Automatic Model-based Face Reconstruction and Recognition

Automatische modell-basierte Gesichtsrekonstruktion und -erkennung

Vom Fachbereich Elektrotechnik und Informatik
der Universität Siegen

zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften
(Dr.-Ing.)

genehmigte Dissertation

von

Dipl.-Inform. Pia Breuer

Siegen – August 2010

Gedruckt auf alterungsbeständigem holz- und säurefreiem Papier.

Als Dissertation genehmigt vom
Fachbereich Elektrotechnik und Informatik
Universität Siegen

Einreichung:	August 2010
Tag der mündl. Prüfung:	21. Dezember 2010
Dekan:	Prof. Dr. M. Pacas
1. Gutachter:	Prof. Dr. V. Blanz
2. Gutachter:	Prof. Dr. T. Vetter
Vorsitzender:	Prof. Dr. U. Kelter

Abstract

Three-dimensional Morphable Models (3DMM) are known to be valuable tools for both face reconstruction and face recognition. These models are particularly relevant in safety applications or Computer Graphics. In this thesis, contributions are made to address the major difficulties preceding and during the fitting process of the Morphable Model in the framework of a fully automated system. It is shown to which extent the reconstruction and recognition results depend on the initialization and what can be done to make the system more robust, e.g. against vague feature positions or occlusions of the face.

Based on the 3DMM, a fully automated algorithm is presented. Support Vector Machines (SVMs) and the Morphable Model of 3D faces are combined for reconstructing a textured 3D model of a face from a single photograph or a raw video stream. The SVM delivers a list of candidates for several facial feature positions, and these are evaluated using a novel criterion that is based on the Morphable Model and a combination of linear projections. To make the algorithm robust with respect to head orientation, this process is iterated while the estimate of pose is refined. Finally, the feature points initialize the model-fitting procedure of the Morphable Model to result in a high-resolution 3D surface model.

Furthermore a new approach called self-adapting feature layers (SAFL) is presented. The algorithm integrates feature detection into the iterative analysis-by-synthesis framework, combining the robustness of feature search with the flexibility of model fitting. Templates for facial features are created and updated while the fitting algorithm converges, so the templates adapt to the pose, illumination, shape and texture of the individual face. The benefit of the proposed method is an increased robustness of model fitting with respect to unavoidable errors in the initial feature point positions. Such residual errors usually create problems when feature detection and model fitting are combined to form a fully automated face reconstruction or recognition system. Several case studies show the benefits of the new concept proposed in this work.

In addition to the SAFL concept, another focus of this work is the importance of contour information for the entire reconstruction result. The overall shape of the face is not exclusively determined by the contour line, but it is substantially influenced by it. Different approaches are presented improving the contour fitting of the 3DMM.

Besides initialization, robustness and better contour adaption, this thesis also addresses the problem of occlusions. Manually or automatically identified occlusions can be marked to exclude

them from the fitting process. Different kinds of automated detection algorithms using the 3DMM, for different kinds of occlusions, are presented.

Finally, capabilities have been investigated for the reconstruction based on multiple images. Here the focus was not only on a better reconstruction of the overall shape, but on distinguishing features, such as wrinkles, birthmarks or freckles. If only one input image is used, these get poorly reproduced. The reconstruction out of multiple images enhances their reconstruction.

The essential approaches are applicable to other model- based approaches to image analysis and they include a number of general strategies to analysis-by-synthesis besides their contribution to the improvement of the 3DMM.

Zusammenfassung

Dreidimensionale Morphable Models (3DMM) sind bekannt als wertvolle Werkzeuge für die Gesichtsrekonstruktion und auch für die Gesichtserkennung. Außerdem wird diese Art von Modellen für Sicherheitsanwendungen und in der Computergraphik verwendet. Die vorliegende Doktorarbeit behandelt im Hinblick auf die Nutzung in einem vollautomatischen System die Hauptschwierigkeiten, die vor und während des Anpassungsprozesses des morphfähigen Modells auftreten. Es wird gezeigt, in welchem Maße die Ergebnisse der Rekonstruktionen und der Erkennung von der Initialisierung abhängen und wie das System robuster gemacht werden kann, beispielsweise im Bezug auf ungenaue Merkmalspositionen oder Verdeckungen des Gesichts.

Basierend auf dem 3DMM von Blanz und Vetter wird in dieser Arbeit ein vollautomatischer Algorithmus präsentiert. Es werden Support Vector Machines (SVMs) und das morphfähige 3D Gesichtsmodell kombiniert, um ein texturiertes 3D Modell eines Gesichts aus einem Einzelphoto oder einem groben Videostrom zu rekonstruieren. Die SVM liefert eine Liste von Kandidaten für verschiedene Gesichtsmerkmalspositionen. Diese werden mittels eines neuen Kriteriums analysiert und die besten ausgewählt. Das neue Kriterium basiert auf dem morphfähigen Modell und einer Kombination von linearen Projektionen. Um den Algorithmus unabhängig von der Orientierung des Kopfes nutzen zu können, wird der Prozess iteriert, während die Posenschätzung verfeinert wird. Schließlich wird die Rekonstruktion mit den ausgewählten Merkmalspunkten initialisiert und die Modellanpassung liefert ein hochaufgelöstes 3D Oberflächenmodell.

Desweiteren wird ein neuer Ansatz vorgestellt: self-adapting feature layers (SAFL). Der Algorithmus verbindet Merkmalserkennung und das iterative Analyse-durch-Synthese Rahmenprogramm. Dadurch wird die Robustheit der Merkmalserkennung mit der Flexibilität der Modellanpassung kombiniert. Während der Anpassungsalgorithmus konvergiert, werden die Vergleichsmuster der Gesichtsmerkmale jeweils aktualisiert. So passen sich die Vergleichsmuster an Form, Farbe, Kopfpose und Beleuchtung jedes individuellen Gesichts an. Der Vorteil der vorgestellten Methode liegt in der größeren Robustheit der Modellanpassung, falls unvermeidbare Fehler bei den initialen Merkmalspositionen auftreten. Wenn man Merkmalsdetektion und Modellanpassung für ein vollautomatisches Gesichtsrekonstruktions- und erkennungssystem kombiniert, verursachen solche verbleibenden Fehler meist Probleme. Verschiedene Fallstudien zeigen die Vorteile des neuen Konzepts, das in dieser Arbeit vorgestellt wird.

Zusätzlich zum SAFL-Konzept wird die Wichtigkeit der Konturinformationen für das Gesamtergebnis der Rekonstruktion untersucht. Die Gesamterscheinung des Kopfes hängt

nicht ausschließlich von der Konturlinie ab, wird durch sie jedoch maßgeblich beeinflusst. In dieser Arbeit werden verschiedene Ansätze vorgestellt, um die Konturanpassung des 3DMM zu verbessern.

Neben der Initialisierung, der Robustheit und einer besseren Konturanpassung liegt ein weiterer Augenmerk dieser Dissertation auf dem Umgang mit Objekten, die das Gesicht partiell verdecken. Verdeckungen, die manuell oder automatisch identifiziert und markiert werden, können bei der Anpassung ignoriert werden. Es werden verschiedene Verfahren vorgestellt, die mithilfe des 3DMM verschiedene Arten von Verdeckungen erkennen.

Abschließend werden die Möglichkeiten untersucht, inwiefern das Rekonstruktionsergebnis durch die Nutzung mehrerer Bilder für eine Anpassung verbessert werden kann. Hierbei liegt der Fokus nicht nur auf einer besseren Rekonstruktion der Gesamtform, sondern besonderer Merkmale wie Falten, Muttermalen oder Sommersprossen. Wird nur ein Eingabebild genutzt, werden diese Merkmale nur unzureichend reproduziert. Die Anpassung an mehrere Bilder verbessert die Rekonstruktion. Die grundlegenden Ansätze dieser Arbeit sind auf andere modell-basierte Verfahren zur Bildanalyse übertragbar. Neben dem Beitrag zur Verbesserung des 3DMM enthält diese Arbeit außerdem grundlegende Strategien, die in anderen Analyse-durch-Synthese-Verfahren genutzt werden können.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

For the reconstruction of 3D faces from image data, there are a variety of approaches that rely on different sources of depth information: some perform triangulation from multiple simultaneous views, e.g. stereo or multiview-video methods. Others use multiple consecutive monocular views in video streams for structure-from-motion or silhouette-based approaches. Finally, there are algorithms that rely on single still images only, for example by exploiting shading information (shape-from-shading) or by fitting face models to single images.

Basis of this thesis is the 3D Morphable Model (3DMM) introduced by Blanz and Vetter [BV99]. The present model can be used for several areas of application. One of the most important scopes is face recognition. On the one hand, it is possible to use the shape and texture coefficients of the adapted model for recognition, and on the other hand, the reconstruction can be used for preprocessing in the following way. Common face recognition applications mostly work with frontal views as target and query images. The recognition gets more difficult, if the query image is a side view, and the database contains only frontal views to compare with. Using the 3D Morphable Model, a synthetic frontal view can be generated out of a side view. This can be compared much better to the frontal views in the database. The method of using the model coefficients for recognition is completely independent from the views in the images because only the characteristics of the heads are compared, and not their poses or rendering conditions.

Another field of application is identity exchange in images, replacing a head by a different one. This works by the same principle as the generation of the synthetic frontal view. The reconstruction of a head out of an image delivers not only shape and texture information, but also information about the pose of the head and the lighting situation. Rendering another model with the same pose and lighting as in the input image, we are able to replace the original head by the new one. For facial animation we are able to add expressions or movements, learned before, to any head model. It is possible to let the models talk, move their eyes, or even to make Mona-Lisa laugh.

The listed possible applications show the diversity of the present model. This thesis concentrates on the need for improvements within the scope of face recognition. There are two major goals: The first goal is to improve the reconstruction, and thus the recognition, and the second goal is to

get able to use the model in an automated application. Not using manual input for initialisation any more, we have to cope with different new challenges. Generally speaking, the model has to become more robust. There are two different ways of looking at the problem. On the one hand, we investigate what can be done before the fitting, and on the other hand, we investigate what is possible to do during the fitting process to automate the reconstruction and to make it more robust. At the general overview in Fig. 1.1 this can also be seen. The blue boxes mark the areas of research addressed in this thesis. The ones in the upper part of the figure refer to components before the fitting. The ones in the right part of the figure refer to the components during the fitting process, or to those which have influence on the cost function to be minimized.

For initialization, the positions of facial features are needed. A common feature detector can be used to find these feature positions. It will be shown that we can also use the 3DMM to detect reliable positions of facial features. This is done in combination with Support Vector Machines (SVMs). The SVMs deliver a list of candidates for several facial feature positions, and these are evaluated using a novel criterion that is based on the Morphable Model and a combination of linear projections. This process takes place before the essential fitting. To make the algorithm robust with respect to head orientation, the process is iterated while the estimate of pose is refined. By this combination of SVMs and the 3DMM, we get a fully automated application, reconstructing a textured 3D model of a face from a single photograph or a raw video stream.

Using automatically detected feature positions, the quality of the fit turned out to depend critically on the precision of the facial features. We present an algorithm inside the fitting process to warrant a reliable reconstruction, even using vague feature positions. The new approach called self-adapting feature layers (SAFL) integrates feature detection into the iterative analysis-by-synthesis framework. Templates for facial features are created and updated while the fitting algorithm converges, so the templates adapt to the pose, illumination, shape and texture of the individual face. Several case studies show the benefits of the new concept proposed in this work.

Face contour adaption is another important part of the reconstruction process. The overall shape is essentially influenced by the contour. It will be shown that improvement of the adaption to the contour is needed. We compare various edge detection algorithms, and present new approaches to enhance the contour adaption, using a combination of edge detection and template matching.

In real world situations, it is also possible to have occlusions in the input image. The 3DMM offers the option to use an additional image for initialisation, which we refer to as occlusion mask. This can also be seen in the general overview (Fig. 1.1). Areas marked in the mask will be ignored during the fitting process, and a reliable result will be ensured. Fully automatic reconstruction, though, requires an automatic detection of occluded areas, too. In this thesis, we present different approaches for automated occlusion detections, for different kinds of occlusions.

At last we investigate the question on how the use of more than one image of a person can lead to more precise reconstruction results. At this the focus is not only on a better reconstruction of the overall shape, but on distinguishing features, such as wrinkles, birthmarks or freckles. These get poorly reproduced, using only one input image. The reconstruction out of multiple images enhances this. We show how to force the shape to reproduce the wrinkles much better, and we

also show how to improve the adapted texture.

The contributions of this thesis are beneficial to the improvement of the reconstructions and the recognition results using the 3DMM. The essential approaches are applicable to other model-based approaches to image analysis and they include a number of general strategies to analysis-by-synthesis besides their contribution to the improvement of the 3DMM. The novel criterion to evaluate candidates for facial feature positions is independent of the specific choice of a feature selection method. The used SVMs can be replaced by other feature detectors like AdaBoost, Template Matching or SIFT. The new type of cost function of the SAFL approach is an innovative contribution on a conceptual level. It combines optimization and feature detection in a novel way. The integration of Template Matching into the fitting process can be adopted one-to-one to other domains like Active Shape Models or Active Appearance Models. The fundamental idea of the new type of cost function will stimulate discussions and further developments in the entire field of research on automatic face reconstruction and face recognition. The improvement of the contour fitting and the automated detection of occluded regions can also be adopted to other model-based analysis-by-synthesis approaches. For the contour fitting, we use a special template matching. The findings of this research can be adopted one-to-one, or will serve a basis for further development. The approaches for the detection of occluded regions show the possibilities of automatic detection using the 3DMM. Other models may be used in a similar way.

The rest of the thesis is organized as follows: Chapter 2 gives a detailed description of the 3D Morphable Model of Blanz and Vetter. In Chapter 3, three important aspects for the introduction to the subject matter of automatic face reconstruction and recognition are reviewed. To get an overview of the state of the art of statistical models of faces, we focus on further developments of basic models like Active Shape Models, Active Appearance Models, and the 3DMM. Separate linear models, and approaches for the reconstruction from video are also introduced, relating to our fully automated approach presented in this thesis. Also relating to this approach, we review the development in the field of face detection and facial feature detection. Finally we address image segmentation. This is related to our research in contour adaption and automated occlusion detection. Chapter 4 gives a survey of essential feature detection algorithms. Four methods are presented in more detail. They represent a benchmark for our feature detection approach in the next chapter. In Chapter 5, we introduce our model-based confidence measure for feature points and a fully automatic reconstruction system. Chapter 6 illustrates our new self-adapting feature layers (SAFL) approach, which leads to independence from reliable initial feature positions. Chapter 7 describes the processed approaches for the improvement of the contour adaption. In Chapter 8, the different approaches for the automated detection of occluding regions are presented, which have been developed in this thesis. Chapter 9 concentrates on improving the reconstruction by using three images, to get more detailed results. Finally, in Chapter 10 the thesis will be concluded.

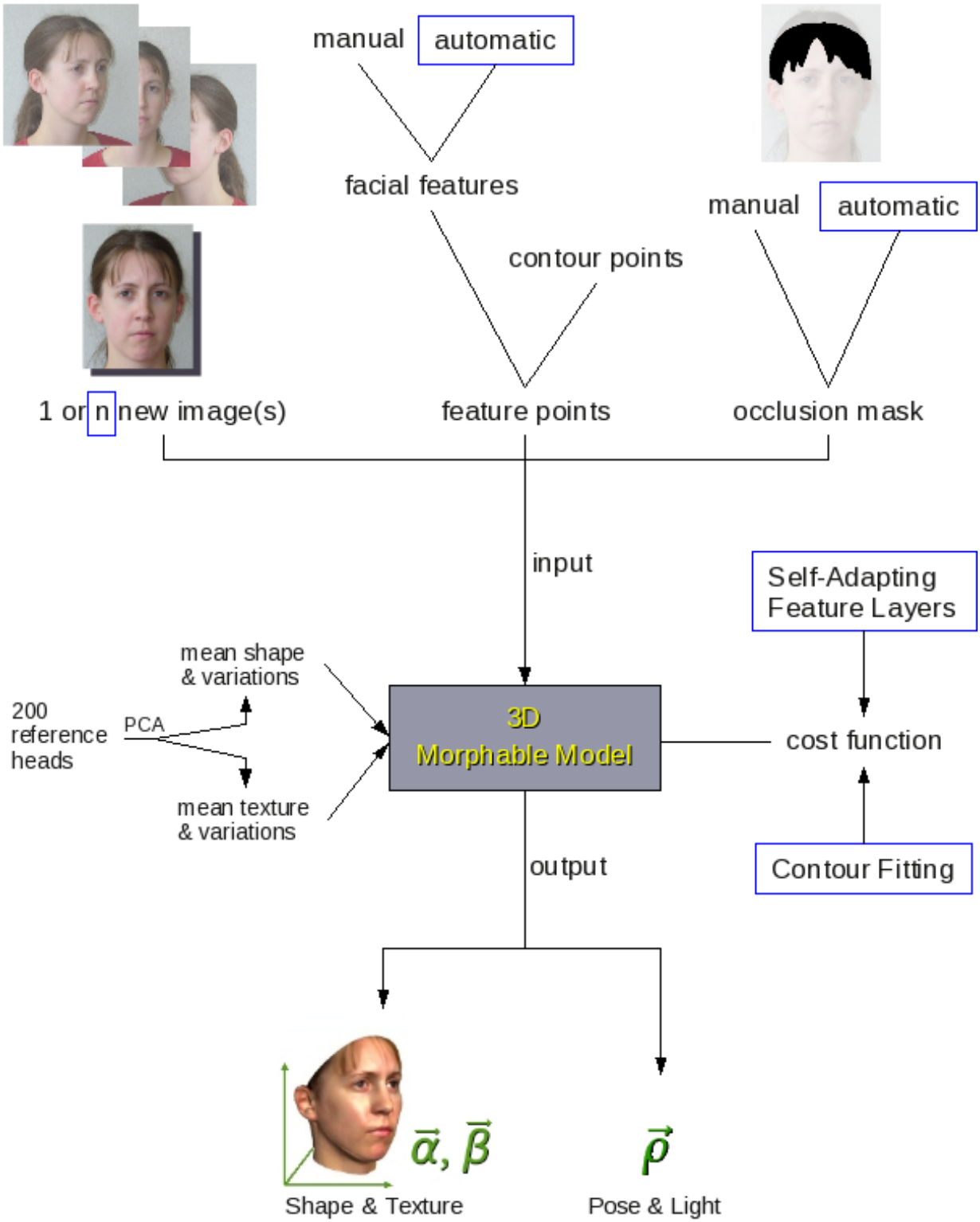


Figure 1.1: **General Overview:** of the face reconstruction system applied in this thesis. The mean shape and texture, and their variations have been learned from 200 reference heads. Based on these, the 3D Morphable Model fits to one or more given input images, minimising a cost function (details in Chap. 2). For initialisation feature points are used, and optionally an occlusion mask can be used to ignore occluded regions during the fitting. The blue boxes mark the areas of research attended in the following chapters of this thesis.

Chapter 2

The 3D Morphable Model

The Morphable Model of 3D faces (3DMM) has first been presented by Blanz and Vetter [BV99]. Learned from examples of 3D faces, the 3DMM is a (statistical) vector space representation of natural human faces for image analysis and for computer graphics. It is used for the reconstruction of a high-resolution 3D mesh. It was built by establishing dense correspondence on a training set of 3D scans of 200 individuals.

Setup

The scans have been recorded with a *Cyberware*^{TM} 3030PS laser scanner, yielding a face shape representation in cylindrical coordinates relative to a vertical axis through the center of the head. On a 512×512 grid representing height h and rotation angle ϕ of the scanner, radius r and color components of the surface texture R,G,B are measured: $r(h,\phi),R(h,\phi),G(h,\phi),B(h,\phi)$ (cf. Fig 2.1).

Shape vectors $\mathbf{S}$ are formed by the x,y,z-coordinates of all vertices $k \in \{1,\ldots,n\}$, $n = 75,972$ of a polygon mesh based on the cylindrical coordinates $(h_k,\phi_k,r(h_k,\phi_k))$, and texture vectors $\mathbf{T}$ are formed by red, green and blue values:

$$\mathbf{S} = (x_1,y_1,z_1,x_2,\ldots,x_n,y_n,z_n)^T \tag{2.1}$$

$$\mathbf{T} = (R_1,G_1,B_1,R_2,\ldots,R_n,G_n,B_n)^T. \tag{2.2}$$

Dense point-to-point correspondence (cf. Fig 2.2) has been established by using optical flow.

Picturing the example faces as data points forming a hyperellipsoidally shaped cloud, PCA is used to identify the principal axes of that hyperellipsoid. The point cloud has to be presented as a square matrix, the covariance matrix. The eigenvectors of this matrix are the principal axes of the point cloud. Because of the covariance matrix being real and symmetric, these eigenvectors

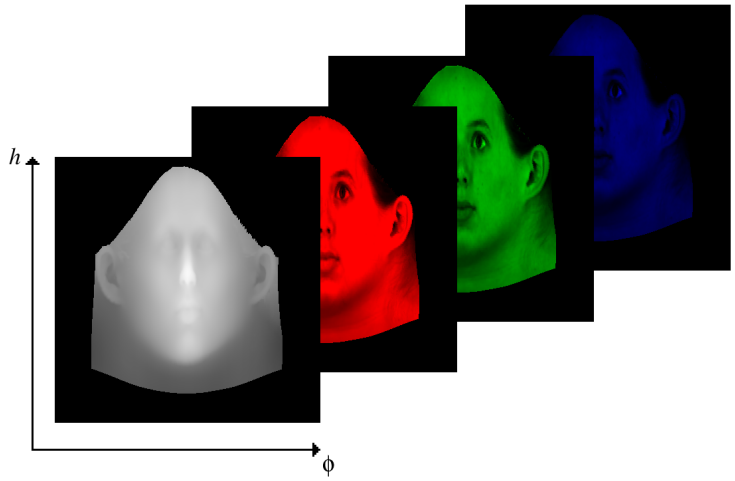


Figure 2.1: **Scanner output:** Example for radius (grey-scale coded depth values) and color components on the 512×512 grid delivered by the scanner.

are orthogonal. They are statistically independent and can be used as a new basis to transform the data to. The eigenvalues are the components of a data point in the basis, and they are called the principal components. Having a normal distribution of data, points of equal probability density are lying on ellipsoids, and after transformation to the new basis they are lying on circles. PCA is also an important method to reduce the dimensions of a data space, restricting attention to those principal axes with the largest principal components (cf. Duda et al.[DHS01]).

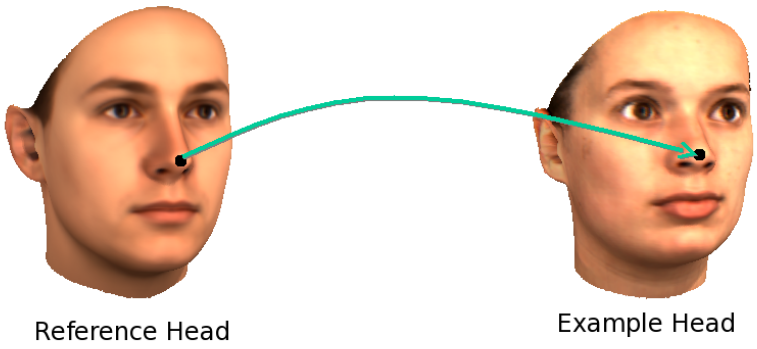
To analyze the face space, PCA is performed on the set of shape and texture vectors, $\mathbf{S}_i$ and $\mathbf{T}_i$, of all example faces $i = 1...m$. Shape and texture are analyzed separately, ignoring their correlation. For shape, first, the mean vector $\bar{\mathbf{s}} = \frac{1}{m}\sum_{i=1}^{m} \mathbf{S}_i$ and the covariance matrix $\mathbf{C}$ have to be computed. Subtracting the average $\bar{\mathbf{s}}$ from each shape vector, $\mathbf{a}_i = \mathbf{S}_i \quad \bar{\mathbf{s}}$, a data matrix $\mathbf{A} = (\mathbf{a}_1, \mathbf{a}_2, ..., \mathbf{a}_m)$ is defined, which forms the covariance matrix

$$C = \frac{1}{m}\mathbf{A}\mathbf{A}^T = \frac{1}{m}\sum_{i=1}^{m} \mathbf{a}_i \mathbf{a}_i^T. \tag{2.3}$$

Using this intermediate step, it is possible to compute the eigenvectors $\mathbf{s}_1, \mathbf{s}_2, ...$ of the covariance matrix by a Singular Value Decomposition (cf. Numerical Recipes [PTVF92]) of $\mathbf{A}$. The eigenvalues $\sigma_{S,1}^2 \geq \sigma_{S,2}^2 \geq ...$, are the variances of the data along each eigenvector.

The same procedure is used to get the eigenvectors $\mathbf{t}_i$ and variances $\sigma_{T,i}^2$ for the texture, obtaining a set of m orthogonal principal components and eigenvectors, forming an orthogonal basis:

$$\mathbf{S} = \bar{\mathbf{s}} + \sum_{i=1}^{m} {}^{1}\alpha_i \cdot \mathbf{s}_i, \quad \mathbf{T} = \bar{\mathbf{t}} + \sum_{i=1}^{m} {}^{1}\beta_i \cdot \mathbf{t}_i. \tag{2.4}$$



$$\mathbf{S}_0 = \begin{pmatrix} x_1 \\ y_1 \\ z_1 \\ \dots \\ x_k \\ y_k \\ z_k \\ \dots \end{pmatrix} \mathbf{T}_0 = \begin{pmatrix} R_1 \\ G_1 \\ B_1 \\ \dots \\ R_k \\ G_k \\ B_k \\ \dots \end{pmatrix} \qquad \mathbf{S} = \begin{pmatrix} x_1 \\ y_1 \\ z_1 \\ \dots \\ x_k \\ y_k \\ z_k \\ \dots \end{pmatrix} \mathbf{T} = \begin{pmatrix} R_1 \\ G_1 \\ B_1 \\ \dots \\ R_k \\ G_k \\ B_k \\ \dots \end{pmatrix}$$

Figure 2.2: **Correspondence:** Each single vertex k represents the same feature on every head. So for example the tip of the nose is represented by the x, y, z and R, G, B values at the same position in every $\mathbf{S}_i$ and $\mathbf{T}_i$.

PCA also provides an estimate of the prior probability density within face space, an approximated multidimensional normal distribution:

$$p_S(\mathbf{S}) \sim e^{\frac{1}{2}\Sigma_i \frac{\alpha_i^2}{\sigma_{S,i}^2}}, \qquad p_T(\mathbf{T}) \sim e^{\frac{1}{2}\Sigma_i \frac{\beta_i^2}{\sigma_{T,i}^2}}. \tag{2.5}$$

Based on this, the distance from one data point x to the mean μ of all data points can be measured: The Mahalanobis Distance is related to the log of the prior probability. Data points with the same Mahalanobis Distance to the mean or the same prior probability lie on the same ellipsoid in the data space.

In this work, only the first 99 principal components of shape and texture are used, because they cover most of the variance observed in the training set. A larger number would increase the computation time while not improving the results significantly.

Fitting

To reconstruct an unknown head, we fit the 3DMM to a 2D image of the person. In an analysis-by-synthesis loop, we find the face vector from the Morphable Model that fits the image best in

terms of pixel-by-pixel color difference between the synthetic image I_{model} (rendered by standard computer graphics techniques), and the input image I:

$$E_I = \sum_{x,y} (I(x,y) - I_{model}(x,y))^2. \tag{2.6}$$

The squared differences in all three color channels are added in E_I. We suppress the indices for the separate color channels throughout this work. The optimization is achieved by an algorithm that was presented by Blanz and Vetter in [BV99, BV03], which we describe in the following: In each iteration, the algorithm evaluates E_I not on the entire image, but only on 40 random vertices, reselected each time. For the optimization to converge, the algorithm has to be initialized with the feature coordinates of at least 5 feature points.

The goal is to minimize the cost function

$$\mathbf{E}(\vec{\alpha}, \vec{\beta}, \vec{\rho}) = \eta_I \cdot E_I + \eta_M \cdot E_M + \eta_P \cdot E_P \tag{2.7}$$

where E_M is the sum of the squared distances between the 2D positions of the marked feature points in the input image $(q_{x,j}, q_{y,j})$, and their current positions in the model (p_{x,k_j}, p_{y,k_j}):

$$E_M = \sum_j || \begin{pmatrix} q_{x,j} \\ q_{y,j} \end{pmatrix} - \begin{pmatrix} p_{x,k_j} \\ p_{y,k_j} \end{pmatrix} ||^2. \tag{2.8}$$

E_P is the Mahalanobis Distance of the current solution from the average face, which is related to the log of the prior probability of the current solution:

$$E_P = \sum_i \frac{\alpha_i^2}{\sigma_{S,i}^2} + \sum_i \frac{\beta_i^2}{\sigma_{T,i}^2} + \sum_i \frac{(\rho_i - \bar{\rho}_i)^2}{\sigma_{R,i}^2}. \tag{2.9}$$

For the measurement of E_P we use the squared Mahalanobis Distance. PCA caused a rearrangement of the data, with the mean (average face) as origin of the coordinate system. For shape and texture, the distance between the current solution (a data point) and the average face (mean), is the absolute value of the data point or the length of the vector to this data point. For the rigid parameters ρ the Mahalanobis Distance is based on the length of the vector between the data point and the mean rigid parameters. Using the squared distance, we do not have to calculate a square root.

η_I, η_M and η_P are weights that are set heuristically: The optimization starts with a conservative fit (η_M and η_P are high), and in the final iterations $\eta_M = 0$ and η_P is small.

Using stochastic Newton optimization the algorithm optimizes the linear coefficients for shape ($\alpha = (\alpha_1, \alpha_2, ...)$) and texture ($\beta = (\beta_1, \beta_2, ...)$), but also 3D orientation and position, focal length of the camera, angle, color and intensity of directed light, intensity and color of ambient light, color contrast as well as gains and offsets in each color channel (combined in ρ).

Figure 2.3: **Segments:** The four regions fitted separately are shown on the average head, mouth (green), nose (red), eyes (blue) and the surrounding area (skin color of the average head)

During the first iteration only the first parameters $\alpha_i, \beta_i, i \in \{1, ..., 10\}$ and all parameters ρ_i are optimized. In further iterations more and more coefficients are taken into account. To be able to reconstruct a larger variety of faces using the given set of examples, different regions of the face have to be fitted separately. The regions are: mouth, eyes, nose and surrounding area (cf. Fig. 2.3). To ensure continuous transitions between the segments an image blending technique interpolating x, y, z and R, G, B in the (h, ϕ)-domain based on the mapping to the reference face is used. After fitting the whole model the segments are optimized separately and combined afterwards.

To be able to reconstruct all details of every face (e.g. moles at any position) a very high dimensional texture space would be needed, which is unfeasible in practice. However, it is possible to map the texture information from the input image onto the reconstructed head. The correspondence problem between model and image is already solved by the fitting. Having the three-dimensional shape and the estimated light, it is possible to distinguish between texture properties and shadows or specular highlights. The equation of the color transformation and the illumination model can be inverted and so shadows and specular highlights can be subtracted when extracting the texture and the 'real' color values of the face are mapped to the model. Using the correspondence between model and input image, all vertices of the model are visited and the color values are taken from the corresponding positions in the input image. Vertices not visible in the input image keep their estimated color values. This illumination fixed texture extraction ensures a correct reproduction of the characteristic features. Using the symmetry characteristics of faces, gaps can be filled in by mirroring of the present extracted texture. So for example an input image showing a rotated view can also yield a fully textured head taking the missing color values from the opposing site.

Chapter 3

Related Work

There are three important aspects for the introduction to the subject matter. This chapter presents an overall picture of significant concepts and ideas of these aspects: Statistical Models of Faces, Detection of Faces and Facial Features, and Segmentation of Images.

3.1 Statistical Models of Faces

Two basic approaches for face reconstruction using statistical models, are the Active Appearance Model (AAM) of Cootes et al. [CET98] for 2D, and the 3D Morphable Model (3DMM) of Blanz and Vetter [BV99] for 3D.

The AAM is based on the Active Shape Model (ASM), developed by Cootes et al. [CTCG95] as well. ASMs are build by learning patterns of variability from a training set of images with feature positions marked by hand. An object is represented as a set of labeled points, their mean position, and a small set of modes of variation. The modes, computed with PCA, describe how the object's shape can change. They build global shape constraints. ASMs can only deform to fit the data in ways consistent with their training set. In an iterative search procedure modeled structures can be located in noisy, cluttered images. AAMs are an advancement of ASMs, using not only the shape information, but also taking into account the grey-level information within the object shape. It is a combination of a model of shape variations and a model of appearance variations in a shape-normalised frame. Based on face images with 122 marked key points, outlining the main features, first a statistical model of shape variations is learned, performing PCA on the aligned point sets. To get the statistical model of grey-level appearance, PCA is performed on the normalised grey-level information. For that, the example images have to be warped, the key points matching the mean shape. To determine the correlation between the shape and grey-level parameters, PCA has to be performed again on the concatenated data. This leads to global appearance parameters. Performing multi-variate regression, the model learns the relationship between the image error and the error in the model parameters. This is used in an iterative algorithm for minimising the difference vector between the current estimate of

appearance and a target image. A reasonable initial starting position of the model is needed. To find the best estimate of appearance parameters, a multi-resolution implementation is used, iterating over different levels. Both ASM and AAM have been used for the detection of various objects, but have been established in the field of face reconstruction.

The 3DMM, as described in detail in the previous chapter (Chap. 2), is formed by a statistical shape model of 3D vertex positions (in contrast to the 2D positions of the AAM) and a shape-free texture model.

3.1.1 Further Development of Active Shape Models and Active Appearance Models

Based on Active Shape Models and Active Appearance Models, different approaches, both in 2D and in 3D, have been developed in recent years.

Langs et al. [LPD+06] present an Active Feature Models, related to AAMs. Contrary to AAMs they do not model the entire texture, but represent the image texture by means of local descriptors. To localize landmarks in images they use these local texture features together with a statistical shape model, which captures the shape variation of landmark configurations. Reliable shape variations have been analyzed with PCA. According to trained relationship between displacement and descriptor responses, pose and shape parameters of the model are updated during search.

The Boosted Appearance Model for face alignment of Liu [Liu07] also works on the basis of AAMs. It consists of a set of trained weak classifiers based on local Haar-like rectangular features, considered as an appearance model. The weak classifiers for the different parts of the face are trained with GentleBoost and the score from the strong classifier is used as distance metric, in contrast to AAMs, where the mean squared error is used. Because of the local rectangular features it is more likely to be robust to partial occlusion.

Using the 3D-AAM from Xiao et al. [XBMK04] as baseline, Ramnath et al. [RKX+08] present a multi-view algorithm for fitting and construction. First they use a naive algorithm, fitting 2D+3DAAM independently to each image, improving this by using the fact that the 3D shape of the face is the same in all views. The 2D shape parameters in each image have to be coupled in a physically consistent manner through the single set of 3D shape parameters. Due to the fact that the multiple images are captured simultaneously with arbitrary located/rotated cameras a scaled orthographic imaging model can be used. While fitting, scaled orthographic camera matrices are computed. Based on this they also investigated the use of human faces as a (non-rigid) calibration grid and outlined a non-rigid motion-stereo algorithm for calibrated multi-view 3D AAM construction.

Morency et al. [MWM08] presented a generalized adaptive view-based appearance model (GAVAM) for head pose estimation and tracking in monocular images. The camera parameters are known and there are six DOFs. The two main components of their work are the view-based appearance model and a series of pose-change measurements. Using a base frame set out of particular key-frames the relative transformation between the current frame and all base frames

is computed and based on this the model is adapted and the current pose gets estimated. The set of key-frames is updated every time a frame different from any other view appears, to ensure a constant tessellation of the pose space in the view-based model.

Saraghi et al. [SLC08] improved deformable face fitting (AAMs) with soft correspondence constraints. To resist two drawbacks of deformable face fitting (for every parameter update an assumption about the correlation of shape and appearance has to be made and also the generalisation is restricted) they only enforce soft correspondences between model and image and use a mixed inverse-compositional-forward-additive parameter update scheme for fitting. The correspondences (working like optical flow) get shape and appearance to be independent, so that for fitting the intrinsic parameters, the extrinsic parameters can be assumed as fixed and only a quadratic problem has to be solved.

To circumvent two other drawbacks of AAMs (prone to local minima and if indeed only few of them lead to reasonable results) Nguyen and De la Torre [NlT08] learn a cost function, with local minima only occurring at places representing correct fitting parameters. They implemented this by enforcing constraints on the gradients of the cost function at the desired position and in its neighbourhood.

Face alignment using a deformable model based on ASM is done by Huang et al. [HLM07]. Unlike the original ASM, here the model consists of parts, which can be aligned separately and combined to a constrained GaussianProcessLatent Variable Model. To ensure the interrelationship between the parts a probability distribution function (PDF) is used, which has to be maximized. The whole error function consists of the PDF of the component parameters, constrained by component shape priors and the local search results. To reach robustness against illumination they use a low pass Gaussian filter and a modified canny edge detector.

3.1.2 Further Development of the 3D Morphable Model

Based on the 3DMM of Blanz and Vetter [BV99] further work has been done over the last years. Romdhani and Vetter [RV03] show how the fitting of a 3DMM can be modified by using an extended 2D algorithm (ICIA - Inverse Compositional Image Alignment), which results in an efficient, robust and accurate but not automatic fitting algorithm. Huang et al. [HBH02, HHB03], Weyrauch et al. [WHHB04], and Heisele and Blanz [HB06b] combined 3DMM and component-based recognition. The 3DMM is fit to multiple (first two and later three) images of a face to generate a 3D head model for each person in face database. Rendering the model under varying pose and illumination conditions they get a large set of synthetic images to train a component-based face recognition on. In a Multi-Features Fitting Algorithm for the 3DMM, Romdhani and Vetter [RV05] use a cost function that adds color difference, edge information and the presence of specular highlights in each pixel. The algorithm is related to our algorithm proposed in Chap. 6.4 [BB10] because multiple features are used and a tradeoff is found for the match of each feature plus a prior probability term. However, the algorithm proposed in this thesis uses features that are derived from facial appearance, and it updates these features during the optimization. Pierrard and Vetter [PV07] localize remarkable irregularities (especially moles) on the face skin for

recognition using the 3DMM. After fitting the 3DMM to the input image some regions, probably containing only skin, are taken for initialization of skin segmentation. For that they use two different methods, one with a simple threshold for maximal difference of grey values and the other using k-nearest neighbour, to detect local unevenness. After segmentation the localization itself is done by a two-stage template matching, first detecting candidates of moles, which then have to be analyzed more precisely using templates with different mole-sizes. By mapping the reconstructed head to the reference head the positions of the moles on the face is detected. These universal coordinates then are used for recognition.

