.00
35	85.85	76.59	91.71	88.59	73.06	90.29	78.33	61.08	67.00	84.00	62.25	73.73	84.21	68.30	80.77
40	85.85	76.10	89.27	87.62	73.06	89.81	78.08	60.59	68.72	81.25	69.75	75.25	83.23	69.90	80.84
45	86.34	75.85	90.00	87.86	67.23	89.08	79.56	57.88	67.73	81.25	63.75	77.25	83.78	66.22	81.08
50	84.39	75.85	89.02	88.83	69.90	87.86	76.85	56.16	67.00	79.50	60.50	72.00	82.43	65.66	79.05

Table A.3.: Precision for various values of *FalseAlarmRate* in %

FalseAlarmRate (*FAR*)	Back			Front			Right			Left			Total		
	HOG	LBP	Haar	HOG	LBP	Haar	HOG	LBP	Haar	HOG	LBP	Haar	HOG	LBP	Haar
0.1	89.63	85.71	75.79	85.65	80.62	59.10	98.71	95.28	92.08	97.58	95.19	89.94	92.12	88.17	75.31
0.3	84.58	83.71	71.46	85.23	77.69	56.70	95.82	95.79	90.73	98.75	93.84	84.11	90.13	86.37	71.85
0.5	84.37	83.62	70.06	80.61	74.16	54.93	98.10	93.75	86.36	98.46	94.19	88.53	89.05	85.00	70.70
0.7	78.70	69.16	45.02	68.02	48.57	27.04	80.81	77.00	52.01	87.20	76.40	51.85	77.95	65.12	40.29
0.9	10.18	8.21	6.20	8.78	3.39	10.47	15.00	5.46	2.63	14.95	5.60	6.79	10.77	5.71	7.05

Table A.4.: Recall for various values of *FalseAlarmRate* in %

FAR	Back			Front			Right			Left			Total		
	HOG	LBP	Haar	HOG	LBP	Haar	HOG	LBP	Haar	HOG	LBP	Haar	HOG	LBP	Haar
0.1	88.54	79.02	93.17	91.26	75.73	92.23	75.62	59.61	68.72	80.50	69.25	78.25	84.03	70.95	83.17
0.3	88.08	81.46	93.41	91.02	75.24	92.48	73.40	61.58	69.95	79.25	64.75	76.75	83.00	70.82	83.23
0.5	89.51	82.20	92.44	89.81	75.24	91.99	76.35	59.11	70.20	79.75	73.00	75.25	83.91	72.42	82.56
0.7	88.29	85.85	93.66	89.32	86.41	93.20	83.99	80.79	79.80	90.25	85.00	84.00	87.96	84.52	87.71
0.9	64.15	27.31	22.93	71.84	11.17	34.71	38.92	11.08	4.68	34.00	11.25	13.25	52.40	15.23	18.98



(a) Back View

(b) Front View

(c) Left Side View

(d) Right Side View

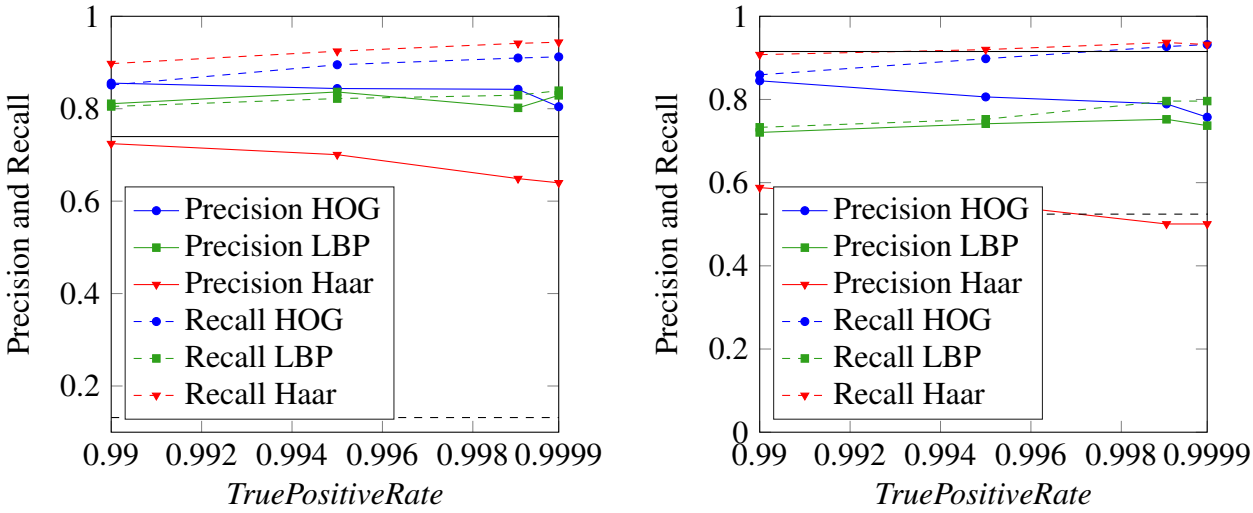
Figure A.1.: Precision and Recall with various values for *NegativeSamplesFactor*. The two black lines represent precision (solid line) and recall (dashed line) of the starting conditions.

