UNIVERSITÄT BASEL

Pose-Invariant Face Detection based on Trees of Wavelet Approximated Vector Machines

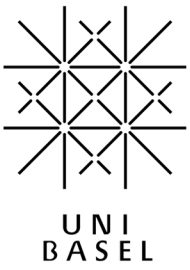
Master Thesis

Natural Science Faculty of the University of Basel
Departement of Computer Science
Graphics and Vision Research Group
http://gravis.cs.unibas.ch/

Supervisor: Prof. Dr. Thomas Vetter
Advisor: Dr. Matthias Rätsch

Michael Fischer

April 30, 2011



I hereby declare that I have created this work completely on my own and used no other sources or tools than the ones listed, and that I have marked any citations accordingly.

Hiermit versichere ich, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie Zitate kenntlich gemacht habe.

Basel, *April* 2011
Michael Fischer

Abstract

One of the main problems in face detection is the large variety of different faces. To handle faces with larger pose we introduce a new approach for pose-invariant face detection. The idea of this project is to divide the feature space into sub spaces for reducing the complexity for hypotheses. A serialization of one detector per sub space is used. These get trained onto different pose regions and can run in sequence. As detector we use the WVM (Wavelet Approximated Vector Machine) for its efficiency. Has the WVM a high probability, the object was found. To optimize the performance the detectors will be arranged in a hierarchical tree structure. A root node covers the union of the sub spaces. If the root node does not reject the patch as non-object, the leaves will be traversed. The leaves are trained for smaller pose regions (e.g. -90 to -40, -40 to 40, 40 to 90 degrees). The decisions of the leaves will be combined by fusion of the responses. With this approach a pose-invariant face detection can be obtained.

Acknowledgements

Firstly, I want to thank my advisor Matthias Rätsch for the wide support for this thesis. For many questions in discussion we found a satisfying solution together. Based on his previous dissertation "Wavelet Frame Accelerated Reduced Vector Machine for Efficient Image Analysis" I could realize my approach for finding a solution for pose-invariant face detection.

I would like to thank my supervisor Prof. Dr. Thomas Vetter for giving me the possibility and the chance to realize this thesis in his research group.

I would also like to thank the people of the Graphics and Vision Research Group for the pleasant atmosphere while doing my project. A "thank you" to Andreas Forster, Tatjana Frank, Patrik Huber and Adam Kortylewski for some discussions we had about face detection and topics of my thesis.

A very special thanks goes to my parents and my sister, for their endless patience and continuous encouragement during my life.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Face detection has become a more important topic in the recent years. Especially for security reasons based on events in the past, people try to build algorithms and methods for finding faces or objects in images, which are recorded e.g. by law enforcement systems. In face recognition systems face detection plays an important role. Before a face can be analyzed by a system, it has to be detected. An example for face recognition is the 3D Morphable Model (3DMM).
The 3D Morphable Model [1] is an approach for face analysis, modeling and synthesis and can also be used for face recognition across variations in pose and illumination. A model is learned from different 3D scans of heads. With statistical methods, the estimation will be achieved by fitting the 3D model of faces to images. The fitting takes several steps. For the initialization, special feature points like eyes, nose and mouth are needed, which have to be found in the images by e.g. particular detectors. Before, the face is to detect in the image. With the current used detector the detection of frontal faces is reliable, but in natural distributed environment about 1/3 of the faces occur with pose which do not pass the current used detector.
A face can assume many different poses in each direction of yaw, pitch and roll angle (Figure 1.1) which makes the face space very large. Consequentially, there exists a lack for the detection of faces with larger pose, that all facial images can be used for face modeling.

Figure 1.1: Example for pose images. Top row: Yaw angle; Second row: Pitch angle; Third row: Roll angle; Fourth row: Yaw and pitch combined.

An approach for pose-invariant face detection was published by Kim et al. [2] "Design and Fusion of Pose-Invariant Face-Identification Experts". They use reference faces in different poses for detecting the faces. In detail, the eyes are used for the correspondency of two faces. With huge pose in yaw angle, one eye disappears (Figure 1.1 fourth row) and the correspondency over the eyes can not be applied.
One possible approach is to split the face space into several sub spaces and to use an expert detector for every sub space. These experts can be run sequentially, which results in longer run-time. In 2006, Sahbi et al. [3] published "A Hierarchy of Support Vector Machines for Pattern Detection". They split the face space into several sub spaces. For every sub space they trained a Support Vector Machine (SVM) for detecting different faces and arranged them in a tree structure. With this approach they reached a slowly pose-invariant face detection for the roll angle (± 20 degrees).

An optimization of the SVM was introduced by Rätsch [4] "Wavelet Frame Accelerated Reduced Vector Machine for Efficient Image Analysis" in 2008. The SVM is approximated in several steps and a run-time performance increase up to a factor of 530 is achieved. We use the tree structure and the Wavelet Vector Machine (WVM) in this thesis.

The main contribution of this thesis is to reach a pose-invariant face detection. The novelty is to use the WVM as efficient classifier and the tree structure to divide the face space into smaller spaces. A significant improvement of the detection results of the Detector Tree is achieved by the developed fusion of responses to obtain the final decision. The procurement and evaluation of face databases is an additional achievement.

This thesis is organized as follows: A deep view over our approach and its background (data, training, and variants of the Detector Tree) are introduced in Chapter 2. Experiments of the single detectors are provided in Chapter 3. The final result is shown in Chapter 4. Some conclusions are drawn in Chapter 5 and finally we discuss further work in Chapter 6.

Chapter 2

Detector Tree

2.1 Divide and Conquer

As introduced in Chapter 1, for frontal faces (± 40 degrees) the current used detector detects the face frequently. The deficit exists for faces with pose larger than ± 40 degrees, special for the yaw angle. The hypothesis space for faces is very large, because of the different appearances in expression, in pose and individual differences. To solve such complex problems efficiently - or sometimes the only way to solve them - is to use the strategy of divide and conquer. Sort algorithms are some of the most popular approaches taking the advantage of the divide and conquer strategy. Sahbi [3] takes the approach to use it for solving the problem of the pose-invariant face detection for the roll angle. Well known is to solve the problem in sequence, but this would be too slow. Sahbi [3] introduced an improvement by the realization of divide and conquer using a tree representation. He focused on the coarse-to-fine processing, which is applied to many problems in computer vision like image registration [5], compression [6], reduction of noise or filtering. In Sahbi's approach [3], the tree is represented with a root node, levels and nodes per level. Every node of the tree is a trained Support Vector Machine [7]. For an image, a decision is made with a combination of the results of every node. In this way, they reached an efficient but slow pose-invariant face detection for roll angle smaller

than ± 30 degrees. We adopt the approach of the divide and conquer strategy for us. What many approaches have shown ([4], [3]) for a given small face space, the detection of faces, e.g. frontal faces, is reliable. We also use the tree structure as Sahbi [3] to divide the feature space. We split the pose, faces can assume, into multiple sub regions. The regions can either be independent or overlapping. Every region is represented in the tree by a node. Based on the information of the nodes of the previous level, the successional levels could be passed. It is not necessary to get only one path through the tree. Especially in the case of objects near the border of two neighboring nodes, high results of both nodes are expected. As novelty in this thesis the final decision to a given object is obtained by a fusion of the responses of the nodes in the different levels.

2.2 Detectors

We use the Wavelet Approximated Reduced Vector Machine (WVM) [4] as detectors for our tree because of its efficiency. The base of the WVM is a Support Vector Machine (SVM) which is approximated by a Reduced Set Vector Machine (RVM). One main advantage of the WVM approach is that we can adjust the complexity of each node from very sparse up till the full complexity of an SVM.

2.2.1 Support Vector Machine

The SVM is a popular classifier for separating different objects into different classes. The classifier is trained with the principal of maximizing the margin [8] with resulting in maximal generalization performance [8] for a given training data set and kernel [9]. Given a labeled train data set with data $x_i, x \in X$ and corresponding labels y_i. The SVM maps the data x_i into a dot product space F with a linear or nonlinear map (2.1)

$$\Phi : X \rightarrow F, x \rightarrow \Phi(x) \tag{2.1}$$

The dot product space has not to be the same as the working space X. It is shown by Mercer's Theorem that there

exist kernel functions $k(x, x')$ of a specific class for computing the dot product in an associated feature space, i.e. $k(x, x') = \langle \Phi(x), \Phi(x') \rangle$. The SVM hyperplane is defined as

$$\psi_{SVM} = \sum_{i=1}^{N_x} \alpha_i \Phi(x_i) \tag{2.2}$$

where α are the weights to the corresponding Support Set Vectors x_i (SSV's). The decision function is defined as

$$y(x) = sgn\left(\sum_{i=1}^{N_x} \alpha_i k(x, x') + b\right) \tag{2.3}$$

with $k(...)$ as the kernel function. An example for a kernel function is the RBF kernel $k(x, x') = exp\left(-\frac{\|x-x_i\|^2}{2\sigma^2}\right)$

The SVM is approximated with the RVM.

2.2.2 Reduced Support Vector Machine

The RVM is an approximation of the SVM. It uses a reduced set of Vectors ($N_z << N_x$) for getting nearly the same accuracy as the SVM (seen in Figure 2.1). The Reduced Set

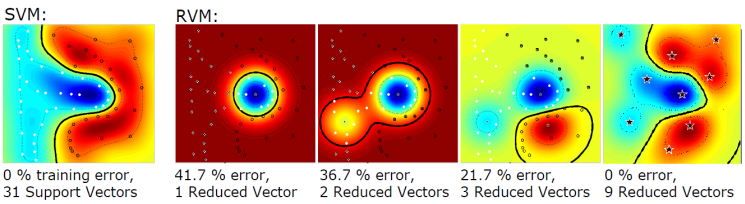


Figure 2.1: The result of the cascaded application of RSV's (stars) to a 2D classification problem (black and white dots), showing (left to right) the original SVM and the result of using 1, 2, and 9 Reduced Set Vectors. (source: [4])

Vectors (RSV's) are not part of the training set as the SSV's for the SVM. They will be generated randomly of the SSV's. The RVM hyperplane is defined as

$$\psi_{RVM} = \sum_{i=1}^{N_x} \beta_i \Phi(z_i) \tag{2.4}$$

where z_i are the Reduced Set Vectors (RSV's) and β the corresponding weights. The decision function is defined as

$$y(x) = sgn\left(\sum_{i=1}^{N_x} \beta_i k(x, x') + b_i\right) \tag{2.5}$$

further, the WVM is built out of the RVM.