Keller et al. [KKV07] investigated the robustness of reconstructing the 3D shape using only the occluding contours (silhouette and inner contours), and how much information is involved in one single contour image. They found out that the resulting head is almost random. Looking at the model coefficients, the distance between the resulting head and the ground truth is similar to the distance between the ground truth and the mean shape. This investigation shows that contours are not strong constraints for the shape, even when using a holistic model, meaning that contour matching is an important part of the whole system but not useful alone. Using the results of Romdhani and Vetter [RV05] as base, Amberg et al. [ARF+07] reconstruct 3D shape and pose from multiple images taken simultaneously by combining monocular and stereo cues. They replace lighting and albedo estimation of the monocular method with the use of stereo information, a colour difference cost, to get higher accuracy and robustness against model albedo effects like make-up, facial hair, moles and cast shadows. Additionally, combining a soft edge detection scheme with a derivative based optimisation algorithm, the visible contour is fit to multiple images simultaneously. For edge detection they use gradient magnitude thresholding with non-maximum suppression and a morphable contour model generated out of the detected contour points of the 3DMM. Integrating the information over a range of edge thresholds they get minima at edges over the full range of thresholds, discerning between strong and weak edges, for their cost surface. To perform expression invariant face recognition Amberg et al. [AKV08] use 3D scans and a model where identity and expression can be treated separately. For recognition the resulting model is normalized by removing the components for pose and expression. The algorithm needs only the tip of the nose as a landmark and uses a kind of non-rigid iterative closest point algorithm (ICP, [BM92]) for optimization. In 2009 Paysan et al. [PKA+09] improved the former models and created the Basel Face Model (BFM), which is publicly accessible now. They reached a higher accuracy in shape and texture due to better scans and reduced correspondence-related artifacts by improving the registration algorithm.

3.1.3 Linear 3D Models

Deploying a combination of symmetric shape from shading by Zhao and Chellappa [ZC99, ZC00], and a statistical approach for facial shape reconstruction by Atick et al. [AGR97], Dovgard and Basri [DB04] recover the 3D structure of faces out of frontal view images, using a linear combination of basis shapes. Zhao and Chellappa [ZC99, ZC00] used symmetry constraints on depth and albedo to obtain another albedo-free brightness constraint and assumed either constant or piece-wise constant albedo. Atick et al. [AGR97] also assumed constant facial albedo.

They used statistical knowledge about shapes and albedos of human faces in general (as well as Blanz and Vetter [BV99]) and enforced a linear constraint on the shape. However the approach of Dovgard and Basri [DB04] accounts for varying facial albedo, but requires known directional illumination and a Lambertian assumption about the face. They provide a closed-form solution solving a single least-squares system of equations, combining constraints for albedo-free brightness and class linearity.

For recovering facial shape out of frontal views Smith and Hancock [SH05] use a needle map, a statistical model which captures variations in surface normal direction. They begin with a set of needle maps and compute the mean field of surface normals to reconstruct a deformable model over the set of surface normals by applying the point distribution model of Cootes et al. [CTCG92]. The fit of the model to intensity data is done by using constraints on the surface normal direction provided by Lambert's law, image irradiance constraints and the azimuthal equidistant projection.

Kemelmacher and Basri [KB06] recover 3D shape and albedo from a single image and a single 3D reference model of a different individual by example using a database of many faces with point correspondences. They solve for lighting, depth and albedo separately, one after another, not simultaneously. Lighting is recovered by searching for the best coefficients that fit the reference model to the image. Following, depth is estimated applying a shape-from-shading approach using the recovered lighting. Both recovered lighting and depth are used for albedo estimation. To smooth the difference between reference and sought face, regularization terms seek for solutions 'near' the rough shape and albedo of the reference model. But the approach is restricted to frontal faces with Lambertian reflectance, roughly aligned to the model.

Another example-based approach for 3D reconstruction is presented by Hassner and Basri [HB06a]. It also has some restrictions: query objects have to be separated from the background and aligned to a preselected image from the database manually, to have the right scaling and inter-plane rotation as starting values. Depth values are determined by comparing the intensity values of the input image with the data base using a window around each pixel. But semantic consistency has to be established by global optimization. Additionally a Laplacian pyramid is used. To cope with different pose and perspective conditions the data base can be updated. Every example head is a 3D object, which can be rendered in different poses to get new views for comparison. Reconstruction of the back of the head is done by a second depth map, which contains the relation between front and back of a head.

Park and Jain [PJ06] use a reduced model of the average head of Blanz and Vetter [BV99] to reconstruct a 3D face from stereo images using linear transformation based on TPS (thin plane spline from Bookstein 1989 [Boo89]) and test its use for recognition. They reconstruct a sparse set of facial landmark points, apply them for coarse alignment, and perform TPS using only 72 points of the model to get the 3D face reconstruction. For texture mapping one of the images has to contain a frontal view. What kind of facial feature points are used, has been selected manually, and also the camera positions have been known. This knowledge together with the point correspondences and a simple equation is used to create a 3D shape of facial feature points.

Bregler et al. [BHB00] recover a Non-Rigid 3D Shape from Image Streams. Like in the 3DMM

they assume a 3D form consisting of K basis shapes. In their approach one special shape is the linear combination of points detected in every frame of the image sequence. They get a tracking matrix using orthographic projection onto the 2D image plane, which can be factorized to get 'pose and face' and the basis shapes out of it. Using SVD 'pose and face' values can be used to determine the weights for the basis shapes, like the coefficients in the 3DMM. Effectively they learn a point-'model' out of an image sequence, describing the object in the scene and its properties (like the principle components in the 3DMM).

An automated algorithm for aligning a 3D face model to a single image has been proposed by Gu and Kanade [GK06]. Based on a Morphable Model of 3D shape and view-based patches located at 190 points, the model is iteratively aligned with the features in the image using an EM algorithm. The Morphable Model, the feature-based approach and the usage of a pose estimate are similar to the system proposed in this thesis (cf. Chap. 5, [BKK+08]). However, they use different feature detection algorithms and a different optimization strategy and for the final reconstruction, they fit all model vertices to the image in an analysis-by-synthesis loop that optimizes all facial details and compensates for lighting and other imaging parameters.

3.1.4 Reconstruction from Video

For reconstruction from video, our system, proposed in this thesis (cf. Chap. 5, [BKK+08]), selects a single, suitable frame from a monocular video automatically, and performs model-based reconstruction from this frame. This is in contrast to previous work in model-based shape reconstruction from monocular video, which involved an analysis of multiple frames, such as model-driven bundle-adjustment (Fua [Fua99]), least-squares reconstruction from multiple successive frames (Dimitrijevic et al. [DIF04]), structure-from-motion with subsequent refinement by a deformable face model (Chowdhury and Chellappa [CC03]), nonrigid structure-from-motion with intrinsic model constraints (Brand [Bra01]) and feature tracking and factorization of the tracking matrix for non-rigid shape estimation (Bregler et al. [BHB00]). Zhang et al. [ZLA+04] presented an algorithm that involves tracking, model fitting and multiple-view bundle adjustment. Many of these algorithms require manual interaction such as a number of mouse clicks.

Unlike previous model-based algorithms for 3D face reconstruction from single images (Blanz and Vetter [BV99, BV03] and Lee et al. [LPMM05]), our combined algorithm no longer requires manual rigid prealignment of the 3D model or manual labeling of features on the 2D image. Due to our automated face and feature detection components, these manual interventions are not necessary for our system. Xiao et al. [XBMK04] presented a combination of Active Appearance Models and 3D Morphable Models that tracks features in realtime in videos, and reconstructs a face mesh for each frame. This system is very impressive, but so far the authors have only used a low-resolution face mesh that does not generate photo-realistic face reconstructions.

Xin et al. [XWT+05] recovered 3D face shape from a video containing a face rotating from frontal view to profile view. Based on image sequence segmentation, they estimate 2D features, head poses and underlying 3D face shape, using a morphable model resembling the one of Blanz

and Vetter [BV99]. In contrast, in our system, we use features for initialisation only, and then fit to grey-levels in the image for more detailed reconstruction.

3.2 Detection of Faces and Facial Features

For initialization, the 3DMM needs the positions of facial features. This chapter reviews related work on both, face and facial feature detection, because in most cases face detection precedes feature detection.

3.2.1 Faces

The best-known methods for face detection indeed are the ones by Schneiderman and Kanade [SK00] and AdaBoost by Viola and Jones [VJ01]. AdaBoost also often is used for facial feature detection, which is described more precisely in Chap. 4.

Schneiderman and Kanade [SK00] statistically observe object occurrence and also object absence employing histogram products of wavelet coefficients and their positions on the object. They are able to detect faces over a wide range of rotation angles using a dyadic strategy. They train different detectors for some distinct object orientations and model the rest of the variations statistically between these detectors.

In the 90s Lades et al. [LVB$^+$93] used the Dynamic Link Architecture to detect faces in grey level images. The idea was the use of synaptic plasticity to be able to group sets of neurons into higher symbolic units. Yet there are two limitations. First the system does not have invariance to orientation to the same degree to which it has translation invariance and second it will usually fail to find the globally optimal solution. Object detection is done by elastic graph matching, taking images as attributed graphs in the image domain and vertices as nodes with attributes attached. The attributes are activity vectors of local feature detectors (GaborWavelets), jets. Binding nodes with similar jets together separates the figure from the ground. After identifying and activating nodes and links in the model domain, the many connections are boilt down to a consistent one-to-one mapping and the match with the lowest cost in both, edge and vertex term, is identified as the model recognized.

Rama et al. [RTBA07] presented a one-stage face detector using a non-linear fuzzy integral operator and extended it in [RTN07] to a four-stage detector. They use Haar features and train the system using a neural network to get the fuzzy measures. Fuzzy integrals are generalizations of integral operators that include non-linear operations on the dataset and map the input set of features to an unique scalar. So a threshold can mark the borderline between face and non-face. Compared with AdaBoost, using the same features, previously selected by AdaBoost, for training of the first stage fuzzy integral classifier their approach performs quite well.

By decoupling feature selection and forming ensemble classifiers, Wu et al. [WBMR08] enhanced cascade detection. They use precomputing for feature selection, so the weak classifiers

are trained only once, and then are stored and reused. Under the assumption that appropriate features have already been selected the asymmetric learning goal is handled as a well-defined constrained optimization problem and solved with the Linear Asymmetric Classifier (LAC) approach. But some limitations are given because of the linearity and the high false positive rate.

A survey about detecting faces in images was presented by Yang et al. [YKA02] in 2002. They categorise and evaluate some algorithms in view of their robustness against variations in lighting, orientation, pose, partial occlusion, facial expressions, glasses, hair on the face and hairstyle. They perceive four categories of algorithms:

- knowledge based top-down algorithms using facial features, texture, skin colour or multiple features, mostly used for localisation
- feature invariant bottom-up algorithms using facial features, texture, skin colour or multiple features, mostly used for localisation
- template matching algorithms using predefined standard patterns for localisation and recognition
- the appearance-based algorithms using statistical analysis and machine learning to compare learned models for recognition

They consider different modes of face detection as the localization of one face for passing it on to a recognition system and the detection of all present faces in one messy black and white scene or in coloured images or in video sequences.

3.2.2 Facial Features

Detection of facial features and its integration into face recognition systems have been studied extensively in recent years. Still, robust feature detection in difficult imaging conditions remains to be a challenge.

In [FK07] by Forczmanski and Kukharev one can find a comparative analysis of simple facial features extractors. They do not concentrate on feature points like corners of the eyes or the tip of the nose, but on physical features and mathematical features. Features are stored as feature vectors, which can be used for face recognition. For physical features, luminance values are used. After image scaling, a feature vector is filled with either all concatenated pixels one after another or just by a small number of points from the face. Mathematical features can be the spectrum of the face image or a pixel intensity histogram.

As AdaBoost of Viola and Jones [VJ01] is a well-known approach also for facial feature detection, Erukhimov and Lee [EL08] use it to first detect candidates for eyes, nose and lips separately. From the candidates, the combination with highest log-likelihood is chosen. Many approaches use coarse-to-fine strategies: Nguyen et al. [NPlTF08] detect the head using AdaBoost and get a first guess of the iris position using linear regression. At the next step a weighted support vector

machine (SVM), using only a small number of pixels of the whole search area, refines the iris position. Zuo and de With [ZdW05] use a cascade of global deformation, texture fitting and feature refinement to refine eye, mouth, nose and eyebrow positions. Tang et al. [TWTP] use a hierarchical face model composed of a coarse, global AAM and local, detailed AAMs for each feature for refinement. This restricts the influence of noise to the features directly nearby, and prevents it from affecting the rest of the face. Wimmer et al. [WMR08] and Ardizzone et al. [ACM09] both use a prior distribution map analyzing AdaBoost face detection output as a starting condition, and refine the feature positions using color values and a decision tree classifier [WMR08], or using a HarrisCornerDetector and a SVM to classify whether the detected corners belong to a feature or not [ACM09]. Kozakaya et al. [KSYY08] find facial features indirectly by using templates for parts of the face in connection with vectors that point from these regions to the positions of the features. The final feature positions are weighted combinations of the vectors.

Instead of refining the positions as in a coarse-to-fine approach, Oh et al. [OKK+05] combine conventional algorithms in a sequence to get better initial values for characteristic points: face detection by skin-color and luminance constraints, eye detection by TM and symmetry enforcement, mouth and eyebrow detection both using luminance and geometry constraints. Other combined approaches have been proposed by Kim and Dahyot [KD08] who classify SURF local descriptors with SVMs: one SVM to decide whether they belong to the face or not, followed by special SVMs for each feature. Celiktutan et al. [CAS] combine four feature detectors (DCT, GaborWavelets, ICA, non-negative Matrix Factorization) on images at a reduced resolution. SVM is performed to get the most reliable positions (highest SVM scores) for each feature, and a graph based post-processing method is used to pick the best combination of feature positions. Refinement of the feature positions at the end is done using DCT again on full resolution.

A number of algorithms introduce local features to Active Shape Models (ASM) and Active Appearance Models (AAM): Milborrow and Nicolls [MN08] extend the ASM by fitting more landmarks, using 2D-templates at some landmark positions and relaxing the shape model wherever it is advantageous. To improve the result further, they use the first alignment as start value for a second fitting with the new ASM. Zuo and de With [ZdW04] combine ASM and Haar wavelets. Cristinacce and Cootes [CC06] use a similar approach to the one proposed in Chap.6.4, yet their 2D AAM model is designed for frontal or nearly frontal views of faces only. They form facial feature detectors from an Active Appearance Model and update them in an iterative search algorithm. In each iteration, they find the feature positions with a plausible 2D configuration (high prior probability) and, at the same time, a high feature detector output. In contrast, our system uses a 3D model that contains additional parameters for pose and illumination, use different methods to create feature detectors and to fit the model, and they integrate the feature point criterion into a cost function that includes overall image difference for a global analysis-by-synthesis.

The first combination of feature detection with 3D morphable models (3DMM) was presented by Huang et al. [HHB03] who created local feature detectors for face recognition from a 3DMM. Unlike the system proposed in this thesis, they first reconstructed a face from a gallery image, created virtual images using the 3DMM and then relied on SVM-based local classifiers for recognition. Gu and Kanade [GK06] presented a patch based approach that is related to our work because it combines local feature detectors and a 3D shape model. In contrast to our algorithm,

however, the feature detectors are trained prior to fitting, and the model fitting minimizes the 2D distances between image points with maximum response of the feature detectors and the corresponding model points. Romdhani and Vetter [RV07] first identify all potential feature points in the image by using SIFT as a criterion for saliency, then reject those that are similar to none of the points in the appearance model, and subsequently find the configuration of features in the image and the mapping to features of the 3DMM that has a maximum likelihood. The resulting feature locations can be used to initialize a 3DMM fitting procedure. Our system (cf. Chap. 4.2 and Chap. 5) uses SVM for detecting faces, estimating pose angles and finding facial features. From a number of nose point candidates, a model-based criterion selects the most plausible position. Then, these data are used to initialize a 3DMM fitting algorithm and compute 3D reconstructions. This may be used in a similar general approach, but with an increased robustness to imprecise initial feature positions.

A similar approach to our facial feature detection algorithm has been proposed in [AZ05] by Arandjelovic and Zisserman, where they restricted the search space of an SVM classifier based on joint configurations. But, unlike Arandjelovic and Zisserman [AZ05], we restrict the search space based on the regression based detection and use both the prior on joint configuration and the approximation of likelihood provided by the SVM to choose the best configuration.

Heisele et al. [HSP+02] used synthetic images to train SVM-based viewpoint-independent feature detectors. They learn discriminative components of objects with SVM classifiers automatically and combine the component-based (face) classifiers to yield a hierarchical SVM classifier.

For more related work in the feature detection literature, which involves SVM-based methods of Osuna et al. and Romdhani et al. [OFG97, RTSB04], we refer the readers to the excellent survey of Yang et al. [YKA02] again.

Eyes

For eyes there have been presented some distinct detectors, most of them using some kind of template matching.

Reinders [Rei97] applied template matching using an automatic codebook generation scheme for eye tracking. To be robust to changes in appearance even applying template matching, he uses a codebook of eye templates. Initialization is needed to create the first templates out of the first input frame, but over tracking iterations the codebook is updated. If the similarity measure gets below a certain threshold (for all present templates) this 'new' template, using the estimated locations in the previous frame as estimator for scale and orientation, is inserted in the codebook.

Using deformable templates Funabiki et al. [FIHO06] detect eye contours. Their template contains three quadratic functions for the eye perimeter and a circle for the pupil stored in ten parameters. Working on hues and color values of the face image they identify the region around the eyes. But template optimization is done on an edge image of the eye region build with a canny edge detector. As a sequential update of parameters guarantees the convergence to a local minimum, they also used a hill-climbing procedure to avoid the convergence to poor local minima at

least. But with a test set consisting of only eight Japanese students, a substantial statement about its robustness is not possible.

Wang et al. [WGJW05] use a hierarchical principle to detect eyes in (nearly) frontal faces. On faces detected with AdaBoost they detect the eyes in the upper half of the face by the average of multiple detection results using recursive nonparametric discriminant analysis (RNDA). Discriminant features to characterize eye patterns are learned and out of their distribution probabilistic classifiers to separate eyes/non-eyes are deduced. Because of the same distribution of weight in both discriminant analysis and AdaBoost they can be associated together and multiple classifiers can be combined in AdaBoost.

Fuzzy template matching and feature-parameter-based judgement is used by Li et al. [LlQjW01] for eye detection in images containing frontal upright faces with open eyes. But they do not locate and extract the boundary of eyes, but rather judge whether the image in a certain window contains an eye or not and if it contains an eye the center of the window is set as the location of eye. The fuzzy template is based on a piecewise boundary with adjacent segments at the eyelid and three other regions for the outside of the eye, the inside of the eye and the eye-tail. Filling each segment with the darkest intensity value within this segment, based on the overlaying template, they get a new image to work on. Performing edge detection and multithresholding on this leads to five feature parameters to decide whether eye or non-eye.

Peng et al. [PCRK05] present an eye detection algorithm on gray intensity images, where the faces are already detected and the eyes can be seen. They combine two existing techniques: a feature based method and a template based method. The feature based method roughly identifies two eye regions on the already detected face and the template based method detects the iris centers in this rough regions using a template size according to the estimated width of the face.

3.3 Segmentation of Images

In this section we review approaches in the scope of image segmentation. This is related to our work on automated occlusion detection, because often segmentation is used to distinguish between foreground and background. Out of the bounds between these, borders and contours are detected, which is also related to our work on contour adaption.

For segmentation and boundary detection Sharon et al. [SBB01a] used multiscale intensity measurements. In a top-down system they combine multiscale measurements of intensity contrast, texture differences and boundary integrity based on segmentation by weighted aggregation (SWA). Using this as basis, Galun et al. [GSBB03] present a bottom-up system combined with a top-down cleaning process, which combines structural characteristics of texture elements with filter responses, which influence each other. The process adaptively identifies the form of texture elements and statistically characterizes them based on features such as size, aspect ratio, orientation and brightness. Filter responses also get statistically observed.

One of the most noted approaches for segmentation of the last years is GrabCut invented by Rother et al. [RKB04]. They use iterated graph cuts for foreground extraction. It is an exten-

sion of the graph-cut approach using Gaussian Mixture Models (GMMs) instead of gray value histograms. Only a part of the background has to be marked to initialize the system. Refinement of the opacities for segmentation is done iteratively, allocating all pixels of the undefined part of the image with their GMM component, learning parameters from the image data and performing global optimization with energy minimization using minimumCut as the graph-cut approach. Romdhani et al. [RPV06] used it also to automatically produce an outlier mask to improve 3D face reconstruction using a 3DMM. But Pierrard and Vetter [PV07] rather used thresholding instead of GrabCut, because GrabCut often did not exceed shadows, detected the chin line as a dividing line and generously classified border areas as non-face areas.

An outlier mask for face reconstruction is also generated by Jia and Martinez [JM08]. They model the skin color using a mixture of Gaussians and improve the result by the use of morphological operators (erosion and dilation) to eliminate isolated segments and fill eroded local areas.

Lee et al. [LASG08] use Markov Random Field Models for hair and face segmentation. Therefore they learn six probability masks for different hair styles and the distribution of hair color and skin color with Gaussian-Mixture-Models in five clusters. Areas with high probability are compared with the distributions and the models are updated online. To find the optimal segmentation they use Graph-Cut or LBP (Loopy Belief Propagation) and iterate model update and segmentation until convergence.

Another application area of the segmentation approach, besides the detection of occlusions, is eyeglass detection and removal. Wu et al. [WLS+02] and Xiao and Yan [XY04] both worked on this aspect. The system of Wu et al. [WLS+02] works on frontal images and is composed of three modules, an eyeglass classifier for recognition, a Morkov chain Monte Carlo approach to locate the glasses and an example based approach to synthesize an image with eyeglasses removed. The eyeglass classifier concentrates on local orientations and gradients. PCA is done on glasses samples and on non-glasses samples respectively. Reconstructing the input image with both eigenspaces, the lower reconstruction error gives the right class. For localization they use an ASM with 15 key points on the glasses frame, maximizing the product of prior and likelihood with Markov chain Monte Carlo. The prior is learned by PCA and the likelihood is learned in a non-parametric way using the response of local edge detectors as features for training. Performing PCA on pairs of corresponding glasses/non-glasses samples, enables to reconstruct the glasses with its eigenspace and to reconstruct the corresponding non-glasses results automatically using the same coefficients for reconstruction. This estimate then is put into the original image and the glasses are removed. But the results are blurry at the eye-region. Xiao and Yan [XY04] also work on frontal images but use a 'face model' based on Delaunay triangulation. They use the binarized image intensity gradient to let the layout of eyes and glasses frames emerge. Out of the 'foreground' pixels they create a face image model using Delaunay triangulation. Related triangles are identified and clustered to groups. From symmetry, size and relative position of any two of these groups and their surrounding areas eyes and glasses frames are identified.

In the following we focus on specific aspects of the above review. Therefore different essential feature detection algorithms are presented in more detail.

Chapter 4

A Survey of Feature Detection Algorithms

For initialization of the fitting process, the positions of facial features are needed. In this chapter, four essential methods for feature detection are presented in more detail: AdaBoost (by Viola and Jones), SVMs for facial feature detection, template matching and SIFT. The results are the basis for our feature detection approach presented in the next chapter.

4.1 AdaBoost

In 2001, Viola and Jones [VJ01] presented their object detection approach *using a boosted cascade of simple features*, today known as AdaBoost, despite a modification of the primary AdaBoost of Freund and Schapire [FS95] makes up only a part of the whole approach.

It consists of three main ideas:

1. Fast feature evaluation of rectangular features using a new image representation, the *integral image*.
2. Association of the most convincing features to strong classifiers by a AdaBoost based learning approach.
3. Combination of classifiers in a *cascade*.

They defined three kinds of rectangular Haar-like features (c.f. Fig. 4.1):

- two-rectangle features
- three-rectangle features
- four-rectangle features

which can be used in different scaling and at different positions of their 24x24 pixel sized detector, amounting to more than 180,000 different potential features. To be able to compute the

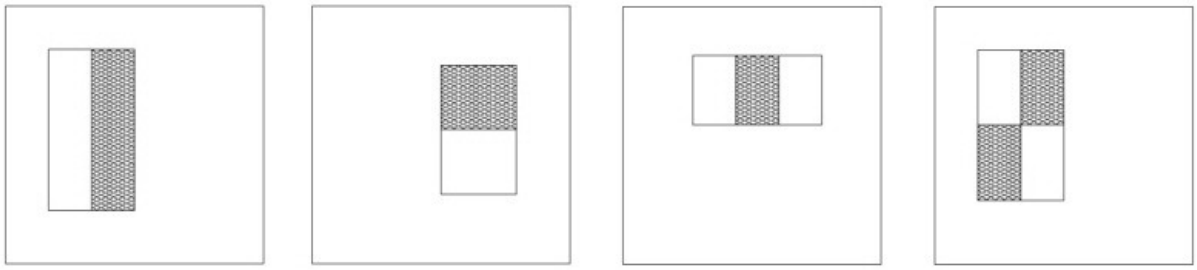


Figure 4.1: **Basic features of the Viola-Jones approach:** From left to right the vertical and horizontal two-rectangle features, one three-rectangle feature and the four-rectangle feature are shown. The images are taken from [VJ01].

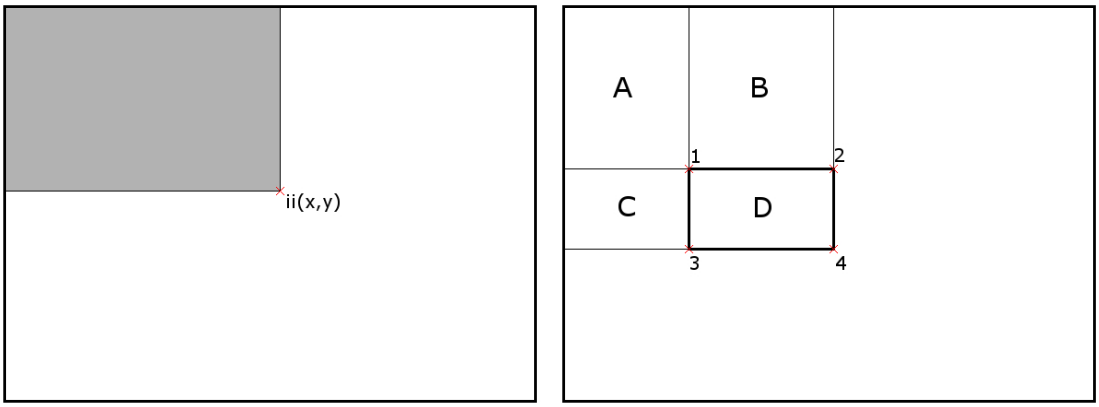


Figure 4.2: **Integral image:** On the left the grey zone marks the pixels whose values are summed up in $ii(x,y)$. On the right one can see: To get the sum of pixels within D only the four array references at the marked corners are needed. Position S_1 ($ii(x-1,y-1)$) contains the sum of pixels in area A. Position S_2 ($ii(x+w-1,y-1)$) contains the sum of pixels in $A+B$. Position S_3 ($ii(x-1,y+h-1)$) contains the sum of pixels in $A+C$. Position S_4 ($ii(x+w-1,y+h-1)$) contains the sum of pixels in $A+B+C+D$. So D has to be computed as $S_4+S_1-(S_2+S_3)$.

features at any scale or location in constant time, they invented an intermediate image representation (closely related to 'summed area table'). This so called integral image can be computed in one pass over the original image and contains at each position (x,y) the sum of pixels above and to the left of the position:

cumulative row sum: $s(x,y) = s(x,y-1) + i(x,y), \quad s(x,-1) = 0$

integral image: $ii(x,y) = ii(x-1,y) + s(x,y), \quad ii(-1,x) = 0$

and any rectangle sum can be computed in four array references (cf. Fig. 4.2) or in six for the two-rectangle features, in eight for the three-rectangle features, and nine for the four-rectangle features because of the involved adjacent rectangular sums.

The total number of 180,000 features is to large to form an effective and efficient classifier. The

majority of features has to be excluded and the critical ones have to be identified. To accomplish this Viola and Jones used a simple modification of the AdaBoost approach of Freund and Schapire [FS95]. They constrained the weak learner to concentrate only on single features per weak classifier and select the feature best separating positive and negative examples, choosing a threshold to yield a minimum number of examples misclassified. At the primary AdaBoost the threshold should yield to a low error rate.

The AdaBoost approach works as follows:

1. Start conditions: Training examples $I_1, \ldots, I_n$ are given, l positive ones with the flag $y_i = 1$ and m negative ones with the flag $y_i = 0$ ($m + l = n$).

2. Initialization: Initialize the weights $w_{1,i}$ using the given flags y_i:

$$w_{1,i} = \begin{cases} \frac{1}{2l} & \text{if } y_i = 1 \\ \frac{1}{2m} & \text{if } y_i = 0 \end{cases} \tag{4.1}$$

   The weights do not have to be normalized, they are already constituting a probability distribution:

$$\sum_{i=1}^{n} w_{1,i} = l \cdot \frac{1}{2l} + m \cdot \frac{1}{2m} = 1 \quad \wedge \quad w_{1,i} \geq 0 \quad \forall i \in [1,n] \tag{4.2}$$

3. For $t = 1, \ldots, T$ (find the T best features / weak classifiers to combine):

   (a) Weak classifiers: Train a classifier c_j for each single feature j and evaluate the weighted error

$$e_j = \sum_{i=1}^{n} w_{t,i} \left| c_j(I_i) - y_i \right|. \tag{4.3}$$

   The lowest error

$$e_t = \min_j e_j \tag{4.4}$$

   marks the best weak classifier c_j.

   (b) Weights update:

$$w_{t+1,i} = w_{t,i} \cdot \beta_t^{1-|c_t(I_i)-y_i|} \quad \text{with } \beta_t = \frac{e_t}{1-e_t} \tag{4.5}$$

   (c) Normalization: The weights have to be normalized to build a probability distribution again:

$$w_{t+1,i} = \frac{w_{t+1,i}}{\sum_{j=1}^{n} w_{t+1,j}} \tag{4.6}$$

Figure 4.3: **First features selected by AdaBoost:** On the left a 24x24 positive example, in the middle the first feature and on the right the second feature are shown. The figure is taken from [VJ01].

4. Strong classifier: Combine the T weak classifiers to one strong classifier

$$C(I) = \begin{cases} 1 & \text{if } \sum_{t=1}^{T} \alpha_t c_t(I) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \quad \text{with} \quad \alpha_t = \log \frac{1}{\beta_t} \\ 0 & \text{else} \end{cases} . \tag{4.7}$$

To get an idea what the selected features are like, in Fig. 4.3 the first two features selected by their approach can be found, the first modelling the fact the region of the eyes often being darker than nose and cheeks, and the second modelling the fact the eyes being darker than the bridge of the nose.

At last they combine multiple boosted classifiers to a cascade. Each stage of the cascade is a strong classifier trained with their modified AdaBoost. To focus the attention on promising regions of the image, subsequent classifiers are trained using those examples, which pass through all the previous stages. The following classifiers face a more difficult task than the previous and each stage reduces the false positive rate and decreases the true positives rate. Features are added to a stage until target detection and false positive rate are met, and stages are added to the cascade until the overall target for the stages is met. To minimize the effect of different lighting conditions, variance normalization is performed on the feature values during training and detection. Whereas the variance can be computed very easily, too, using a second integral image of the original image squared to get the sum of squared pixels.

The detection process, using the cascade of boosted classifiers, works like a degenerated decision tree rejecting the majority of subwindows with the first classifiers, to use more complicated classifiers later to achieve low false positive rates. A positive result of a classifier triggers the evaluation with the next one (the next stage) and a negative result causes an immediate rejection

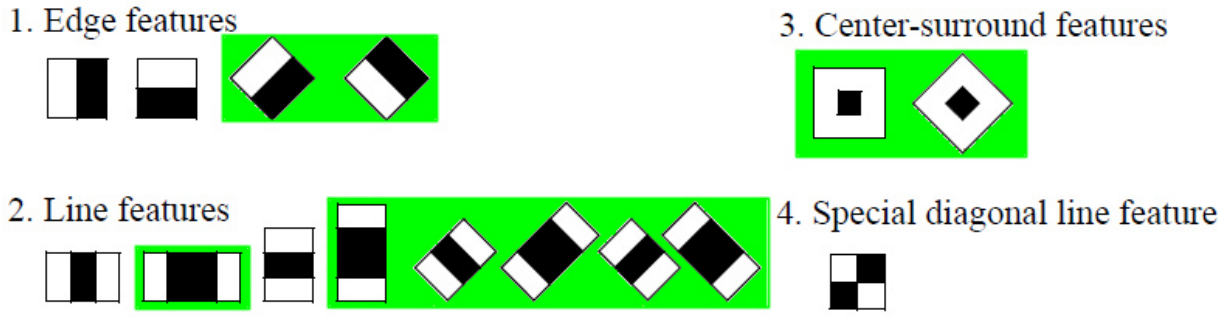


Figure 4.4: **Extended Features:** Here basic and new features are shown. The images are taken from [LKP03]. The new features are marked with a green background. The two-rectangle features have been rotated (1.) and new center-surround features have been added (3.). The three-rectangle features have been enlarged in the middle and also have been rotated (2.).

of the subwindow. The final detector than scans across the query image at multiple scales and locations. Because of being insensitive to small changes in translation and scale, more than one location may be detected in a given face. To overcome this, the result has to be post-processed combining overlapping detections into a single one building the average of bounds.

The detector presented in their paper, had been trained with 9832 positive examples and 10,000 negative examples, all of a resolution of 24x24 pixels. The face images had to be manually labeled, scaled and aligned. The non-face subwindows were randomly taken out of 9544 images not containing faces, at each stage in any case taking the false positives of the previous stage, replenishing to 10,000 examples with random subwindows.

The Open Computer Vision Library [Ope06] contains a full implementation of the ViolaJones approach extended by Lienhart et al. [LKP03]. The publicly available package contains a trained face detector and a training and detection system implemented by Lienhart.

Lienhart et al. [LKP03] improved the object detection performance, especially for objects exhibiting diagonal structures, extending the over-complete set of Haar-like features by a set of 45° rotated ones. The basic features had been modified in their dimensions and new center-surround features (following the 'early features of the human visual pathway') had been added to extend the domain knowledge (cf. Fig. 4.4). Although frontal faces exhibit little diagonal structures, their extension yielded an about 10% lower false alarm rate at comparable hit rates. To be able to evaluate the rotated features as fast as the axis aligned ones, only one auxiliary adjusted integral image - the so called Rotated Summed Area Table - was needed. Using this to get the sum of pixels within a rectangle, also only four array references are needed (cf. Fig. 4.5).

Comparing different boosting variants (Discrete AdaBoost, Real AdaBoost and Gentle AdaBoost [FS95]), using only the basic feature set and stumps as weak classifiers, the authors came to the result that Gentle AdaBoost outperforms the others with respect to detection accuracy and computational complexity. They also experimented with other weak classifiers and different sizes of input patterns. Based on this, they found 20x20 to be the optimal input pattern size and

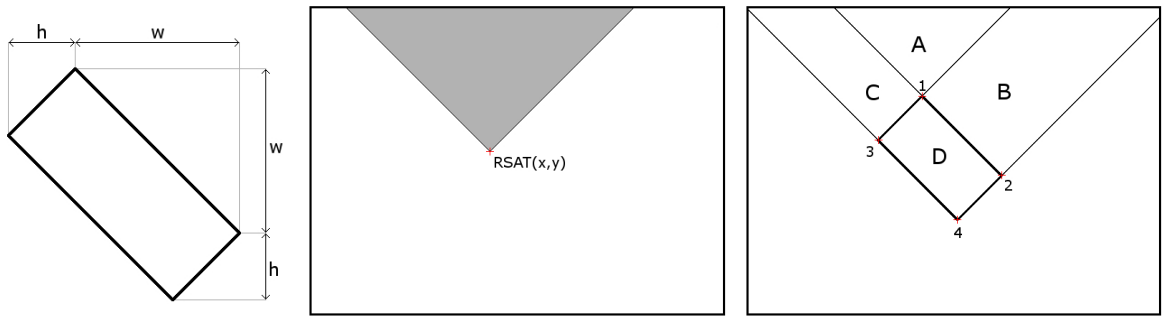


Figure 4.5: **Rotated Summed Area Table:** On the left it is shown how width and height are measured for the rotated features. In the middle the grey zone marks the pixels whose values are summed up in $RSAT(x,y)$. On the right one can see: Here D has to be computed as $S_4 + S_1 - (S_2 + S_3)$, too, with $S_1 = RSAT(x, y-1)$, $S_2 = RSAT(x+w, y+w-1)$, $S_3 = RSAT(x-h, y+h-1)$ and $S_4 = RSAT(x-h+w, y+w+h-1)$

Gentle AdaBoost using small CART trees as weak classifiers outperforming Discrete AdaBoost and stumps.

A review about how the *boosted cascade of simple features* has been used over the last years for face and facial feature detection can be found in the previous section on related work (Chap. 3.2).

4.2 Support Vector Machines

Support Vector Machines can be used for classification and regression problems. The detection of objects (e.g. faces) is a typical classification problem. The main idea of SVMs concerning classification can be described as follows: "It maps the input vector x into a high-dimensional feature space Z through some nonlinear mapping, chosen *a priori*. In this space, an optimal separating hyperplane is constructed." (cf. Vapnik [Vap00])

On the theoretical level, the algorithm is motivated by the principle of structural risk minimization. This means, the possibility of generalization is determined, not only by the training error but also by the complexity of the used model.