Table A.5.: Precision for various values of *NegativeSamplesFactor* in %

Negative Samples Factor	Back			Front			Right			Left			Total		
	HOG	LBP	Haar	HOG	LBP	Haar	HOG	LBP	Haar	HOG	LBP	Haar	HOG	LBP	Haar
1	88.02	64.97	67.78	84.01	58.57	52.26	99.03	98.19	93.17	99.38	96.47	92.93	91.57	72.30	71.03
2	84.37	83.62	70.06	80.61	74.16	54.93	98.10	93.75	86.36	98.46	94.19	88.53	89.05	85.00	71.16
3	83.49	87.10	69.80	82.48	78.01	55.80	95.88	92.31	82.72	96.09	90.80	86.72	88.73	86.56	71.00
4	85.28	88.04	69.87	82.85	84.51	57.06	95.21	91.54	80.33	96.12	88.33	81.89	89.27	88.02	80.77

Table A.6.: Recall for various values of *NegativeSamplesFactor* in %

Negative Samples Factor	Back			Front			Right			Left			Total		
	HOG	LBP	Haar	HOG	LBP	Haar	HOG	LBP	Haar	HOG	LBP	Haar	HOG	LBP	Haar
1	87.80	87.32	93.90	90.53	81.31	92.72	75.12	53.45	63.79	80.00	61.50	72.25	83.42	71.01	80.59
2	89.51	82.20	92.44	89.81	75.24	91.99	76.35	59.11	70.20	79.75	73.00	75.25	83.91	72.42	82.31
3	87.56	79.02	92.44	90.29	72.33	91.02	80.30	65.02	71.92	86.00	76.50	80.00	86.06	73.22	84.21
4	89.02	79.02	92.20	90.29	72.82	92.23	83.25	71.92	71.43	86.75	79.50	78.00	87.35	75.80	79.22



(a) Back View

(b) Front View

(c) Left Side View

(d) Right Side View

Figure A.2.: Precision and Recall with various values for *TruePositiveRate*. The two black lines represent precision (solid line) and recall (dashed line) of the starting conditions.

Table A.7.: Precision for various values of *TruePositiveRate* in %

TruePositiveRate (*TPR*)	Back			Front			Right			Left			Total		
	HOG	LBP	Haar	HOG	LBP	Haar	HOG	LBP	Haar	HOG	LBP	Haar	HOG	LBP	Haar
0.99	85.54	81.08	72.44	84.49	72.08	58.81	98.34	92.25	87.40	97.67	95.05	88.51	90.13	82.77	70.50
0.995	84.37	83.62	70.06	80.61	74.16	54.93	98.10	93.75	86.36	98.46	94.19	88.53	89.05	85.00	68.16
0.999	84.20	80.19	64.87	78.93	75.23	50.06	97.74	91.45	80.65	97.49	94.80	85.64	88.47	84.53	66.70
0.9999	80.43	82.89	63.97	75.74	73.71	50.07	97.79	93.33	80.09	98.09	95.00	80.73	86.53	85.16	77.15