2.2.3 Wavelet Approximated Reduced Vector Machine

The RVM is used as input for the WVM. Each RSV is approximated by several levels of Wavelet Approximated Reduced Set Vectors (W-RSV's). An example is shown in Figure 2.2. This has the advantage to reject specific patches

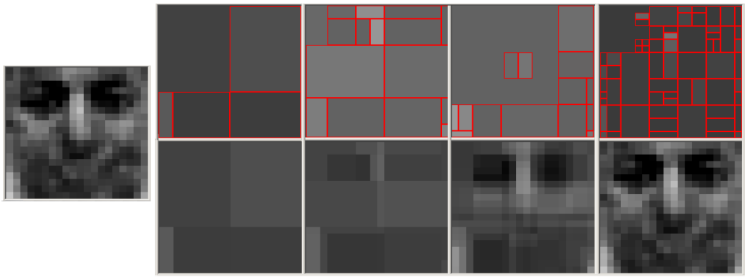


Figure 2.2: Left: a RSV; right, bottom row: W-RVS's at different resolution levels (left to right: level 0,1,9,18); top row: related wavelet approximated residuals (left to right: level: 0,1,9,18). (source: [4])

early, for example homogeneous regions. The kernel is evaluated efficiently using Integral Images. The advantage of Integral Images is in computing the sums in the kernel function. The computation of the sum of pixel values of an input image in an rectangle of any size can be made by four additions. For this reason, the convolution of a patch with one RSV can be decreased. Finally, to find the best representation of the W-RSV's, an Over-Complete Wavelet Transform [10] is applied during the training of the WVM. Summarizing, the key principals of the WVM are:

1. Support Vector Machine: An optimized SVM that is known to have optimal generalization capabilities.

2. Reduced Support Vector Machine: Which is an approximation of the SVM with much less RSV's with nearly the same accuracy as the SVM.

3. Double Cascade: Approximation of each RSV by W-RSV's for rejecting non-candidates.

4. Integral Image: Evaluation of the W-RSV's kernel.

5. Wavelet Frame: Find best representation by overcomplete wavelet system.

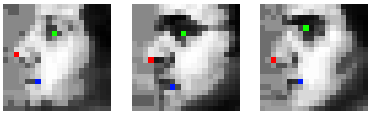
For further and more detailed information about the WVM we refer the reader to [4].

We use the WVM algorithm for building our detectors.

2.3 Training-/Validation Data

The use of "good" training- and validation/test data is essential for the training and the optimization of efficient detectors. The data has to fulfill the following properties:

1. Position of face: The features of a face (e.g. eyes, nose, mouth) shall always be at the same position in the patch. That means the eyes, nose and mouth of the same pose should optimally be found, for example in case of yaw angle of -90 degree, as seen in Figure 2.3. Consequential that most of the patch is filled with the face and not with background noise which should not be learned.



(a) (b) (c)

Figure 2.3: Position of facial features in training data. Red: Nose; Blue: Mouth; Green: eye.

2. Label for the pose of the face: The pose of the patch has to be known for assigning the patch to an angle range.

3. Variation of face: The patches have to have many variations in subjects, which are present, optional in light and expression.

2.3.1 Position of the face

A logic is used based on the coordinates of the facial features (eyes: outer corner, inner corner or center; nose: tip of the nose; mouth: either the inner center or the mouth corners) to achieve the first condition. The feature points have to be labeled. Next step is to define the face box $(x_{min}, y_{min}, x_{max}, y_{max})$ from top left to bottom right for cropping the faces. Given an example where the eyes' outer and inner corners and the mouth's corners are labeled. The centers of the eyes (2.6) and the mouth (2.7) are calculated first:

$$center_{reye/leye} = \frac{outercorner_{reye/leye} + innercorner_{reye/leye}}{2} \tag{2.6}$$

$$center_{mouth} = \frac{leftcorner_{mouth} + rightcorner_{mouth}}{2} \tag{2.7}$$

Second, the center between the eyes is calculated:

$$center_{eyes} = \frac{center_{leye} + center_{reye}}{2} \tag{2.8}$$

It is part of the virtual face center axis between the center of the eyes (green point) and the center of the mouth (blue point) as seen in Figure 2.4.

The height of the face box is set with a distance logic between the euclidean distance of $center_{reye}$ (2.6) to $center_{left}$ (2.6) or the euclidean distance of $center_{eyes}$ (2.8) to $center_{mouth}$ (2.7). The maximum of these two is taken and multiplied by two. With larger pose in yaw the distance between the eyes tendy towards zero $dist_{leye/reye} \rightarrow 0$ and the $dist_{leye/reye}$ carries authority (2.9).
For regularization, a parameter α is added for the case that

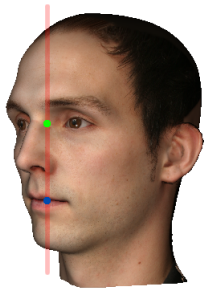


Figure 2.4: Shown in red: The face center axis for the given face. Green: Center of eyes; Blue: Center of mouth.

the distances between the centers are small, which results in a too small face box. This occurs for larger pitch angles.

$$height = max(\alpha, max(dist_{leye/reye}, dist_{eyes/mouth}) * 2) \tag{2.9}$$

α is tuned manually and can vary for different image sizes. x_{max}, y_{max} are set in dependency of $x_{max} = x_{min} + height$ and $y_{max} = y_{min} + height$. y_{min} is valued with $y_{min} = center_{eyes} - height/4$, resulting in a face box starting over the eyes.
If the pose is getting larger to +- 45 degrees yaw angle, the center of the face have to be displaced for the cropping as seen in Figure 2.6. The red line is the normal computed face center axis (between the $center_{eyes}$ (2.8) and the $center_{mouth}$ (2.7)) without displacement (as seen in Figure 2.6(a)) with the corresponding green box. This results in a patch with about 30% background noise, which should not be learned. For this reason the face center axis is displaced as seen in Figure 2.6(b). A displacement function is used in dependency of the pose for moving the axis (2.10) with a regularization parameter β tuned manually for different image sizes.

$$length_{xmin} = \left| \sin(pose) * (\frac{height}{\beta}) \right| - 90 < pose < 90 \tag{2.10}$$

The function is shown in Figure 2.3.1 for the values of -90 to 90 degrees, constant height of 118 and β of 4.2 which is mostly used in this thesis. The x axis describes the yaw angle in degrees, the y axis the displacement in pixels.

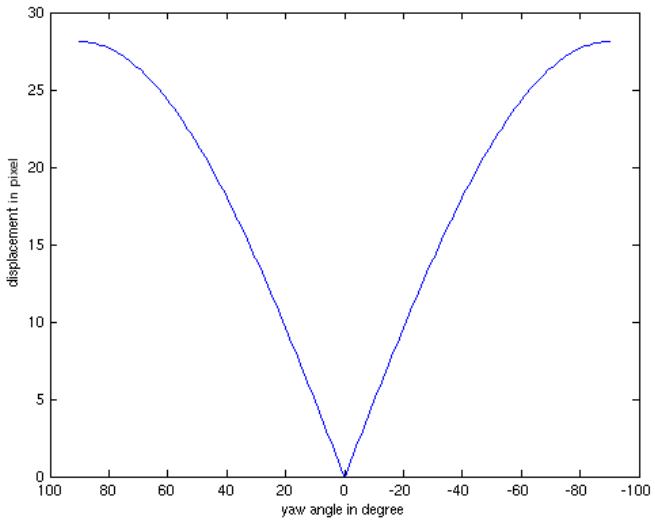


Figure 2.5: Function for displacement of the face center axis for pose of -90 to 90 degrees, height of 118 and β of 4.2.

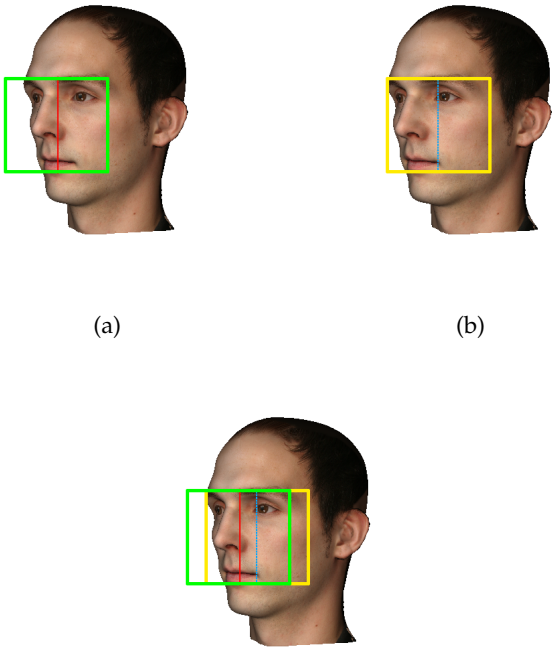
Finally, x_{min} is set as

$$xmin = \frac{x_{eyecenter} + x_{nosetip}}{2} \pm \frac{length_{xmin}}{2} \tag{2.11}$$

where the center of the eyecenters (2.8) and the tip of the nose are taken and x_{min} is displaced by the $length_{xmin}$ positive or negative, based on the sign of the pose. With this method we achieved a consistent cropping of the faces.

2.4 Data sets

Several different data sets are used in this thesis and build several sets. The sets are used for training, validation and measuring of the detection performance. It has to be differed between synthetic data and real world images. Both are used in this thesis because of their different advantages (e.g. accuracy in pose, variety).