The Main Idea of SVMs in Detail

We have a set $\mathcal{X}$ of training samples in form of vectors x_i, and for each of them we know the class it belongs to:

$$\{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_m, y_m) \mid \mathbf{x}_i \in \mathcal{X}, y_i \in \{-1, 1\}\} \tag{4.8}$$

y_i marks the class where the corresponding $\mathbf{x}_i$ belongs to. The SVM has to find a hyperplane in this vector space, which separates best the vectors of two different classes. The minimum distance (margin) of the hyperplane to samples of both classes, has to be maximized: If you want

to classify new samples, which are not similar to the training samples, a bigger margin results in a more reliable separation. The vectors located on the margin, are called *support vectors*: To get an exact mathematical description of the hyperplane, only these *support vectors* are needed. All the other vectors do not have any influence. After training, the hyperplane is used as decision function, to classify new samples:

$$f(x) = sgn(\langle \mathbf{w}, \mathbf{x} \rangle - \mathbf{b}) \tag{4.9}$$

with **w** representing the normal vector of the hyperplane, and a bias **b**.

If the data can not be parted in a linear way, a kernel-trick can be used to find a non-linear hyperplane to part the two classes. The kernel-trick transfers the training vectors to a higher-dimensional space. In this space, it is possible to find a linear solution to part the classes. Re-transformation of this hyperplane into the original vector space, results in a non-linear solution. The most popular function for the kernel trick, is a radial basis function (RBF) like this:

$$K_\gamma(|x - x_i|) = e^{-\gamma|x-x_i|^2}. \tag{4.10}$$

Using the kernel trick, results in the following decision function to classify a new x:

$$f(x) = sgn\left(\sum_{i=1}^{N} \alpha_i K_\gamma(|x - x_i|) - b\right) \tag{4.11}$$

with $\alpha_i \neq 0$ only for the *support vectors*, and $\alpha_i = 0$ for all other vectors.

Regression- and Classification-based Approach for Facial Feature Detection

For facial feature detection exist alternative algorithms using SVMs, which already have been discussed in the related work section (Chap. 3.2). In the following, we have a more detailed look at the regression- and classification-based approach for facial feature detection, presented in [BKK$^+$08], which we coauthored.

At first, we have to define two terms (*Kernel Ridge Regression* and *regressor*) to ensure a better understanding of the regression- and classification-based approach:

In a controlled learning problem we have sample pairs (x_i, y_i) with $x_1, .., x_n \in \mathcal{X}$ and $y_1, .., y_n \in \mathbb{R}$. We want to learn the function $g : \mathcal{X} \mapsto R$.

Kernel Ridge Regression (KRR) can be used for both, regression and classification. Like in other kernel-based methods, we have to learn the parameters for

$$g(x) = \sum_{i=1}^{n} k(x, x_i)\alpha_i \tag{4.12}$$

in the training step with a positive definite kernel matrix

$$K = (k(x_i, x_j))_{i,j=1}^{n}. \tag{4.13}$$

Using the kernel matrix K and the label vector $Y = (y_1, ..., y_n)$, α can be computed as follows:

$$\alpha = (K + \tau I)^{-1} Y \tag{4.14}$$

where $\tau \geq 0$ is a regularization parameter. Then, ridge regression computes the hyperplane:

$$\alpha = \arg\min_{\alpha \in R^d} (X_i^T \alpha - Y_i)^2 + \tau ||\alpha||^2. \tag{4.15}$$

which minimizes the quadratic error, and penalizes the quadratic norm of the weight vector α. For more information about Kernel Ridge Regression we refer to [STC04].

Everingham and Zisserman [EZ06] use a *regressor* to accurately localize the eyes in face images extracted by a face detector. The region output of a face detector is used as input for the regressor. This linear regressor maps the input image to the predicted eye position:

$$f(x) = W^T \phi(x) \tag{4.16}$$

A generalization of this regression method is used in the following approach.

In the rest of this section, we quote the paper [BKK+08], which has been coauthored by us:

For facial component detection, we trained an array of regressors for each component, as opposed to only one regressor used in other algorithms. The detection results are then refined by a classification-based approach which combines SVM-based component detections and the prior distribution of their joint configurations. We trained and tested the classifiers based on the results of the regression-based method. This helps to filter out image regions far from the facial components and accordingly prevents the training of an SVM from being disturbed by irrelevant image information. The component detectors are view-specific, and they were trained on synthetic data produced by the 3DMM.

As a first step, a face detector is applied to the input image or video. For this purpose, two publicly available face detection libraries for Matlab were considered: an approach based on SVMs (Kienzle [KBFS05]), and an implementation of the widely used Boosting based detector by Viola and Jones [VJ04]. We chose the SVM implementation which also returns a confidence value together with each detected face. The confidence estimates are used to resolve ambiguities: if there are more than one detection in an image or a video, we discard all but the one for which the detector is most confident. The input image is cropped to a square region around the most confident position, and is then rescaled to 200×200 pixels. This is the reference coordinate system used in all subsequent processing steps.

Faces show a significant variation in shapes, projecting them onto the viewing plane, depending on the 3d rotations. To facilitate the training of subsequent *view-based* component detection, the face images are categorized based on the rotation angles such that a component detector is trained for each category. This allows each component detector to focus on similar views. Training categories are found by uniformly quantizing the interval of tangent values of horizontal rotation $[-1, 1]$ (corresponding to $[-45°, 45°]$) into seven bins.

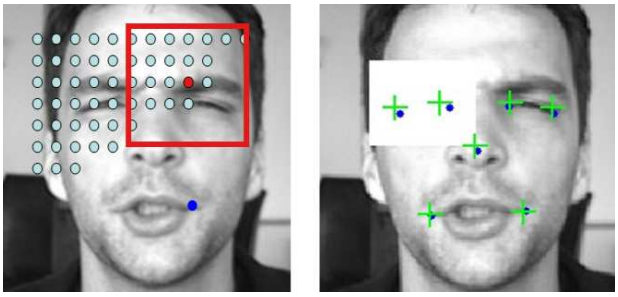


Figure 4.6: **Regression-based detection of facial components:** *Left*: illustration of the regressor array. For each component (e.g., a corner of the mouth, here marked dark blue) 144 regressors are learned. Each one operates on a different image region, centered at one of the light blue points. *Right*: prediction on a corrupted test image (the left eye region is covered with a white rectangle). Green + indicate desired component locations.

For testing, the rotation angle of the input face image is estimated by regression. The input face image is scaled down to 40×40 and the Kernel Ridge Regression (KRR) is applied to get the tangent of the angle. The estimated angle is then used to choose a proper component detector. Because the KRR provides only rough estimation of angles with an average error rate around $7°$, the estimated angles are refined later by iteration through alternating the component detection and face fitting (cf. Chap. 5).

The third stage computes position estimates of eye and mouth corners, which we here refer to as the Components Of Interest (COI), in the 200×200 image. For this purpose, we developed a novel algorithm, which can be seen as a generalization of the regression method proposed by Everingham and Zisserman [EZ06]. It predicts the position of a COI from pixel intensities within a $k \times k$ window. Invariance under intensity changes is achieved by subtracting the mean value from each window, and dividing it by its standard deviation. The KRR is adopted for this purpose. The novelty of our approach is that for each facial component we train an array of $12 \times 12 = 144$ regressors, as opposed to only one (Everingham and Zisserman [EZ06]). All of them predict the same quantity, but they are trained on different $k \times k$ regions on the 200×200 image, evenly spaced on a 12×12 grid (see Fig. 4.6, left image). To predict the position of that component in a test image, all 144 estimates are computed, and then binned into 1-pixel-sized bins. The bin with the most votes is chosen as the predicted location. The rationale behind this is that faces cannot be arbitrarily deformed, and thus the appearance of facial regions away from the component in question can be informative about its position. The use of 144 regressors makes the detector extremely redundant and therefore robust to occlusions and other local changes. This effect is shown in Fig. 4.6.

The regression-based approach is fast and robust. However, it turns out that its accuracy is not sufficient for subsequently fitting the 3D face model. So we present a classification-based method which is built on top of the regression-based component detection. The basic idea is to scan the input face image I with a small window and classify the central pixel of the window, using an SVM, as belonging to either the COI, or background.

We generated training examples for the classifier by sampling small windows from locations with predefined distances of the ground truth locations. Positive examples have their reference point in a 3×3 window around the ground truth location; negative examples, on the other hand, have it inside a 29×29 window, excluding the central 9×9 such that the training is not affected by ambiguous patterns. The 29×29 window for sampling negative examples is motivated by the typical location error of the regression-based method which lies within the range of 0 to 12 pixel distances from the true COI locations.

In the classification stage, instead of scanning the entire face image, the search space for a given COI is restricted as a 25×25 window surrounding the regression-based detection. While it is more accurate than the regression-based method, the problem of the classification method is that it does not automatically single out a detection. Instead, it either produces a detection blob around the COI or sometimes produces no detection. Thus, if the detection blob for a given COI is too small (or zero), we adjust the decision threshold of the SVM such that the blob size is larger than or equal to a predetermined threshold r.

In the next step, the individual detection results need to be combined, taking into account a prior in the space of joint configurations. A configuration of eye and mouth corners constitutes a 13-dimensional vector $H = (h_1, \ldots, h_6, h_\phi) = ((h_1^x, h_1^y), \ldots, (h_6^x, h_6^y), h_\phi)$ ((x,y)-coordinates values for six components and the horizontal rotation angle ϕ). Then, the best detection vector is obtained by first generating 100,000 random vectors (by sampling two dimensional vectors ((x,y)-coordinates)) from each component blob and concatenating them to constitute 12(+1)-dimensional vectors, and then choosing the maximizer of the following objective function.

$$C(H) = \alpha \sum_{i=1,\ldots,6} \log\left(\frac{1}{1+\exp\left(-g^i(W_i)\right)}\right) - M(H), \tag{4.17}$$

where $g^i(W_i)$ is the real-valued output of the i-th SVM for the input image window W_i corresponding to the coordinate h_i, and $M(H)$ is the Mahalanobis distance of configuration H to the mean of a Gaussian distribution estimated based on training configurations. We motivate this cost function as follows. Suppose we want to obtain the most probable configuration

$$H^* \quad = \quad arg\max P(H = O|I), \tag{4.18}$$

where $O = (o_1, \ldots, o_6, o_\phi)$ is the unknown ground truth configuration. The cost function (4.17) is then obtained as a result of the following series of approximations

$$\begin{aligned} C(H) &\approx P(W_1, \ldots, W_6|H = O)P(H = O) \\ &\approx \prod_{i=1,\ldots,6} P(W_i|h_i = o_i, h_\phi = o_\phi)P(H = O) \\ &\approx \sum_{i=1,\ldots,6} \log P(W_i|h_i = o_i, h_\phi = o_\phi) - M(H), \end{aligned} \tag{4.19}$$

where the first line applies the Bayes formula and replaces the image by the small windows (W_i), the second line corresponds to an independence assumption of the component likelihoods plus

precategorization of face images, and the third line assumes a Gaussian distribution of the configurations H. The last step is to substitute $P(W_i|h_i = o_i, h_\phi = o_\phi)$ with the component detection posterior $P(h_i = o_i, h_\phi = o_\phi|W_i)$ calculated based on Platt's method [Pla99], assuming a uniform distribution over h_i, h_ϕ within a small region generated by the rough detector.

This technique of replacing the likelihood by the posterior estimated from a discriminative classifier has shown to improve the discrimination capability of generative models (Schenkel [SGH95] and Cohen et al. [CFM+92]).

In the computation of $P(H)$, the (x,y)-coordinate of the outer corner of the left eye was used as the origin $(0,0)$, and the width and height of the bounding box for the joint configuration were normalized by dividing by the width of the original box. Accordingly, H is a 11-dimensional vector.

In addition to the eye and mouth corners, we also use nose tip for fitting the 3D face model. However, it turns out that the nose is very hard to identify by using local features alone which both the regression-based and classification based methods rely on. Accordingly, the nose detection is performed in a separate step which will be explained in Chap. 5. To facilitate this, we generate several nose candidates based on detected eye and mouth corners. A conditional Gaussian model of nose tip location given the eye and mouth corners are estimated from which the nose candidates are obtained by thresholding based on Mahalanobis distance.

In the preliminary experiments, the number of support vectors for SVM-based component detectors ranged from around 2,000 to 5,000. This resulted in processing time of more than 10 minutes per image. To reduce the time complexity, our approach adopts the *reduced set method* (Burges and Schölkopf [BS97] and Schölkopf et al. [SMB+99]) which finds a *sparse* solution

$$g(\cdot) = \sum_{i=1}^{N_\mathbf{b}} \alpha_i k(\cdot, \mathbf{b}_i) \tag{4.20}$$

as an approximation of the original SVM solution f. In the original reduced set method, the approximation error is measured by $\mathcal{F}(g) = \|g - f\|^2_\mathcal{H}$, where $\mathcal{H}$ is the *Reproducing Kernel Hilbert Space* corresponding to the kernel function of SVM. However, this does not consider the distribution of data $P(\mathbf{x})$. In this paper, we find the solution $g(\in \mathcal{H})$ as the minimizer of a new cost functional

$$\mathcal{G}(g) = \sum_{\mathbf{x}\in\mathbf{X}} (g(\mathbf{x}) - f(\mathbf{x}))^2, \tag{4.21}$$

where $\mathbf{X}$ is a finite set of data sampled according to $P(\mathbf{x})$. Informally, this cost functional allows us to find the sparse solution by putting more emphasis on high-density regions.

The solution g is found by first initializing the basis $\{\mathbf{b}_1, \ldots, \mathbf{b}_{N_\mathbf{b}}\}$ by k-means on a set of *unlabeled* data $\mathbf{X}_U$, and then performing the gradient descent on the regularized cost functional

$$\mathcal{F}(g) = \lambda\|g\|^2_\mathcal{H} + \sum_{\mathbf{x}\in\mathbf{X}_U} (g(\mathbf{x}) - f(\mathbf{x}))^2, \tag{4.22}$$

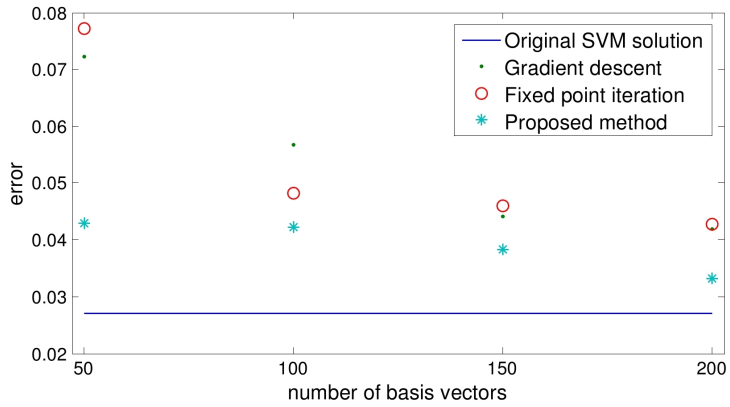


Figure 4.7: **Comparison between different reduced set methods**

where $f(\mathbf{x})$ denotes the output of the SVM at $\mathbf{x}$. λ is obtained by cross-validation. For all component detectors, we set the number of basis vectors $N_{\mathbf{b}}$ to 200. To gain an insight into the performance of the proposed reduced method, the classification error rates for a test set which is disjoint from the training set is compared with two implementations of existing reduced set method for case of the outer corner of left eye component: fixed point iteration (Schölkopf et al. [SMB$^+$99]) and gradient descent (Burges and Schölkopf [BS97]). Fig. 4.7 summarizes the results.

For training the rough angle estimator and the component detectors, we used 3,070 synthetic images obtained from 307 3D faces. 107 of these 3D faces were reconstructed from single images of the FERET database [PWHR98] using the algorithm described in [BV03] by Blanz and Vetter, and 200 3D faces were from laser scans [BV03]. The synthetic images were rendered from the 3D faces by varying the azimuth angle randomly in $[-45°, 45°]$ with a uniform distribution, varying pitch in $[-20°, 20°]$, and the azimuth and height of the light in $[-45°, 45°]$. Also, the relative contribution of directed and ambient light was varied randomly. Gaussian kernels ($K(\mathbf{x}, \mathbf{y}) = \exp(-\gamma\|\mathbf{x}-\mathbf{y}\|^2)$) were utilized for the KRRs and SVMs whose parameters were found by cross-validation. For the KRR component detector, the size of the regression input k and a scaling factor s by which the image was downscaled before sampling the $k \times k$ window were set to 9 and 0.1, respectively. For each SVM detector, around 20,000 to 40,000 training patterns are collected from these 3,070 faces. The input window size for eye and mouth detection was determined as (31×31) which for the eye case, roughly corresponds to the average length of the eye in (200×200)-size frontal face images. The blob size threshold r and α in Eqn. (4.17) were empirically set to 25 and 4, respectively. We did not find the parameters to affect the results significantly, but would expect that a future choice by cross validation could somewhat improve the performance.

The threshold for choosing nose candidates from eye and mouth corner detection is set to be 1.1. This value ensures that the resulting candidate set includes desired nose points for the entire training set. However, instead of investigating all the candidates, we use only a small subset sampled with a regular interval (3 pixels for each dimension) in an image domain so that on average, the number of actual candidates are around 100.

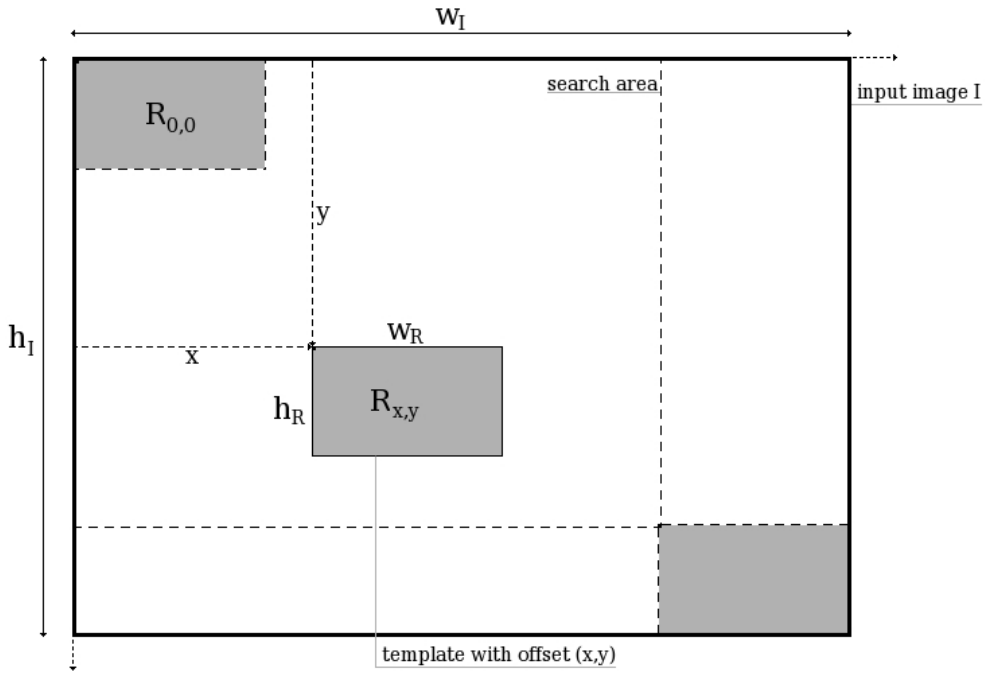


Figure 4.8: **Main idea of template matching:** The template is moved about the offset (x, y) over the input image using the origin as reference point. The sizes of input image $(w_I \times h_I)$ and template $(w_R \times h_R)$ define the search area. The figure is following Burger and Burge [BB08].

4.3 Template Matching

In image space, a template is a 2D array containing a wanted structure with some background in its direct neighbourhood. The main idea is to move the template (reference image) R over the input image I and compute the difference to find the right offset (x, y) where the similarity of the moved reference image $R_{x,y}$ and the input image is at the maximum (cf. Fig. 4.8). Whole regions or even only edges can be matched.

To use Template Matching (TM) the shape of the wanted structure or feature has to be known quite exactly. Aside from that, the matching is depending on scaling and rotation. So the scaling and rotation parameters have to be known or the searched parameter space has to be extended and the similarity function has to be computed in a 4D space of planar translation, scaling and rotation.

In the following we presume that rotation and scaling are known and have a look at the different possibilities to measure the degree of similarity depending only on the offset (x, y).

A pixel-by-pixel comparison seems to be necessary for matching subimage templates. The most simple measurements are the sum of absolute differences (Eqn. 4.23), the maximum of absolute differences (Eqn. 4.24) and the sum of squared differences (Eqn. 4.25).

$$\sum_{(p,q)\in R} |I(x+p,y+q) - R(p,q)| \tag{4.23}$$

$$\max_{(p,q)\in R} |I(x+p,y+q) - R(p,q)| \tag{4.24}$$

$$\sum_{(p,q)\in R} (I(x+p,y+q) - R(p,q))^2 \tag{4.25}$$

They all deliver small values for high similarity. Multiplying out the sum of squared differences

$$= \sum_{(p,q)\in R} (I(x+p,y+q))^2 + \sum_{(p,q)\in R} (R(p,q))^2 - 2 \cdot \sum_{(p,q)\in R} I(x+p,y+q) \cdot R(p,q) \tag{4.26}$$

shows that it can be divided into three parts: the sum of squared values of the current image part, the sum of squared values of the template and the linear cross correlation. Assuming equipartition of energy over the whole input image, causing nearly constant results for the sum of squared values of any part, the minimum distance can be measured by maximizing the linear correlation. Viewing the correlation as a convolution of template and input image using Fourier transformation a fast computation is possible in the spectral space.

These kinds of measurements are fast to compute and quite robust against noise but not really useful for use in practice. They are susceptible for differences between template and input image in terms of the mean brightness or the contrast between foreground and background, and a template fitting very closely to the input image is need. The use of the linear correlation is only possible having an equipartition of energy, which is highly unlikely in practice, too.

To get more robust against scaling and shifting of the grey-scale spectrum and against intensity changes in the input image, the normalized correlation coefficient

$$\frac{\sum_{(p,q)\in R} I(x+p,y+q) \cdot R(p,q)}{\sqrt{\sum_{(p,q)\in R} (I(x+p,y+q))^2} \cdot \sqrt{\sum_{(p,q)\in R} (R(p,q))^2}} \tag{4.27}$$

has to be used, taking the energy of the current underlying image part into account.

Better robustness against global intensity and contrast changes can only be reached computing the difference in relation to the local mean value of the template and the local mean value of the underlying image part

$$\frac{\sum_{(p,q)\in R} (I(x+p,y+q) - \bar{I}(x,y)) \cdot (R(p,q) - \bar{R})}{\sqrt{\sum_{(p,q)\in R} (I(x+p,y+q) - \bar{I}(x,y))^2 \cdot \sum_{(p,q)\in R} (R(p,q) - \bar{R})^2}} \tag{4.28}$$

with the mean values $\bar{I}(x,y) = \frac{1}{N} \sum_{(x,y)\in R} I(x+p, y+q)$ and $\bar{R} = \frac{1}{N} \sum_{(x,y)\in R} R(x,y)$ and N for the number of elements of the template. This yields a pixel by pixel comparison of the template and the current underlying part of the image, also robust against small differences between the images to compare. The only disadvantage is the fact that this local measurement has to be computed in the image space because there is no simple and efficient possibility to realize it in the spectral space.

Rearranging the formula, there is a more efficient way of computing the distance measure:

$$\frac{\sum_{(p,q)\in R} (I(x+p, y+q) \cdot R(p,q)) - N \cdot \bar{I}(x,y) \cdot \bar{R}}{\sqrt{\sum_{(p,q)\in R} (I(x+p, y+q))^2 - N \cdot (\bar{I}(x,y))^2 \cdot \sigma_R}} \tag{4.29}$$

with $\sigma_R^2 = \sum_{(p,q)\in R} (R(p,q) - \bar{R})^2 = \sum_{(p,q)\in R} (R(p,q))^2 - N \cdot \bar{R}^2$ which has to be computed only once as well as $\bar{R}$, making it being beneficial to the use in practice.

Minimizing the template size keeping the amount of background pixels as small as possible minimizes potential errors caused by wrong background information within the template. It is also possible to use not only rectangular templates but also circles, triangles, elliptical ones, cross- or x-shaped or completely different shapes such as for some eye detection algorithms (cf. Chap. 3.2). Using a so called 'window matching', allocating different weights within the template, makes it possible to give more importance to some parts, e.g. to concentrate more on the center of the template than on the margin.

To match binary images the introduced correlation based measurements do not work well. The difference between two binary images is small only if there is a substantial pixel-by-pixel congruence between them. There are no continuous junctions between the intensity values. This fact leads to a lot of local peeks in the distance function depending on the relative offset of the template. Edge comparisons are the better choice using the measurements presented in the following.

Edges are shape features. It does not matter what an object looks like. Here it is only taken advantage of the fact that conformable objects have similar edge features.

To get robust matching on edges preprocessing to extract significant edges (e.g. with a Canny Edge Detector) is necessary. Than, the similarity is given by the distance between corresponding edges. Small distances between corresponding edges yield high similarity, whereas the correspondences have to be defined first using Chamfer- or Hausdorff-Distance.

Direct matching requires strict conformity between the edges of the template and the input image. But this is highly unlikely in practice and also asymmetry and missing correspondences are main problems to be solved. Most of the time the input image contains more edges than the template and also edges of the model (in the template) may be missing in the input image because of occlusions and artifacts.

To overcome this, correspondences have to be determined asymmetrically. Each point of an edge of the model corresponds to a point on the nearest edge in the image and vice versa. The similarity measure for a given position than results from the sum of shortest distances between model and image edges. Having a binary image differentiating between foreground and background the distance of any pixel to the nearest edge can be precomputed using Chamfer- or Hausdorff-Distance.

This kind of matching applies to detection of traffic signs [Gav99] (using circles and triangles in different resolutions) but also to facial feature detection and contour matching.

To match at different resolutions it is feasible to use a mean sized template first to detect 'interesting' areas. Using a hierarchical structure (a tree) for the different resolutions from there different templates can be taken to examine the interesting areas more precisely.

Taking occlusions into consideration, the correspondence assumption has to be changed taking the template size into account. Otherwise points of the template not contained in the input image are matched to wrong positions. Forward and backward distances have to be treated differently using the template size to mark the region to search for correspondences only. The forward distance is given by the maximum of the shortest distance from a point on an edge of the template to an edge in the image. The backward distance measures the maximum of the shortest distance from edge points of the image to the template edges. But this backward distance has to be modified. Only distances to points that are not further away than the size of the template, are taken into account.

The basic information of this section are mainly taken from Tönnies [TÖ5] and Burger and Burge [BB08].

A review about how TM has been used over the last years for face and facial feature detection can be found in the related work section (Chap. 3.2) including the use of different kinds of templates (like deformable ones) with a closer look at eye detection, and Chap. 6 shows the use of it for increasing the robustness of the 3DMM.

4.4 SIFT

In this section another essential method is described in detail: SIFT (scale-invariant feature transform). SIFT is used to find essential features, characterizing the input image. The features consist of keypoints and vectors (as will be described in the following). To detect a certain facial feature like the corner of an eye, you need the SIFT features of a sample, to compare with. It will be shown, that SIFT has been used in several approaches of facial feature detection, even for initialization of a Morphable Model. We present SIFT here to show its capabilities. Our approach presented in the next chapter (Chap. 5: Model-Based Confidence Measure for Feature Points) is not limited to measure the plausibility of feature configurations detected by SVMs. It is also possible to analyze a set feature positions detected with SIFT.

In 1999 Lowe first presented his SIFT approach transforming image data into scale-invariant coordinates relative to local features [Low99] for object recognition. He has also an US patent on the scale-invariant feature transform (SIFT) and further improved the approach in 2004 [Low04].

Outline of the algorithm:

1. Blurring the original image step by step with a Gaussian $1 \leq \sigma \leq 2$. This results in a series of blurred images, the so called scale-space.

2. Blurred images have to be subtracted from their direct neighbours (according to σ), yielding a new series of Differences of Gaussian (like a scale normalized version of Laplacian of Gaussian). At least three images per octave are computed and a new octave is started with half of the resolution (cf. Fig. 4.9).

   Main steps:

3. Detect extrema in the scale space (cf. 2.). Like in blob detection each pixel is compared with its eight neighbours and with the corresponding pixels and their neighbours (nine each) of the upper and lower images of the scaling pyramid (cf. Fig. 4.10).

4. Choose keypoints from the extrema in scale space (cf. 3.): At each candidate location, a detailed fit of a 3D quadratic function to the nearby data is done, to get location, scale and the proportion of the principle curvatures. This information allows to reject points that have low contrast (and are therefore sensitive to noise) or are poorly localized along an edge.

5. To assign an orientation to a keypoint, in a 16×16 window around it the histograms of gradients (direction and size) are computed using bi-linear interpolation.

6. At the end the keypoints are represented as 128-dimensional vectors. The location is quantized into a 4×4 location grid computed from the 16×16 window and the gradient angle is quantized into 8 orientations in each part of the grid: $4 \times 4 \times 8 = 128$. To obtain illumination invariance the descriptors are normalized by the square root of the sum of squared components.

Invariance of the descriptors is given by the circumstance that the coordinates and the gradient directions are given relatively to the orientation of the keypoints.

To use SIFT for object detection or matching different views of an object or scene, a nearest-neighbour algorithm followed by a Hough transformation using an efficient hash table has to be performed. First SIFT features have to be extracted from a set of sample images. Matching a new image is done by comparing the features of the new image with the stored ones using Euclidean distance. This delivers candidates for related features. Features of the background can be eliminated by building related sub sets of keypoints fitting to an object with this position, scaling and orientation. Any feature belonging to no sub set has to be discarded. If three or more

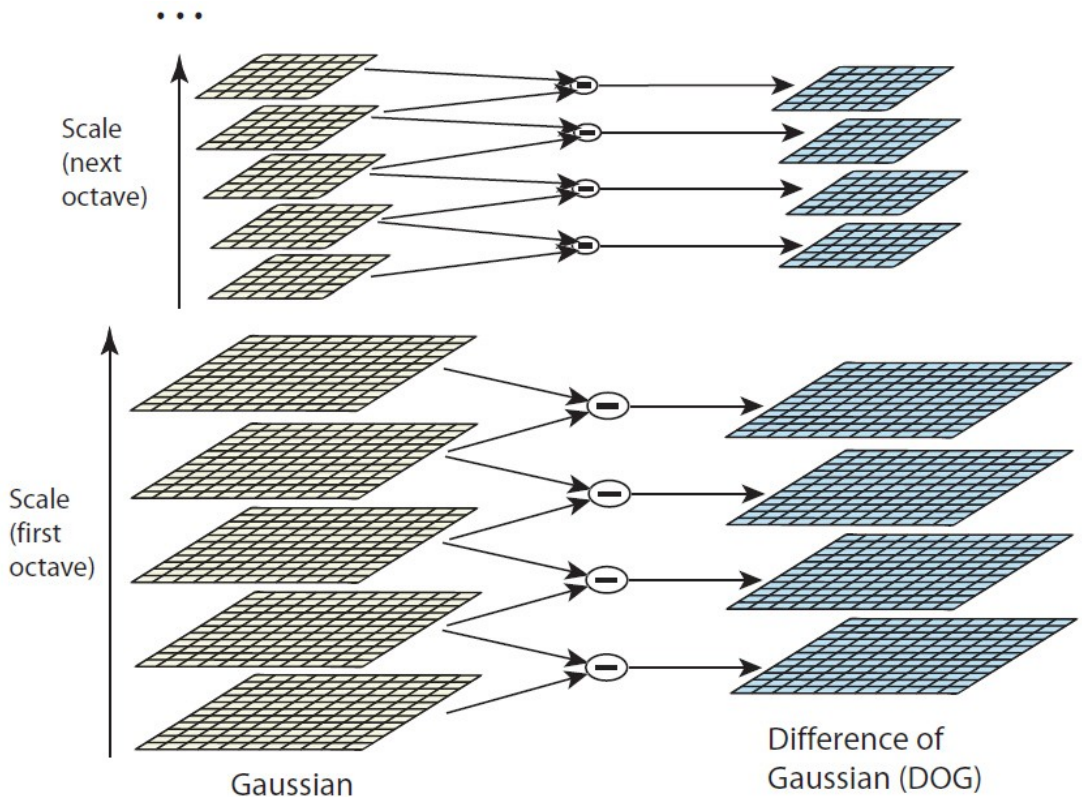


Figure 4.9: **Octaves of DoGs:** The initial image is convolved repeatedly with Gaussians (left side) and adjacent results are subtracted to get the Difference of Gaussian images (right side). The figure is taken from [Low04].

features fit to an object it has to be verified in more detail: Get an affine approximation to the object pose by least-squares estimate and search for consistent pose parameters. The accuracy of the fit and the number of false matches yield to the probability of having detected a certain object.

The presented SIFT approach has been used by several scientist since its invention.

For example, to recognize object classes (mostly faces) Helmer and Lowe [HL04] used SIFT features grouped by a probabilistic model. Initialized with a few parts of the object, further parts are learned incrementally adding them during the process of training. To update the whole model and learn the parameters they use the EM algorithm in a Maximum Likelihood framework.

The probabilistic model represents an object class by a set of local parts modelling their appearance, relative location, co-occurrence and scale. Based on the approach of Fergus et al. [FPZ03], where the parts have been learnt simultaneously, they performed an incremental learning and improved the model to get invariant against translation and scale. Images each are decomposed to a set of SIFT features containing shape, scale and appearance. Beginning with a few accurate parts (with tight distributions on scale, appearance and location), parts increasing the likelihood the most are added to the model, after transforming the feature locations of every image into the

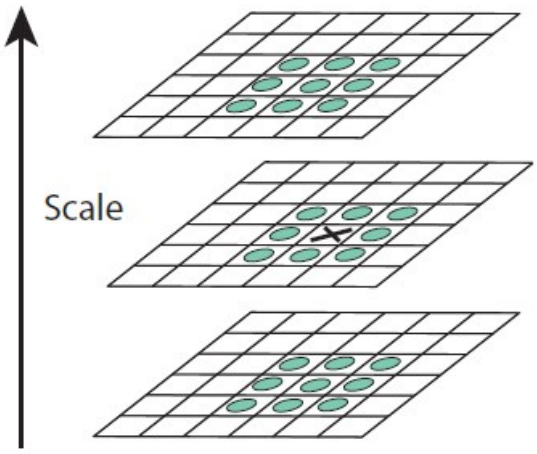


Figure 4.10: **Neighbours compared with for extrema detection:** The *X* marks the pixel to compare with its neighbours (the green circles) at the current and adjacent scales. The figure is taken from [Low04].

model coordinate space (using the most likely matches from features to parts). Using the EM algorithm with a random initialization would yield a solution that is a local minimum, but using some accurate parts for initialization and adding only parts when there is a number of features in the dataset that match them closely keeps the variance small. High variance in location and scale after a few iterations indicate that the approach has arrived at a poor model and the system needs a restart.

Mikolajczyk and Schmid [MS05] compared the performance of ten different local descriptors on grey-scale images with the result that SIFT based descriptors performed the best. They used SIFT descriptors computed with the code provided by Lowe [Low99, Low04] and compared it with other descriptors (PCA-SIFT, spin image, steerable filters, differential invariants, complex filters) using the criteria distinctiveness, robustness against changes in viewing directions, against errors of the detector and recall-precision in matching and recognition. All descriptors were computed for normalized image patches chosen by different region detectors. They also presented an extension of the SIFT descriptor and showed that it outperforms the original method. The extension is called GLOH (gradient location-orientation histogram) and is designed to increase the robustness and distinctiveness of SIFT. They compute the SIFT descriptors for log-polar location grids with three bins in radial direction and eight in angular direction (cf. Fig. 4.11), resulting in 17 location bins. Quantizing the gradient orientations in 16 bins they get a 272 bin histogram for each descriptor. Reducing the size with PCA (estimated on 47000 image patches collected from various images) only the 128 largest eigenvectors are used for the description in the end. To measure the similarity between the descriptors Euclidean distance is taken for GLOH, too. Testing the performance on six image transformations (rotation, scale change, viewpoint change, image blur, JPEG compression, illumination) GLOH obtained the best results in most of them, closely followed by the original SIFT.

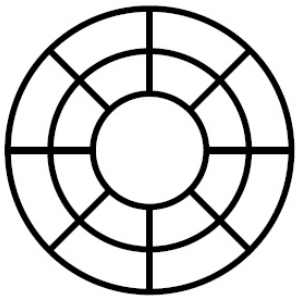


Figure 4.11: **Log-polar grid used for GLOH:** with three bins in radial direction and eight in angular direction

Bicego et al. [BLGT06] investigated the application of the SIFT approach in the context of face authentication by selecting and grouping the extracted features according to the face geometry. They assume the knowledge of the position of a few facial landmarks (mouth and eyes) to deliver a concept of the geometry. They presented three different matching strategies. First they used only the Minimum Pair Distance of only one pair of features to estimate the matching score of two images. This approach is based on the idea to have one unique distinctive feature of the subject which can also be found in the testing image. But this approach is independent of locations, the knowledge of the geometry is not used. Observing subimages of the eye-region and the mouth and computing the matching score as the average of the Minimum Pair Distances of both regions lead to better matchings. Having both eyes in one region, though, matching is still not location specific because features at the left eye of the one image could be matched with features at the right eye of the other image. To resolve this ambiguity they finally do the matching on a regular grid obtained from a registration process. Using a registered query image and a registered reference image, these can be subdivided in lots of subimages and the minimum pair distances on corresponding ones can be computed. The average distance of this location dependent matching marks the most reliable match score, but always localization information is needed to register the images.

Using SIFT features to perform face identification has been tried by Meng and Tiddeman [MT06]. They worked on blurred images and computed the SIFT features only at the edges. Using their own Gradient-Based Edge Detector the local descriptors mostly appear in the eyes, nostrils, at the top of the nose and at the corners of the lips. They tested the repeatability and stability of the SIFT features on a pair of images (one being the rotated version of the other) and compared each keypoint of the original image with all keypoints in the transformed image. The first two 'qualified' pairs of the keypoints were chosen to set the transformation, describing the mapping of an original image point to a transformed image point. This way they showed the stability and how the SIFT features move along with the rotation of the image, but an explicit matching between different faces has not been presented.