Table A.8.: Recall for various values of *TruePositiveRate* in %

TPR	Back			Front			Right			Left			Total		
	HOG	LBP	Haar	HOG	LBP	Haar	HOG	LBP	Haar	HOG	LBP	Haar	HOG	LBP	Haar
0.99	85.12	80.49	89.76	85.92	73.30	90.78	58.37	58.62	56.40	73.25	52.75	65.50	75.74	66.40	76.04
0.995	89.51	82.20	92.44	89.81	75.24	91.99	76.35	59.11	70.20	79.75	73.00	75.25	83.91	72.42	82.99
0.999	90.98	82.93	94.15	92.72	79.61	93.69	85.22	76.35	86.21	87.25	82.00	88.00	89.07	80.22	90.42
0.9999	91.22	83.90	94.39	93.20	79.61	93.20	87.19	75.86	87.19	89.75	80.75	88.00	90.36	80.04	82.00

Table A.9.: Precision and Recall for best parameter combination in %

Evaluation results	best for precision				best for recall			
	Back	Front	Right	Left	Back	Front	Right	Left
Precision	93.29	92.47	98.08	98.78	9.85	10.12	33.40	32.41
Recall	71.22	83.50	62.81	81.00	54.15	64.56	79.31	78.75

Table A.10.: Combined default detectors applied to gray-scale evaluation images. Precision and Recall in %.

Categories	Back View		Front View		Right Side View		Left Side View	
	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
Back View Detector	97.55	77.80	68.88	77.95	64.78	45.35	78.30	45.77
Front View Detector	79.78	66.21	96.97	85.44	77.32	52.33	87.71	65.67
Right Side View Detector	57.69	6.85	57.38	15.56	98.75	78.08	30.77	3.98
Left Side View Detector	39.39	2.97	59.77	11.61	23.81	1.16	99.07	80.25

Table A.11.: Precision and recall for detectors with default values, trained on RGB and gray-scale images, in %.

	image type	Back			Front			Right			Left		
		HOG	LBP	Haar	HOG	LBP	Haar	HOG	LBP	Haar	HOG	LBP	Haar
Precision in %	RGB	97.55	93.21	87.31	96.97	87.65	82.05	98.75	92.86	91.44	99.07	94.10	86.21
	gray-scale	84.37	83.62	70.06	80.61	74.16	54.93	98.10	93.75	86.36	98.46	94.19	88.53
Recall in %	RGB	77.80	73.66	83.90	85.44	68.93	85.44	78.08	57.64	65.76	80.25	67.75	75.00
	gray-scale	89.51	82.20	92.44	89.81	75.24	91.99	76.35	59.11	70.20	79.75	73.00	75.25

List of Acronyms

BB Bounding Box 14
BB_{dt} detected Bounding Box 25
BB_{gt} ground truth Bounding Box
FAR *FalseAlarmRate*(training parameter name) 55
FAST Features from Accelerated Segment Test 4
FN False Negative 9
FNR False Negative Rate 17
FP False Positive 9
FPR False Positive Rate 10
HOG Histograms of Oriented Gradients 4
LBP Local Binary Pattern 5
MSER Maximally Stable Extremal Regions 4
numFN number of False Negatives 25
numFP number of False Positives 25
numTN number of True Negatives 26
numTP number of True Positives 25
PCBR Principal Curvature-Based Region 4
PPV Positive Predictive Value
ROC Receiver Operating Characteristic 10
SIFT Scale-Invariant Feature Transform 5
SUSAN Smallest Univalue Segment Assimilating Nucleus 4
TN True Negative 9
TP True Positive 9
TPR True Positive Rate 10
TPR *TruePositiveRate*(training parameter name) 60

List of Figures

List of Tables

Bibliography

[1] C. Solomon and T. Breckon, "Fundamentals of digital image processing - a practical approach with examples in matlab," 2011.

[2] J. Canny, "A computational approach to edge detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. PAMI-8, no. 6, pp. 679–698, Nov 1986.

[3] S. M. Smith and J. M. Brady, "Susan - a new approach to low level image processing," *International Journal of Computer Vision*, vol. 23, pp. 45–78, 1995.

[4] E. Rosten, G. Reitmayr and T. Drummond, "Real-time video annotations for augmented reality," *International Symposium on Visual Computing*, pp. 294–302, 2005.

[5] J. Matas, O. Chum, M. Urban and T. Pajdla, "Robust wide baseline stereo from maximally stable extremal regions." in *Proc. of British Machine Vision Conference*, 2002, pp. 384–396.