(a) (b)

(c)

Figure 2.6: Displacement: (a): Face center axis and face box without displacement; (b): Face center axis and face box with displacement; (c): Both together.

2.4.1 Synthetic data

The synthetic data is generated by the Morphable Model Toolbox [1]. The 3D Morphable Model includes a fitting function, which generates from a 2D image of a person a 3D face shape. The vertices of the 3D shape are registered to the corresponded points in the 2D image. Further, from this 3D shapes 2D images are generated.
Several images are produced for different data sets. An advantage of the synthetic data is the precision in location and environment. For every image the pose, the light direction and type, and the background can be assigned. Also the exact location of the feature points in the image for cropping the faces with the method described in Section 2.3.1 are known.

2.4.2 Natural facial databases

Primarily, the following databases are used in this thesis:

1. FERET
2. CAS-PEAL
3. CMU Multi-PIE
4. LFW
5. PubFig
6. TRW
7. Head Pose
8. FacePix
9. Caltech

FERET

The FERET database [11] was published in 1993 and updated several times. It contains both, colored and gray scale images. The gray scale images are present in many different poses and recorded with one camera moving around the subjects, which they had to look into. It includes different subjects with many different accessories like glasses and beards. For this reason this database is used for training data, image- and test data.

CAS-PEAL

The CAS-Peal-R1 Face Database is a Chinese database and was published in 2004 and contains 30'900 gray scale images of 1040 individuals in six different positions of yaw, and three in pitch angles. These are recorded simultaneously with six cameras per pose of pitch angle. For the variety of subjects a sub set is used for the training of the detectors. The centers of the eyes are labeled. "The research

in this paper use[s] the CAS-PEAL-R1 face database collected under the sponsor of the Chinese National Hi-Tech Program and ISVISION Tech. Co. Ltd." [12].

CMU Multi-PIE

The CMU Multi-PIE database is the successor of the PIE database and published in 2009. It is a large database with more than 750'000 colored images of 337 people, recorded with 15 different cameras. This database is used in this thesis because of the accuracy of the angle. The people are recorded simultaneously, consequently there is no movement of the head between the recording. A disadvantage of the database are the missing labels of the facial features. We labeled parts of it manually.

LFW

Labeled Faces in the Wild (LFW) [13] is a database recorded from the internet using the Viola-Jones face detector. It contains more than 13'233 colored images of 5'749 individuals. This database is used for measuring the detection performance of the Detector Tree on images. The images show everyday situations, e.g. speaker on a conference or while having a conversation. They are used because of their realism.

PubFig

The Public Figures Face Database (PubFig) was published in 2009 by the University of Massachusetts Amherst (UMASS Amherst [14]). It contains 58'797 colored images of 200 famous people. They also show everyday situations and is an extension to the LFW with different subjects and is therefore used in this thesis.

TRW

The Real World (TRW) set is a face database created by University of Basel containing about 340 images. This database has a wide variation in image type (real taken poster to phantasy images) and size. Parts of it are used for evaluation of the Detector Tree.

Head Pose

The Head Pose Image Database [15] is a small database containing 2'790 colored images of 15 different subjects with 93 different poses between (± 90) yaw angle and (± 90) pitch angle. It is the only database in our sets with a high variation in the pitch angle. The database is used to show the limits to the pitch angle of the Detector Tree.

FacePix

The FacePix database was published in 2008 ([16], [17]) and is also a small database with 30 different people in 181 poses of -90 to 90 degree yaw angle. A camera was moving around the subjects and took one image per degree. So there is a high accuracy in the pose.

Caltech

The Caltech contains many different image types like faces, planes, leaves and one special set with non-face images. These images are used to generate negative patches for the detectors.

Detailed informations on the databases are shown in Table 2.4.4.

2.4.3 Negative patches

For an efficient detector it is essential to have good negative patches. To get a variability of different negative patches, the following method is used, e.g. for patches of the size $psize = 20 * 20$. First, an image img is chosen randomly of a set of images without a face. Second, a random position x_{min}, y_{min} inside the image is taken (2.12).

$$x, y_{min} = imlentgh_{x,y} * rnd \tag{2.12}$$

$imlength$ is the length of the image in x and y direction, rnd a random number between 0 and 1. This is the left upper start point of the face box. The lower right end point x_{max}, y_{max} will be chosen with the same formula (2.12) as left upper start point randomly. For final cropping some conditions have to be fulfilled:

1. $x_{min}, y_{min} < x_{max}, y_{max}$
2. $x_{max}, y_{max} \leq imlength_{x,y}$
3. $dist(x_{min}, y_{min}; x_{max}, y_{max}) > psize$

$dist$ is the euclidean distance. If one of these failes, new x_{max}, y_{max} will be chosen. If this conditions are not passed in several iterations, a new x_{min}, y_{min} will be taken until a valid solution is found and the cropping can take place. At the end, the cropped patch is resized to $spatch$. With this method we can produce patches in different scales for having a variation in the negative examples.

2.4.4 Histogram equalization

The technique of normalizing the data introduced by Huber [18] is used for getting robust data units to e.g. illumination.

Name	Source web	Source gravis
FERET	http://www.itl.nist.gov/iad/humanid/feret/	/export/facedb/feretR2/
CAS-Peal-R1	http://www.jdl.ac.cn/peal/	/export/facedb/CAS-PEAL-R1/
CMU Multi-PIE	http://www.multipie.org/	/export/facedb/multipie/
LFW	http://vis-www.cs.umass.edu/lfw/	/export/mauritania/lfw/
PubFig	http://www.cs.columbia.edu/CAVE/databases/pubfig/	/export/mauritania/PubFig/
TRW	not yet	/export/mauritania/TRW/
Head Pose	http://www-prima.inrialpes.fr/perso/Gourier/Faces/HPDatabase.html	/export/mauritania/HeadPose/
FacePix	http://www.facepix.org/	/export/facedb/FacePix/
Caltech	http://www.vision.caltech.edu/html-files/archive.html	/export/mauritania/caltech/

Table 2.1: Overview databases part I

Name	# of images	# of subjects	# labeled images	[a]feature labeled	[b]pose	colored/gray	comment
FERET	ca. 20'000	ca. 500	2846	le,re,nt,lm,rm	y:-90:90 p: 0 r: 0	yes/yes	labeled manually
CAS-Peal-R1	30'900	1040	30'900	le,re	y: -90:90 p: -30:30 r:0	no/yes	only Chinese
CMU Multi-PIE	ca. 750'000	337	1984	le,re,nt,mc	y:-90:90 p: 0 r: 0	yes/no	labeled manually
LFW	13'233	5749	0		y:-90:90 p: 0 r: 0	yes/no	web images, detected with viola jones
PubFig	58'797	200	58'797	fb	y:-90:90 p: 0 r: 0	yes/no	web images, detected with viola jones
TRW	340	about 300	0		y:-90:90 p: -60:60 r: -10:10	yes/yes	not official, pose not labeled
Head Pose	2790	15	2790	fb	y:-90:90 p: -90:90 r: -90:90	yes/no	
FacePix	5430	30	0		y:-90:90 p: 0 r: 0	yes/no	
Caltech	ca. 3000					yes/yes	many different objects

Table 2.2: Overview databases part II

[a]le = left eye; re = right eye (center or corners); nt = nose tip; lm = left mouth corner; rm = right mouth corner; mc = mouth center; fb = facebox

[b]y: yaw angle; p: pitch angle; r: roll angle

2.5 Training of the detectors

After generating data units, optimized detectors can be trained. This takes several steps.

2.5.1 SVM

Assuming the training data are labeled pairs $(x_i, y_i), i = 1, ..., N$ where $x_i \in \mathbb{R}$ and $y \in \{1, -1\}^N$. The SVM is the first stage which has to be trained. It requires the solution of the following optimization problem

$$\text{minimize} \quad W(\alpha) = -\sum_{i=1}^{N} \alpha_i + \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j k(x_i, x_j) \tag{2.13}$$

$$\text{subject to} \quad \sum_{i=1}^{N} y_i \alpha_i = 0 \qquad \forall i : 0 \leq \alpha_i \leq C \tag{2.14}$$

with N being the number of training examples. w is a vector with N variables, containing for every component α_i a corresponding training pair (x_i, y_i). With minimizing (2.13) the result is a vector w^* with respect to the constraints (2.14). By defining the matrix Q as $Q_{i,j} = y_i y_j k(x_i, x_j)$ the optimization problem (2.13) can be written as vector form:

$$\text{minimize } W(\alpha) = -\alpha^T 1 + \frac{1}{2}\alpha^T Q \alpha \tag{2.15}$$

$$\begin{aligned} &\text{subject to } a^T y = 0 \\ &0 \leq \alpha \leq C_1 \end{aligned} \tag{2.16}$$

This results in a hyperplane $\psi_{SVM} = \sum_{i=1}^{N} \alpha_i \Phi(s_i)$ (s are the SSV's) with maximal generalization performance [7] and its following decision function $y(x) = sgn\left(\sum_{i=1}^{N} \alpha_i k(x, x') + b\right)$.
Further information can be found in [19, 9].
A few criteria are used for finding the optimal kernel parameter g for the rbf kernel (Section 2.2.1) and regularization parameter C. The evaluation happens on a test set $(x_i, y_i), i = 1, ..., N$ where $x_i \in \mathbb{R}$ and $y \in \{1, -1\}^N$. The parameters are optimized for the given test set for getting the optimal classification results. The criteria are:

1. FAR
2. EER
3. ROC

False Accepted Rate

The False Accepted Rate (FAR) is an indicator for the detection performance of a machine. Given two or more machines $m_1, m_2, ..., m_n$ with different g_n, C_n and thresholds b_n. By adapting the threshold b_n to a specific FAR (2.17), e.g. 0.01 %, the True Accepted Rate (TAR) (2.17) is an indicator for the detection performance of a machine to a given

test set.

$$FAR = \frac{\text{\# of False Alarms (FA)}}{\text{total \# of negative objects}}$$

False Alarms (FA): negatives objects, classified as positives

$$FRR = \frac{\text{\# of False Rejections (FR)}}{\text{total \# of positive objects}}$$

False Rejections (FR): positive objects, classified as negatives

$$TAR = \frac{\text{\# of True Alarms (TA)}}{\text{total \# of positive objects}}$$

True Alarms (TA): positive objects, classified as positives

$$TRR = \frac{\text{\# of True Rejections (TR)}}{\text{total \# of negative objects}}$$

True Rejections (TR): negative objects, classified as negatives (2.17)

With higher TAR the detection performance on a given test set increases. So $maximum(TAR_1, TAR_2, ..., TAR_n)$ characterizes the best machine. With this criterion a first coarse decision for machines can be made, resulting in these having the same TAR by given FAR. One problem of this criterion is the missing global information of the behavior of the rates. The next criterion gives a further decision for the remaining machines.