Fernandez and Vicente [FV08] also used SIFT for face recognition in combination with multiple interest point detectors, namely a Harris-Laplace detector (a scale invariant form of the Harris-Corner detector) and Difference of Gaussians. To get the best match they performed three

measurements. First they determined the best and second best match between a query image and each one out of the database to compare. Only the images from the database with a ratio of best to second best match bigger than a certain threshold are kept for further examination. Using Hough transformation images with features coherent in orientation and scaling are detected and finally the relative positions of the features are compared to identify the image best matching to the query image.

Grujić et al. [GILF08] used SIFT features on MSER regions for facial pose estimation by image retrieval. Features probably belonging to the background are sorted out. Configurations of valid features are stored in a so called vocabulary tree. For retrieval images this tree can efficiently be searched for the most similar feature configuration, this way providing a list with best matching poses. The poses correspond to different people (showing this pose) in the database. To find out which person corresponds best to the person in the query image, they match the features between every retrieved image of the list and the input image. From the matches they compute a global motion using RANSAC [FB81] for each image of the list. The more consistent the motion is over all features of a retrieved image, the more geometrically consistent is the person in the pose.

Using SIFT as a criterion for saliency Romdhani and Vetter [RV07] first identified all potential feature points in an image rejecting those that are similar to none of the points in their appearance model. Subsequently finding the configuration of features in the image and the mapping to features of the used 3DMM that has a maximum likelihood. The resulting feature locations could then be used to initialize a 3DMM fitting procedure.

GLOH has been used for pose invariant face recognition, too. Sarfraz and Hellwich [SH08] analyzed face images detected with AdaBoost from ViolaJones using the gradient location-orientation histogram (GLOH). They learned how to create a frontal view out of different other poses $I_{frontal} = B \cdot I_{pose}$ and by computing the likelihoods they estimate the probability of two face images showing the same person.

Chapter 5

Model-Based Confidence Measure for Feature Points

In this chapter, we use hypothetical 2D locations of facial components as feature points and compute a 3D-based confidence measure for the plausibility of a configuration, using the Morphable Model. This is important because plausible feature positions are needed for the initialisation of the 3DMM to get a reliable full reconstruction. Especially for the nose it is relevant to consider 3D geometry. Eyes and mouth positions are approximately coplanar. We treat the following feature points: the tip of the nose, the corners of the mouth, and the external corners of the eyes. The inner corners of the eyes turned out to be dispensable for model fitting.

For each of the feature points $j = 1, ..., 5$, we have the image positions $(q_{x,j}, q_{y,j})$ and we know which vertex k_j of the model it corresponds to. We can now find the linear combination of examples and the 3D rotation, scale and translation that reproduces these positions best. We do this with an efficient, quasi-linear approach (Blanz et al. [BMVS04]) that we summarize below. Unlike previous work, we are now using the Mahalanobis Distance from the average face as a measure of 3D distortion.

To assess how well the reconstructed face fits to the pixel values in the image, we modify the quasi-linear algorithm: After shape fitting, we can look up the desired color or grey values of the image for each vertex. Unlike the algorithm in Chap. 2, we assume simple ambient illumination here. For finding the optimal nose position, it has turned out to be best to use only vertices in the nose region. The color values $(R_{k_j}, G_{k_j}, B_{k_j})$ for vertices k_j are reconstructed by the textures of the Morphable Model using the algorithm described below. Again, Mahalanobis Distance is used as a confidence measure. For grey-level images, we replace colors in the Morphable Model by grey-levels.

Both the coarse shape and texture reconstruction is achieved by a Maximum a Posteriori estimate [BMVS04]. In the following, let either $\mathbf{v} = \mathbf{S}$ or $\mathbf{v} = \mathbf{T}$ (shape or texture vectors and

$$\mathbf{x} = \mathbf{v} - \overline{\mathbf{v}}, \qquad \overline{\mathbf{v}} = \frac{1}{m} \sum_{i=1}^{m} \mathbf{v}_i. \tag{5.1}$$

so the data matrix $\mathbf{X}$ of sample faces is

$$\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m) \in I\!R^{n \times m}. \tag{5.2}$$

and the covariance matrix is

$$\mathbf{C} = \frac{1}{m}\mathbf{X}\mathbf{X}^T = \frac{1}{m}\sum_{j=1}^{m} \mathbf{x}_j \mathbf{x}_j^T \in I\!R^{n \times n}. \tag{5.3}$$

In this unified notation, let $\mathbf{s}_i$ be the eigenvectors of C from PCA, and σ_i the standard deviations which we include as explicit factors in the expansion

$$\mathbf{x} = \sum_{i=1}^{m'} c_i \sigma_i \mathbf{s}_i = (\sigma_1 \mathbf{s}_1, \sigma_2 \mathbf{s}_2, \ldots) \cdot \mathbf{c} \tag{5.4}$$

so the estimated normal distribution based on PCA takes the simple form

$$p(\mathbf{c}) = \nu_c \cdot e^{-\frac{1}{2}\|\mathbf{c}\|^2}, \qquad \nu_c = (2\pi)^{-m'/2}. \tag{5.5}$$

$\|\mathbf{c}\|^2$ is the Mahalanobis Distance from the average.

Now let a reduced set of model data be a vector $\mathbf{r} \in I\!R^l$ of 3D coordinates of 5 feature points, or color values of the vertices in the nose region. These are given as input to our confidence algorithm, and mathematically they are related to $\mathbf{v}$ by a projection operator. In addition, we may perform orthographic projection, rotation and scaling to geometry to get 2D coordinates, or change contrast in the color channels. For the moment, assume that these operations are known, and they combine to a linear operator

$$\mathbf{r} = \mathbf{L}\mathbf{v} \qquad \mathbf{L} : I\!R^n \mapsto I\!R^l. \tag{5.6}$$

$$\mathbf{y} = \mathbf{r} - \mathbf{L}\bar{\mathbf{v}} = \mathbf{L}\mathbf{x} \tag{5.7}$$

The least-squares solution of this problem would be to minimize

$$E(\mathbf{x}) = \|\mathbf{L}\mathbf{x} - \mathbf{y}\|^2. \tag{5.8}$$

Let $\mathbf{q}_i = \mathbf{L}(\sigma_i \mathbf{s}_i) \in I\!R^l$ be the reduced versions of the scaled eigenvectors, and

$$\mathbf{Q} = (\mathbf{q}_1, \mathbf{q}_2, \ldots) \in I\!R^{l \times m'}. \tag{5.9}$$

In terms of model coefficients c_i from (5.4), (5.8) is

$$E(\mathbf{c}) = \|\mathbf{L}\sum_i c_i\sigma_i\mathbf{s}_i - \mathbf{y}\|^2 = \|\mathbf{Q}\mathbf{c} - \mathbf{y}\|^2. \tag{5.10}$$

However, it has been shown that this simple approach would produce significant over-fitting artifacts [BMVS04], so we use a Maximum Posterior Probability (MAP) approach [BMVS04]: Given the observed vector $\mathbf{y}$, we are looking for the coefficients $\mathbf{c}$ with maximum posterior probability $P(\mathbf{c}|\mathbf{y})$. As an intermediate step, consider the likelihood of measuring $\mathbf{y}$, given $\mathbf{c}$: In the noiseless case $\mathbf{c}$ would define the vector

$$\mathbf{y}_{model} = \mathbf{L}\sum_i c_i\sigma_i\mathbf{s}_i = \sum_i c_i\mathbf{q}_i = \mathbf{Q}\mathbf{c} \tag{5.11}$$

We assume that each dimension j of the measured vector $\mathbf{y}$ is subject to uncorrelated Gaussian noise with a variance σ_N^2. Then, the likelihood of measuring $\mathbf{y} \in \mathbb{R}^l$ is given by

$$P(\mathbf{y}|\mathbf{y}_{model}) = \prod_{j=1}^{l} P(y_j|y_{model,j}) \tag{5.12}$$

$$= \prod_{j=1}^{l} \nu_N \cdot e^{-\frac{1}{2\sigma_N^2}(y_{model,j}-y_j)^2} = \nu_N^l \cdot e^{-\frac{1}{2\sigma_N^2}\|\mathbf{y}_{model}-\mathbf{y}\|^2} \tag{5.13}$$

with a normalization factor ν_N. In terms of the model parameters $\mathbf{c}$, the likelihood is

$$P(\mathbf{y}|\mathbf{c}) = \nu_N^l \cdot e^{-\frac{1}{2\sigma_N^2}\|\mathbf{Q}\mathbf{c}-\mathbf{y}\|^2}. \tag{5.14}$$

According to Bayes Rule, the posterior probability is

$$P(\mathbf{c}|\mathbf{y}) = \nu \cdot P(\mathbf{y}|\mathbf{c}) \cdot p(\mathbf{c}). \tag{5.15}$$

with a constant factor $\nu = (\int P(\mathbf{y}|\mathbf{c}') \cdot p(\mathbf{c}')d\mathbf{c}')^{-1}$.

Substituting (5.5) and (5.14) yields

$$P(\mathbf{c}|\mathbf{y}) = \nu \cdot \nu_N^l \cdot \nu_c \cdot e^{-\frac{1}{2\sigma_N^2}\|\mathbf{Q}\mathbf{c}-\mathbf{y}\|^2} \cdot e^{-\frac{1}{2}\|\mathbf{c}\|^2}, \tag{5.16}$$

which is maximized by minimizing the cost function

$$E = -2 \cdot logP(\mathbf{c}|\mathbf{y}) = \frac{1}{\sigma_N^2}\|\mathbf{Q}\mathbf{c} - \mathbf{y}\|^2 + \|\mathbf{c}\|^2 + const. \tag{5.17}$$

To simplify the calculation, we introduce a regularization factor $\eta = \sigma_N^2 \geq 0$ and minimize

$$E = \|\mathbf{Qc} - \mathbf{y}\|^2 + \eta \cdot \|\mathbf{c}\|^2. \tag{5.18}$$

Using a Singular Value Decomposition $\mathbf{Q} = \mathbf{UWV}^T$ with a diagonal matrix $\mathbf{W} = diag(w_i)$, it can be shown [BMVS04] that the optimal coefficients are

$$\mathbf{c} = \mathbf{V} diag(\frac{w_i}{w_i^2 + \eta})\mathbf{U}^T\mathbf{y}. \tag{5.19}$$

Our confidence measure for feature points is $\|\mathbf{c}_{shape}\| + \|\mathbf{c}_{texture}\|$.

In order to deal with unknown position, orientation and scale, we use the method of Blanz et al. [BMVS04], which is to treat not only translation, but also rotation and scaling as additive terms, and add a set of vectors $\mathbf{s}_i$ and coefficients c_i to the system. For rotation, this is a first-order approximation only. From c_γ, c_θ, c_ϕ, we recover the angles γ, θ, ϕ, then update $\mathbf{L}$ and iterate the process, which gives a stable solution after the second pass [BMVS04]. For the estimation of texture, we apply the same method to deal with gains and offsets in the color channels, caused by face characteristics as well as by different light directions.

Fully Automated System

Using this confidence measure in combination with the SVM based feature detection presented in Chap. 4.2 we are able to build a system for 3D reconstruction that

- can be applied either to single still images or to raw monocular video streams,
- operates at a wide range of poses and illuminations,
- involves zero user interaction,
- produces close-to-photorealistic 3D reconstructions.

The processing steps of our algorithm (Fig. 5.1) are:

1. Face Detection using SVM. (Chap. 4.2)
2. For video data: selection of the best frame using SVM. (Chap. 4.2)
3. Coarse estimate of pose (azimuth angle ϕ) based on regression. (Chap. 4.2)
4. Facial component detection using regression and classification, which have been trained for the estimated pose (ϕ). (Chap. 4.2)
5. Selection of the most plausible combination of components based on Gaussian distributions. (Chap. 4.2)
6. Selection of the most plausible nose position from a array of candidates from step 5, based on the Morphable Model. (Confidence measure presented in this chapter (5). In the flow diagram (Fig. 5.1) this step is marked yellow).

7. *Fast fit* of the 3DMM for pose estimation, to estimate ϕ again. (Fitting as described in Chap. 2, with less iterations.)
8. Iteration of step 4 to 7 until pose estimation is stable. If the angle estimated by the 3DMM is the same as the angle used for facial component detection (step 4), the most plausible combination of components is found, and we can proceed with step 9. If the angles are different, we have to start again at step 4 with regression and classification, that have been trained for the angle, now estimated by the model.
9. *Full fit* of the 3DMM. (3D reconstruction as described in Chap. 2.)

In the algorithm, the fitting algorithm of the 3DMM is used twice: In a *fast fit* with a reduced number of iterations (item 7), the system estimates the azimuth angle ϕ to compare with the angle estimated by the SVM. Given the optimal set of feature points, the final reconstruction is then computed by a *full fit* (item 9). Since the linear combination of textures $\mathbf{T}_i$ cannot reproduce all local characteristics of the novel face, such as moles or scars, we extract the person's true texture from the image and correct its illumination (Blanz and Vetter [BV99]).

Unlike previous model-based algorithms for 3D face reconstruction from single images (Blanz and Vetter [BV99, BV03] and Lee et al. [LPMM05]), the combined algorithm no longer requires manual rigid prealignment of the 3D model or manual labeling of features on the 2D image.

5.1 Results

We tested our algorithm on 6 videos and 870 images corresponding to the 87 individuals from the FERET database [PWHR98] who were not in the training set. The face detection algorithm succeeded in 705 out of 870 images, at a computation time of less than 50 ms per image on a standard PC. A single step of detecting the facial components and classification-based refinement took around 7 seconds per face. The component detection results were evaluated by measuring the average Euclidean distance of the 7 components from manually labeled ground truth within the rescaled face images (200×200 pixels). For comparison, four variants of existing classification-based methods were implemented. The independent SVM-based method scans the input image within a candidate region for each COI (Component Of Interest - based on the estimated Gaussian distribution of the COI location in the training set, hereafter referred to as Gaussian prior) and produces the detection as the location with the highest SVM score. The connected components method, inspired by Arandjelovic and Zisserman [AZ05], first generates a binary image for each COI with the pixel values corresponding to the SVM classification labels (COI or background). The location of the COI is then obtained as the mean of the connected component of the COI labels yielding the highest evidence, assuming the above-mentioned Gaussian prior. The Bayesian component detector, inspired by Everingham and Zisserman [EZ06], models pixel values within a window around each COI and background as Gaussian distributions. The location of a COI is then obtained as the position of the window that yields the greatest log-likelihood ratio among a set of candidate locations within the smallest rectangle encompassing

all the training data locations. The pairwise reinforcement of feature responses (PRFR) is based on the idea of Cristinacce et al. [CCS04], i.e., the individual COI detections obtained from each SVM are refined based on their distributions conditioned on the other components. Details of the refinement procedure can be found in [CCS04] by Cristinacce et al. .[1] Table 5.1 summarizes the results.

Given four feature points for the corners of the eyes and mouth, along with around 100 candi-

[1] In the original work of Cristinacce et al. [CCS04], a boosting classifier is utilized while we used the SVM to facilitate comparison between different methods.

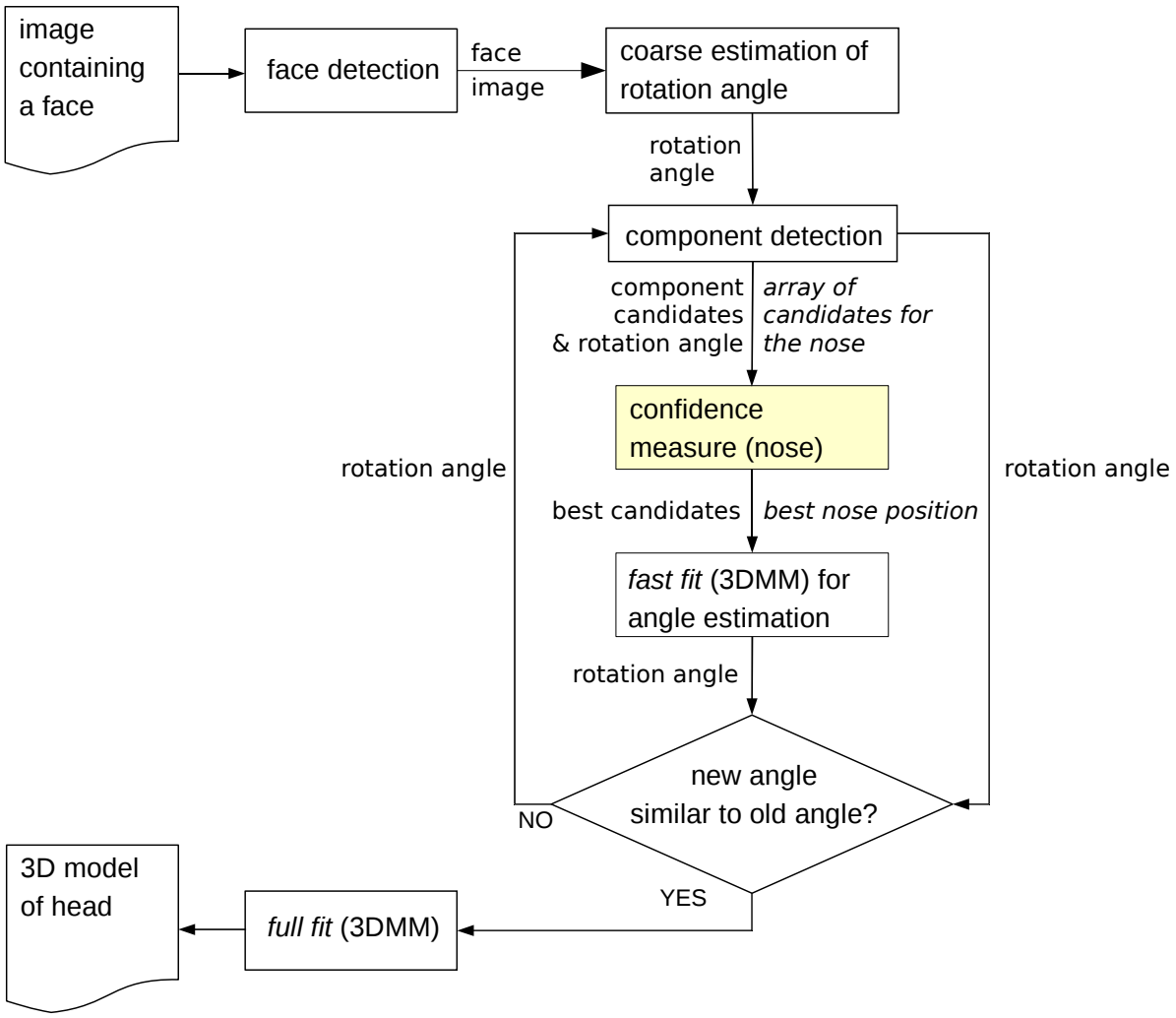


Figure 5.1: **Flow diagram:** of the automatic face fitting system. The rectangle marked yellow represents the module using the proposed new confidence measure (item 6 of the listed processing steps). The previous modules are part of the SVM based feature detection presented in Chap. 4.2. *fast fit* and *full fit* are done with the original 3DMM algorithm.

component detection method	average error (pixels)
independent SVM	8.30
connected components	7.05
PRFR	8.17
Bayesian	6.50
proposed (rough detection)	6.51
proposed (single iteration)	4.56
proposed	4.52

Table 5.1: **Performance of different component detection methods:** 'rough detection' refers to the KRR-based detection; 'single iteration' stands for a single iteration of the component detection and fitting (with the rotation angle obtained from the rough estimator).

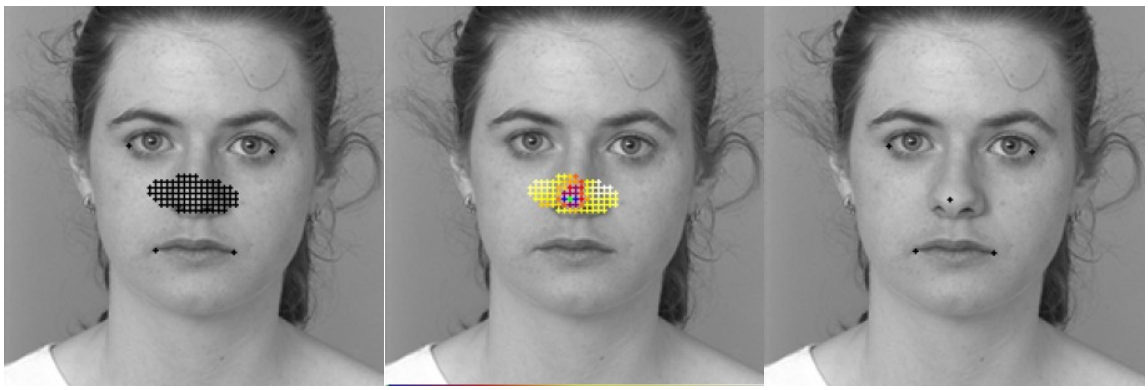


Figure 5.2: **Nose candidates:** Left: nose candidates given by the component detection; Middle: color coded results of the model-based measure (the darker the better, best position marked with green) Right: nose position chosen by the model-based measure.

dates for the nose, the model-based confidence measure returns the most likely nose position, as illustrated in Fig. 5.2.

Tables 5.2 and 5.3 show the quality of the returned feature positions listing the average errors compared to manually labeled features for different feature types and for different viewing angles of the original images. The angles in Table 5.3 are estimates from [BV03] by Blanz and Vetter.

Reconstruction was based on four points given by the facial component detection (external corners of the eyes, and corners of the mouth) and the nose position returned by the model-based confidence measure. The computation time is approximately 3 minutes.

	left eye o	right eye o	nose	left mouth	right mouth	left eye i	right eye i
error	4.78	4.97	6.09	4.39	4.53	3.71	4.20

Table 5.2: **Average differences per feature over all views:** The differences are given in pixel-units. *o* marks the outer corner and *i* marks the inner corner.

	ba 1.1°	bb 38.9°	bc 27.4°	bd 18.9°	be 11.2°	bf −7.1°	bg −16.3°	bh −26.5°	bi −37.9°	bk 0.1°
error	3.92	6.03	4.89	5.14	4.19	3.35	3.66	4.82	6.29	4.40

Table 5.3: **Average differences over all features per view *ba* to *bk* of the FERET database:** The differences are given in pixel-units.

	very good	good	acceptable	bad
%	12.86	24.09	28.93	34.12

Table 5.4: **Average rating of 200 examples:** by 49 participants.

5.2 User Study

Our system does not produce veridical, but plausible and photorealistic 3D models consistent with the input images. Hence we evaluated it only using a perceptual experiment that reflect the demands of many potential real-world applications.

We randomly selected 200 out of all automatically reconstructed models, and collected ratings by 49 participants. Those participants came from the broader environment of the University of Siegen: research assistants, students, and their relatives with a wide range of backgrounds in computer science (from average customers to scientists of different subject areas). Each participant saw the original image and two views of the reconstructed face on an online questionnaire. Some participants rated only a subset of faces, so we obtained an average of 24 ratings per face (4,815 ratings in total) according to the following instructions: "The following 3D reconstructions are supposed to be used as personalized avatars in a game. Divide the results into four groups: very good, good, acceptable and bad." Their ratings are shown in Tab. 5.4, and typical examples are shown in Fig. 5.3. Given the instructions, the criteria and standards were deliberately left open to the judgements of the participants. Still, we found that the ratings were overall positive and promising. To demonstrate the quality and variability of the results, images of the reconstructed faces without textures are shown in Fig. 5.4.

In the second experiment, we showed the automated reconstruction and the reconstruction from manually labeled features side by side in random order for each face, along with the original image. Participants were instructed to select the reconstruction that looks better. The results in

participant	better	same quality	worse
1	10%	26%	64%
2	22%	24%	54%
3	16.5%	29%	54.5%

Table 5.5: **Side by side comparison of reconstructions:** Percentage of faces where participants found the automated reconstruction better, equal or worse than the reconstruction from manual labeling in a side by side comparison. Participants did not know which was the automated and which was the manual result.

Table 5.5 indicate that the automated algorithm is competitive in many cases, even though it does not fully match the quality of the manual initialization.

The six videos were recorded with a webcam (Logitech QuickCam pro 4000). Each video shows a moving person, e.g. turning their heads, taking glasses off and on, moving forward and backward, etc. The recording speed was 30 frames/sec. and the resolution of each frame was 320×240. Our face detection algorithm attempts to detect faces in every frame, and returns the single frame with maximum detection score. Component detection, confidence measure for the feature points and finally the reconstruction proceed the same way as for the still images. For all video examples we got similar results, one of which is shown in Fig. 5.5.

By combining Support Vector Machines and 3D Morphable Models, we have addressed the problem of fully automated 3D shape reconstruction from raw video frames or other images. The system has proved to be robust with respect to a variety of imaging conditions, such as pose and lighting. The results and the rating scores by human participants demonstrate that the system produces a high percentage of photo-realistic reconstructions which makes it useful for a wide range of applications.

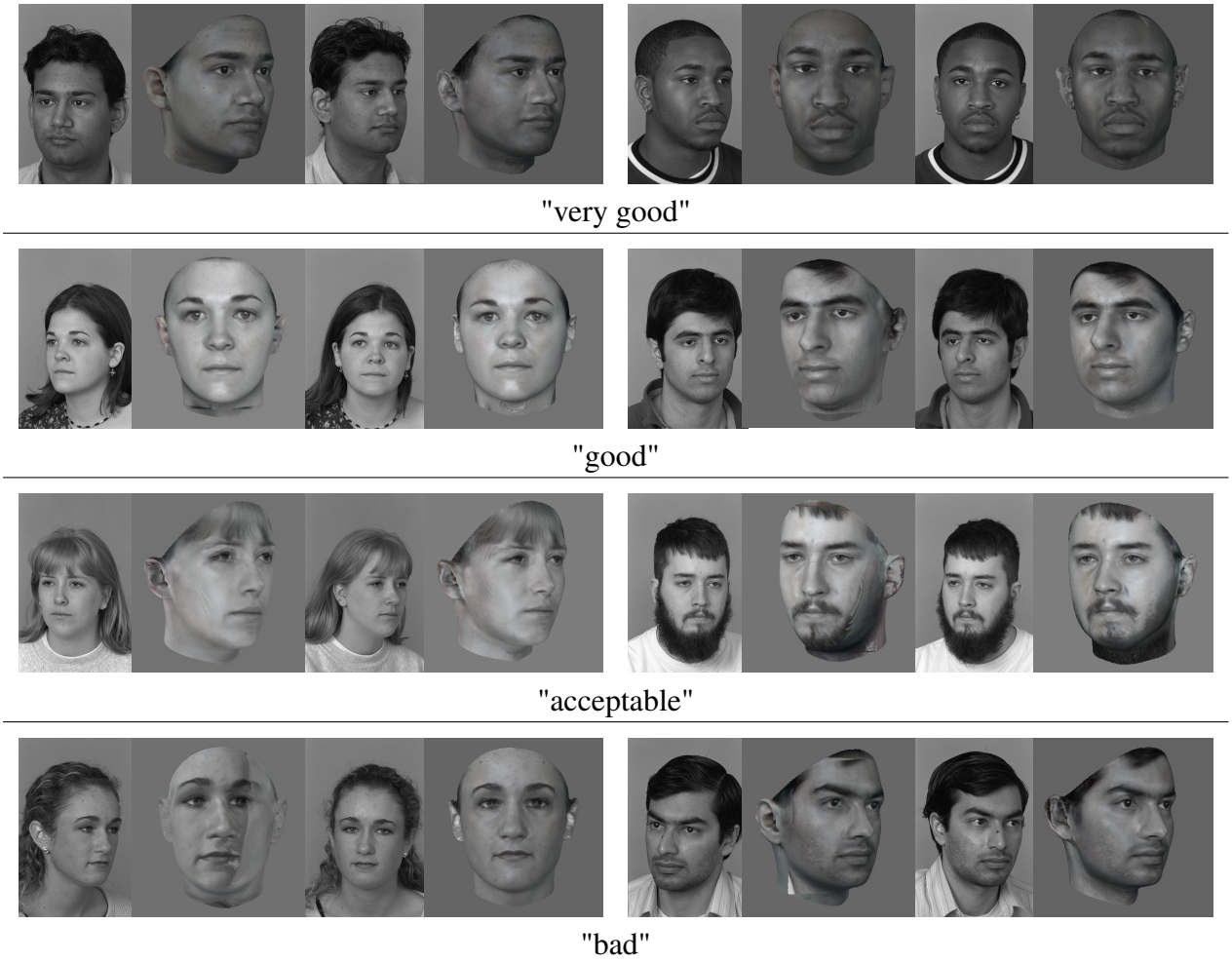


Figure 5.3: **Reconstruction from still images:** Two typical examples from each rating score level. From left to right: original image, novel view of the automated reconstruction, original second view of the person for comparison, novel view of the reconstructed 3D model if features are labeled manually.

Figure 5.4: **Heads:** Images of the automated reconstructions without texture. Same faces as in Fig. 5.3 line by line.

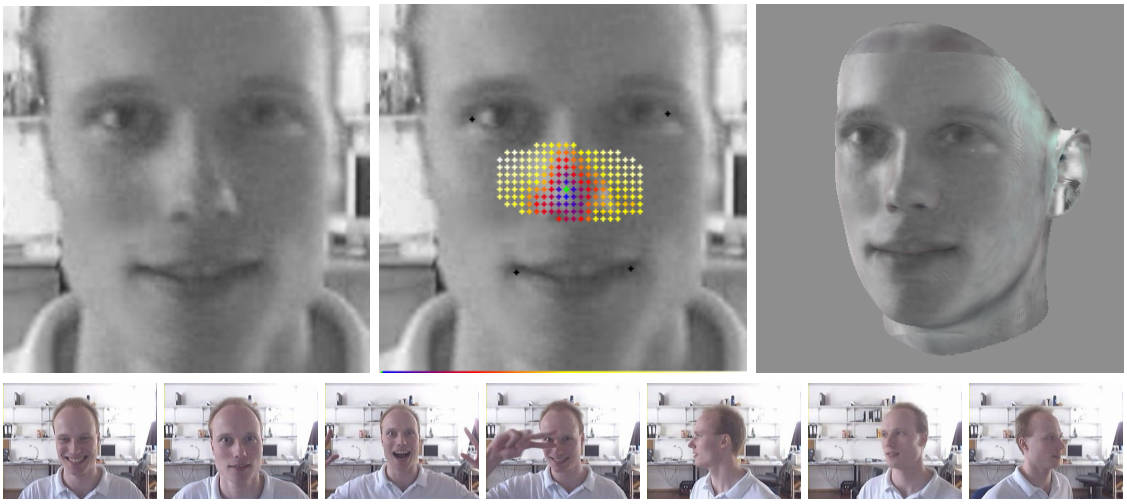


Figure 5.5: **Reconstruction from video:** Fully automated reconstruction from an automatically detected single frame of a webcam video. From left to right: original frame, color coded results of the model-based measure (the darker the better, best position marked green) and the automatically reconstructed head. The bottom row shows 7 sample frames of the video.

Chapter 6

Independence from Reliable Initial Feature Positions

A bottleneck in the development of automated fitting algorithms is the initialization of the optimization. While early work has started from a coarse alignment (Blanz and Vetter [BV99]), later versions have relied on manually defined feature point positions (Blanz and Vetter [BV03]). Recently, a fully automated 3DMM fitting algorithm has been presented (cf. Chap. 5,which uses Support Vector Machines (SVM) for the detection of faces and facial features. However, the quality of the fit turned out to depend critically on the precision of the facial features.

The goal of the work presented in this chapter is to integrate feature detection into the 3DMM fitting procedure and to do a big stride towards independence from manual initialization or precise automatic initialization. In order to leverage the fact that 3DMM fitting can be applied to any pose and illumination, it is important to have feature detectors that are either invariant, to rely on a set of different detectors, or - as we propose here - to have adaptive feature detectors. In our approach, the feature detector is updated by rendering an image of the current estimate of the face at the current estimate of the imaging parameters several times during the optimization, and forming templates from (predefined) face regions. Unlike more powerful feature detectors such as SVM or AdaBoost [VJ01], template matching (TM) does not require multiple training samples.

At first, three different approaches have been pursued, all using TM to detect the optimal feature position: The first approach relies on the assumption that only the rough position of the head (detected by AdaBoost [VJ01]) and its rotation angle are used for initialization and that further rough alignment has to be done by the model. The second approach proceeds based on the assumption that some feature positions are already used for initialization, which however can or even have to be improved. The third approach disengages from delivered features and their positions. The feature positions are only used for initialization and predefined significant feature position are refined afterwards during the fitting.

At last, in the main approach (the fourth approach) of this chapter, feature detection is used to find not only the optimal feature position, but the detector response is included as additional

term to the cost function of the 3DMM fitting. Here the predefined features are used again. The approach is called *self-adapting feature layers* (SAFL) because each feature detector response forms an additional 2D array, or layer, that is used along with the three color channel layers which form the input image.

The predefined significant features can be found in Fig. 6.1.

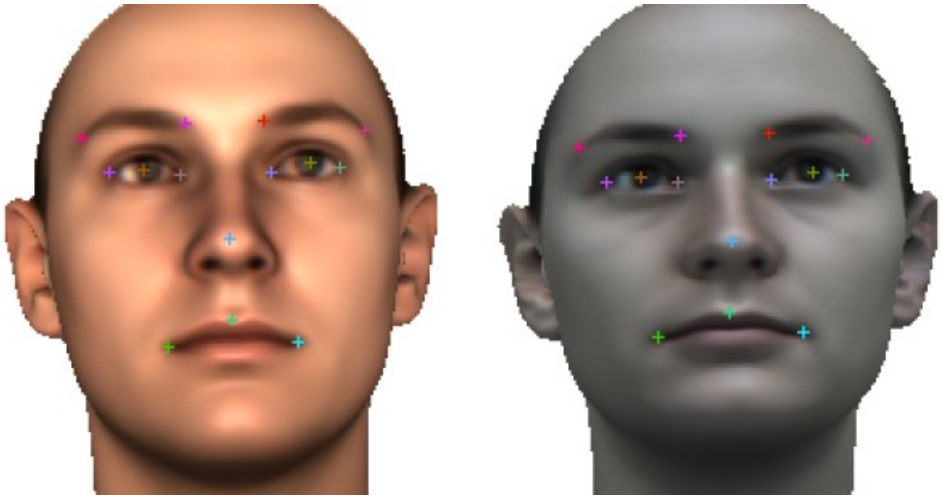


Figure 6.1: **Positions of significant features:** On the left the positions are marked on an early fitting result. On the right the positions are marked on a later fitting result. The head has adapted its color to a grey scale image. They significant features are namely (line by line from top to bottom): left corner of the left eyebrow, right corner of the left eyebrow, left corner of the right eyebrow, right corner of the right eyebrow, left corner of the left eye, sclera of the left eye, right corner of the left eye, left corner of the right eye, sclera of the right eye, right corner of the right eye, tip of the nose, left corner of the mouth, Cupid's bow, right corner of the mouth.

The templates are cut out of the rendered entire current fitting result I_{model}. The match score $\mathbf{C_F}$ is measured using normalized cross correlation (Burger and Burge [BB08]), which we found to be more reliable than alternative choices (cf. Chap. 4.3):

$$\mathbf{C_F}(x,y) = \frac{\sum_{(p,q)\in R} (I(x+p,y+q)\cdot R(p,q)) - N\cdot \bar{I}(x,y)\cdot \bar{R}}{\sqrt{\sum_{(p,q)\in R} (I(x+p,y+q))^2 - N\cdot(\bar{I}(x,y))^2\cdot \sigma_R}} \tag{6.1}$$

where I is the original image and $\bar{I}(x,y)$ its local mean value around the current position (x,y) in a template-sized area, R is the current template (or reference image) and $\bar{R}$ its mean value (over all (p,q)), σ_R is the variance of the template values and N is the number of template values ($width\cdot height$). Only $\bar{I}$ has to be computed for every (x,y). The other three components ($\bar{R}$, σ_R, N) can be precomputed. Note that $\forall(x,y)\in I : \mathbf{C_F}(x,y)\in[-1,1]$, with 1 representing a maximum match and -1 a maximum mismatch. For color images, $\mathbf{C_F}(x,y) = \frac{1}{3}(\mathbf{C_{F,red}}(x,y) + \mathbf{C_{F,green}}(x,y) + \mathbf{C_{F,blue}}(x,y))$.

The optimal feature position (x_o, y_o) than is represented by the position of the maximum matching score:

$$(x_o, y_o) = \max_{(x,y)} \mathbf{C_F}(x, y) \tag{6.2}$$

and the updated position gets integrated into the existing gradient descent procedure and enforces the model to move there by directional constraints.

6.1 Detection of Feature Positions for Rough Alignment

In the first approach big templates are needed for the rough alignment. Templates are not cut out around each significant feature but the current positions of them after initialization are used for three big templates only: left eye, right eye and the mouth. The dimensions of the eye templates are assigned by the positions of the corners of the eyebrows and the corners of the eyes. The dimensions of the mouth template are assigned by the corners of the mouth and the position of Cupid's bow. So the template size is not only depending on the size of the head but also on its orientation. This is because rotation changes the proportions. However, this strategy ensures always including the important parts to the template and searching for them.

The position of the maximum match score marks the middle of the template and has to be converted to get the positions of the significant features included. These new feature points get integrated into the existing gradient descent procedure (contributing to E_M of the cost function, Eqn. 2.7) and enforce the model to move there by directional constraints.

Fig. 6.2 shows some matching results. The position of the maximum match score and the converted feature positions are marked. Also the templates of one example with their corresponding matching scores are shown. The first two examples show good matching results. Only the corners of the mouth of the first example are not exactly at the right positions. The mouth template is a little bit too small. The center of the mouth (the center of the template) marked with a green X is detected right but converting the feature positions the corners of the mouth are marked too far to the middle. In the third example the left eye was not detected right. The initialization angle is smaller than the real rotation angle of the head in the input image. This causes an eye template looking a little bit different to the eye in the image. At the matching scores in the lower row on the left it can be seen that there is an area of high scores further up at the right (also marked red) besides the maximum (marked with a blue X). There would be the right position. There is not the maximum match score and so the other position is taken.

This approach is only suited for the improvement of the rough alignment at the beginning of the whole fitting process. It has to be followed by one of the other approaches afterwards.