[6] H. Deng, W. Zhang, E. Mortensen, T. Dietterich and L. Shapiro, "Principal curvature-based region detector for object recognition," in *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, June 2007, pp. 1–8.

[7] R. Haralick, "Ridges and valleys on digital images," *Computer Vision, Graphics, and Image Processing*, vol. 22, pp. 28–38, 1983.

[8] J. Crowley and A. C. Sanderson, "Multiple resolution representation and probabilistic matching of 2-d gray-scale shape," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. PAMI-9, no. 1, pp. 113–121, Jan 1987.

[9] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1, June 2005, pp. 886–893 vol. 1.

[10] D. Lowe, "Object recognition from local scale-invariant features," in *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, vol. 2, 1999, pp. 1150–1157 vol.2.

[11] (2015) Cbcl pedestrian database no1. Center for Biological and Computational Learning at MIT and MIT. [Online]. Available: http://cbcl.mit.edu/software-datasets/PedestrianData.html

[12] T. Ojala, M. Pietikainen and D. Harwood, "Performance evaluation of texture measures with classification based on kullback discrimination of distributions," in *Pattern Recognition, 1994. Vol. 1 - Conference A: Computer Vision amp; Image Processing., Proceedings of the 12th IAPR International Conference on*, vol. 1, Oct 1994, pp. 582–585 vol.1.

[13] X. Wang, T. Han and S. Yan, "An hog-lbp human detector with partial occlusion handling," in *Computer Vision, 2009 IEEE 12th International Conference on*, Sept 2009, pp. 32–39.

[14] A. Haar, "Zur theorie der orthogonalen funktionensysteme," *Mathematische Annalen*, 1910.

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

[16] R. Lienhart and J. Maydt, "An extended set of haar-like features for rapid object detection," in *Image Processing. 2002. Proceedings. 2002 International Conference on*, vol. 1, 2002, pp. I–900–I–903 vol.1.

[17] (2015, 07). [Online]. Available: https://upload.wikimedia.org/wikipedia/commons/2/26/Precisionrecall.svg

[18] W. Xu-hui, S. Ping, C. Li and W. Ye, "A roc curve method for performance evaluation of support vector machine with optimization strategy," in *Computer Science-Technology and Applications, 2009. IFCSTA '09. International Forum on*, vol. 2, Dec 2009, pp. 117–120.

[19] (2015, 07) vision.cascadeobjectdetector system object. MathWorks. [Online]. Available: http://de.mathworks.com/help/vision/ref/vision.cascadeobjectdetector-class.html?

[20] (2015, 07) Train a cascade object detector. MathWorks. [Online]. Available: http://de.mathworks.com/help/vision/ug/train-a-cascade-object-detector.html?

[21] P. Viola and M. Jones, "Robust real-time object detection," in *International Journal of Computer Vision*, 2001.

[22] P. Dollar, C. Wojek, B. Schiele and P. Perona, "Pedestrian detection: An evaluation of the state of the art," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 34, no. 4, pp. 743–761, April 2012.

[23] R. E. Schapire, "The strength of weak learnability," in *Machine Learning*, 1990.

[24] A. Opelt and A. Pinz. (2004) Graz-02 database. Institute of Electrical Measurement and Measurement Signal Processing, Graz, University of Technology, Austria. [Online]. Available: http://www.emt.tugraz.at/~pinz/data/Graz_02/

[25] Makehuman. [Online]. Available: http://www.makehuman.org/

[26] Blender. [Online]. Available: https://www.blender.org/

[27] (2015, 07) Training image labeler. MathWorks. [Online]. Available: http://de.mathworks.com/help/vision/ref/trainingimagelabeler-app.html

[28] M. Everingham, L. Van Gool, C. Williams, J. Winn and A. Zisserman, "The pascal visual object classes (voc) challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2010. [Online]. Available: http://dx.doi.org/10.1007/s11263-009-0275-4

[29] H. Kruppa, M. Castrillon-Santana and B. Schiele, "Fast and robust face finding via local context," *Proceedings of the Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, 2003.

[30] A. Opelt and A. Pinz. (2003, 03) Graz-01 database. Institute of Electrical Measurement and Measurement Signal Processing, Graz, University of Technology, Austria. [Online]. Available: http://www.emt.tugraz.at/~pinz/data/GRAZ_01/