Equal Error Rate

The Equal Error Rate (EER) is more general for comparing machines. The EER is reached if $FAR == FRR$. The EER is visualized by a schematic drawing (Figure 2.7). The x axis is the threshold b from 0 to a max value depending on its range and y axis the error in %. The FAR and the FRR are written as function of b. The EER is the intersection between FAR and FRR. The lower the EER, the better is the detection performance of a machine. The main problem of finding the EER is given by the test set. In case of having large non-symmetric data (ratio between object classes is very high, e.g. 1'000 $class_1$, 100'000 $class_2$) small changes in the threshold b have large effect on the error of the percentage of the larger class, because of their closest examples to the hyperplane. In this case finding the optimal threshold b manually where $FAR == FRR$ can take a lot of time.

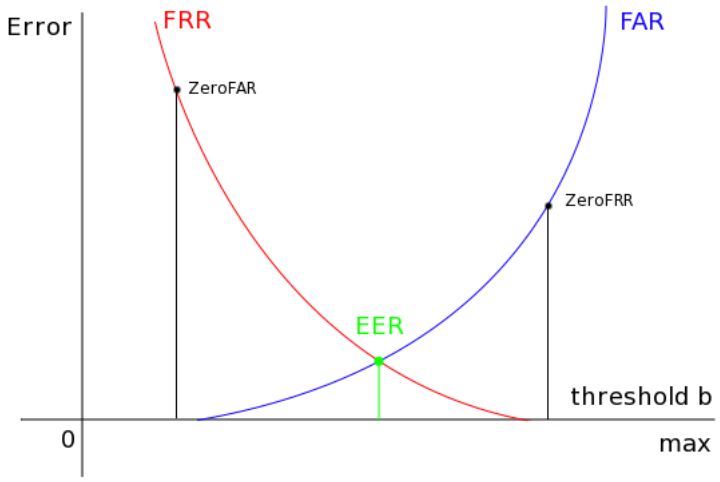


Figure 2.7: Equal Error Rate (EER): Intersection between FAR and FRR (green point).

Also the global information about the behavior of the rates is missing. A solution is the ROC curve.

ROC

The Receiver Operating Characteristic (ROC) curve is one established criterion for measuring the detection performance for a detector. The fundamental is to variegate the threshold b in a defined range. In case of a detector, the range can be set as e.g. $20\% \leq FAR \leq 95\%$. By variegation of the threshold the output is a curve. This ROC is a measurement for the quality of a detector and is built to a given test set, an example is shown in Chapter 3. The x axis describes the Detection Rate (DR), the y axis the FAR. The DR is defined as $DR = (1 - FRR)$. In observation of all rates given a specific threshold b, the EER (introduced above) is a point on the curve. The curve shows a continuing expansion of the relation of the FAR to DR. Given a test set, it is a direct comparison of the detection performance of different machines. A disadvantage is the computation of the ROC. With large test sets ($> 100'000$ examples) the classification can hold up several minutes to hours (in respect to the size of the detector with e.g. 20'000 SSV's, also depending on

the implementation and the hardware). Consequently, the very coarse evaluation of detectors shall take place with the equal FAR. The next finer distinction is the EER and finally the ROC for showing the global detection performance.

2.5.2 RVM and WVM

For the RVM a number of RSV's is chosen manually (used in this thesis is a number of 100). The RVM is optimized based on a ROC for the given test set. Further the WVM approximates the RSV's. The number of operations used for rejecting the non-face patches $\frac{\#\text{ of operations}}{\#\text{ of rejections}}$ of the RVM and WVM have to be minimized. This criterion is a measurement for the quality of RVM and WVM and is built to a given test set, an example is shown in Chapter 3. More detailed informations about the ROC are shown in [4].

2.6 Detector Tree based on face detection

The face detection application is implemented in C++ and Matlab by [4]. The Detector Tree is integrated in this application. First off, an overview about the stages of an WVM detector:

1. WVM cascade over an image
2. First overlap-elimination
3. Full SVM for final decision over remaining candidates
4. Second overlap-elimination
5. Final decision

For simplification, in the following WVM is known as the cascade stage of the WVM detector. The execution flow of the WVM algorithm acts as follows:
A sliding window with the size of the detector (dimension of the training patches, e.g. 20x20 pixels) is moved over

the image. The image is rescaled several times by a specific scaling factor $sfac$ chosen manually to detect faces in different sizes. In this way an image pyramid is built. For every level of the pyramid the sliding window procedure takes places. Every patch is classified by the WVM with a probability whether a face is present. This results in an amount of remaining candidate patches having a high probability. The next step is the overlap-elimination, where the candidates are compared to their location and clusters are built out of them for minimizing the remaining candidates and finding the best location for a possible face. Next stage is to classify these patches with the full SVM (probabilistic SVM, see [4] Section 3.4), which gives a final probability for a given patch. Further step is to do the second overlap-elimination for the patches classified by the full SVM as face. The highest result is chosen (in case of several faces in an image, several results are chosen according to their probability) for being the patch with the face. The WVM algorithm is used for the Detector Tree.
The Detector Tree consists of two layers l_1, l_2, one root node r and three leave nodes n_1, n_2, n_3 as seen in Figure 2.8. The

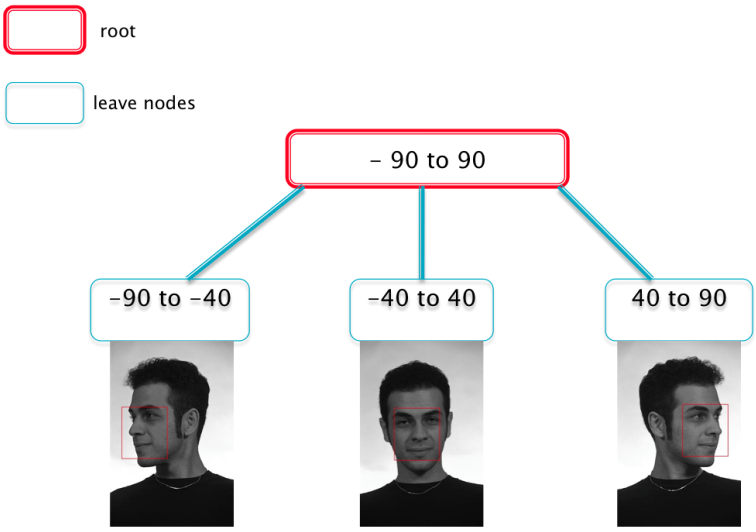


Figure 2.8: Representation of the Detector Tree with one root node and three leave nodes.

whole range of yaw angle is divided into three regions. First, -90 to -40 (in the following known as left), second -40 to 40 (hereafter known as middle) and third 40 to 90

(in the following known as right) degrees. Every leave caps one of these regions. On top is the root node which covers the whole region of -90 to $+90$ degree yaw angle. Pitch and roll variegate between -15 and $+15$ degree in root and leaves.
Projected to the execution flow, the Detector Tree covers the stages seen in Figure 2.9. As seen in Figure 2.9 the root

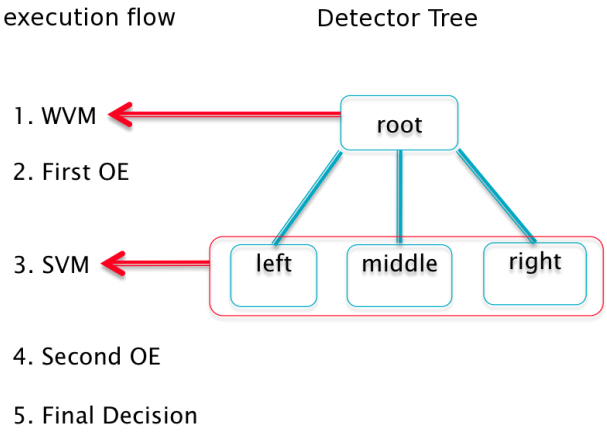


Figure 2.9: Projection of the Detection Tree to the execution flow.

node consists of an WVM (or several seen further) with the assignment to reduce the candidates for the leaves. The leaves consist of separated SVM's which evaluate the probability for a face for every patch passed by the root node. To prevent the SVM's of computing a decision for every patch in the leaves, it is important to find good candidates with less computations in the root. The root can be represented in several variants.

2.7 Root node

There are several approaches to defining the root node. The first and most intuitive approach is to generate WVM's out of the SVM's in the leaves. (Figure 2.10).

2.7.1 Root I: WVM's of the leaves

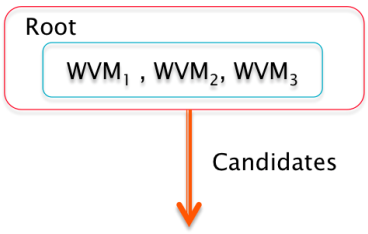


Figure 2.10: First representation of the root node with the separated WVM's of the leaves.

The patches become classified by the WVM's of the leaves. Every patch runs through every WVM. If one WVM gives a high response for a face, the patch gets inserted to the possible candidates. In our case it is not important which of the detectors gives the highest response, because all of the remaining patches pass every SVM in the leaves. We expect the best candidates with this approach, because of the exact feature space for every detector. The worst run-time performance is also expected because three WVM's evaluates every patch. Assuming all detectors have the same number of cascades, the complexity would be n^3 instead of n. The reduction of the complexity can be done over reducing the number of WVM's.