6.2 Refinement of Given Feature Positions

To refine given feature positions at the second approach templates are cut out around each of them. The template size is predefined relative to the head size s_H (distance between a vertex

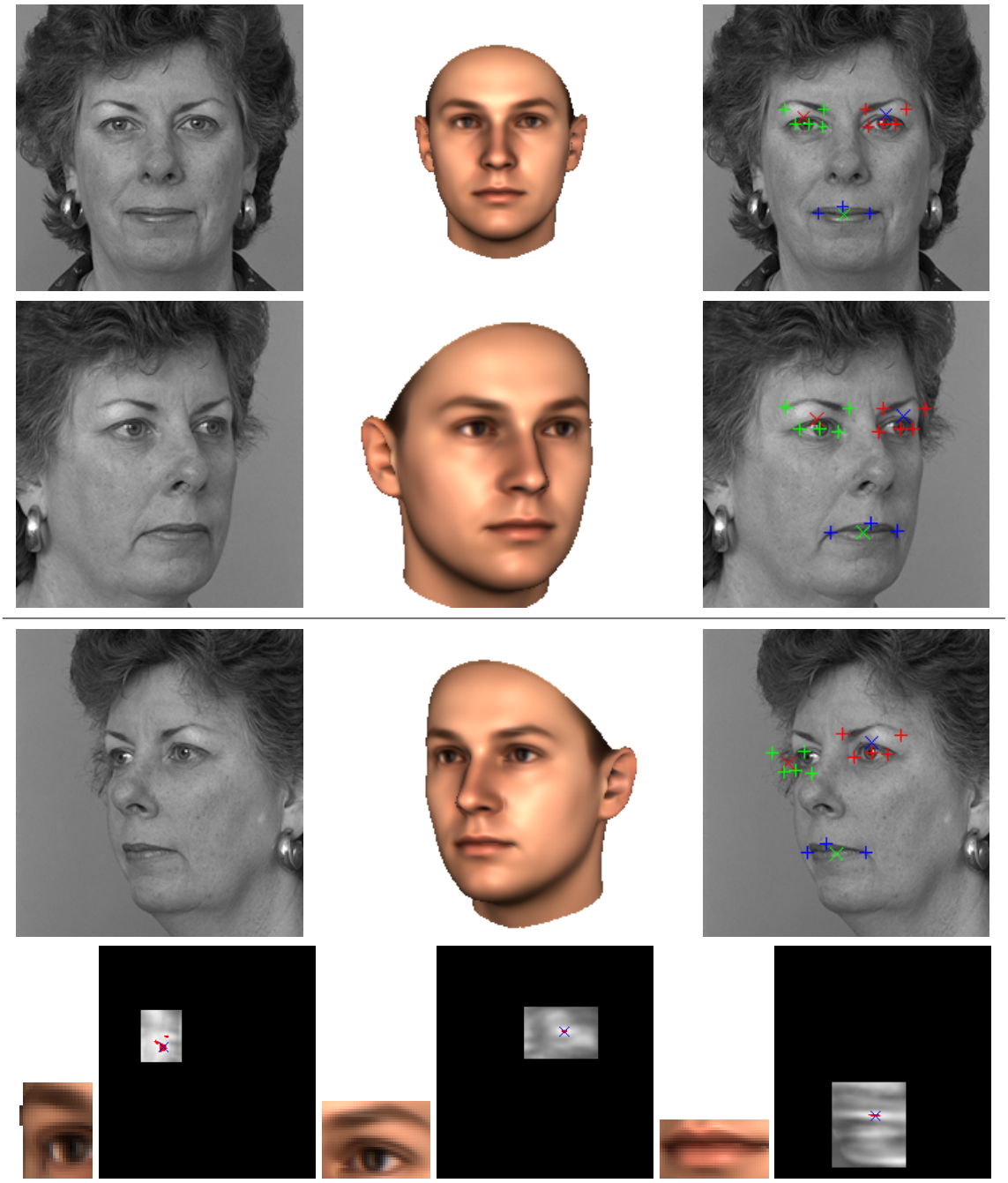


Figure 6.2: **Matching results using big templates for rough alignment:** In the first three rows from left to right the input image (face region detected by AdaBoost [VJ01]), the start model rotated by the passed over angle and positioned depending on the face region and the detected feature positions marked on the input image are shown. The positions of the maximum match scores are marked with X and the converted feature positions are marked with $+$. The last row shows the cut out templates of the third example and the corresponding matching results. The matching scores are measured only in a certain region of interest (ROI) with high possibility for the feature position corresponding to the face region and the rotation angle. The values within the ROI are rescaled to the range [0,255]. The areas marked red contain values ≥ 245 and the maximum (255) is marked with a blue X.

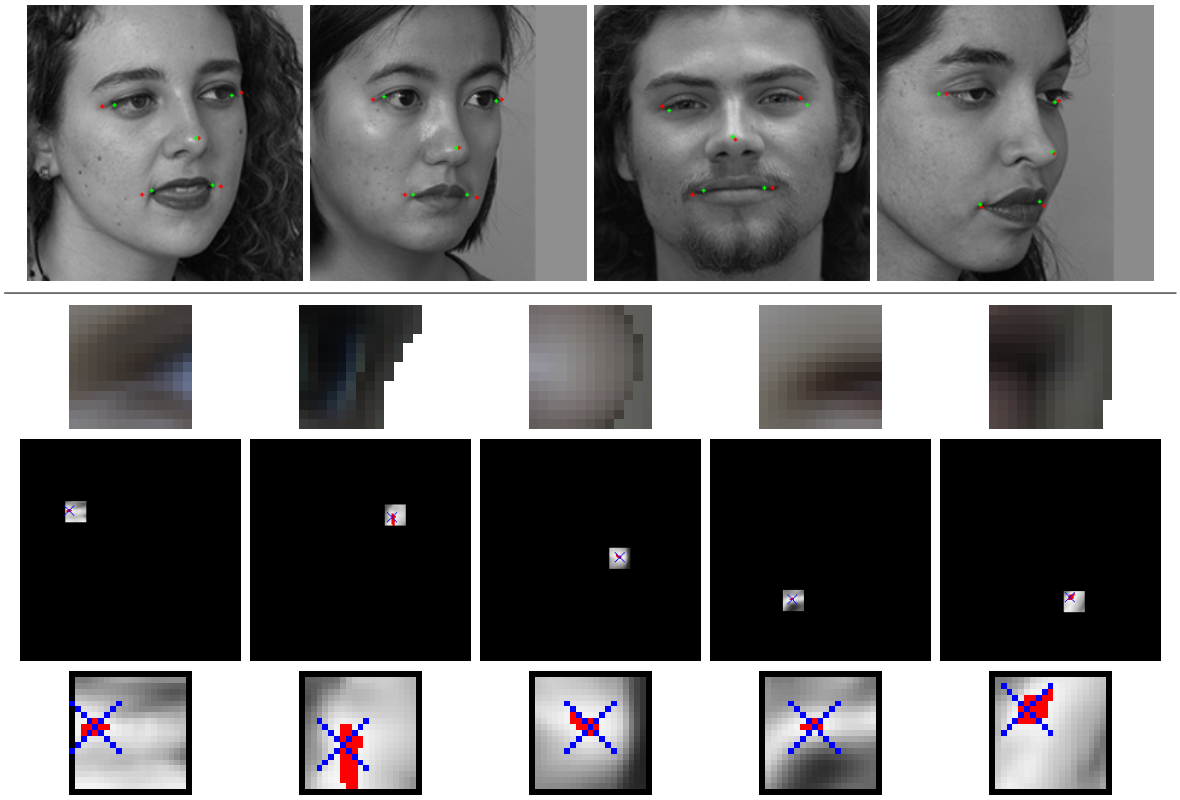


Figure 6.3: **Refinement of given feature positions:** The top row shows four examples with refined feature positions. The given positions are marked red and the new positions after TM are marked green. In the lower rows we see the cut out templates (of the given features, from left to right: outer corners of the left and right eye, the tip of the nose and the corners of the mouth), the corresponding matching scores, and close-ups of the respective ROIs. The templates belong to the first example. The values within the ROIs are rescaled to the range [0,255]. The areas marked red contain values ≥ 245 and the maximum (255) is marked with a blue X.

on the top of the forehead and one on the bottom of the chin, in pixel units). The same size $(\frac{1}{18}s_H) \times (\frac{1}{18}s_H)$ is taken for all because it is not known what kind of features are given and what dimensions would be the most valuable (like a larger horizontal dimensioning for the mouth or a larger vertical dimensioning at the eyes including a part of the eyebrows to the template). If there are any contour points given, they are retained and the updated positions of the other features are used to enforce the model to move there by directional constraints again.

Results of the feature refinement are shown in Fig. 6.3. At the first two examples all feature positions get refined quite well. At the third example the corner of the right eye is detected too far to the right and the corners of the mouth are detected a little bit too far to the middle. At the fourth example the corner of the left eye is detected too far to the left. The matching scores corresponding to this can be found on the left of the lower row. The maximum (marked with a blue X) is positioned in the middle of the highest values (marked red). On the right there are also

high values (white areas). There the real corner is located but not the maximum matching score. At the matching scores of the corner of the right eye (second template) and the right corner of the mouth (last template) the areas with high values are quite big. At the eye this is because of having background information included in the template. The most remarkable characteristic of the template is the difference between foreground and background. So the TM searches more for the contour than for the corner of the eye. At the corner of the mouth the area with high matching scores is placed around the right border of the mouth. But even with the human eye the right position can not be detected precisely.

At this point the performance is heavily dependent on the first given features points (no matter whether they had been detected automatically or by hand). The main problem is that it is not known what kind of features are present. So the template dimensions cannot be chosen optimally.

6.3 Refinement of Predefined Feature Positions

At the third approach the previously defined significant features are used again, to become independent from delivered features and their positions, and to be able to scale the dimensions optimally. Template sizes are predefined relative to the head size s_H once more but with different dimensions for different kinds of features: eyes: $(\frac{1}{9}s_H) \times (\frac{2}{9}s_H)$, nose: $(\frac{1}{18}s_H) \times (\frac{1}{18}s_H)$ and mouth: $(\frac{2}{9}s_H) \times (\frac{1}{9}s_H)$. We chose these sizes to make sure that each template contained enough diagnostic features, such as part of the eyebrows in the eye template.

Results of the refinement of the predefined features are shown in Fig. 6.4. In most of the examples the detection of Cupid's bow did not work well. But even for the human eye it is not always easy to find the right position. Therefore using Cupid's bow as a significant feature is not useful any more. The positions of the corners of the mouth can be detected better. They are so robust that the additional third feature is not needed to yield a good fitting result. At the first example the corner of the eyebrow has not been detected. The eyebrow merges into the hair. Their colors are quite similar. High matching scores can only be found at the area where the hair color gets lighter. The template contains a part of the eyebrow and a part of the background. So the difference between foreground and background is more significant than the eyebrow itself. At the second example the position of the left corner of the right eye is detected too far to the top. The matching scores show a wide area of high scores (white area). But the maximum is at the very top of the ROI. At the third example the right corner of the right eyebrow is detected too far to the left. But there is also a big area with high matching scores containing the right position but the maximum is at the very left of the ROI. Looking at the last three examples the position of the maximum score and the correct feature position are not at the same place. But the right feature position is always contained in an area with high matching scores. So just looking at high values and not only at the maximum the correct positions could be reached. At the last example the left sclera is not detected right. There are two areas with very high matching scores (marked red) but the maximum is at the very left of the ROI. The correct position is located in the area at the right of the maximum.

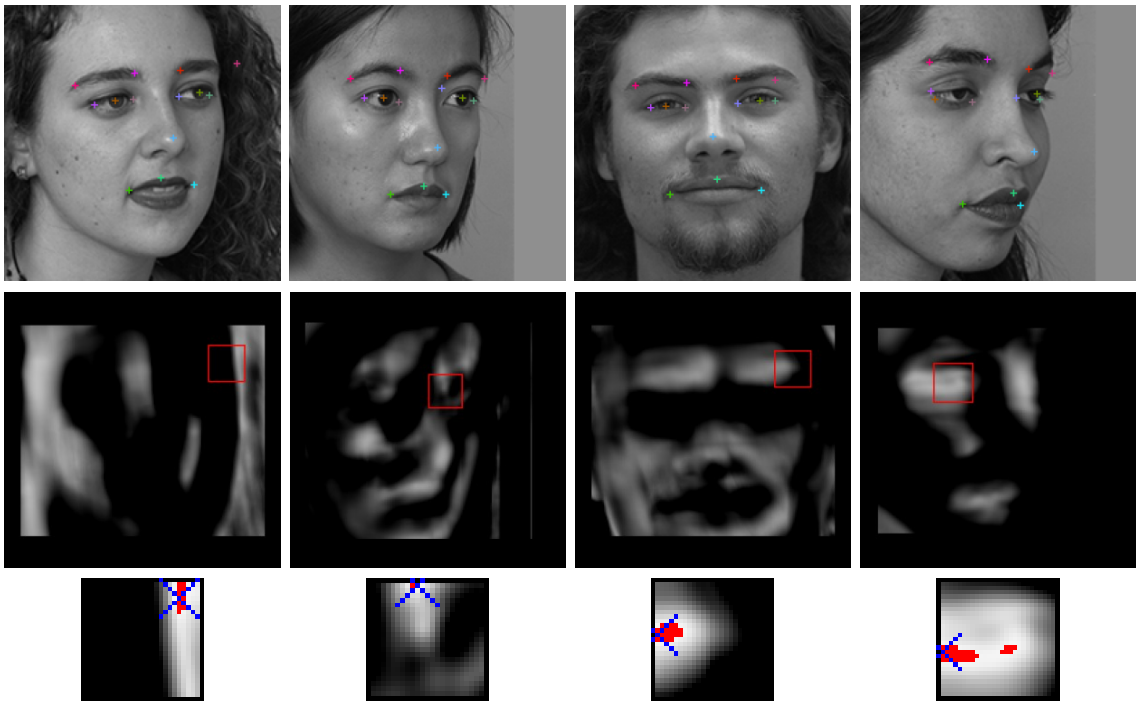


Figure 6.4: **Refinement of predefined features:** From top to bottom we see the input images with the new positions of the significant features marked on, matching scores of single features, over the whole image with the ROIs marked on, and close-ups of the respective ROIs. For each example only the scores are shown which correspond to features not being detected correctly: 1. right corner of the right eyebrow, 2. left corner of the right eye, 3. right corner of the right eyebrow, 4. sclera of the left eye. The values within the ROIs are rescaled to the range [0,255]. The areas marked red contain values ≥ 245 and the maximum (255) is marked with a blue X.

6.4 Self-Adapting Feature Layers

Most existing algorithms (as the ones presented above, too) find the image position with maximum response of the feature detector and pull the corresponding point of the model towards this position. However, on difficult images, it may occur that the feature response at the correct position is not the global maximum (as seen at the last examples). Therefore, we propose a strategy that forms a tradeoff between high feature detector response and a plausible overall configuration. This is achieved by including the value of the feature detector response as an additional term in the cost function of 3DMM fitting, rather than the 2D distance between the current feature position of the model and the position of the global maximum. Each feature detector response forms an additional 2D array, or layer, that is used along with the three color channel layers which form the image. On a more general level, the approach presented in the this section introduces a new, high-level criterion for image similarity to analysis-by-synthesis strategies. In fact, this can be implemented with any feature detector or any other local descriptor of image properties.

Here we present a novel way to include facial features into model fitting. Our proposed self-adapting feature layer (SAFL) approach is built on top of the 3DMM and introduces a novel criterion in the cost function. The goal is to reduce the influence of the (potentially unreliable) initial feature positions that are used in E_M: They are only used for the first coarse alignment of the head, and discarded later. After coarse alignment and a first estimation of the illumination, the term E_M in the cost function (Eqn. 2.7) is replaced by new E_{F_i}, that will be explained below, with $i = 1 \ldots 7$, for the set of 7 feature positions to be refined, weighted with η_F:

$$\mathbf{E} = \eta_I \cdot E_I + \eta_F \cdot [\sum_{i=1}^{7} E_{F_i}(x_{F_i}, y_{F_i})] + \eta_P \cdot E_P \tag{6.3}$$

The features are: the tip of the nose, the corners of the mouth, and the inner and outer corners of the eyes. For feature point i, we know which vertex k_i of the model it corresponds to, and using perspective projection we get the current position (x_{F_i}, y_{F_i}) in the image I_{model}.

Once every 1000 iterations, the entire current fitting result I_{model} is rendered, and templates are cut out around the current feature positions (x_{F_i}, y_{F_i}). Template sizes are predefined relative to the head size s_H like at the approach using the directional constraints: eyes: $(\frac{1}{9}s_H) \times (\frac{2}{9}s_H)$, nose: $(\frac{1}{18}s_H) \times (\frac{1}{18}s_H)$ and mouth: $(\frac{2}{9}s_H) \times (\frac{1}{9}s_H)$.

The new E_{F_i} in (6.3), based on TM, are

$$\mathbf{E_{F_i}}(x_{F_i}, y_{F_i}) = 1 - \mathbf{C_{F_i}}(x_{F_i}, y_{F_i}). \tag{6.4}$$

where $\mathbf{C_{F_i}}$ is the normalized cross correlation (cf. Eqn. 6.1) again.

The weight η_F is constant and scales the sum of all E_{F_i} to the same range of values as the image difference E_I.

Templates R and cross correlations $\mathbf{C_{F_i}}$ are updated once every 1000 iterations of the fitting algorithm. To reduce computation time, $\mathbf{C_{F_i}}(x,y)$ is calculated for each feature i only in a region of interest (ROI: $\frac{1}{9}s_H \times \frac{1}{9}s_H$) around the current position (x_{F_i}, y_{F_i}). Even in the first iteration, these positions can be assumed to be approximately correct, and also the head size s_H that defines the (constant) template size will be in the right order of magnitude, due to the vague initial feature coordinates.

In intermediate iterations, $\mathbf{C_{F_i}}$ remain fixed, but the positions (x_{F_i}, y_{F_i}) for looking up $\mathbf{C_{F_i}}(x_{F_i}, y_{F_i})$ will change. This reflects the fact that the locations of feature points may change faster during the optimization than the appearances of features do. Figure 6.5 gives an overview of the algorithm.

Figure 6.6 shows how the templates (here: outer corner of the left eye) change over fitting iterations when fitting to different images (rows in Fig. 6.6). Over the first six template-adaptions, not much change is observed. At the seventh template at each example, the change is already visible at the eyebrow, after the eighth and ninth adaption the whole templates changed significantly. The major change can be observed at step eight and nine, because this is where fine adjustment starts: The head model is broken down in different regions, and these are optimized separately (see Chap. 2 and [BV99]).

Figure 6.7 shows an example of how the cross correlation result, matching the left corner of the left eye, changes over the fitting iterations. The detail belongs to the result of the third line of Fig. 6.13. There, first the position of the corner of the eye has been displaced to the right to evaluate robustness. As the fitting proceeds, it moves to the left and upward until it reaches the correct position eventually. The position of the ROI also shows where the feature has been positioned when computing the cross correlation. The ROI position reveals the drift of the feature to the correct position.

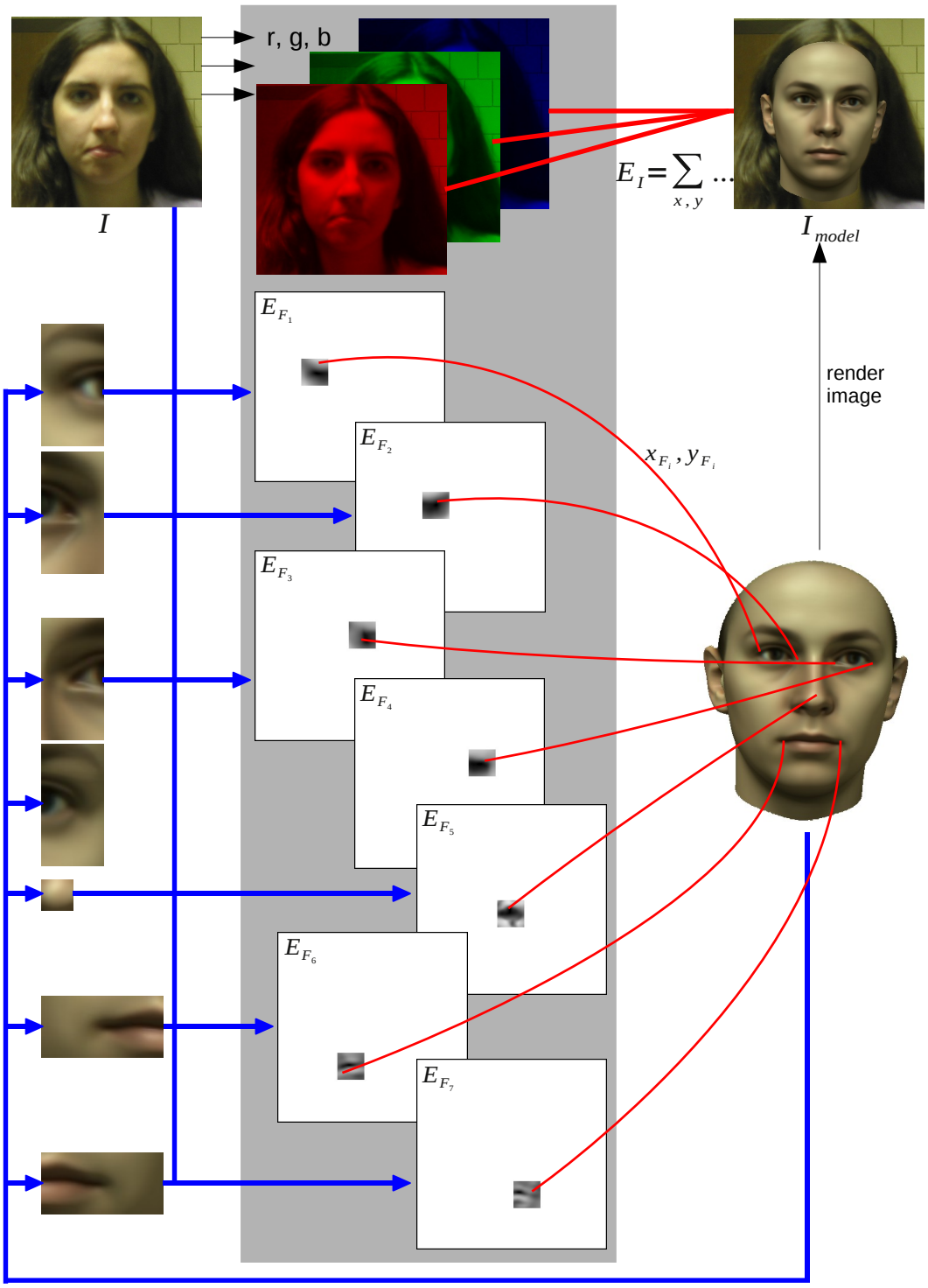


Figure 6.5: **Self-adapting feature layers:** Blue arrows show actions performed only every 1000 iterations: Templates are cut out from the current fitting result. They are compared to the original image I using normalized cross correlation at a certain ROI and from these, the 'feature layers' are generated. Red lines show actions performed every iteration: I_{model} is compared to I, and for each feature i the error value E_{F_i} is taken from the corresponding 'feature layer' at the current feature position (x_{F_i}, y_{F_i}).

It can be seen that the cross correlation turns into a single, wide optimum as the template adapts to the appearance in the image. Note that if the model adapts perfectly to the feature in the input image, $\mathbf{C}_{\mathbf{F_i}}(x,y)$ will converge to the auto-correlation function, and the width of maxima and minima will be determined by the frequency spectrum of the template.

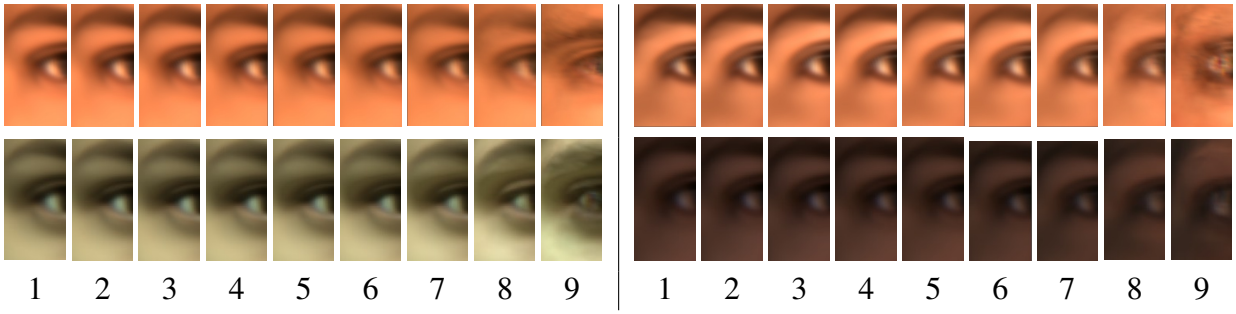


Figure 6.6: **Templates:** changing over fitting iterations. Each 'row' is from fitting to one input image. In the first two examples (on the left), convergence was correct, in the last two (on the right), the corner of the eye moved to the eyebrow.

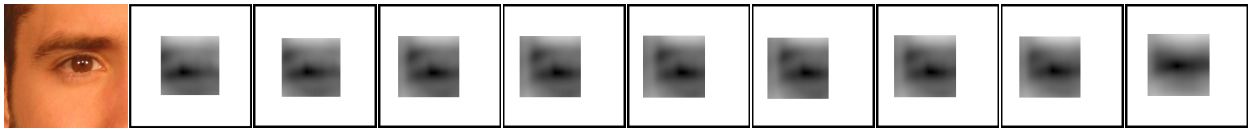


Figure 6.7: **Cross correlation results:** in the ROI, changing over fitting iterations. Dark pixels indicate good matches. These results correspond to the templates in the upper left of Fig. 6.6.

6.4.1 Evaluation of the SAFL Approach

We tested our algorithm on 300 randomly chosen images from the FRGC database [PFS+05], using three images per person and a set of 50 women and 50 men. The only constraint in random selection was that the person did not show an extreme facial expression. Typical examples of the randomly chosen samples, some of them in difficult imaging conditions (focus, illumination, expressions) can be found in Fig. 6.8. The database contains front view images only. We show results on non-frontal views later in this section.

For ground truth in every image, five feature positions (outer corners of the eyes, nose and corners of the mouth) were labelled manually. To simulate scenarios with an unreliable initial feature detector, we perturbed the feature positions randomly:

1. randomly select two (of the five) features to be perturbed
2. randomly select a displacement direction for each feature position
3. displace feature positions by a fixed distance

Figure 6.8: **Typical examples of images per person:** Each line shows three pictures of the same individual [PFS+05]. Here not the whole pictures, but only the facial regions (scaled to the same size) are shown.

We chose controlled perturbation to be able to have an isolated (and repeatable) view on the fitting.

In three different test conditions, we used distances of 5%, 12% and 25% of the vertical distance between eyes and nose. This corresponds to distances of 0.2cm, 0.48cm and 1.0cm in reality on an average sized head. The perturbation ranges are visualized as the radii of the circles in Figure 6.9. Figure 6.10 shows a typical example for each test condition. By using displacement distances relative to the eye-nose distance in the image, rather than fixed pixel distances, we were able to use images at different resolutions.

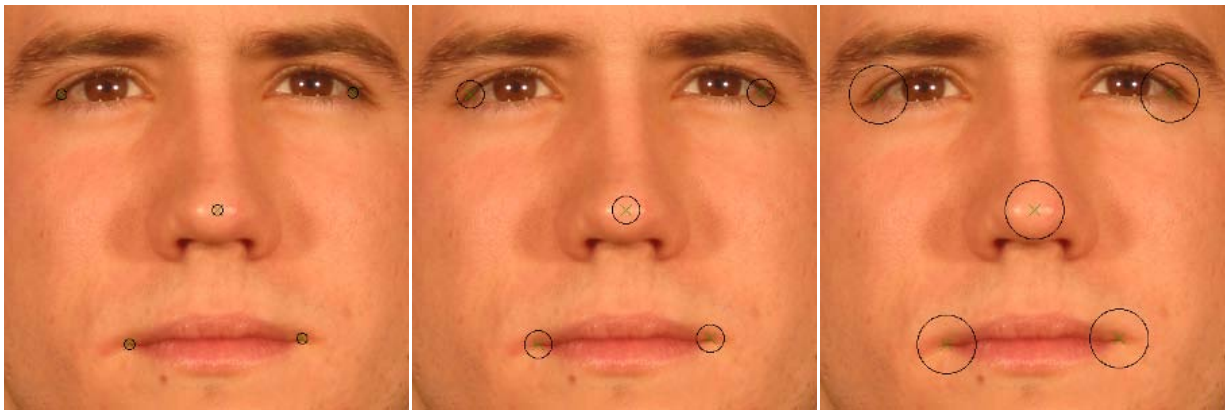


Figure 6.9: **Perturbation ranges:** Circles mark the perturbation ranges of the three different test conditions, from left to right: 5%, 12% and 25%.

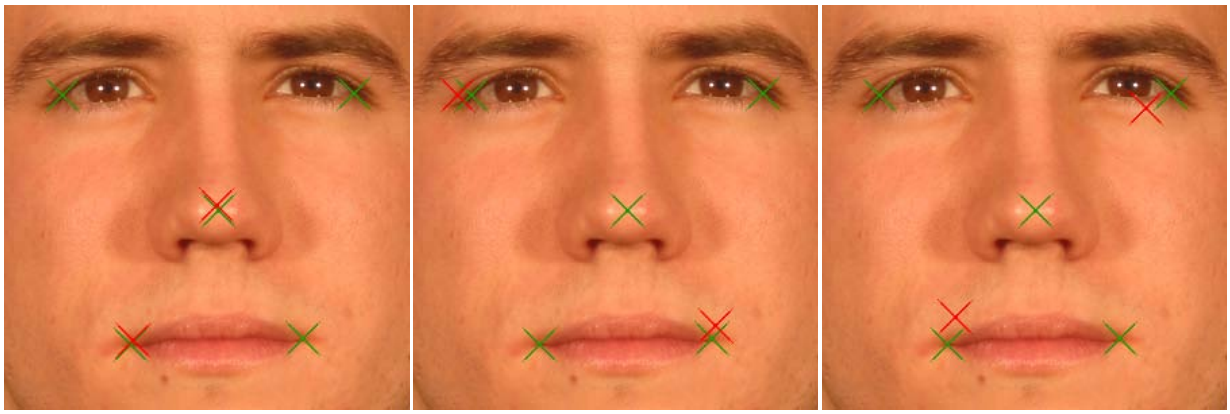


Figure 6.10: **Typical examples of perturbed positions:** Green crosses mark manually labelled feature positions and red crosses mark perturbed positions.

Evaluation Criterion 1: Recognition Performance

To have an independent criterion for the quality of the reconstructions, we evaluate recognition rates from model coefficients in an identification task. Given the linear 3DMM coefficients for shape and texture of the entire face and the facial regions (eyes, nose, mouth and surrounding area), which are concatenated into coefficient vectors $\mathbf{c}$, the algorithm finds the individual from the gallery set with a minimum distance, measured in terms of a cosine criterion $d = \frac{\langle \mathbf{c}_1, \mathbf{c}_2 \rangle}{\|\mathbf{c}_1\| \cdot \|\mathbf{c}_2\|}$ (see [BV03, BKK$^+$08]). For each probe image, a comparison with the other two images of that person and with all three images of all 99 other individuals is performed.

Recognition is tested with the standard 3DMM fitting algorithm ([BV03], see Chap. 2) and with our new SAFL approach for the manually marked feature positions and each perturbation range. The percentages of correct identification can be found in Fig. 6.11.

Due to the difficult imaging conditions, the overall recognition rate is below 50%. In the unperturbed case, both the standard algorithm and the new SAFL approach deliver similar results, indicating that SAFL does not downgrade the system when correct feature positions are given. However, with perturbed features, the recognition rate for the standard algorithm rapidly decreases as the displacements get larger. In contrast, SAFL identification rates remain stable. This demonstrates that SAFL increases the robustness of the fitting for face recognition.

Evaluation Criterion 2: Movement of Feature Positions

We have also evaluated the distances between the ground truth feature positions and the optimized positions after fitting. Figure 6.12 shows the distribution of the average 2D distances of the five features in each test image: The vertical axis is the absolute number of test images (out of a total of 300) where the average feature distance is below the distance threshold indicated on the horizontal axis.

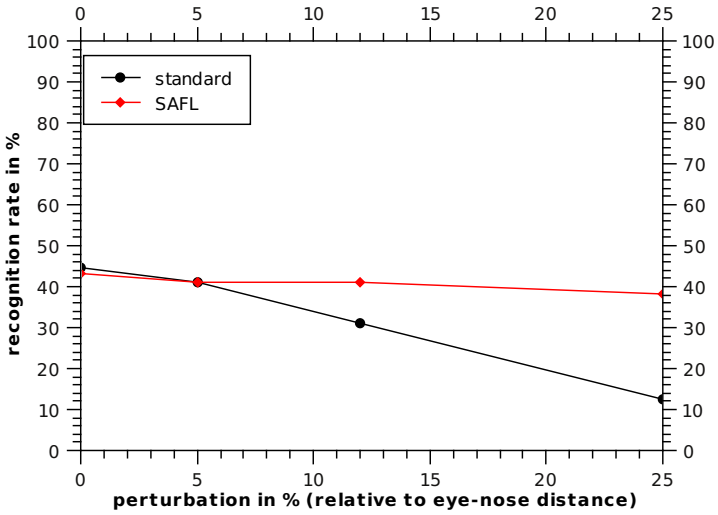


Figure 6.11: **Comparison of recognition rates:** measured in terms of correct identifications (one out of $n = 100$ individuals).

If we do not perturb the starting positions of features, most test images have an average error in final feature positions of 5% to 10% of the vertical distance between eyes and nose, which corresponds to approximately $2mm$ to $4mm$. The standard algorithm performs slightly better than SAFL because E_M keeps the features fixed to the ground truth positions during part of the optimization. It should be noted that the ground truth positions may have some residual uncertainty, because it is difficult to identify corresponding feature positions (pixel in the image - vertex on the model) exactly by hand. This may explain why the benefit of SAFL in this evaluation criterion becomes visible only on a larger scale of feature distances, i.e. when larger perturbations are applied (Figure 6.12, second diagram). These results are consistent with the face identification rates in Fig. 6.11, where we found similar performance for unperturbed initial features, but a significant improvement for perturbed features.

Evaluation Criterion 3: Comparison with Alternative Versions of the Algorithm

To confirm our choice of making the features self-adapting and of using the image layer approach rather than considering only the position of maximum feature response, we evaluated some alternative versions of the algorithm:

a. Standard algorithm, but the initial (perturbed) feature positions (which contribute to E_M) are replaced after iteration 1000 by the position of the maximum output of template matching (TM). The idea is that a single TM early in the process would be enough to refine the perturbed feature positions. The template is created after a coarse estimation of pose, lighting and appearance.

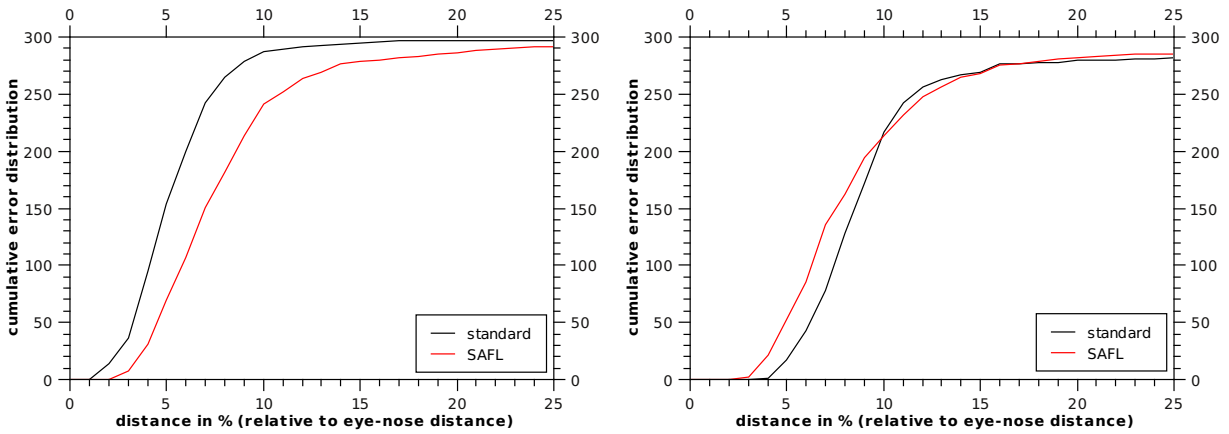


Figure 6.12: **Movement of feature positions:** Left diagram: The manually labelled feature positions (without perturbation) were used for initialization of the reconstruction. Right diagram: 25% perturbation. x-axis: Average distance between the manually labelled positions and the resulting positions after reconstruction for a given test image. y-axis: Cumulative error distribution (absolute number of images with distance below threshold, total 300.) Black line: standard 3DMM fitting algorithm, red line: proposed algorithm SAFL.

b. Use self-adapting templates that are adapted every 1000 iterations, but consider only the maximum TM output rather than layers E_{F_i}, and use it in E_M instead of the initial features (standard algorithm). This condition tests whether the layer approach E_{F_i} is superior to E_M which just pulls features to the positions of maximum TM output.
c. Compute the cross correlation results only once for each feature, and use this to get $E_{F_i}(x_{F_i}, y_{F_i})$ for the rest of the reconstruction algorithm. This condition verifies the benefit of adaptiveness in the SAFL algorithm.

Table 6.1 shows the recognition rates of the standard algorithm, the proposed SAFL and the additional tested versions a, b and c listed above. The results of the versions (a) and (b) are much lower than all others, indicating that the cost function E_{F_i} performs better than searching for the maximum output of TM only. The recognition rates of setting (c) come close to the ones of the SAFL approach, but are still inferior, showing that self adaptation is useful. We conclude that both main ideas proposed in this section (making the features self-adapting and using the image layer approach) make a significant contribution to the stability of 3DMM fitting in a recognition scenario with partially unreliable initial features.

The computation times for the different approaches according to different facial region sizes can be found in Tab. 6.2. When TM is used, computation times depend critically on the size of the facial region. It would have been worth considering to perform template matching at a lower image resolution.