2.7.2 Root II: Merge of input spaces

Instead of using the leave SVM's for generating three WVM's for the root, an alternative is to merge their input spaces and train a new large classifier. As we said in Section 2.1, for a final decision the hypothesis space for one classifier is too large. But for pre-processing the patches and to find good candidates the idea of one classifier is promising, regarding the complexity of this classifier is low. This "super detector" has to be reliable and have a good detection performance not to lose any face candidates which would not pass the SVM. There is the problem with the trade off between not losing faces for getting the optimal detection performance and rejecting many non-face candidates for lowering the computations in the leaves. With the WVM as

classifier we can influence the strictness of rejections with some preferences. Empirically, too many candidates pass to the leaves. Further approach is to reach optimal run-time performance with optimal detection quality.

2.7.3 Root III: Combination

For achieving best quality on the one hand and best run-time performance on the other hand one possibility is to combine the two approaches introduced above in a tree structure as seen in Figure 2.11. The WVM_{super} is a trained

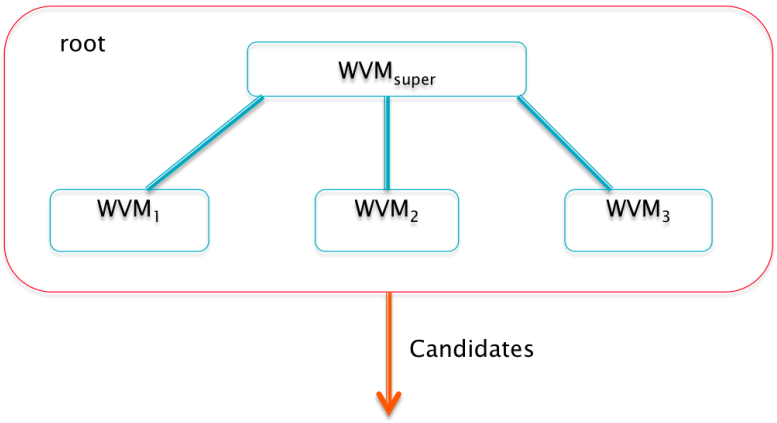


Figure 2.11: Combination of a single detecter as pre classifier and following sub classifiers.

classifier with the input space of SVM_1, SVM_2, SVM_3. The top node makes a first decision d_r. Based on this information further steps will be defined. If $d_r > t$ with t as threshold for a face (e.g. 98 %), the patch become passed to WVM_1, WVM_2, WVM_3. In other words, if the probability for patch to be seen a non-face is high, the patch is rejected. If the super node is not sure about the result for having a face, it also passes the patch to WVM_1, WVM_2, WVM_3, which make a final decision for inserting the patch to the possible candidates. In this way the loss of faces of Root II can be minimized and the run-time performance of Root I can be optimized by rejecting sure non-face patches in a first decision d_1 like background patches. It is not truly an additional layer in the Detector Tree, because the decision

of the root node has no direct impact to the later decision in the leaves. So the leaves do not know which classifier in the root node made the decision for including a patch into the possible candidates.

2.8 The leave nodes

After the root node made its decision for every patch, the following task is to make final decisions about the candidates.

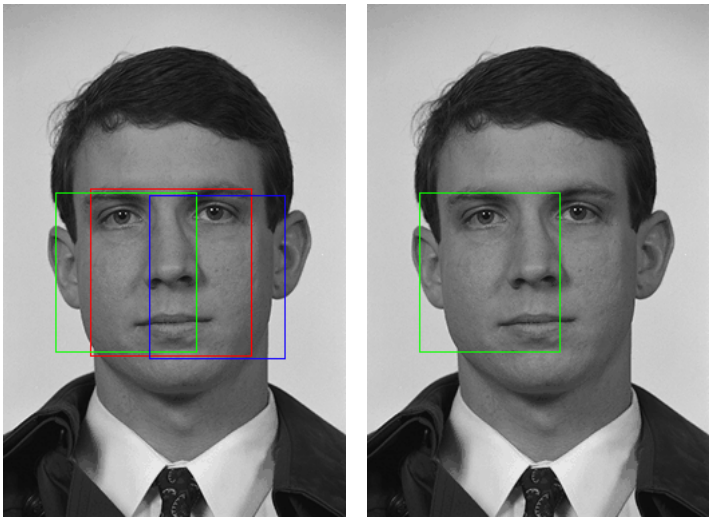
2.8.1 Simple maximizing

The leave nodes take the candidates as input for making a decision. Every decision per detector d_1, d_2, d_3 is independent of the others and has its probabilities p_1, p_2, p_3. One fast moving method for getting a final decision for a patch is to take the maximum of the responses $max(p_1, p_2, p_3)$. It is to expect, that every detector d_1, d_2, d_3 has a high response in a patch, which belongs to its sub space, given the FA's go to zero $FA \rightarrow 0$. In practice it is the case, that if the probability of detector 1 is high (e.g. $p_1 > 95\%$) for a given patch of its sub space, the probability of the other detectors ($p_{2,3} < p_1$) are lower on that patch. Consequently, for a single patch the divide and conquer principle is fulfilled.
A problem can occur when expanding the decisions to an amount of patches.

2.8.2 Overlapping decisions

With the sliding window procedures the decisions are made based on local information in the patch with no global knowledge. This can result in a true decision in false context. In Figure 2.12 such a case is shown.

The red box is the response of the middle detector. The green of the right and the blue of the

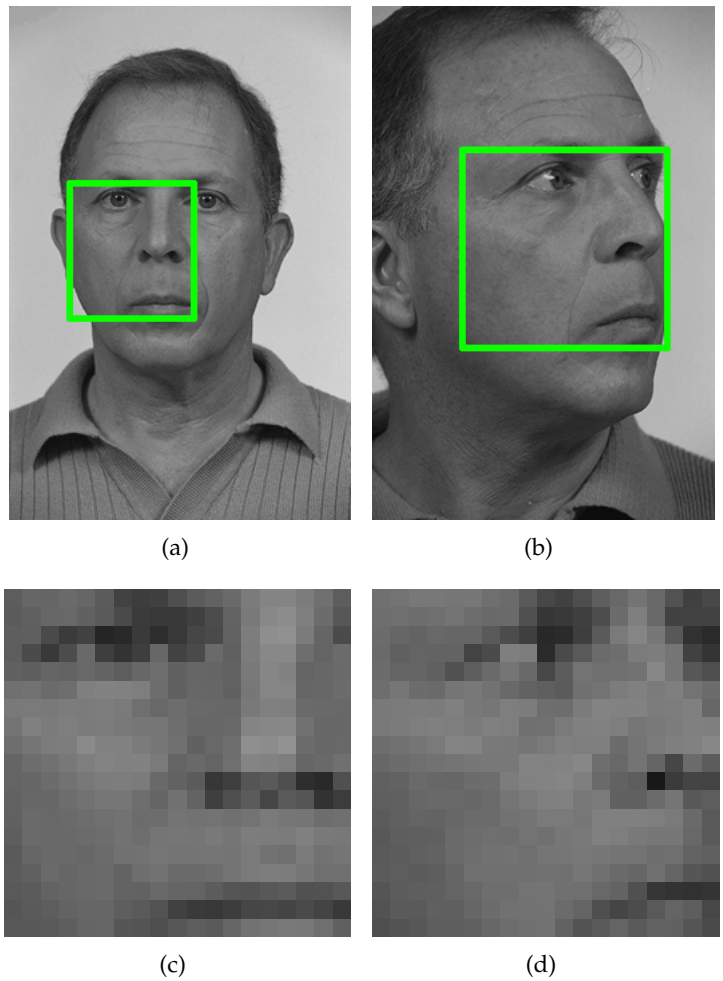


(a) (b)

Figure 2.12: Left: Response of all detectors (Red: Middle; Green: Right; Blue: Left). Right: Final decision based on max principal.

left detector. If we express this in probabilities, $p_m = 0.999856, p_r = 1.0, p_l = 0.999329$. It is seen that the right has the highest response of these. So the final decision will be made based on this information resulting in a wrong face box 2.12(b). A possible explanation is seen in Figure 2.13. Assuming top left 2.13(a) is the image in which the face shall be detected. On the top right is the original image of which the training data is extracted. On bottom right is the rescaled training patch, similar to the support vectors of the right detector. In the bottom left Figure is the rescaled patch of the green box of 2.13(a). It is seen that the two patches (2.13(c), 2.13(d)) are very close in intensity per pixel. That means that the right detector has a high response in 2.13(c) which is, on his local view, a correct decision.

With this explanation, all detector responses (Figure 2.12(a)) are comprehensible. As novelty to known approach using trees we developed in this thesis a method for dealing with this problem to get a correct facebox.



(a) (b)

(c) (d)

Figure 2.13: (a): To detecting image; (b): Example training patch; (c): Resized patch of the green box of (a); (d): Resized patch of the green box in (b).

2.8.3 Fusion of responses

Given the responses of the detectors middle, right, left and the corresponding probabilities $p_m = 0.999856, p_r = 1.0, p_l = 0.999329$ as seen Figure 2.12(a). Every patch has a position $pos(x, y)$, a scale s, in which level of the image pyramid the patch is located, and a scaling factor $sfac$, which represents the down scaling level of the input image. Experiments show that the problem occurs for frontal faces. So a condition for the method is $p_m > z$ with z is the threshold tuned manually for a given image set (e.g.: $0.95 \hat{=} 95\%$ having a face). If this condition holds, the percentage of every $p_{m,r,l}$ to the sum over the responses (2.18) is calculated.

$$pA_n = \frac{p_n}{\sum_{i=1}^{n} p_i} \tag{2.18}$$

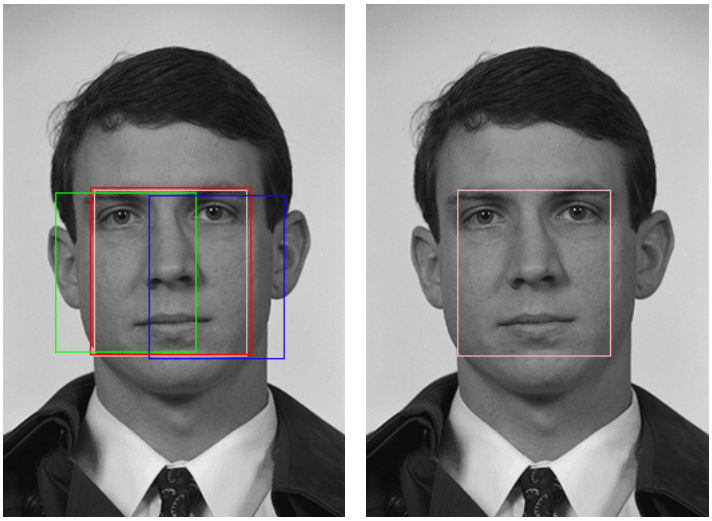
where n is the number of decisions, in our case three. With this percentage for every $p_{m,r,l}$ a new position $pos_{new}(x, y)$ and a new scale s_{new} for the new patch will be calculated:

$$pos_{new}(x) = \sum_{i=1}^{n} pA_i * x_i \tag{2.19}$$

$$pos_{new}(y) = \sum_{i=1}^{n} pA_i * y_i \tag{2.20}$$

$$s_{new} = \sum_{i=1}^{n} pA_i * s_i \tag{2.21}$$

The resulting patch is the final aggregated and improved decision seen in Figure 2.14. We imply that no FA's are present and every response of the detector is optimal for its sub space. An improvement to prevent outliners can be made by weighting each detector result focused to the overlapping area between them. First experiments show promising results. More details are in Chapter 6.



(a) (b)

Figure 2.14: Left: response of all detectors (Red: Middle; Green: Right; Blue: Left; Purple: Fusion); Right: Final decision of fusion.

Chapter 3

Outcome

We produced several sets for the detectors and the Detector Tree. We differ between training sets to train our detectors, test sets to optimize our detectors and image sets for measuring the detection performance on images.
We use two sets of synthetic data for training and testing. One set with a pose range of -90 to 90 degrees yaw angle, 0 degree pitch and 0 degree roll angle, containing 88'672 images of about 400 different subjects. We also produced a test set of 12'328 images with these characteristics. The second set for training contains 30'762 images of the range of -90 to 90 degrees yaw angle, -10 to 10 degrees pitch and 0 degree roll angle. Another test set was produced with 4'077 images with these characteristics.
Out of the FERET database, we took 906 profil images (± 90) of 453 different subects which we labeled manually for our training set. One image set has 1940 images of 194 different subjects (in -60, -40, -25, -15, 0, 15, 25, 40, 60 yaw angle with two images with 0 degree yaw angle but different illumination). All images with -60, -40, 40 and 60 degrees yaw angle are separated and form a part of the test set (776 images). Some other images of the FERET database are also used for the image sets.
The CAS PEAL set includes 2280 images in pose of -67.5, -45, 45 and 67.5 degrees yaw angle. We put these in our training set.
We extracted about 0.5% out of 13'233 LFW images with -90 to 40 and 40 to 90 degrees yaw angle independently of

their pitch and roll angle for the image sets. Also for our training set we labeled 1984 images of Multi Pie manually with angle range of -90, -75, -60, -40, 40, 60, 75, 90 degrees yaw angle in natural illumination.
To get more training data, all training examples with pose ≤ 40 are mirrored and put to the training set of the left detector.
PubFig, TRW are mixed into some image sets for variability.
Headpose database forms an own image set with 186 images with maximal angles (± 90 degrees pitch and yaw, 0 degree roll) for measuring the limits of the detectors and the Detector Tree. With the Caltech database we produced 400'000 negative patches with the method introduced in Section 2.4.3. For optimizing the prevention of FA's we put the positive examples of the right detector to the negative examples of the left detector.

Because of the symmetry of faces, the left or right detector can be trained and mirrored (W-RSV's, SSV's). The middle detector is inherited from a former project and is trained with the "big" training- and test set, containing:

Name	#pos	#neg
vir-5-train.mat	20'504	13'504
vir-2-fp-test.mat	5'003	23'077
good_web6.m	4'911	5'604
f-nf-train.mat	3'194	20'275
f-nf-test.mat	828	5'375
bdisc-20x20-testset.mat	7'742	100'000
bdisc-20x20-rbfb2-svs-noBound-8291.mat	3'379	4'912
bdisc-20x20-rbfb1-svs-noBound-4359.mat	1'611	2'748
bdisc-20x20-rbf2-svs-noBound-2467.mat	1'266	1'201
patches_ind_nonface.mat	0	93'192

Table 3.1: Data sets: Middle detector

For more details we refer the reader to [18]. We produced the following sets for our detectors:

Training set left

Name	#pos	#neg	set file name	[a]comment
Synthetic data	36'021	0	synth-40.mat	
FERET	906	0	feretplpr.mat	pl with mirrored pr
CAS PEAL	2'280	0	caspeal-40.mat	with mirrored +40
CMU Multi Pie	1'984	0	mp-40.mat	with mirrored +40
Caltech	0	130'001	negativePatches.mat	

Test set left

Name	#pos	#neg	set file name	comment
Synthetic data	4'936	0	synthtest-40.mat	
FERET	776	0	feret-40.mat	with mirrored +40
Caltech	0	30'001	negativePatches.mat	

Training set root

Name	#pos	#neg	set file name	comment
Synthetic data	119'434	0	synthtest-40.mat	
FERET	906	0	feretplpr.mat	pl with mirrored pr
CAS PEAL	14'546	0	caspeal900000.mat	with mirrored
CMU Multi Pie	3'968	0	multipie.mat	with mirrored
Caltech	0	400'000	negativePatches.mat	
Big Set	159'968	34'961	big-train.mat	

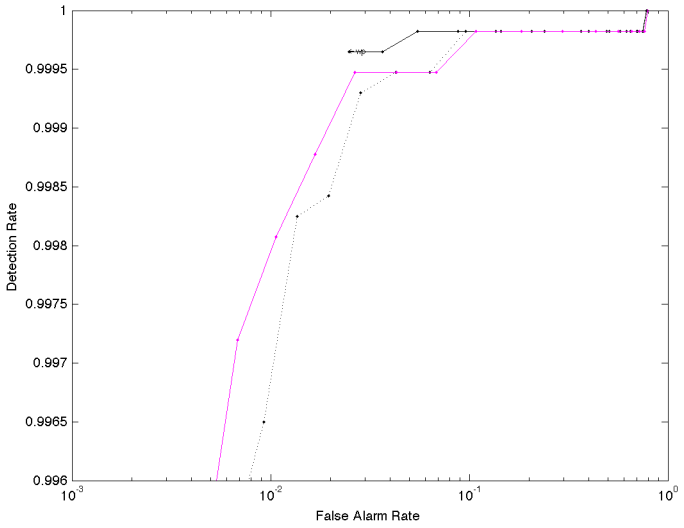
Test set root

Name	#pos	#neg	set file name	comment
Synthetic data	16'405	0	synthtest-40.mat	
FERET	776	0	feret-40.mat	with mirrored +40
TS107742	7'742	100'000	ts107742.mat	with mirrored
Caltech	0	50'000	negativePatches.mat	

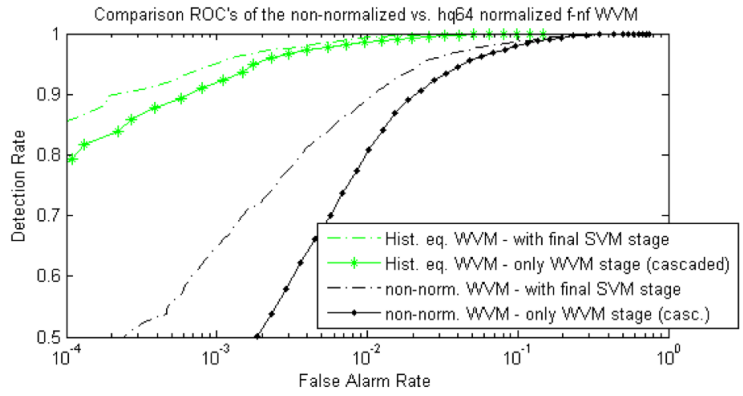
Table 3.2: Data sets: Left and root detectors.

[a]pl = profile left (-90 degree); pr = profile right (90 degree); -40 = patches with pose < -40; +40 = patches with pose $> +40$; mirrored = patches are mirrored for extend the positive examples

All the detectors are trained with SVMlight [19] and optimized with the criteria introduced in Section 2.5.1. We achieved the detection performance for the left SVM on the test set shown in Figure 3.1. X axis describes the FAR, Y



(a)



(b)

Figure 3.1: 3.1(a): SVM of left in black, RVM of left in dotted black, WVM of left in violet; 3.1(b): SVM of middle in dotted green (source [18]).

axis the DR. As seen in Figure 3.1(a) (black curve) the detection rate of left SVM does not go below 99.97 % on the test set. This set is easy to classify for the left detector. An improvement can be done by expanding the test set. Af-

ter optimizing the SVM, the SSV's are approximated with the RVM. For the RVM training the implementation of [4] was used. The ROC of the RVM is shown in Figure 3.1(a) (black dotted curve). Finally, the WVM was built with the implementation given by [4]. We reached a detection rate of over 99.8% with a FAR of about 1% 3.1(a) (violet curve) and with about 300 operations we rejected 100% of the non-faces 3.2 (violet curve). In 3.1(b) the black curves describes a non-normalized WVM and the green curves a normalized WVM of the middle detector with histogram equalization introduced by Huber [18]. As seen in Figure 3.1(b) the normalized WVM is more efficient than the non-normalized. For this reason we also use the normalization in our approach.