Figure 6.13 shows 4 reconstruction results. We show only one perturbation level per example in this figure in order to give an idea of what the perturbations look like and what the reconstructions

perturbation in %	0	5	12	25
standard	44.6	41.0	31.0	12.3
a	25.0	25.3	25.0	19.6
b	14.0	14.3	11.3	11.3
c	42.0	40.6	39.0	33.0
SAFL	43.3	41.0	41.0	38.3

Table 6.1: **Recognition rates:** percentage of correct identifications for all algorithms tested on all perturbation ranges.

facial region	std.	a	b	c	SAFL
541^2px	64	92	103	146	160
1054^2px	67	120	232	331	456

Table 6.2: **Computation times:** in seconds, measured on an Intel^R Core^{TM}2 Duo CPU E8300 @ 2.83GHz (single threaded).

using this feature positions look like. In the left column, the feature positions are marked on the input images with colored crosses: green for manually labelled positions and red for perturbed positions. The second column shows close-ups of the features randomly chosen for perturbation. In the first row, the manually labelled feature positions were used. Here the SAFL approach got into a local minimum, moving the eyes to the eyebrow positions. In the second row, two randomly chosen feature positions were perturbed 5%. Here the perturbation is quite small, but SAFL outperforms the standard algorithm in reconstruction. In the third row, two randomly chosen feature positions were perturbed 12%. Here it can be seen how much the perturbed feature positions influence the standard algorithm. The reconstruction using SAFL is plausible. In the fourth row, two randomly chosen feature positions were perturbed 25%. We would like to add that both algorithms may produce suboptimal results occasionally, and we selected four typical examples here.

In Fig. 6.14 more reconstruction results are presented, showing examples for each perturbation range for which the new algorithm delivered the same result as the standard algorithm, improved the result or sometimes made it even worse.

At the upper row (no perturbation) it can be seen that this can also cause defective reconstructions both using the standard algorithm and the new algorithm. This may occur due to the fact using only five feature positions as start values but no more guiding features for the model to cling to like a point on the earlobe or some contour points. The results on the left and in the middle show that for each perturbation range it is also possible to get convincing reconstructions even using the standard algorithm. The middle column shows what kind of errors may sometimes occur using the new algorithm, feature positions drifting to wrong positions, like eyes moving to the

eyebrows or the whole face moving away. Indeed the left and right columns also show how the new algorithm preserves or even improves the reconstruction results, what has also been verified by the recognition rates yet.

Non-frontal Faces

To have also quantitative results on non-frontal faces we did some additional tests on the FERET database. The setting was chosen like for the FRGC data: for ground truth in every image, only five feature positions (outer corners of the eyes, nose and corners of the mouth) were labelled manually. Two (of the five) features have randomly been selected for perturbation and have been displaced by a fixed distance to a randomly selected direction. In a rank 1 identification experiment (1 out of 194) we used *ba* images as gallery and *bb* (rotated views with a mean rotation angle ϕ of 38.9°, cf. [BV03]) as query images also considering perturbation ranges from 0% to 25%. The percentages of correct identification can be found in Fig. 6.15. The low recognition rates compared to [BV03] are due to the fact that we do not rely on contour points in this evaluation. Our goal here is to demonstrate the usefulness of the new algorithm compared to the standard one.

Results of non-frontal views are shown in Figure 6.16. We show only this randomly chosen examples with a perturbation range of 12% in this figure in order to give an idea of what the reconstructions look like. At the upper line the SAFL approach improved the reconstruction. On the left side it is obvious but on the right side it can only be seen at the forehead and at the chin. At the lower line the SAFL approach does not really improve the reconstruction. On the left side the model fits better to the image (it is rotated) but it is still too small and on the right side it fits better at the forehead and at the ear but chin and nose are deformed.

Real World Scenario

To test the new algorithm in a *real world scenario* we chose the feature detector of [BKK$^+$08] to automatically detect the feature positions on the 300 faces taken from the FRGC database. Performing a rank 1 identification experiment again the standard algorithm delivers a recognition rate of 29.$\bar{6}$% and the new algorithm yields a recognition rate of 39.0%. This results are comparable to the recognition rates of the former experiment with random perturbation of 12%.

Both the results of the *isolated view on the fitting* (using fixed perturbation ranges to move the feature positions to random directions) and the results of the *real world scenario* (using a feature detector for initialization) show that the approach of the self-adapting feature layers is a big stride towards independence from manual initialization or precise automatic initialization.

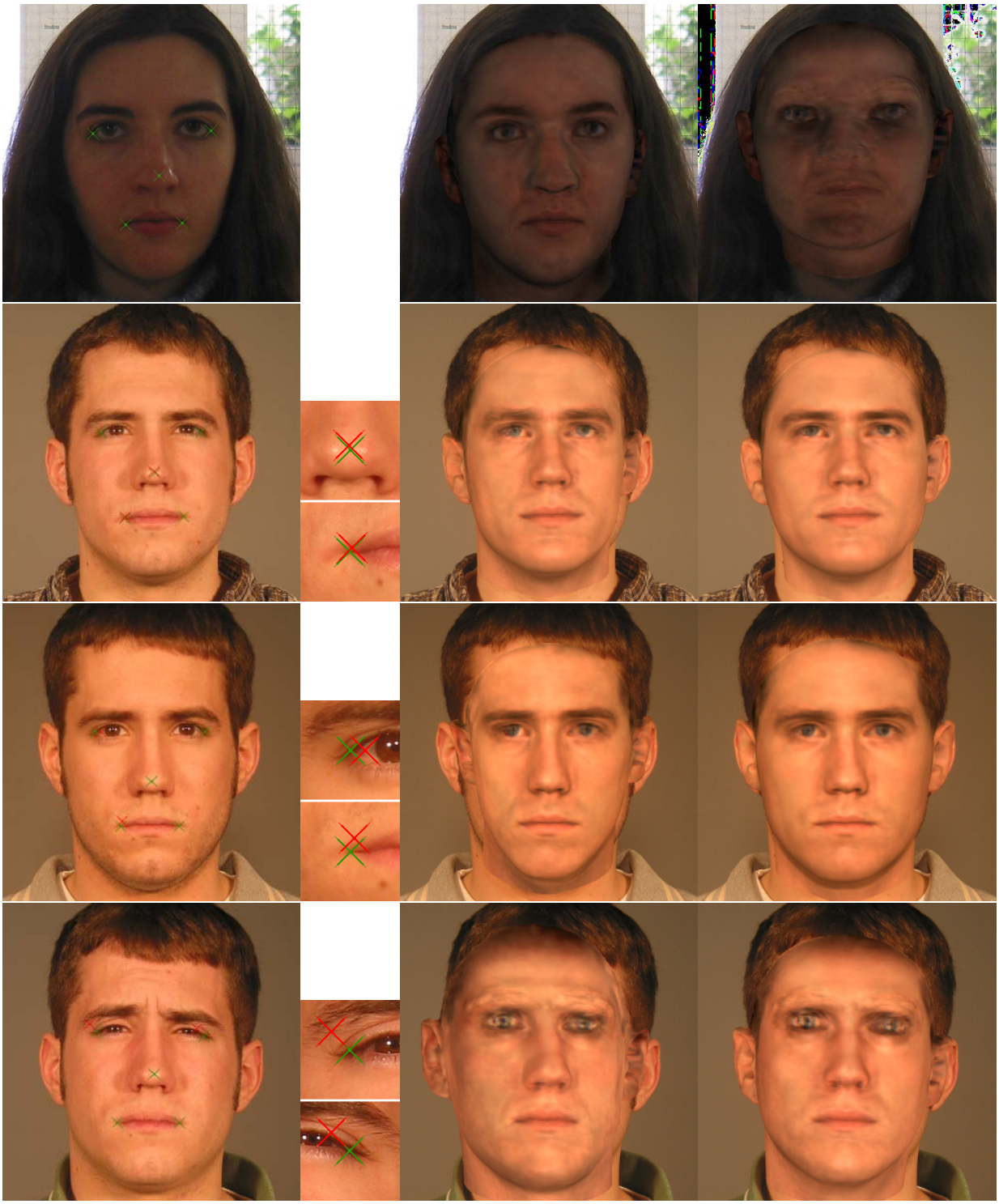


Figure 6.13: **Reconstruction examples:** Each row shows 3DMM fitting for one test image. In the examples in row 1, 2, 3, 4, we used 0%, 5%, 12% and 25% perturbation, respectively, relative to eye-nose distance. From left to right: positions marked on the input image, close-ups of the perturbed feature positions (green: manually labelled, red: perturbed position used), reconstruction with the standard algorithm, reconstruction with the new SAFL approach.

Figure 6.14: **More reconstruction results:** Each pair of images shows the reconstruction result of the standard algorithm on the left and the result of the new algorithm on the right. From top to bottom always two rows belong to one perturbation scenario from 0% to 25%. From left to right pairs where the results are nearly the same, pairs where the standard algorithm was even better and pairs where the new algorithm improved the reconstruction, are shown.

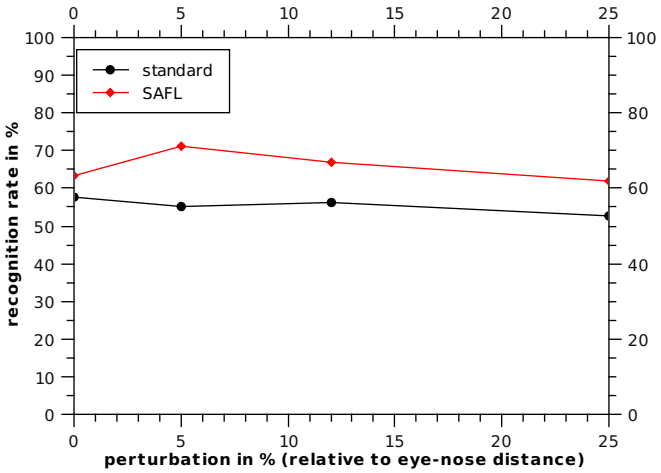


Figure 6.15: **Comparison of recognition rates including non-frontal faces:** measured in terms of correct identifications (one out of $n = 194$ individuals) using images from the FERET database *ba* as gallery and *bb* as query images.

Figure 6.16: **Reconstruction examples of rotated views:** Examples were chosen randomly out of the reconstruction results with a perturbation range of 12%. Each pair of images shows reconstructions using the standard algorithm on the left and using SAFL on the right.

Chapter 7

Contour Fitting

Adapting to the contour of the face is an important part of the whole reconstruction process. The overall shape of the face is not exclusively determined by the contour line but essentially influenced by it. For improving the contour fitting of the 3DMM, different approaches are presented and analyzed in the following.

If a current fit is larger than the real face in the input image, this gets penalized by the color difference E_I having very different color values at the border area of the model compared with the underlying background. If the model is smaller than the face, it is hard to detect any errors based on color information because all color values inside the model are valid skin color.

To overcome this, the original 3DMM by Blanz and Vetter [BV99, Bla00] already contains a simple contour adaption. In addition to the 40 random vertices of the model per iteration to compute the color difference E_I, 8 vertices on the contour of the model are picked randomly. At each of these contour vertices a look at the values along its normal (projected onto 2D) yield an additional error term. A contour triangle (later reduced to a contour vertex, positioned at the middle of the triangle) of the current model can be detected by the fact that its normal has to be perpendicular to the viewing direction. To exclude inner edges another criterion has to be fulfilled: at least one vertex of the triangle must belong to the background. Therefore the z-buffer, containing the depth information of the whole scene, is used. If there is no depth information available (z-buffer = 0) for a vertex, it belongs to the background. For excluding the artificial edges of the model (forehead, neck and back of the head), which would deliver false results, the orthogonality of the normal and the viewing direction is checked. To ensure that face color merges into any other color at the contour line of the model, the difference between the color value of the contour vertex k and the value of the input image at a position moved along the normal is squared and subtracted from the color difference E_I to form the new E_I':

$$E_I' = (I_{model,k} - I_{image}(\vec{x}_k))^2 - 0.5 \cdot (I_{model,k} - I_{image}(\vec{x}_k + \varepsilon \cdot \hat{n}_k))^2 \tag{7.1}$$

with $I_{model,k}$ representing the color value of the current vertex, $\vec{x}_k$ containing its coordinates and $\hat{n}_k$ being its normal projected into the image plane. $\varepsilon = 0.01$.

Besides that, it is possible to pass over some contour positions as features. To 'help' the model, pixels of the input image lying on the contour can be marked as such, constraining the model contour to move to it. Manually (or automatically) defined feature positions at the contour contribute to E_M. The contour is not only taken into account for the image error E_I', like described above.

Keller et al. [KKV07] also worked with the 3DMM and showed how important contour matching is for the whole system, but they also stated that it is not expedient using it exclusively. They investigated the robustness of reconstructing the 3D shape using only the occluding contours (silhouette and inner contours), and how much information is involved in one single contour image. They found out that the resulting head has more or less a random shape. For feature extraction they used a universal edge detector (CannyEdge). To get a 3D reconstruction of a human face from occluding contours they needed to optimize the model and the camera parameters by minimization of the distance between the contours of the model and the edge image. The main problem is, that the edge detector delivers edges, but not contours. Although most of the contours along the edges get detected, they are noisy and incomplete. To overcome this, they used a more robust distance function ignoring both not-comparable edges and not-comparable contours. Their distance function contains only the sum of Euclidean distances between edge pixels and closest contour pixels. Therefore they use linear Euclidean distances with a plateau, limiting the maximum measured distance. All distances larger than a threshold are set to the value of the threshold. Only the closest contour pixels are important. The exact values of larger distances do not matter. They used a reduced version of the 3DMM (with only less than one third of the vertices) and maximum 30 principal components. To glean the contours of the current fitting result, first some candidates are found. These candidates are the edges between two opponent polygons of the mesh. If one polygon is front facing and the other is back facing, their shared edge possibly belongs to the contour. Covered edges are excluded by using the z-buffer. The remaining edges are projected onto the image plane to get an edge image comparable to one of an edge detector.

In 2007, Wang [Wan07] compared two different Laplacian Operator-Based Edge Detectors, both using a convolution with a RoundMeanFilter and a convolution with a NoiseSmoother combined for generating one edge image. Taking the known Laplacian filter as baseline, a Laplacian of Gaussian (LoG), he first presented an optimal edge-matching filter-based edge detector (OED) replacing the NoiseSmoother by another (existing) edge filter. Second, he introduced a multistage median filter-based edge detector (MED) replacing the NoiseSmoother by a multistage median filter, keeping the details and smoothing the noise at the same time. Parting the filter into four subwindows, for each window the median has to be computed and also minimum and maximum value out of all four medians have to be computed. The median of the original value, and the measured minimum value, and maximum value yields the filter output. To reduce the detection of noise here, like in other edge detection algorithms, a threshold is introduced to eliminate small detected amplitudes. Wang uses a Maximum A Posteriori estimate to get the optimal threshold. However, if the weighted median ($\frac{\text{median}}{0.6745}$) of all values of the current edge detection result is bigger than the mean of all squared values of the current result, the threshold becomes infinity and nothing is detected at all. He tested the algorithms on an image showing a pile of peppers,

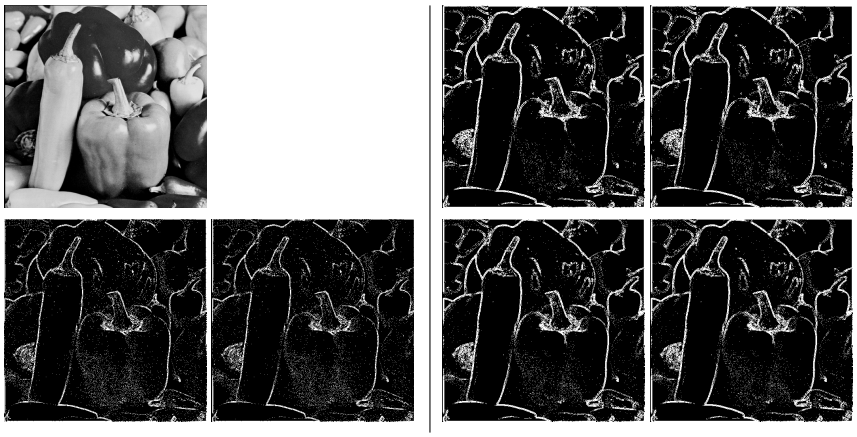


Figure 7.1: **Laplacian Operator-Based Edge Detectors:** The used input image and the edge detection results of the three different methods LoG, OED and MED are presented. Each pair contains both the result of the pure edge detection and the result using the optimal threshold. For better sight all values > 0.1 are set to 1. The figure is following [Wan07].

and showed their effectiveness (cf. Fig. 7.1).

Wang tested only on this image showing a pile of peppers, and an image showing a balloon. We tested both the detector MED presented by him and CannyEdge (with two different thresholds) on typical face images used in our work, determining the most suitable kind of detector for our purpose. The results can be found in Fig. 7.2. Looking at the outer image pairs, MED detected the edges quite good, but a lot of other details not representing edges are detected, too. However, using the optimal threshold, as defined by Wang, the threshold tends to infinity in three of four cases and no edges are detected any more. At the image pair in the middle, many little edges are detected, which are not helpful for identifying a contour. Changing the threshold does not improve these edges significantly. CannyEdge yields continuous good results. The evaluation showed that the CannyEdge-Detector is the most suitable for this kind of purpose and thus will be used in the following steps.

Romdhani and Vetter [RV05] used the CannyEdge-Detector to get a binary mask of the edges of the input image. They included the edge information into their cost function. They extended the cost function using the distance between the contour points of the model and their next neighbours (detected by ICP) in the binary mask.

7.1 Contour Fitting through Edge Detection

In our first approach we are using the CannyEdge-Detector as well, but we do not include the whole contour into the cost function. As at the original algorithm we take only 8 contour points into account but not for every iteration but only once every 2000 iterations. The cost function is not extended from E_I to E_I' but the upgraded contour positions are treated like the given feature points contributing to E_M and enforcing the model to move there by directional constraints.

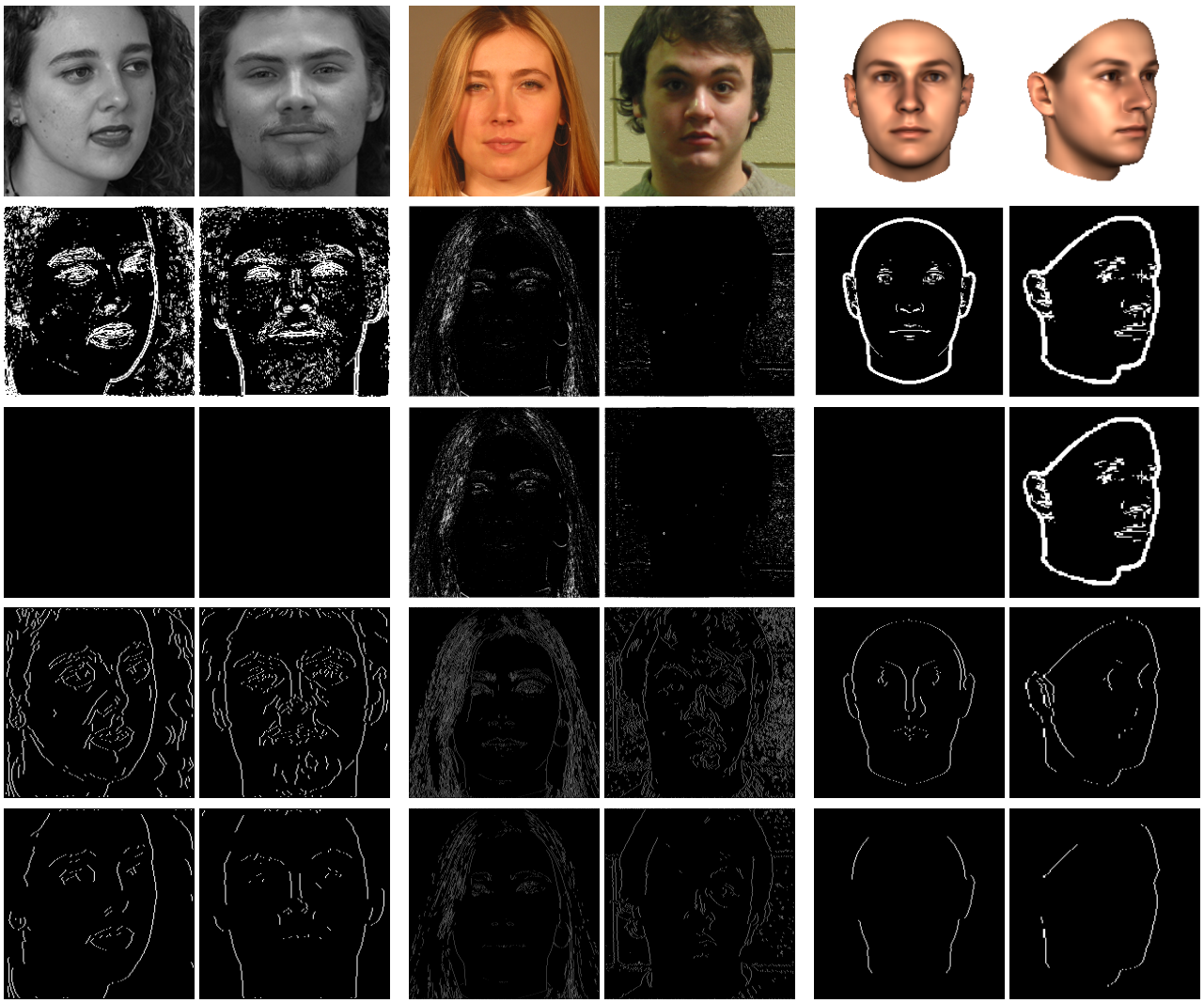


Figure 7.2: **Detection on face images:** From top to bottom the rows show the used input images and the edge detection results of the two different methods MED (both without and with Wang's optimal threshold) and CannyEdge (first with thresholds at 0.4 and 0.75 and second with strong thresholds at 0.6 and 0.92). The first two images are taken from the FERET database [PWHR98], the second are taken from the FRGC database [PFS+05] and the third are renderings of the average head of the 3DMM at different views.

Starting from the position of a contour point of the current fit it is searched along the x-direction (forward and backward in the range $[x_c \quad \varepsilon, x_c + \varepsilon]$) for an edge in the input image resp. in the CannyEdge image. If an edge has been located, its position is set as a contour feature point, which means that it gets integrated into the existing gradient descent procedure.

Figure 7.3: **Contour alignment using CannyEdge image:** From top to bottom the rows show (a) the input images, (b) the results using E'_I, (c) the results using a CannyEdge image, and (d) the CannyEdge images (black lines) overlayed by the contour of the results from (c) (red lines).

Results

Fig. 7.3 shows fitting results using this approach. Here only five automatically (by Breuer et al. [BKK+08]) detected feature points were used for initialization. No hand marked contour points had been passed to the algorithm. At the first example no difference can be observed when comparing the fitting results. Both are aligned to the real contour quite good, which can also be seen at the overlayed edge image. In the second example the new contour alignment improves the adaption, but at the right eye and at the chin the contour adapted to the hair. This is because the alignment is not based on color information but only on the edge image, where the

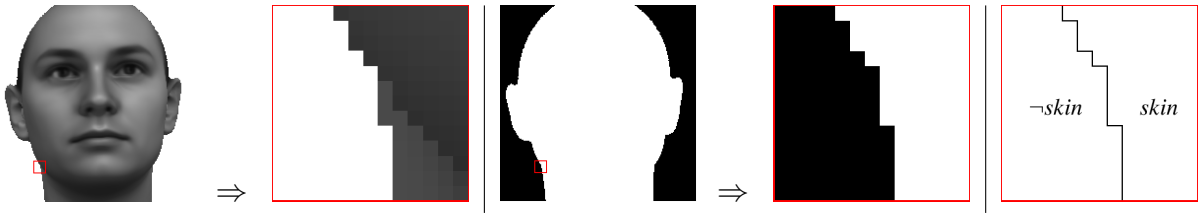


Figure 7.4: **Templates for contour alignment:** On the left and at the middle we see the current fitting result and the corresponding z-buffer output with their cut out templates at one of the contour points. On the right the *skin* and $\neg$*skin* areas are marked on the template.

crossover from hair to background also has been detected. At the two examples at the right the new alignment clearly improves the adaptation. Only at the chin of the woman and at the right ear of the man differences can be found, probably caused by too small sampling at this regions. These examples show, that using only five feature points and no hand marked contour points for initialization can result in faces that are too big, even when E_I' is used. The new contour alignment using a CannyEdge image is able to improve the adaption. But as seen at the second example, the use of edge information only is not free from producing errors.

7.2 Contour Fitting with Special Templates

In the last section (7.1), we have seen how difficult it is to detect reliable edge information in images. Amberg et al. [ARF+07] phrased it like this: 'no general edge detection method is able to perfectly find the silhouette in real world images'. In this section (7.2), we look closely at the use of templates for contour fitting. We are not only searching for edges or contours in the input image. We search for contour structures in the input image, that are similar to templates cut out from the rendering of the current fitting result.

We use special templates with different peculiarities, which will be described in detail in the following: The template size is predefined relative to the head size s_H (distance between a vertex on the top of the forehead and one on the bottom of the chin, in pixel units, like in Chap. 6.4) and set to $(\frac{1}{18}s_H)^2$. Each template is cut quadratic around a contour point of the model. Not only the color information is taken (like for usual template matching), but also the information about the affinity of the underlying pixels to foreground (*skin* area) or background ($\neg$*skin* area). The definition of these areas is done by using the z-buffer. Fig. 7.4 shows an example of a cut out template, the corresponding z-buffer output, and the marked areas.

Starting from the position of a contour point of the current fit, the template is moved along the x-direction (forward and backward in the range $[x_c \quad \frac{1}{18}s_H, x_c + \frac{1}{18}s_H]$, based on the template size) and the underlying values of the input image are observed, according to their belonging to one of the two areas *skin* and $\neg$*skin* (cf. Fig. 7.4):

- mean value of all pixels in the *skin* area of the input image:

$$\varnothing(skin_I) = \frac{1}{\#(p \in skin_I)} \cdot \sum_{\forall p \in skin_I} val(p) \tag{7.2}$$

- mean value of all pixels in the $\neg skin$ area of the input image:

$$\varnothing(\neg skin_I) = \frac{1}{\#(p \in \neg skin_I)} \cdot \sum_{\forall p \in \neg skin_I} val(p) \tag{7.3}$$

- mean value of all pixels in the *skin* area of the current fit of the model:

$$\varnothing(skin_M) = \frac{1}{\#(p \in skin_M)} \cdot \sum_{\forall p \in skin_M} val(p) \tag{7.4}$$

Based on this kind of templates (cf. Fig. 7.4) different evaluation schemes have been implemented:

1. $\|\varnothing(skin_I) - \varnothing(\neg skin_I)\|$, search for the maximum dissimilarity of the mean value of the *skin* area and the $\neg skin$ area of the input image
2. $\|\varnothing(skin_M) - \varnothing(skin_I)\|$, looking at the similarity of the mean value of the *skin* area of the current fit of the model and the mean value of the *skin* area of the input image, searching for the crossover (the step) from high similarity to high dissimilarity
3. $\|\varnothing(skin_M) - \varnothing(\neg skin_I)\| - \|\varnothing(skin_M) - \varnothing(skin_I)\|$, searching for both maximum similarity of the mean value of the *skin* area of the current fit of the model and the mean value of the *skin* area of the input image and at the same time maximum dissimilarity of the mean value of the *skin* area of the current fit of the model and the mean value of the $\neg skin$ area of the input image

Fig. 7.5 shows the assumed ideal evaluation functions for the schemes. The graph on the left represents the selection of the optimal contour position using scheme 1 or 3 and the graph on the right represents the second scheme searching for the crossover from skin to background.

The so detected (best) contour position again is set as a contour feature point, which means that it gets integrated into the existing gradient descent procedure and enforces the model to move there by directional constraints.

Fig. 7.6 shows examples for measured evaluation functions using the three different schemes. It can be seen that the measured functions in the minority of cases are identical to the assumed (ideal) functions. Looking at the second scheme first, a distinct step cannot be found. A reasonable approach would be to use the location of the steepest gradient. This works well for the first and the third example but not for the others. In the second and the third example the functions of

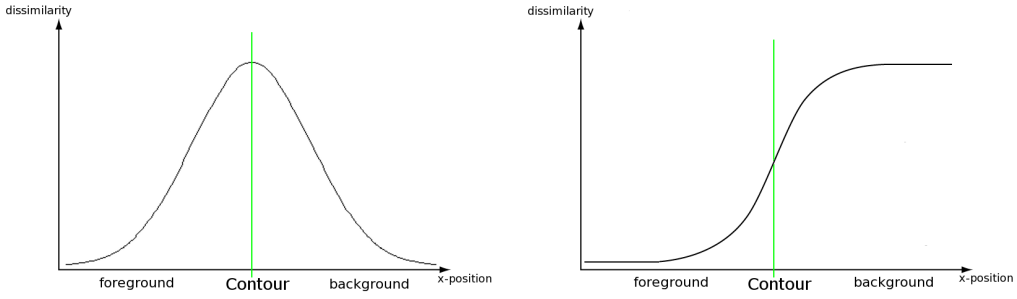


Figure 7.5: **Evaluation functions (assumed):** On the left we see the function for schemes 1 and 3 searching for the maximum output, and on the right we see the function for scheme 2 searching for the step of the function.

schemes 1 and 3 look very similar, respectively, but at the other examples the maximum of the third function is rather at the correct position, than the maximum of the first function. Hence in the following we use the third evaluation scheme.

It can be seen how different the ranges of values of the functions are. To be sure to detect not only the best position in the scrutinized area but a real contour point, a threshold value had to be identified experimentally. Only if the measured maximum exceeds this threshold (set to 20), a new contour feature point is set. The four examples in Fig. 7.7 show again measured functions and additionally the positions that have been detected to be set as new contour feature points. In the first example the maximum value is clearly below the threshold which leads to not setting a new contour feature point. The position of the maximum value is marked in blue. In the third and fourth example the maximum values exceed the threshold and the new contour feature points are set to the right positions (both marked in green). In the second example the hair simulates a present contour, which causes the maximum value to exceed the threshold and sets a new contour feature point (marked in green). The real contour is covered by the hair and the detected position is the best one to be reached.

Results

Fig. 7.8 shows fitting results using this contour alignment compared with the results using E_I', both initialized with manually marked feature positions on the head, but not at the contour. At the upper left side the new alignment approach delivers a better result, but the contour at the chin is not optimal yet because too few contour points have been observed at this region. At the upper right side the contour at the chin moved towards the contour of the hair. The difference between skin color and background color was big enough to detect an edge, but the difference between the skin color and the hair color was to small to recognize that this is not the real contour. Further on, the whole contour on the right moved along with it, not able to 'find back' to the right contour. At the lower left side improvement clearly can be seen both at the ear and at the forehead. At the lower right side the results look quite similar, but using the new contour alignment moved the contour at the right ear to a better position.

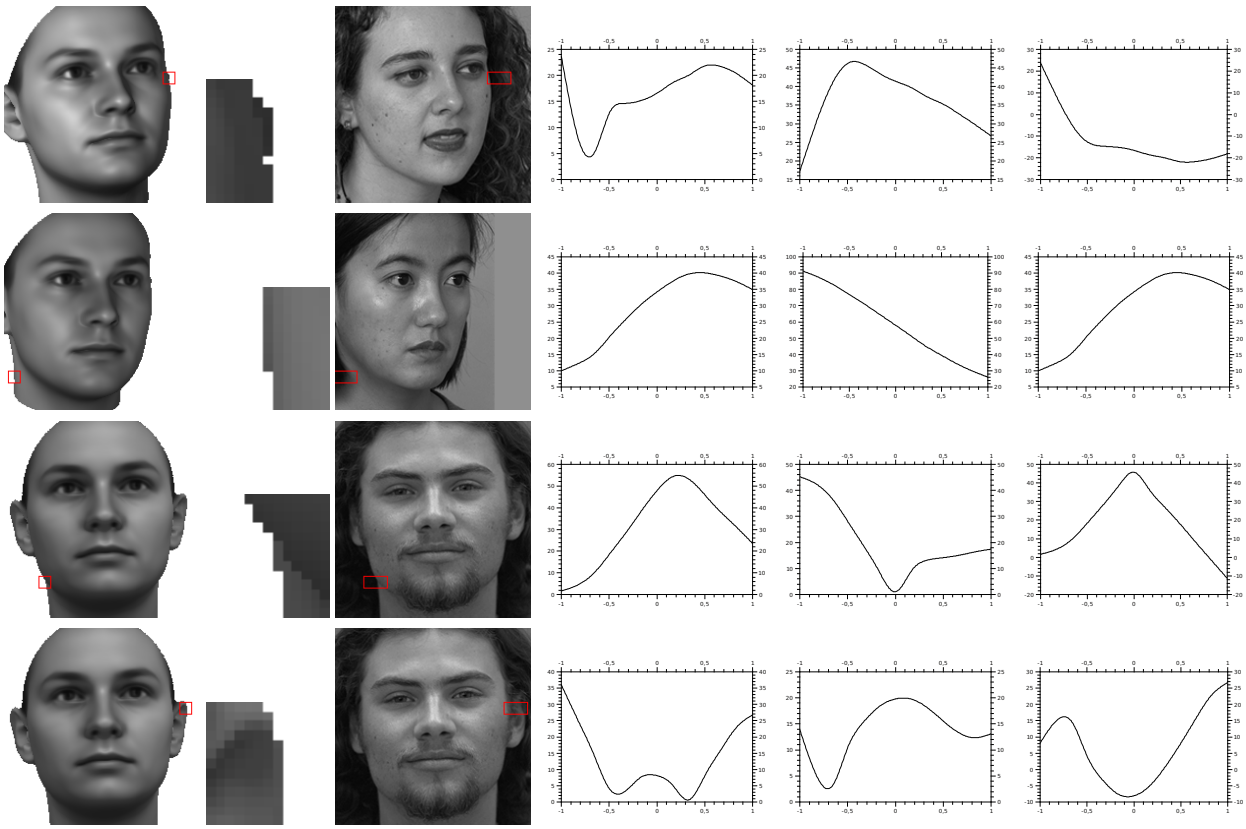


Figure 7.6: **Evaluation functions (measured):** On the left side we see the current fitting result with the template position marked on it, the cut out template and the area to search in marked on the input image. On the right side the measured evaluation functions of the three schemes are shown in the range $[(\quad 1)\cdot\frac{1}{18}s_H, (+1)\cdot\frac{1}{18}s_H]$ around the current contour position x_c.

Another problem which may occur is the adapting of the contour to the edge between cheek and ear. Sometimes there appears a sharp-edged crossover, which is not detected as an inner edge, but is detected as contour. Besides that, the misalignment at the hair could even be worse than at the upper right of Fig. 7.8. This is a general problem of the 3DMM approach and can be counteracted by a previous occlusion detection, excluding the region of the hair from the fitting. Fig. 7.9 shows two examples for this kind of difficulties. On the left side the contour at the ear of the model adapted to the edge between cheek and ear in the input image. There is indeed a sharp-edged crossover and the inner edge is detected instead of the outermost contour. On the right side the texture of the model adapted to the hair. So their color became the *skin* color of any contour template in that region and the contour certainly adjusted to the hair still more.

Changing the Search Direction

At the former approach to measure E_I' at each of the 8 randomly chosen contour triangles a step into the direction of the corresponding normal was done. In the previous tests with the new template based contour adaption, the search direction was the x-direction. The search along the normal direction is the more intuitive one. Hence, the tests also have been done searching forward and backward along the normal direction. Fig. 7.10 shows the siting and orientation of the normals on an intermediate fitting result. The area to search in (its range) has also been adjusted based on the orientation of the normalized normal $\hat{n}_k = \begin{pmatrix} x_n \\ y_n \end{pmatrix}$. If $(x_n \geq y_n)$, the range remains $[x_c - \frac{1}{18}s_H, x_c + \frac{1}{18}s_H]$, and if $(x_n < y_n)$, the range changes to $[y_c - \frac{1}{18}s_H, y_c + \frac{1}{18}s_H]$.

Fig. 7.11 shows reconstruction results using the search into forward and backward direction of the normal. The results are similar to the ones using search in x-direction, but at the first example the contour at the chin adapted even better. Before it had been to small, but to fit the right contour the chin had not only to move but also to grow. This could only be achieved by detecting a contour point at the point of the chin. There are not so many contour triangles involved but to hit upon the real contour is more likely when using search in normal direction than searching in x-direction. At the second example the contour did not adapt to the hair again and at the third example it adapted as good as before but this time the eyes look even better. The area of the eyes is not as deformed as before using E_I' or search in x-direction. At the fourth example overall shape fits quite well, only at the ears the adaption did not work as well as before.

Effectiveness

To demonstrate the effectiveness of our approach of using directional constraints and considering a certain number of contour points every 2000 iterations, concluding tests using an optimal contour line have been done on the assumption that there would be an ideal contour detector. In these tests a new contour feature point is set when crossing the optimum contour line searching in normal direction. Fig. 7.12 shows the optimal contour lines and the corresponding reconstruction results. The tests show how good the model is able to adapt when the ideal contour line is known. At the first example at the chin the improvement of the adaption against the former contour search E_I' is clearly recognizable. Also at the upper cheek, aside the eye, and at the ear improvements can be seen. Using the new contour search with 24 points instead of 8 points improves the adaption further, visible e.g. at the contour of the ear. The second example illustrates the improvement at the whole contour on the left side of the face. For the former contour search it is certainly not possible to adapt to the right contour not knowing the occlusion. Based on the knowledge of the ideal contour line the new contour search using 24 points per search adapts best. With these concluding tests we have shown the effectiveness of our new contour search approach and the previous tests showed the use of the new contour search in an automatic scenario. The automatic detection is still not as good as when the ideal contour line is known, but the results are promising and afford a good base for further research.