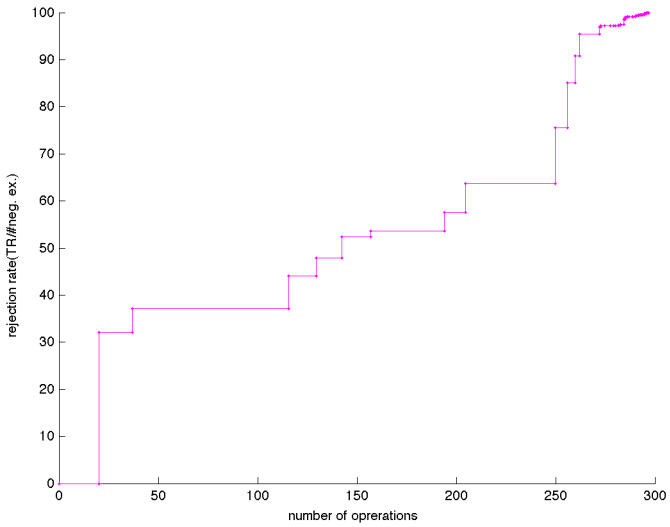


Figure 3.2: Number of used operation per rejection of the WVM left.

3.1 Results of the trained detectors

The detection performance of the trained detectors is shown first on some example images and later verified accordingly on an amount of image sets. In Figure 3.3 the top row shows the detection result of the right detector on three images. Below are the results of the middle detector on the given images. In the third row are images detected by the left detector and, for comparison, the bottom row with the corresponding results of the middle detector. The red box symbolizes a high probability for a detected face on the given patch, further, the black box symbolizes a low probability for a detected face on the given patch. It is seen that the detection of the middle detector fails, but the left and right detectors detect the faces with pose.

3.2 Results on sets of images

To evaluate the detection performance we built several image sets. A set (”set-left”) with images only in range of -40 to -90 degrees, a second set (”set-middle”) with images containing only frontal images between -20 and 20 degrees and a third set (”set-right”) with images of 40 to 90 degrees. The images are taken from LFW, TRW, Pubfig, Multi Pie and FERET for getting optimal results under natural conditions and having a high variation in subject and environment. The resolution variegates between 180x180 and 1160x120 pixels for guaranteeing faces in different sizes. The images are chosen randomly in respect to their pose. ”set-left” contains 111, ”set-middle” 103 and ”set-right” 129 images. To compare the results with each other, the configuration parameters of the WVM and SVM are not tuned for one of these sets. In all performance measurements the down scaling factor of the image is 0.85, the number of levels uses for the pyramid is 18 and the minimum of the face size is 90x90 pixels. In Table 3.3 the detection performance of every detector on every set is shown. #pos is the number of truly detected faces. This implies that the facial features have to be inside the face box, the size of the face box must not differ too far of the ground truth and the face has to be found

Figure 3.3: Top row: Result of right detector; Second row: Result of middle detector; Third row: Result of left detector; Bottom row: Result of middle detector.

nearly in the middle of the box. #neg is the number of not or false detected faces, which are faces, where a feature is outside of the face box (e.g. eye, tip of nose), the size is not acceptable or the face is not in the middle of the face box. Some examples for not accepted faces are shown in Figure 3.4. Seen in Table 3.3 every detector has a high DR of over 90% on its region. Higher DR's can be achieved by tuning the configuration parameters. In the next Section the detection performance of the detectors combined in the Detector Tree is shown.

Figure 3.4: Three not accepted faces. Left: Eye is not included; Middle: Position of face and face box to large; Right: Face box to large.

"left-set"

Detector	#pos	#neg	Detection Rate
left	106	5	95.50%
middle	27	84	24.32%
right	24	87	21.62%

"middle-set"

Detector	#pos	#neg	Detection Rate
left	36	67	34.95%
middle	99	4	96.12%
right	27	76	26.21%

"right-set"

Detector	#pos	#neg	Detection Rate
left	16	113	12.40%
middle	34	95	26.36%
right	120	9	93.02%

Table 3.3: Results on sets of images

Chapter 4

Results

For measuring the detection performance of the Detector Tree, random images of the data sets introduced in Section 3.2 are merged to one image set ”all-set” with 140 images. As said above, it contains images with a high variation in sizes, subjects and environments. The configuration parameters used for WVM and SVM are a down scaling factor of the image of 0.85, the number of levels for the pyramid of 18 and the minimum of the face size of 90x90 pixels. Each single detector as well as the Detector Tree are measured on the ”all-set”. In Figure 4.1 the result is shown. The x axis describes the detection rate in %. We

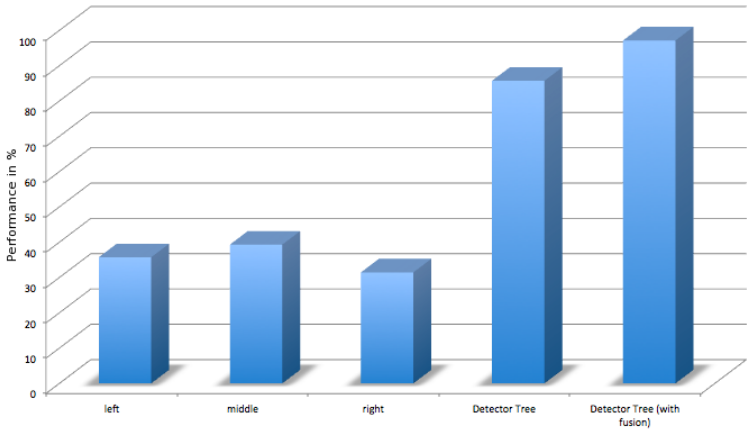


Figure 4.1: The single detectors in comparison with the Detector Tree without and with fusion of responses on the ”all-set”. Increase of the detection performance of over 46 %.

could improve the detection rate of ”all-set” from about 39% (old middle detector) to over 97% on images with a high variation in pose (Table 4.1).

	”all-set”		
Detector	#pos	#neg	Detection Rate
left	50	90	35.71%
middle	55	85	39.29%
right	44	96	31.43%
Detector Tree	120	20	85.71%
Detector Tree (with fusion)	136	4	97.14%

Table 4.1: Results of the detectors and the Detector Tree on the ”all-set”

To test the limits of the Detector Tree we used an image set with one subject of the Head Pose Database with 186 images of -90 to +90 degrees variation in pitch and yaw angle and 13 images per pitch range. The set is split into two parts, one with glasses, the other without glasses. In Table 4.2 the detection results are shown. It is seen that the detection results for glasses are worth than without glasses. The reason is the minority of faces with glasses in the training set. A few examples for accepted and non-accepted faces are shown in Figure 4.2. The Detector Tree is trained for a range of pitch angles of ± 15 degrees and is for the region of ± 30 degrees robust. With the Detector Tree we achieved a robust pose-invariant detection algorithm.

Additional to comparing the different variants of root node introduced in Section 2.7 we did experiments on a sub set: ”left-set”, ”middle-set” and ”right-set”. The sub set includes four images of each set and every image is chosen from different databases for getting a high variability. The aim is to minimize the number of candidates which reach the leaves. The run-time and the number of candidates are measured after the first overlap-elimination (first OE), where the candidates for the leaves are present and non-object patches are rejected. The average is calculated out of all detections and the result shown in Table 4.3. All variants for the root node detected the set correct. The single root produced the most candidates but with nearly half of the run-time costs of the other approaches. In comparison, there is only a small difference of number

"with glasses"

Pitch angle	#pos	#neg	Detection Rate
-90	0	1	0%
-60	1	12	7.69%
-30	10	3	76.92%
-15	13	0	100%
0	10	3	76.92%
15	10	3	76.92%
30	11	2	84.62%
60	1	12	7.69%
90	0	1	0%

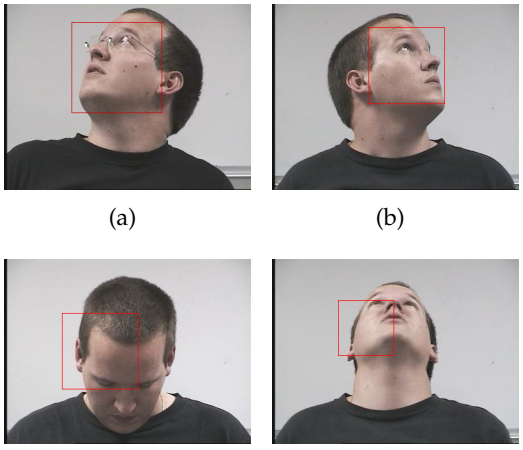
"without glasses"

Pitch angle	#pos	#neg	Detection Rate
-90	0	1	0%
-60	2	11	15.38%
-30	9	4	69.23%
-15	13	0	100%
0	13	0	100%
15	13	0	100%
30	13	0	100%
60	12	1	92.31%
90	0	1	0%

"both combined"

Pitch angle	#pos	#neg	Detection Rate
-90	0	1	0%
-60	3	23	11.54%
-30	19	7	73.08%
-15	26	0	100%
0	23	3	88.46%
15	23	3	88.46%
30	24	2	92.31%
60	13	13	50.00%
90	0	1	0%

Table 4.2: Limits of the Detector Tree in focus to the pitch angle



(a) (b)

(c) (d)

Figure 4.2: Four examples for accepted(a)(b) and not accepted detections(c)(d).

of candidates between the single leaves and the combined approach. Seen in Table 4.3 ”only one root WVM”, many candidates are passed to the leaves, so it is to expect that in the combined approach many candidates have to run through the root and leaves WVM’s. Further step would be to optimize the top node WVM for getting a higher rejection performance, which would result in less run-time for the computation of the combined approach.

only one root WVM

image number	time in sec per detection	#of candidates
1	0.08	201
2	0.12	217
3	0.12	227
4	2.27	2554
5	1.98	2352
6	1.16	1489
7	0.96	1202
8	0.04	58
9	0.03	75
10	0.03	81
11	1.28	1560
12	0.19	266
average	0.69	856.83

only leave WVM's

image number	time in sec per detection	#of candidates
1	0.31	72
2	0.31	62
3	0.27	46
4	2.67	229
5	2.71	343
6	2.07	314
7	2.09	393
8	0.1	22
9	0.14	44
10	0.14	52
11	2.1	349
12	0.53	61
average	1.12	165.58

root and leave WVM's combined

image number	time in sec per detection	#of candidates
1	0.24	56
2	0.36	61
3	0.31	44
4	3.21	221
5	2.93	305
6	2.07	283
7	1.88	306
8	0.09	19
9	0.13	41
10	0.12	43
11	2.23	304
12	0.35	48
average	1.16	144.25

Table 4.3: Run-time performance of the different root variants

Chapter 5

Conclusion

The main contribution of this thesis was to reach a pose-invariant face detection. We took the advantage of the divide and conquer principal and divided the large face space into several sub spaces. For each sub space, we trained a detector based on the data on its sub space. We arranged these detectors in a hierarchical tree structure. As detector we used the WVM On top of the tree is a root node, which makes a first coarse decision for a given patch. Based on this information, the leaves can be traversed. For patches which pass the root node, the leaves compute a decision. A final decision is made. As novelty we introduced the fusion of responses to combine different responses and to get an optimal result. Important for the efficiency of the detectors is the variability in the train- and test data, on which they are optimized. For this, the procurement and evaluation of different face databases was an additional achievement. With the Detector Tree, a pose-invariant face detection for yaw ± 90 and pitch ± 30 degrees has been reached.

Chapter 6

Further Work

There are several extensions which could be done.

In Section 2.8.3 we introduced the fusion of responses with the assumption that no FA's are present, which would result in a falsified face box. To prevent the fusion for FA's, a possible solution is to analyze the overlapping areas between the responses. Lets assume the sliding window size is ws, the scale factor sf, the scale level l, h is the height of the patch and $p(x, y)$ the center of the patch. The height h can be calculated like:

$$h = \frac{ws}{sf^l} \tag{6.1}$$

This results in a rectangle with corners $p^{*1}(x - \frac{h}{2}, y - \frac{h}{2})$ and $p^{*2}(x + \frac{h}{2}, y + \frac{h}{2})$. In consideration of only x dimension $p^{*1}(x) = a$ and $p^{*2}(x) = b$. For every response of the detectors this can be calculated and results in three intervals $[a_1, b_1] = O$;$[a_2, b_2] = P$;$[a_3, b_3] = Q$. The overlapping area is

$$\begin{aligned} O \cap P \cap Q \cap &= [max(a_1, a_2, a_3), min(b_1, b_2, b_3)] \\ &= [c, d] \\ &\text{respect to } c, d \in O, P, Q \end{aligned} \tag{6.2}$$

An example is illustrated in Figure 6.1. The length of the interval is $l_1 = d - c$. The calculation of the intersection can be done in the y dimension, too, resulting in l_2. If the responses are overlapping, the area between the patches is

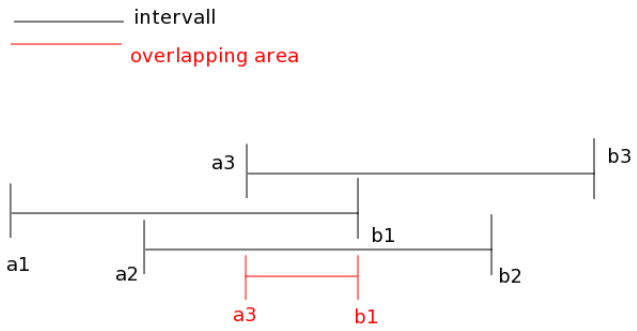


Figure 6.1: The different intervals in one dimension with a overlapping interval [a3,b1].

$l_1 * l_2 > 0$. This can also be done for only two detectors. The overlapping area between the detectors can be used for weighting the portion to the fusion of the responses. If the area between two detectors is large, they will get a higher weight than the outliner. If the to be detected faces are estimated with a variability only in yaw angle, an additional analysis for the y value can be made.

One method for preventing FA's could be to detect facial features in the patch after detecting the face box. Every sub space has a specific arrangement of features e.g. for left detector the half of the mouth, only one eye or a nose which looms outside of the face contour. These features can be analyzed and a decision of an expert can either be confirmed or discarded. Also could an analysis of facial features after the first OE for the remaining patches gives information about a possible expert for a sub space. This would result in more complexity for detecting the face, because many detections of features have to be run before the final detection of the expert is made.

Another approach for assigning a patch to an expert could be done by estimating the pose for a given patch by regression. Or the other way around, if a patch is flagged as a face by an expert, the regression could give an accurate result for the pose on the face of a given patch. And if the result does not cover with the sub space of the expert, the patch can be rejected. The fusion of the responses of the leaves could be improved optimally by combining the evaluated probabilities, the locations and the pose estimations.

One additional approach for assigning a patch to an expert could be to use a multi class SVM for the remaining patches after the root node. This SVM can be trained with the input spaces of the experts. This would be an accurate identification of the expert, but would be an additional SVM which results in an increased run-time. In compare to the approach seen above this would reduce the costs of the computation because of reducing the number of the leaves to one expert. The multi class SVM could also be extended to the fusion of responses and could be one additional information which can be integrated in the final decision.

For reaching a detection of ± 90 pitch and roll angle, the Detector Tree could be extended with some more layers and more accurate experts with a smaller sub space. To make a decision with the responses, the Hierarchical Mixtures of Experts [20] could be used to consider all leaves and their predecessor.

Instead of using the Detector Tree with the sub spaces of angles, the detectors could also be trained for facial features. For example, the root node classifies both, left and right eyes (which are very similar for frontal faces). If the patch is not rejected, the leaves decide if it is an left or right eye. The result would be an invariant feature detection. Or e.g. several different scales for a patch, which would result in a scale invariant detection.

To upgrade the detectors in general, an improvement of the negative examples could be done. Using the Detector Tree for detecting the faces in images, we could cut off the patches around the face box as seen in Figure 6.2 (bootstrapping). This would help to define the area around the face during the detection and could prevent

FA's around the face.

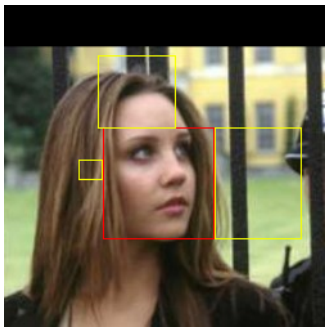


Figure 6.2: Yellow boxes: Negative patches cut near the face box.

Bibliography

[1] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3d faces, 1999.

[2] Tae kyun Kim and Josef Kittler. Design and fusion of pose-invariant face-identification experts. *IEEE Transactions on Circuits and Systems for Video Technology*, 16:1096–1106, 2006.

[3] Hichem Sahbi, Donald Geman, and Pietro Perona. A hierarchy of support vector machines for pattern detection. *Journal of Artificial Intelligence Research*, 7:2087–2123, 2006.

[4] Matthias Rätsch. *Wavelet Frame Accelerated Reduced Vector Machine for Efficient Image Analysis*. PhD thesis, University of Basel, 2008.

[5] J.Y. Jiang, S.F. Zheng, A.W. Toga, and Z.W. Tu. Learning based coarse-to-fine image registration. In *CVPR08*, pages 1–7, 2008.

[6] Ho Lee, Guillaume Lavoue, and Florent Dupont. Adaptive coarse-to-fine quantization for optimizing rate-distortion of progressive mesh compression. In *Vision, Modeling, and Visualization Workshop(VMV)*, pages 73–81, November 2009.

[7] Corinna Cortes and Vladimir Vapnik. Support-vector networks. In *Machine Learning*, pages 273–297, 1995.

[8] Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. A training algorithm for optimal margin classifiers. In David Haussler, editor, *Proceedings of the 5th Annual Workshop on Computational Learning Theory (COLT '92), July 27-29, 1992, Pittsburgh, PA, USA*, pages 144–152. ACM Press, New York, NY, USA, 1992.

[9] Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

[10] Matthias Rätsch, Sami Romdhani, Gerd Teschke, and Thomas Vetter. Over-complete wavelet approximation of a support vector machine for efficient classification. In *In Pattern Recognition 27th DAGM Symposium, volume 3663 of LNCS*, pages 351–360. Springer, 2005.

[11] P. Jonathon Phillips, Hyeonjoon Moon, Syed A. Rizvi, and Patrick J. Rauss. The feret evaluation methodology for face-recognition algorithms, 1999.

[12] Wen Gao, Bo Cao, Shiguang Shan, Delong Zhou, Xiaohua Zhang, Debin Zhao, Wen Gao, Bo Cao, Shiguang Shan Et. Al, Wen Gao, Bo Cao, Shiguang Shan, Delong Zhou, Xiaohua Zhang, and Debin Zhao. The cas-peal large-scale chinese face database and evaluation protocols. Technical report, Joint Research and Development Laboratory, CAS, 2004.

[13] Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007.

[14] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar. Attribute and Simile Classifiers for Face Verification. In *IEEE International Conference on Computer Vision (ICCV)*, Oct 2009.

[15] N. Gourier, D. Hall, and J. L. Crowley. Estimating Face Orientation from Robust Detection of Salient Facial Features. In *Proceedings of Pointing 2004, ICPR, International Workshop on Visual Observation of Deictic Gestures*, 2004.

[16] John A. Black, M Gargesha, K Kahol, P Kuchi, and Sethuraman Panchanathan. A framework for performance evaluation of face recognition algorithms, 2002.

[17] Greg Little, Sreekar Krishna, John Black, and Sethuraman Panchanathan. A methodology for evaluating robustness of face recognition algorithms with respect to changes in pose and illumination angle. Ottawa, Canada, October, 2005 2005.

[18] Patrik Huber. Normalization and feature space reduction of a support- and reduced vector machine. Bachelor-Thesis, 2009.

[19] Thorsten Joachims. *Making large-scale support vector machine learning practical*, pages 169–184. MIT Press, Cambridge, MA, USA, 1999.

[20] Michael I. Jordan. Hierarchical mixtures of experts and the em algorithm. *Neural Computation*, 6:181–214, 1994.