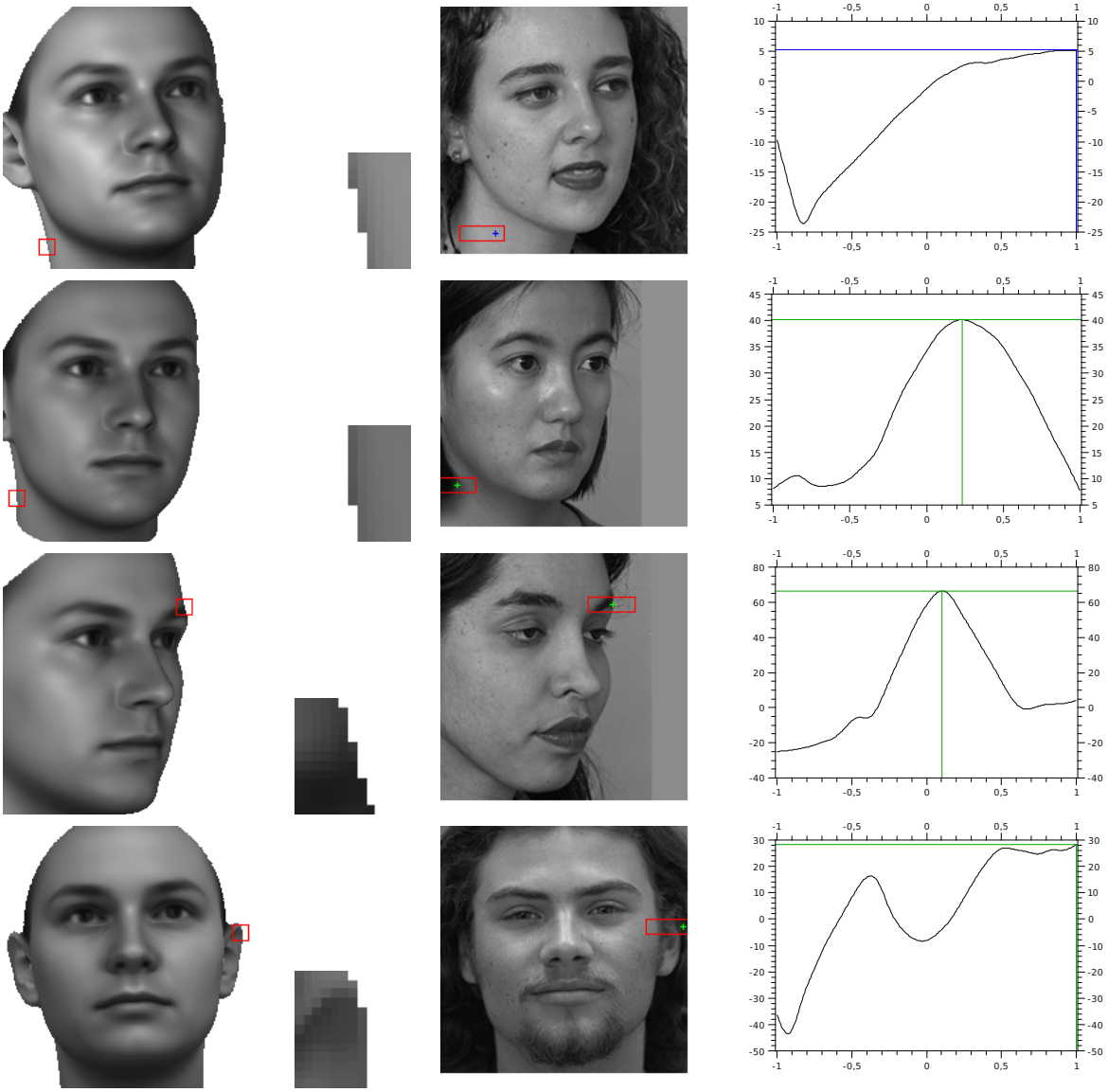


Figure 7.7: **Positioning of contour points:** From left to right we see in each row the current fitting result with the template position marked on it, the cut out template, the input image with the marked search area, and the corresponding evaluation function in the range $[(\quad 1)\cdot\frac{1}{18}s_H, (+1)\cdot\frac{1}{18}s_H]$ around the current contour position x_c. The lines on the graphs mark the maximum values and the crosses in the search area of the input image mark their corresponding positions. They are color coded: Green stands for a maximum value above the threshold where a contour feature point is set and blue stands for a maximum value lower than the threshold where no contour feature point is set.

Figure 7.8: **Contour alignment using templates:** Each pair shows the reconstruction results using E_I' on the left and using the new template based contour alignment on the right.

Figure 7.9: **Difficulties:** Each pair shows a part of an input image on the left and the fitting result using the new template based contour alignment on the right. The two examples demonstrate difficulties for the contour alignment. The outer contour of the model aligned to edge between face and ear on the left and on the right the model adapted to the hair.

Figure 7.10: **Normal directions:** Here the 2D-projections of all normals at the current contour triangles are shown. The green part points towards the normal direction (away from the face) and the red part points towards the reverse direction.

Figure 7.11: **Contour search into normal direction:** Here we see reconstruction results using the new contour alignment searching in direction of the normal, to be compared to Fig. 7.8.

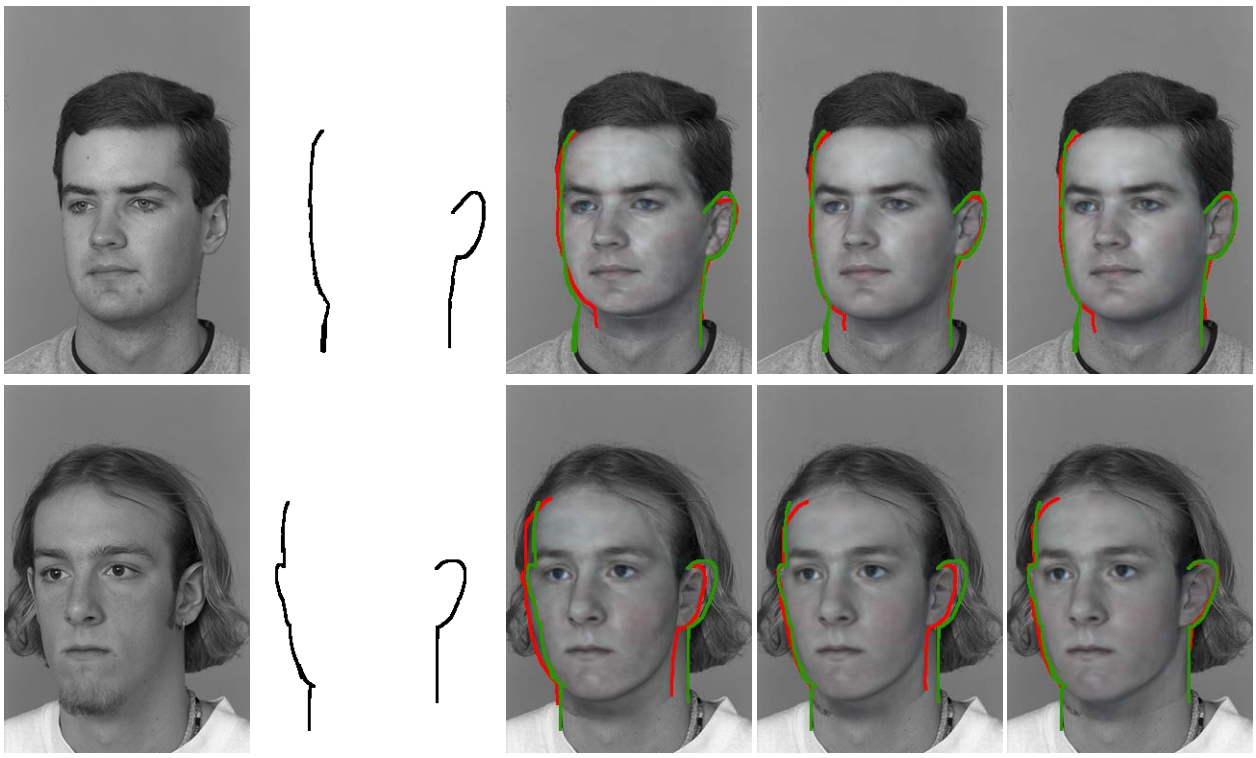


Figure 7.12: **Fit to optimal contour:** From left to right we see the input image, the optimal contour lines, the reconstruction results using E_I', contour search in normal direction at 8 contour points every 2000 iterations, and contour search in normal direction at 24 contour points every 2000 iterations. On the reconstruction results the optimum contour is marked green and the final contour of the reconstructed model is marked red.

Chapter 8

Automated Detection of Occluding Regions

In the majority of cases, face images in real world situations are not taken under optimal conditions and they are far from matching standards like the ones for visas, travelling documents or identification cards (cf. Forczmanski and Kukharev [FK07] based on [ISO04]): natural ambient light, neither too bright nor too dark, minimum dimensions of 320 $\times$ 240 pixels, face in frontal orientation, facial area vertically oriented covering at least 85% of the whole image, distance between centers of eyes not less than 60 pixels, neutral expression with both eyes opened normally and mouth closed, glasses shall be clear and transparent so the pupils and irises are clearly visible, no lighting artifacts of flash reflections on glasses.

Often there are occlusions or perturbations causing pixels not pertaining to the proper face. Examples are hair, glasses, reflections or artificial occlusion (objects in front of the face, covering it), which can be found in Fig. 8.1.

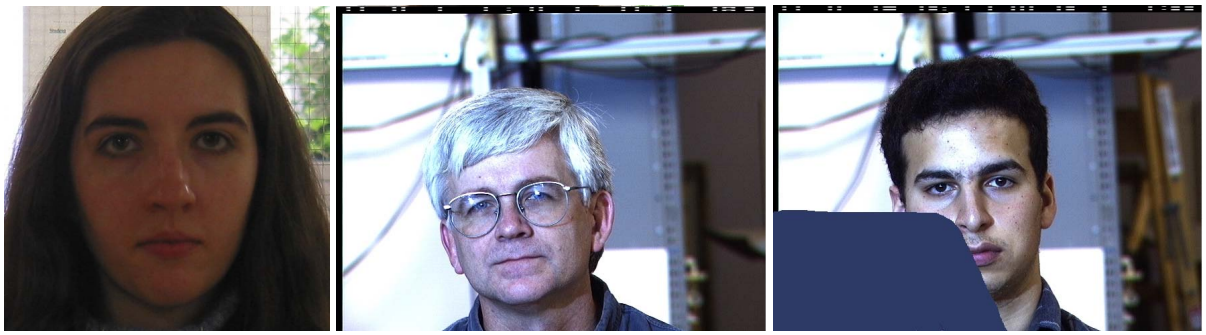


Figure 8.1: **Typical examples of occlusions**

When the fitting algorithm reconstructs a 3D model from an image where part of the face is occluded, it may reproduce part of the occluded pixels in order to minimize the cost function,

which is not the desired result here. It can be seen as a case of overfitting. Two examples are shown in Fig. 8.2: In the first example both ears are covered by hair but the model adapts to them

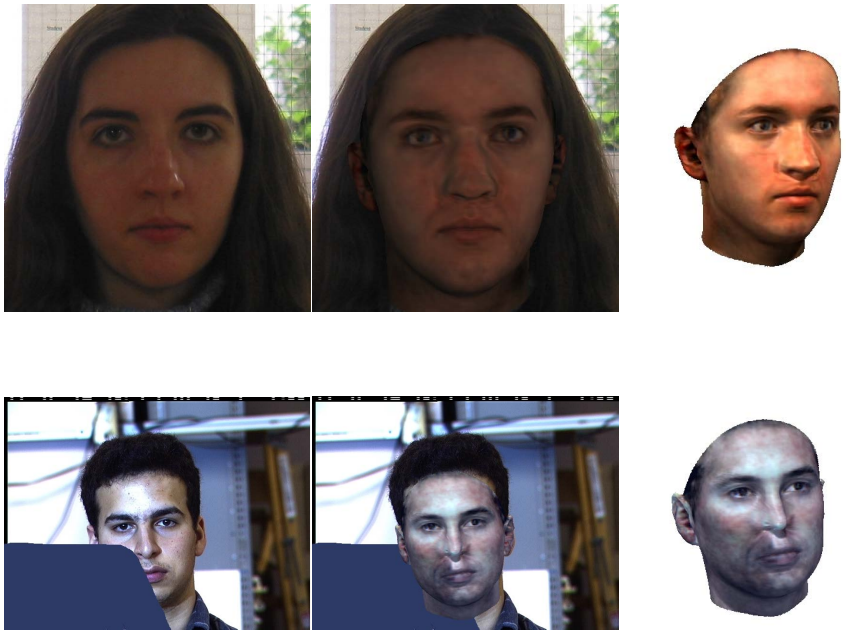


Figure 8.2: **Reconstructions of occlusions**

anyway, thus the reconstructed texture is also dark at the ears. A much bigger influence of the occlusion also to the rest of the face can be seen in the second example. Even though the nose is not covered, the reconstruction is affected by the artificial occlusion anyhow and the nose is squeezed.

To improve the reconstruction it is possible to use an occlusion mask. It is binary-coded and each pixel of the input image is marked either as important for the reconstruction or as an occlusion. The pixel-by-pixel image difference (cf. Eqn. 2.6) is changed to take only pixels into account not belonging to the occlusion, to let the model fit to the accurate parts of the image only:

$$E_I = \sum_{(x,y)\notin occMask} (I(x,y) \quad I_{model}(x,y))^2 \tag{8.1}$$

In addition, it is possible to use the occlusion mask to complete the imperfect original texture of the surface based on the 3D model reconstruction. After reconstruction of shape and texture, the original texture out of the input image can be mapped onto the 3D model and based on the occlusion map the undesirable occlusions may be discarded and replaced by the considered texture of the model.

Related work in the fields of segmentation, generating outlier masks and removing occlusions can be found in Chap. 3.3.

To decide whether a pixel belongs to an occlusion or not, there are some criteria. Doing a coarse fit of the model to the input image gives a clue what a full reconstruction would look like if there would not be any occluding structures. In a pixel-by-pixel comparison of a rendering of the current model to the original image, totally different colour values indicate a potential occlusion. But for this we have a typical chicken and egg problem: First you have to adapt to an imperfect image, to compare the result with the original image, to decide whether there are 'suspicious' pixels and where they are. But how much adaption is necessary to fit the 'good' areas so that in the comparison they are detected as 'good' and not mistaken for 'suspicious'? And how much of adaptation is permitted so that the model has not fitted to the erroneous areas and they can not be detected by comparison any more? Further indications for occluding areas can be seen in Fig 8.3. First there is the extension of a region beyond the face boundaries. Apart from glasses, no occlusion can be restricted to the front of the face, but it has to cover both the face and the background. These do not have to be artificial necessarily. Hair can also create such a kind of occlusion. Additional cues could be a contour in the image which is not found on the model, a contour on the model which is not found in the image or t-junctions.

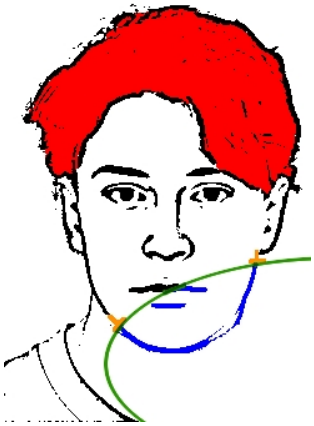


Figure 8.3: **Indications for occluding areas:** red: region stretching out over the boundaries of the face, green: contour in the image but not on the face/model, blue: contour on the face/model but not in the image, orange: t-junctions

Different potential approaches arise to generate an occlusion mask automatically. They all use similar coarse fits as basis and will be outlined in the following sections: 8.1 presents a learning based approach, and in 8.2 occlusion detection is done by bootstrapping. In 8.3 predefined occlusion masks are used, and 8.4 shows two approaches working on a first, raw occlusion mask, which grows to the right dimensions. Mainly, we have developed and evaluated the PIE database [SBB01b]. The results of the automatic occluder detection in the sections 8.1-8.3 are always shown on six typical examples, unless explicitly stated differently. The original six images can be found in Fig. 8.4.

Figure 8.4: **Typical images worked on:** images in the first row are taken unmodified from the PIE database. Images left and in the middle of the second row are taken from the PIE-DB, too, but have been modified by painting an artificial occlusion on them. The last two small images show close ups, one with huge remarkable reflections on the glasses and one with a moustache.

8.1 Learning Based Detection

This approach is based on the concept of using variations of model coefficients and rigid parameters to distinguish plausible variations of individual pixels from unexpected variations which indicate that there is an occlusion.

Doing a conservative fit and varying the model coefficients and rigid parameters result in different assumptions to which direction the fitting process might move during the next iteration steps. So these variations can be used to learn the variation of each individual pixel with respect to a plausible degree of variation of the model parameters as a criterion per pixel for plausibility of pixel errors. For example, at the eyebrow a high degree of variation is possible. If the shape coefficients change only a little bit, it could still be possible that the whole brow moves up or down and a dark pixel turns brighter because of first being located on the eyebrow and later being located above or below.

After fitting coarsely to the input image, we generate a set of $n = 104$ variations of the reconstructed model coefficients and rigid parameters. The first 10 shape coefficients and the first 20 texture coefficients, separately are varied along their corresponding principal component (s_i or t_i), in positive and negative direction both, in the range of the corresponding σ ($\sigma_{S,i}$ or $\sigma_{T,i}$). The 7 rigid coefficients on translation, rotation and scaling, and the 15 light coefficients, are also varied separately in positive and negative direction in the range of $\pm\sigma_{R,i}$. For each variation the resulting model is rendered into a separate image (variation image). In a pixel-by-pixel analysis over all n variation images we get knowledge about plausible variations.

This knowledge can be used in different ways. The first naive approach is to compute just

minimum and maximum value per pixel over all generated variations, in order to get lower and upper bounds of plausible pixel values. If a pixel value from the original image is not between the bounds of the variations, this is an indication for an occlusion. But min and max values can also be analyzed statistically. Around pixels with big differences between min and max it is admissible to have discontinuities in the original image. If there are discontinuities anywhere else, these pixels have to be assumed to be suspicious.

Using only upper and lower bound (c.f. Fig. 8.5) helps to detect most of the glasses frames and the moustache as occlusions, but it does not work well for the artificial occlusions. The artificial occlusions taking place on cheeks or at least on an unstructured part of the face are detected but at the periphery of the face, at the mouth and at the chin they are not detected. In these regions, the changes in shape and texture during the variations cause a wide range in possible pixel values. The reflections on the glasses are recognized quite well, but in the majority of cases a lot of other un-occluded areas are detected as an occlusion as well. Fig. 8.6 shows the results analyzing the values statistically. Here also most of the frames of glasses are detected but the reflections on the glasses are not fully identified as an occlusion at the eyes . They are placed near the eyes which might 'move around' over the different variations in shape and texture. So at some pixels at certain variations the almost white sclera and at other variations the dark pupil is found, which causes high differences between min and max values preventing the detection of the occlusion. The moustache is not detected, too, because it is located near the mouth, where shape and texture also change much over the variations. The artificial occlusions are not detected for two reasons. At the periphery, at the mouth and at the chin there are the high differences between min and max value so that at this areas it is not searched for an occlusion. At the cheeks only little changes in the pixel values are caused by the variations but the occlusions are uniform themselves and so there are no discontinuities within them, so they are not detected. Even where occlusions are present, this criterion serves rather as edge detector than as occlusion detector.

Figure 8.5: **Min and max values used as upper and lower bounds**

To use the full range of information from the variations and not just the upper and lower bounds,

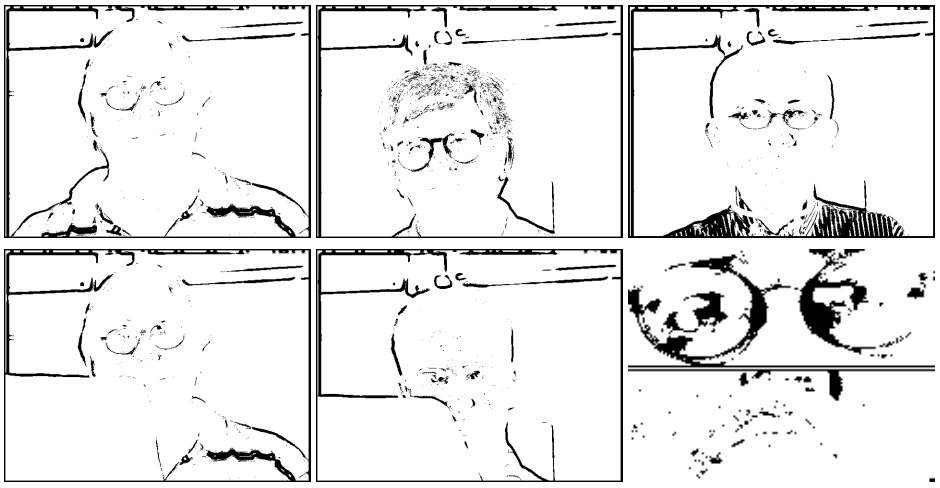


Figure 8.6: **Detecting non-admissible discontinuities using min-max differences**

the variance has to be computed. Using the criterion that around pixels with large variances it is allowed to have discontinuities in the original image and if there are discontinuities anywhere else, we infer that there is an occlusion, according to expectations we get occlusion masks similar to the previous ones (c.f. Fig. 8.6 and Fig. 8.7). Using the full range of information and not just the lower and upper bound, improves the result only marginally.

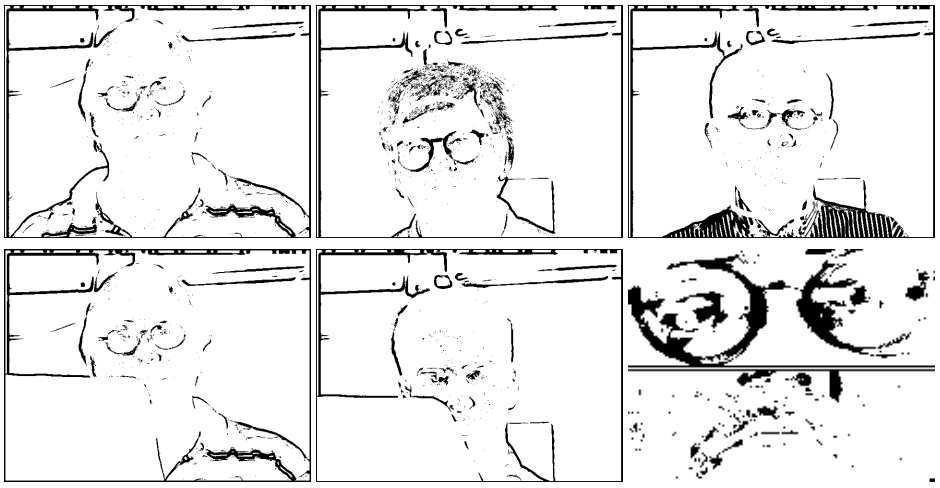


Figure 8.7: **Detecting non-admissible discontinuities using variance per pixel**

The variance can also be used to treat the pixel-by-pixel differences, comparing the first coarse fit and the original image, relatively to the plausibility of pixel errors. Scaling down the squared differences using the variance at each pixel

$$\frac{(r_{model} \quad r_{image})^2}{\sigma_r^2} \qquad \frac{(g_{model} \quad g_{image})^2}{\sigma_g^2} \qquad \frac{(b_{model} \quad b_{image})^2}{\sigma_b^2} \tag{8.2}$$

only large quotients indicate a potential occlusion.

The resulting occlusion masks can be found in Fig. 8.8. The main parts of the reflections on the glasses are detected but the frames of the glasses are not fully detected. Where the frames of the glasses are placed in areas exposed to considerable fluctuations (at the eyebrows, near the nose, and at the periphery of the face) the difference in the color values is reduced by the scaling and is not detected any more. The area around the mouth is also exposed to considerable fluctuations which causes the moustache not being detected as well. So only salient occlusions may be detected using this method.

Figure 8.8: **Differences scaled by standard deviation**

Another approach involves not just processing the variations with themselves and using the result, but exploiting all variations to the same extent by building the difference of each variation to the original image. Computing the mean discrepancy between the input image and all different assumptions to which direction the fitting process might move during the next iteration steps, it can be concluded from that where more than likely an occlusion takes place because a high value in the mean discrepancy signifies that in many cases a difference between one of the variation images and the original image is present.

The generated occlusion masks can be found in Fig. 8.9. Reflections on the glasses and the glasses frames have been identified well again, but the moustache is only partly detected.

All in all these learning based approaches are useful for the detection of glasses frames and reflections on the glasses. The moustache has only been identified using the first criterion and the artificial occlusions have never been detected completely. An occlusion mask generated like this would be useful as a *first guess* for some of the following approaches. But the learning based

Figure 8.9: **Average of differences per color channel:** Here the differences have been computed separately for each color channel and the combined results are presented. The shown colors indicate in which channel an occlusion has been detected. white: nowhere, black: in every color channel, cyan: red channel only, magenta: green channel only, yellow: blue channel only, blue: red and green channel, green: red and blue channel, red: green and blue channel

approaches are very time-consuming because for every new input image the coarse fit and all the variations have to be generated again. So faster possibilities to generate 'first' occlusion masks will be presented in the next sections.

8.2 Bootstrapping

Here we have a typical chicken and egg problem: Adapting to an imperfect image, the result is compared with the original image, to decide whether there are suspicious pixels and where these pixels are. For solving this typical problem, a suitable approach is bootstrapping. Besides, bootstrapping can be seen as an attempt to improve the resulting masks of the previously presented approaches.

We use a coarse fit and a simple threshold or one of the former presented approaches to generate a *first* occlusion mask. Making use of this mask, another conservative fit (adapting a bit more than at the first fit) is done. Comparing the new result with the input image again leads to the *next* occlusion mask. Iterating this procedure should result in occlusion masks getting better and better.

In Fig. 8.10 the first and second row show the results of using one of the learning based approaches to generate the occlusion mask. In the third row a simple threshold is used for occlusion mask generation. In the examples in the two upper rows, the main parts of the reflections on the glasses and the moustache have been detected as an occlusion at the first step. No further big

areas are detected over the next steps (little green areas in the difference image on the right side of Fig. 8.10). But some parts, not covered at all (at the nose and at the eyebrow), are detected as non-covered over time (red areas). At the third example the occlusion mask grows or shrinks strictly into the wrong direction. The region containing the artificial occlusion is decreasingly detected as an occlusion over the iterations and the rest of the face, not covered at all, gets classified as an occlusion more and more.

If small occlusions are detected already at the *first guess*, the marked areas are retained over the progress of the masks. Falsely detected areas after all are detected as non-covered over time. However, if there are bigger occlusions which are only partially detected at the *first guess* it is not possible to recover this even using bootstrapping. It rather gets worse. With every step the head is adapted more to the non-detected part of the occlusion. It gets more similar to it. Comparing this with the input image the occluded area is detected less reliably than before. Therefore, bootstrapping in this way does not seem to be a good strategy.

8.3 Predefined Masks

A less time-consuming approach is the use of predefined occlusion masks. Doing a conservative fit with each of the different predefined masks leads to different reconstructions which have to be compared to the input image. Using the pixel-by-pixel difference with a certain threshold as a criterion this leads to different assumptions of possible intermediate occlusion masks which have to be beneficially combined to a final mask. Here we use four rectangular masks covering the upper side, the lower side, the left side or the right side of the input image. They can be found in the top row of Fig. 8.11. The configuration proceeds based on the assumption that the face is placed in the middle of the input image (like in face detection results) and the occlusion covers not more than a quarter of the face area.

If the predefined mask is located in the area where the occlusion takes place, the model adapts to the non-covered part and the difference between conservative fitting result and the input image gets conspicuous, which can obviously be seen in Fig. 8.11: Using the first (left side covered) or the last (lower side covered) predefined mask the occlusion has been detected the most. Taking place in the lower left side of the input image it is covered by these two masks. If the occlusion is covered (marked black in the mask) the model adapts more to the un-occluded part of the face. So the difference between the fitting result and the input image (containing the occlusion) becomes more obvious. Using the second (right side covered) or third (upper side covered) predefined mask big parts of the face are detected as occlusions. These two mask cover the un-occluded part of the face and the model adapts more to the occluded part. The difference between the fitting result and the occluded part of the face gets smaller. But the difference gets bigger for the un-occluded part because the whole model is influenced by the color values of the occlusion. A certain area that is repeatedly detected using different masks, more than likely indicates a real occlusion. In Fig. 8.12 the resulting occlusion masks are shown using different thresholds and different combinations of the intermediate masks.

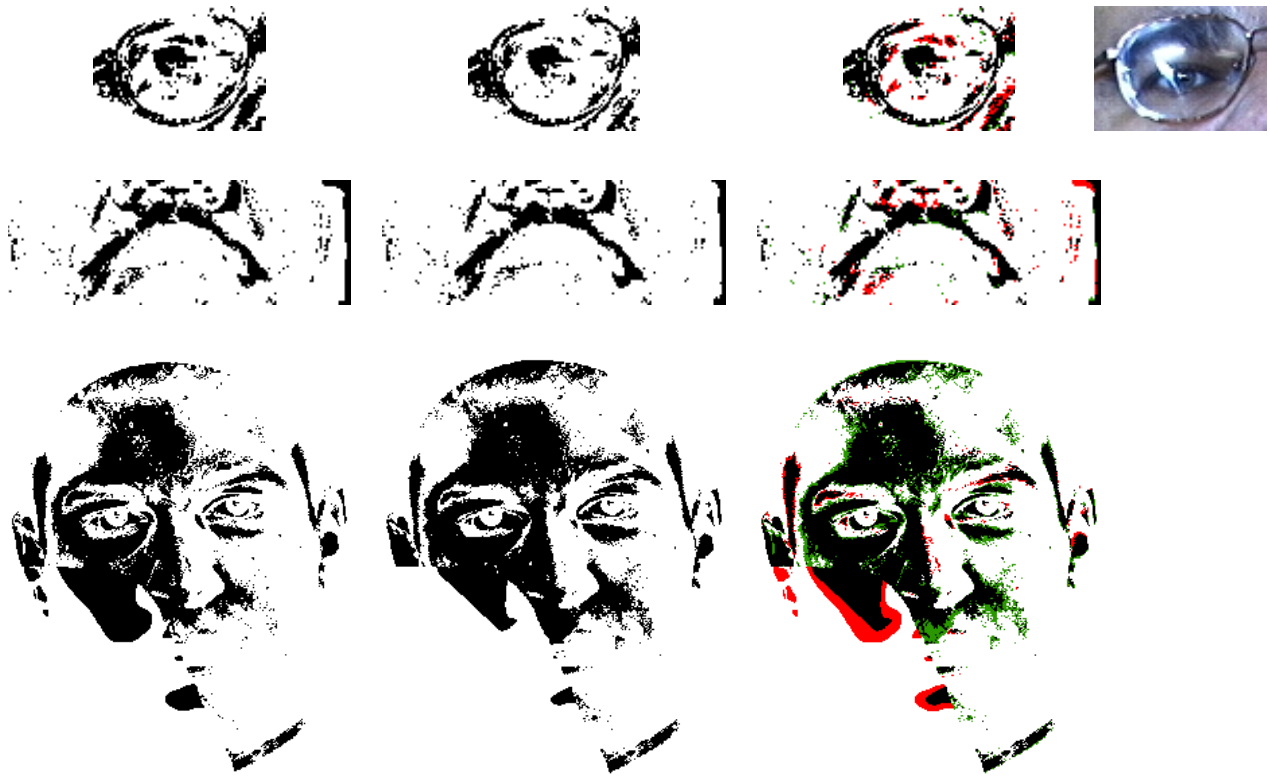


Figure 8.10: **Bootstrapping:** On the left the occlusion mask at the first iteration is shown and in the middle the result after a few iterations. The images on the right show the differences between the first and the later occlusion mask, where white areas have never been detected as an occlusion and black areas have been detected as an occlusion over all iterations. Areas marked red show what has been detected as an occlusion at the first iteration but has been discarded over the iterations and the green areas show what has been detected additionally over the iterations. At the very right of the upper row the part of the input image used here is shown. The input images used for the second and third row can be found in Fig. 8.4

Using the lower (strong) threshold requires using a harder criterion at the combination of the masks: a pixel is marked as an occlusion only if it is seen as an occlusion in three or more cases (middle column). When using the higher (weak) threshold, it is sufficient that a pixel is detected as an occlusion more than twice to mark it as an occlusion in the final mask. Looking at the second and third example, the approach presented in the last column obtains the best results. Only at the first example nothing of the artificial occlusion gets detected then. Here the mask presented in the middle column would be a better choice. The first and the last row show that this approach does not only work for artificial occlusions, but also for glasses and reflections.

Since only parts of the painted occlusions have been detected, the masks need to be improved and at most can be used as a *first guess* for the region growing approach described in the next subsection.

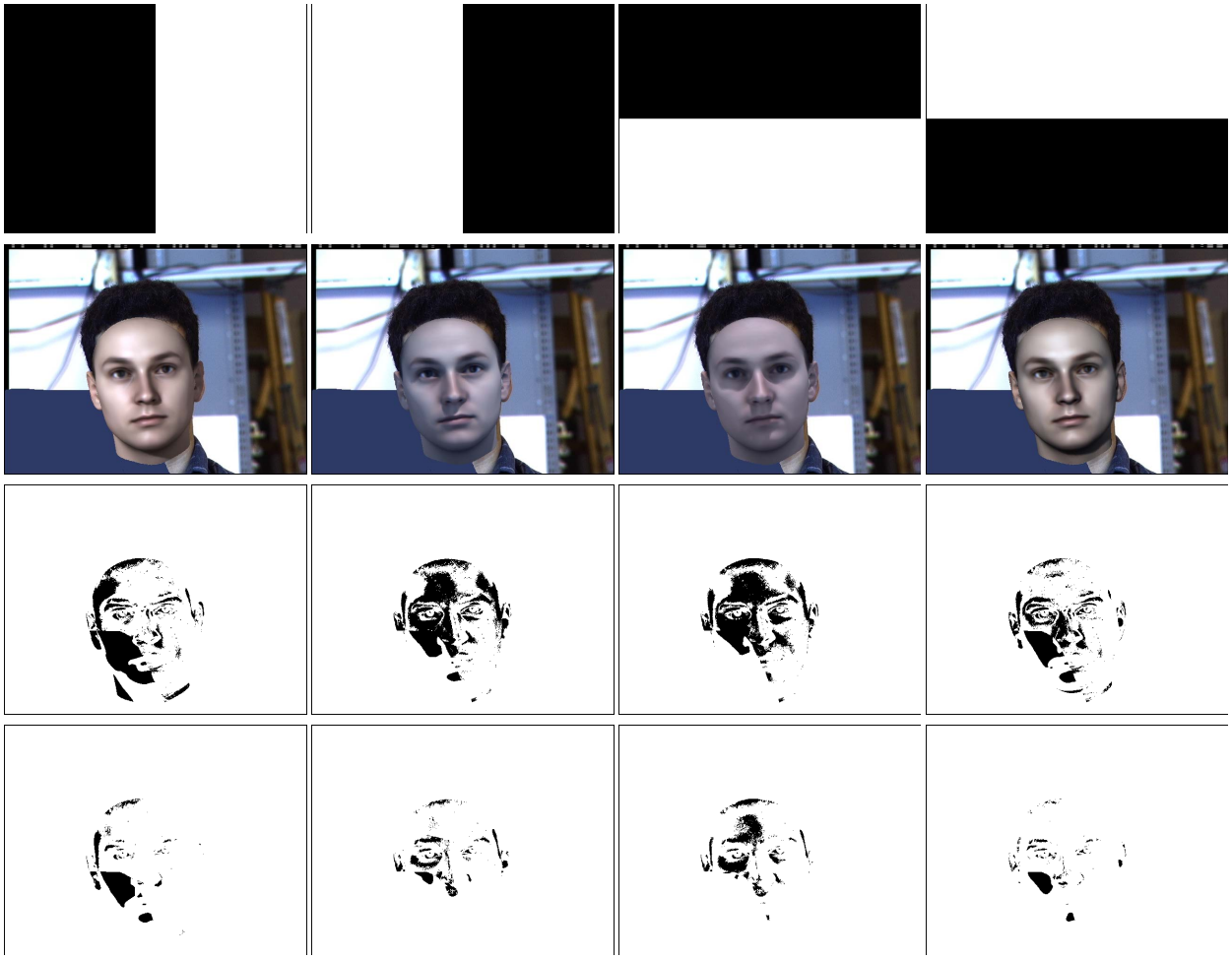


Figure 8.11: **Predefined occlusion masks:** first row: the four predefined occlusion masks covering different parts of the whole image, second row: results of a coarse fitting using the predefined masks rendered back into the input image, last two rows: generated occlusion masks using different thresholds ($t = 30$ and $t = 65$)

8.4 Region Growth

In this section two approaches using a *first guess* of the occlusion mask and region growth are presented, based on the idea to use a weak threshold at first to get a clue where the occlusion might have taken place and then let the occlusion mask grow using the indications shown in Fig 8.3. To generate the *first guess* occlusion mask any of the previously proposed approaches could have been taken, but here the quick and simple pixel-by-pixel comparison using a threshold is chosen. The first approach presented is suitable to detect central occlusions like glasses, specular highlights and objects in front of the face and the second approach is most applicable to peripheral occlusions like hair or clothes.

Figure 8.12: **Resulting masks using four predefined ones first:** left and right: if a pixel in two or more cases is seen as an occlusion, it is finally seen as an occlusion ($t = 30$ and $t = 65$), middle: if a pixel in three or more cases is seen as an occlusion, finally it's seen as an occlusion ($t = 30$)

8.4.1 Central Occlusions

This subsection focuses on central occlusions like glasses, specular highlights and objects in front of the face. As we have seen in the previous sections, a lot of the presented approaches are useful to detect frames of glasses and specular highlights. Here we will show how to detect this kind of occlusions using a simple criterion and the right threshold. Widespread occlusions are caused by objects covering the face in the image plane. This kind of occlusions can be characterized by stretching out over the boundaries of the face, and not only taking place in front of the face. This characteristic together with two other indications for occluding areas (contours present in the image but not on the model, contours present on the model but not in the image, cf. Fig. 8.3) will be exploited in the following, to detect widespread occlusions of any kind. An overview over the workflow described in this subsection is given in Fig. 8.13.

The first step of the algorithm is a conservative (coarse) fit. Given several landmarks we perform only less then half of the iteration steps of the complete fitting algorithm and render the result into the original image. Also the z-buffer of this fit is saved to know whether a pixel belongs to the face (foreground) or to the background, which is needed in the final step.

To get a *first guess* of the occlusion mask, we compare the image of the first fit with the original one. Differences in colour values are computed and if they are above a certain threshold, these pixels are marked to belong to the occlusion. We choose two different thresholds, $t = 60$ and $t = 65$.

As even the coarse reconstruction has adapted somewhat to the occluded parts of the face, the image comparison only provides a rough clue of where occlusions may be and only detects small parts of the whole occlusion. To get the complete regions of occlusion we have to let grow the areas detected first. The neighbours of the already marked pixels have to be analyzed, whether they belong to the occlusion, too. The indications for that can be found in Fig. 8.3 again. If a neighbouring pixel is identified as also being part of the occlusion, it is marked (set to black) in the occlusion mask, too. Iterating this, the marked areas of the occlusion mask grow until the entire occlusion is detected.

Generating the first occlusion mask with $t = 60$ ensures the detection of sufficient areas to let grow, without too many 'false alarms'. For the detection of glasses (for a possible use at the final step) it is satisfactory to choose $t = 65$. To ensure a controlled growth of the mask, we use edge detection. The marked regions of the mask shall not grow in every direction. If they reach the border of the occlusion they should stop growing. For this we perform a Canny edge detection on the original image as well as on the first occlusion mask ($t = 60$). Edges existing in the occlusion mask that have no counterpart in the original image are suspicious. Missing edges indicate that the detected region is not as big as the corresponding region in the original image. It is not complete and has to grow at this edge. At the edges, we only let the areas grow one pixel per iteration to get the start occlusion mask for the next step. There we compare the edges of the new start occlusion mask to the original edges again, grow and repeat this until there are no suspicious edges left or the maximum number of iterations (here 50) is reached.

Often, growth causes stripes in the occlusion mask because the edges are not continuous most of the time. Hence we have to fill up the stripes. If there is a white pixel with black pixels above and below or black pixels left and right, we turn it into black.

Finally, to check the second criterion which is whether a detected region extends beyond the face, at first a connected component analysis is done. Pixels of same regions are labelled as connected. After this, every connected region has to be compared to the z-buffer from the beginning, to identify whether it belongs only to the foreground or only to the background or to both of them. Only regions extending beyond the face are retained (marked black in the occlusion mask), the rest of them are deleted (marked white in the occlusion mask).

If the person in the original image does not wear glasses, we can take this as the final occlusion mask. To account for glasses as well, which are entirely on the face, but are easy to detect with a large threshold ($t = 65$), we add the start occlusion mask ($t = 65$), which contains glasses, to the final occlusion mask.

Fig. 8.14 and 8.15 show the results of the reconstructions for three examples using only the *remaining regions* (extending beyond the face) as occlusion mask and using the *remaining regions* together with the *first guess* respectively. In Fig 8.15 also a close up is shown to demonstrate what completion of the imperfect original texture of the surface based on the 3D model reconstruction is like.

Results

The test images are taken from the PIE database [SBB02] and have been equipped with occlusions by hand. In the upper row of Fig. 8.14 the main part of the dark bar disappears. Only at the side of the nose a small part has not been detected because there could have been present a shadow as well. In the middle row the upper part of the right part of the face is totally covered. The occlusion is detected fully whereby the head adapts only to the valid parts and the part under the occlusion is modified correlatively. After texture extraction from the input image the missing part is filled by mirroring the available texture and at the remaining parts the reconstructed texture of the model can be seen. Given the fact that the human face is nearly symmetric, using the mirroring it leads to a result close to the original that as well can be used for recognition again. Exceptions appear if there are remarkable points in the mirrored area. They will be mirrored, too. If there is a birthmark in the mirrored area, in the final texture there will exist two birthmarks on the opposite parts of the face. In the lower row a small part of the right part of the face and one ear are occluded and the occlusion is fully detected. This time also a part of the glasses frame is detected because it extends beyond the face. In Fig. 8.15 the particular *first guess* is added to get the final occlusion mask. In the first and second row some small parts are detected as occlusions, at the eyebrows and a little bit at the eyes, although they are not occluded at all. But this does not have a negative effect on the reconstruction. In the third row also the frame of the glasses and the reflections on the glasses have been detected at the *first guess* whereby they disappear after reconstruction and texture extraction. At the close up the result shows the reflections on the glasses vanished. But the frame of the glasses still can be conjectured because at some parts not the whole occlusion has been detected or the original texture and the fitted texture do not merge into each other optimally. Fig. 8.16 shows an example were the proposed algorithm failed. The difference between the colour of the occlusion and the face was within the threshold and therefore too small to detect. This example demonstrates how difficult it is to find the right trade-off between detecting the occlusion and not detecting too much of the rest.

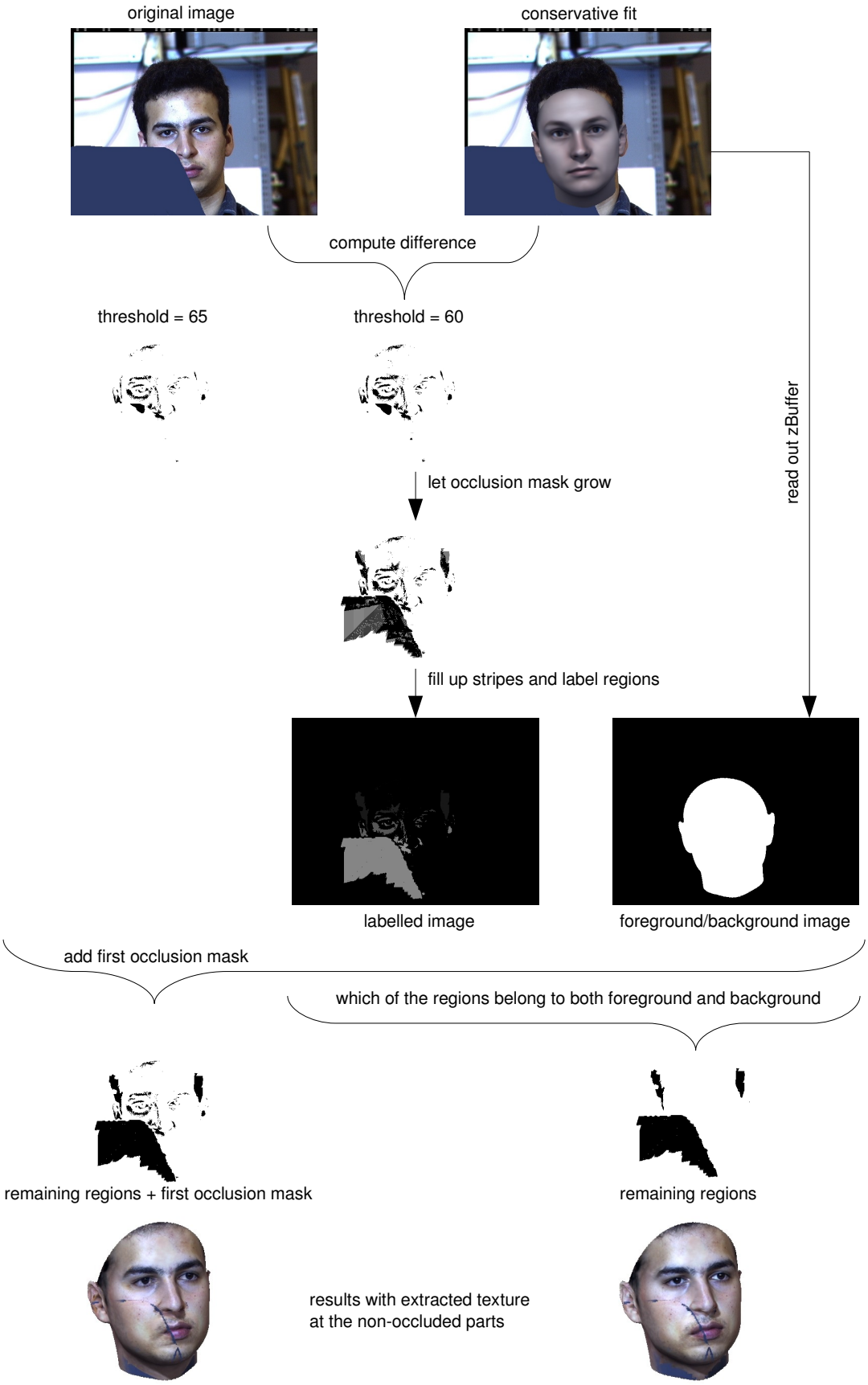


Figure 8.13: **Workflow of the region growth approach for central occlusions**

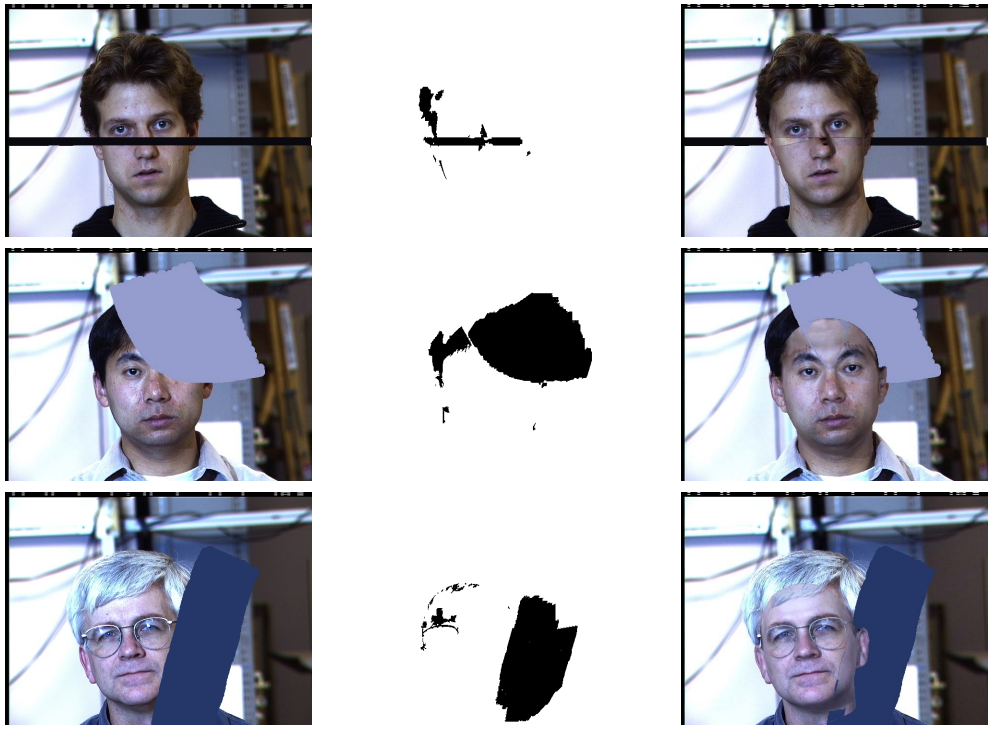


Figure 8.14: ***Remaining regions* as occlusion mask:** left column: input images with painted occlusions, middle: automatically generated occlusion masks (only the *remaining regions* extending beyond the face), right: reconstruction results with completed textures rendered back into the input image

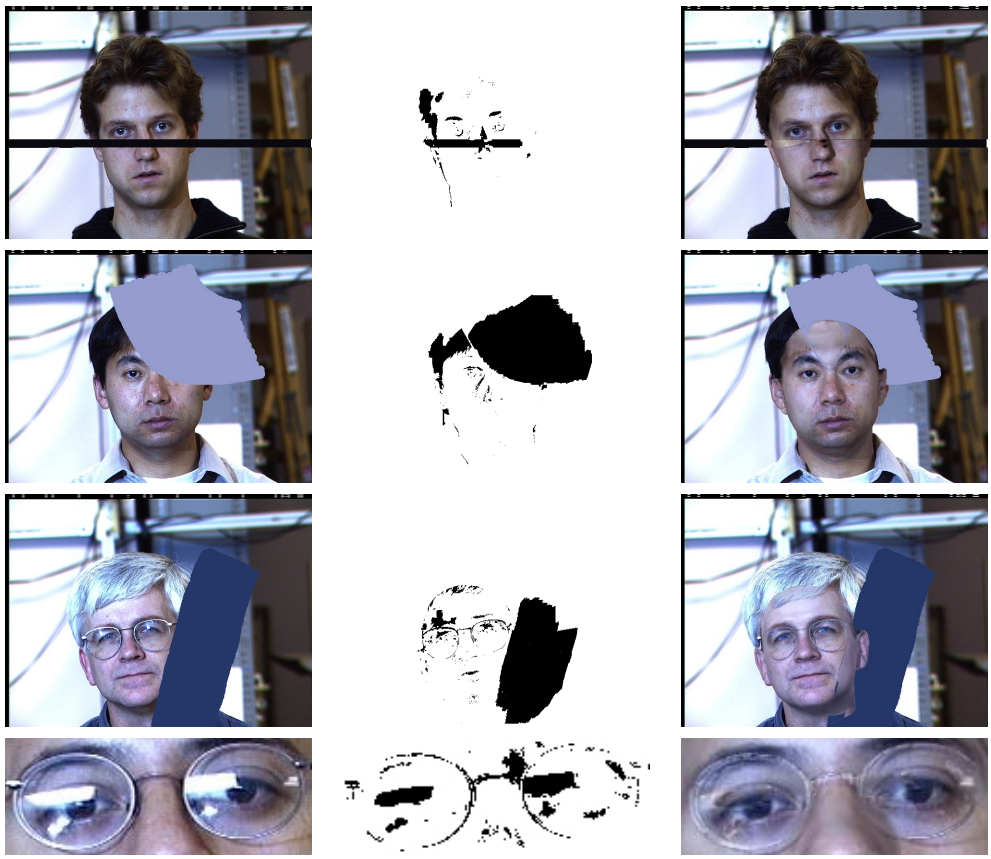


Figure 8.15: ***Remaining regions + first guess* as occlusion mask:** left column: input images with painted occlusions (and at the bottom a close up of an eye region), middle: automatically generated occlusion masks (*remaining regions* extending beyond the face combined with the *first guess*), right: reconstruction results with completed textures rendered back into the input image

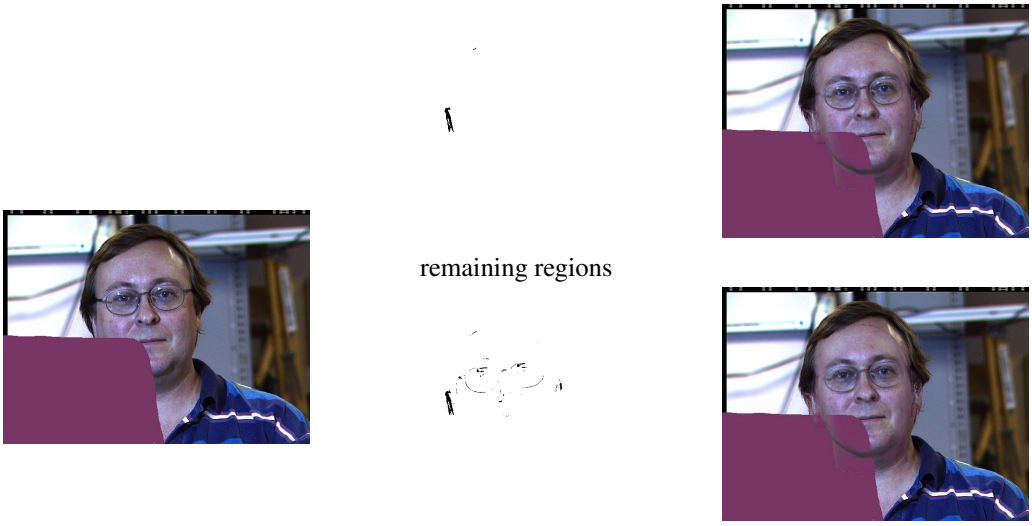


Figure 8.16: **Negative example:** left: input image with painted occlusion, middle: automatically generated occlusion masks (*remaining regions* extending beyond the face only, and combined with the *first guess*), right: reconstruction results with completed textures rendered back into the input image

8.4.2 Peripheral Occlusions

This subsection focuses on peripheral occlusions around the face, more precisely on forehead, neck and ears, where hair or clothes can interfere with the fitting algorithm. In the previous subsection (8.4.1) it has been shown how to detect glasses and specular highlights using a simple criterion and the right threshold. The same criterion has also been used to measure a *first guess* of the occlusion mask. Widespread occlusions, caused be objects in front of the face, have been detected by using a controlled growth of the first mask. Hair and clothes are a kind of widespread occlusions, too. In this subsection we concentrate on the peripheral areas of the face. Peripheral occlusions are stretching out over the boundaries of the face, too, but they do not widely extend into the facial area. Focusing on this special kind of widespread occlusions, we are able to detect them more precisely. An overview over the workflow described in this subsection is given in Fig. 8.17.

The first step of the algorithm is a conservative (coarse) fit as described in Chap. 8.4.1. But this time not the whole head is adapted to the input image, but only the inner part: forehead, neck and ears are ignored (cf. Fig 8.18). So the head adapts more precisely to the inner part of the face and the difference between the resulting model and the occluding areas in the image becomes more conspicuous.

At this approach the three color channels are not used separately but both, original image and rendered image, are transformed into grey scale images. If an manually defined occlusion mask already exists, it is taken into account, too. Also the z-buffer of the resulting head is stored to know whether a pixel belongs to the face (foreground) or to the background, which is needed in the final step.

For the comparison there are two options: working on vertices or on triangles. For both we have to check first whether a vertex or triangle is visible in the resulting image using the z-buffer (at a side view half of the face could be self-occluded or at a frontal view some parts of the neck could be self-occluded, too). Then the comparison is done by a loop over all visible vertices or visible triangles corresponding to one of the previously ignored areas (for triangles it suffices if one corner is part of the area). If the computed difference in the grey value is above a certain threshold (here 28), the pixel position of the rendered vertex or every pixel corresponding to the rendered triangle are marked as an occlusion (set to black in the occlusion mask):

$$\mathbf{occMask}(\mathbf{x_i}, \mathbf{y_i}) = \begin{cases} \text{set}, & \text{if} \quad (\mathbf{I}_{orig}(x_i, y_i) - \mathbf{I}_{model}(x_i, y_i)) > 28 \\ \text{not set}, & \text{else} \end{cases} \tag{8.3}$$

with $i \in S = \{\text{indices of visible vertices/triangles belonging to the prev. ignored parts}\}$

This yields a *first guess* of the occlusion mask.

For growing, we use a single pixel-by-pixel comparison as described in the first step, instead of edge detection. From each side of the image a search to the middle is initiated until a foreground pixel is hit. If this pixel or any pixel in the neighbourhood (two more are checked in search direction) is already marked as an occlusion, a new comparison in the opposite direction (towards

the borders of the image) is started from this point. If the computed difference in the grey value is above the threshold, too, this pixel is also marked as an occlusion (set to black in the occlusion mask). Remaining holes have to be filled in a final loop over the whole occlusion mask. Unmarked pixels near marked ones have to be checked if they have nearly the same color as the already marked occlusion. If this is the case, the pixel is also marked as part of the occlusion mask.

Results

This approach has been tested on several images of the FRGC database [PFS+05] and the FERET database [PWHR98]. Fig. 8.19 shows how the reconstruction gets 'better' (the shape changes and the texture at the ear gets brighter) using the intermediate occlusion masks of the different stages of the approach. In Fig. 8.20 some more results are shown. Here not only the hair, but also parts of the clothes interfering with the fitting are detected, discarded and replaced by the considered texture of the model. It can be seen as well that the occlusions do not only influence the texture reconstruction but also the shape. In the third row the differences in shape can be seen easily with the naked eye. Here, although the occlusion is placed at the neck, it influences the fitting result at the middle of the face. Not using an occlusion mask results in a peaked nose ((a) and (c)). Using the automatically generated occlusion mask improves the fit and the rounded shape of the nose is adapted better ((b) and (d)).

All examples in this chapter showed the importance of occlusion detection for good reconstruction results. Different approaches have been presented, to automate the detection. The approaches based on a *first guess* combined with region growth turned out to be the best.

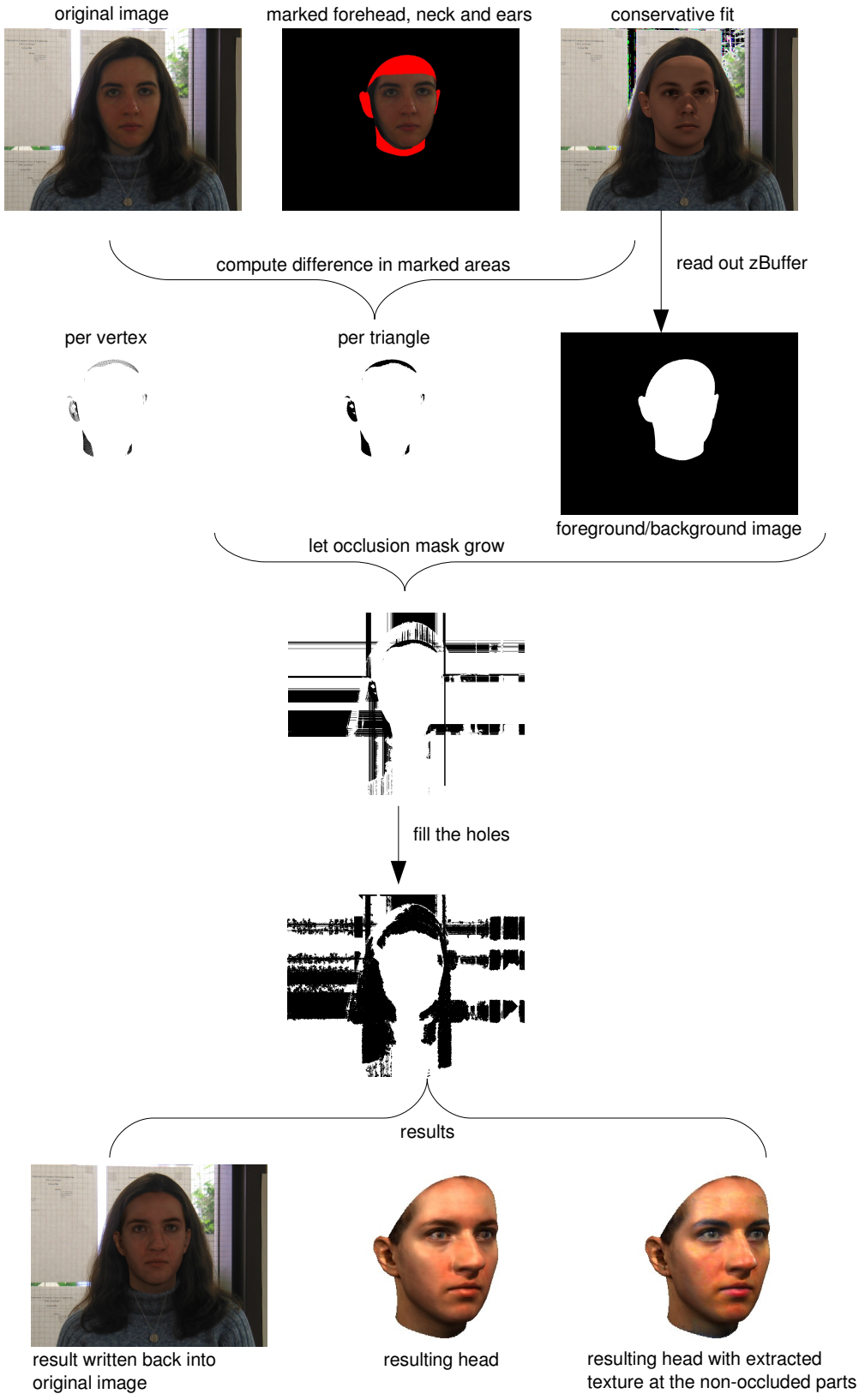


Figure 8.17: **Workflow of the region growth approach for peripheral occlusions**

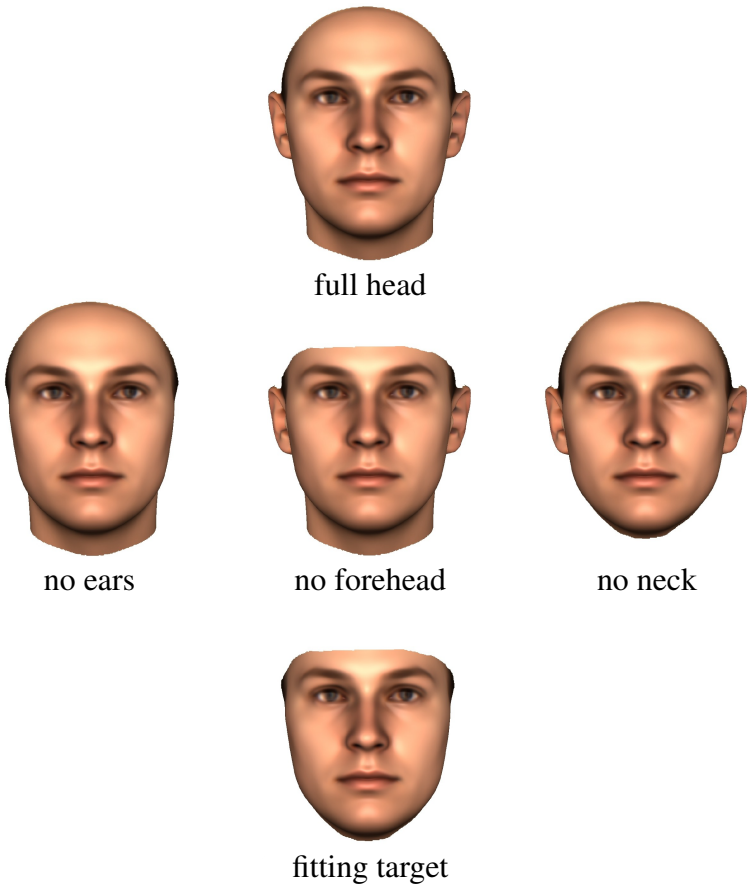


Figure 8.18: **Ignored parts**

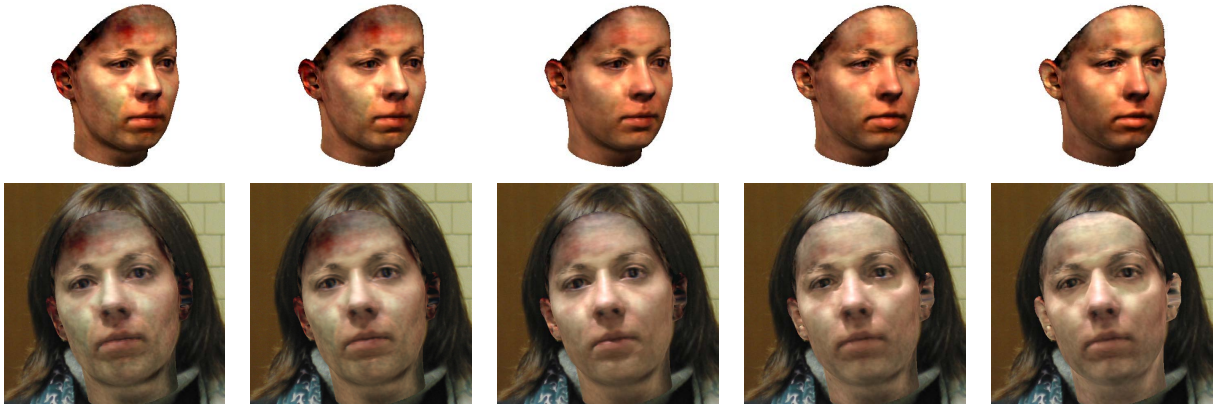


Figure 8.19: **Intermediate results:** Full reconstructions using occlusion masks at different stages of the algorithm. From left to right: no occlusion mask, *first guess* per vertex, *first guess* per triangle, after growing, final occlusion mask (after filling holes). First row: reconstructed heads, second row: reconstructions written back into original image

(a) (b) (c) (d)

Figure 8.20: **Full reconstructions with and without occlusion mask:** The left column shows close ups of the input images. From left to right the remaining columns show: (a): the reconstructed head not using an occlusion mask, (b): the reconstructed head using the occlusion mask generated with the approach in Chap. 8.4.2, (c): the same head shape as in (a) with the whole texture extracted from the input image on it, (d): the same head shape as in (b) with the original texture from the input image on it, but only at the non-occluded parts.

Chapter 9

Improving Reconstruction by Using Three Images

In this chapter, we introduce a new approach to get a better reconstruction out of three images instead of a single one.

As already mentioned in the related work chapter (Chap. 3.1.2), Amberg et al. [ARF+07] combined monocular and stereo cues to reconstruct 3D shape and pose from multiple images taken simultaneously. By replacing lighting and albedo estimation of the monocular method with the use of stereo information, and a colour difference cost function, they get higher accuracy and robustness against model albedo effects like make-up, facial hair, moles and cast shadows. Combining a soft edge detection scheme with a derivative based optimisation algorithm, the visible contour is fit to multiple images simultaneously. Faggian et al. [FPS08] use only the feature points for shape reconstruction but not the texture. To fit the model to multiple views they extend the error function from Blanz et al. [BMVS04], fit to the first input image and use the other images for shape update only. However, using the basic 3DMM algorithm of Blanz [Bla00] there is also already the option to reconstruct a head out of a number of images at the same time (*MultiView-Fit*):

In that approach the images are represented by a identical shape and texture, so they have the same parameters α_i, β_i but different rigid parameters $\rho_{i,1}, \rho_{i,2}, ...$ The cost function (Eqn. 2.7 in this thesis) includes the sum over all particular image distances

$$E(\vec{\alpha}, \vec{\beta}, \vec{\rho_1}, \vec{\rho_2}, ...) = E_{I,1} + E_{I,2} + ... + \eta \cdot E_P. \tag{9.1}$$

The stochastic Newton gradient descent is made according to the derivatives of E with respect to all parameters $\alpha_i, \beta_i, \rho_{i,1}, \rho_{i,2}, ...$ To simplify the implementation, at every single iteration of the gradient descent the image errors $E_{I,1}, E_{I,2}...$ are used in turn, and α_i, β_i, and $\rho_{i,j}$ are updated (here j corresponds to the index of the currently used image). For small step sizes this corresponds to the expected value of a minimization of the cost function (cf. Eqn. 9.1).

One deficit of this approach of Blanz [Bla00] is the fact that it does not exploit distinguishing features (wrinkles,...) well, as you can see at the upper left side of Fig. 9.3: If you compare the resulting texture to the original image in Fig. 9.1 (upper left side) it strikes that the wrinkles on the forehead disappeared.

To get a better alignment for this and other distinguishing features, we used the following new approach, where intrinsically a quasi stereo algorithm becomes operative: At first a full reconstruction only based on the frontal view is done. Then the resulting head and the extracted texture are used as new start values for fitting to the other views. The start head for the second step is close to the wanted one, and the further fitting can concentrate more on the shape details and it aligns the texture details. The start-texture is extracted from the frontal view out of the first input image. This texture can be retrieved in the other views of the same person. If the start head and the head in the second input image have the same texture, it is like using two registered images and a stereo algorithm. So the image error in the second step is more sensible for changes in the details and the shape of the head fits even better to the given images.

To test the approach example data was chosen from the Face Recognition Vendor Test (FRVT) [PGM$^+$03]. This database contains different viewing directions for each person: up, straight ahead (frontal view), down, right and left. Therefrom persons with remarkable characteristics where selected: three with striking wrinkles and three with moles and birthmarks. Not all views were used, but only frontal, right and left per person. The frontal views can be found in Fig. 9.1. The samples are named from (a) to (g) to refer to them later.

Different reconstruction scenarios were tested to show that the new approach really appears to be the best one to use. Fig. 9.2 shows the different considered reconstruction scenarios:

(I) Only the frontal view is used as input image. The reconstruction result is exploited by other scenarios as start value (*start head*).

(II) Only the frontal view is used as input image. Additionally, the previously computed *start head* (shape and texture) is used. This delivers a result as if the head has been fitted twice to the same image. This serves as a control condition.

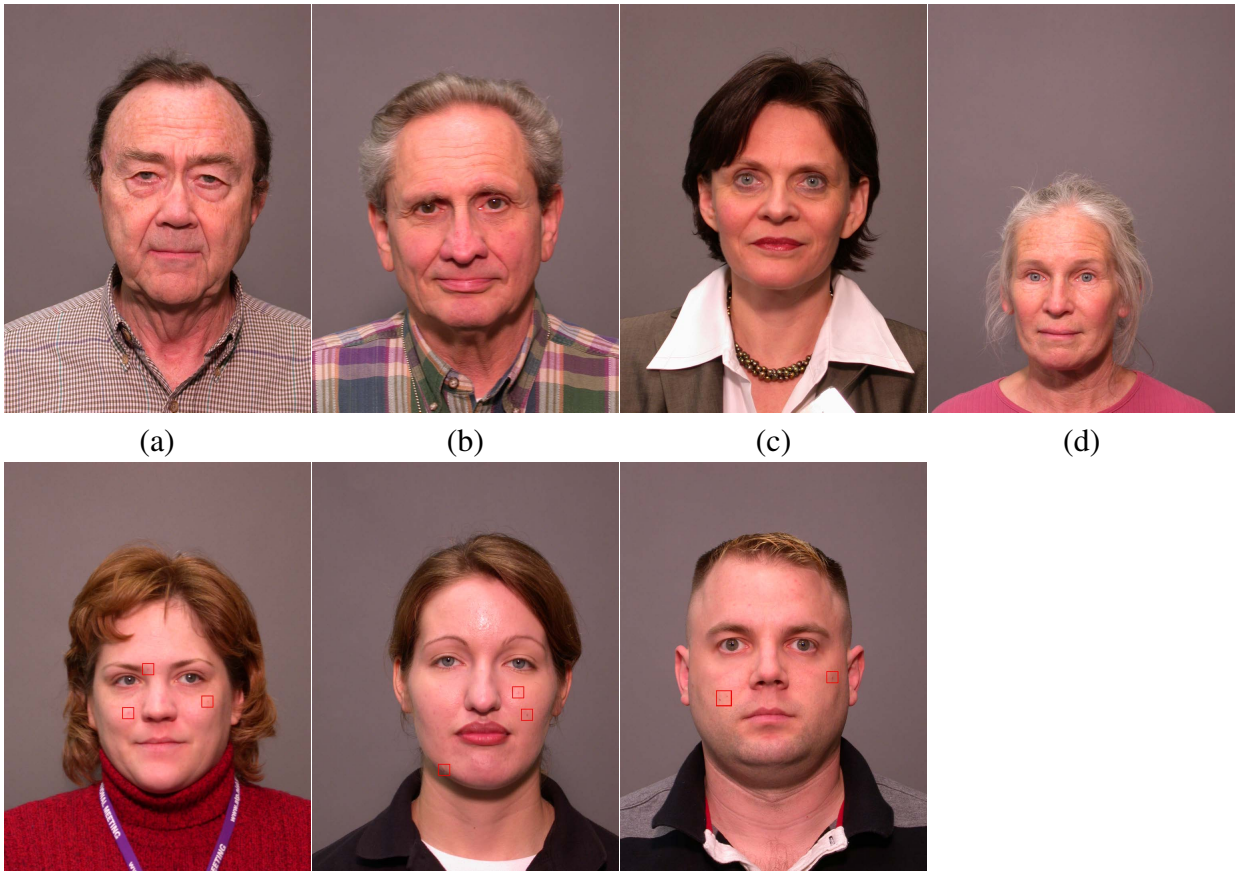
(III) All three views (frontal, left and right) are used as input images. This yields the result of the already existing *MultiView-Fit* (see above and [Bla00]).

(IV) Only the side views are used as input images. Additionally, the previously computed *start head* (shape and texture) is used.

(V) Only the side views are used as input images.

(VI) All three views are used as input images. Additionally, the previously computed *start head* (shape and texture) is used.

Fig. 9.2 also shows the resulting textures of each scenario and the resulting shapes (untextured heads) of the scenarios (I), (III) and (VI) using the example of person (c). For 'ground truth' (to compare with) we used the results of the reconstruction from the frontal view only (scenario (I)) and the existing *MultiView* reconstruction approach (scenario (III)).



(a) (b) (c) (d)

(e) (f) (g)

Figure 9.1: **Selected persons:** only frontal views are shown, first row: wrinkles, second row: moles and flecks (marked with red rectangles)

The resulting textures and untextured heads of sample (a) can be found in Fig. 9.3. However, there an additional texture image is shown. The image in the third row results also from the sixth scenario but there cast shadows were eliminated (in the following we refer to this with VI*). Sometimes the lighting estimated causes cast shadows under or besides the nose or at the ear. When extracting the texture from the input image the lighting is subtracted out, based on the lighting estimation. That is why at the areas with cast shadows white spots arise. This can be seen very clearly at the nose and under the ears of sample (d) and (f). Prohibiting the assumption of cast shadows at the fitting the white spots can be prevented. For the remaining samples ((b) to (g)) the resulting textures of scenario (I), (VI) and (VI*) can be found in Fig. 9.4 and the resulting heads of scenario (I) and (VI) in Fig. 9.5.

In Fig. 9.2 and 9.3 at the resulting textures of scenario (I) and (II) 'bleeding' of the hair can be seen: The color of the hair diffuses into the face area below. This arises from the fact that the viewing direction is perpendicular to the normals at this part of the head and the color values at the corresponding pixels of the input image can not be clearly correlated. At the textures of

scenario (IV) and (V) it can directly be seen that these are just composed of two parts. The two parts do not fit together correctly and even the start head does not help because the definite texture is composed only out of the last used input images. The textures of scenario (III) and (VI) look very similar because they are both composed of the same three images, but because of the more precise reconstruction of shape and light parameters, scenario (VI) delivers the better texture extraction: At sample (c) in Fig. 9.2, the texture from scenario (VI) is less blurry than the texture from scenario (III), in the area above the eyes, and at the part of the nose between the eyes. The texture from scenario (III) shows a shadow under the nose. This shadow does not appear in the texture from scenario (VI). At sample (a) in Fig. 9.3 the texture from scenario (III) shows some areas which are brighter than their surrounding, located around the eyes, and between nose and mouth. They are caused by imprecise reconstruction of light parameters. The texture from scenario (VI) does not show these brighter areas, and the wrinkles on the forehead are more distinguishable here than in the texture from scenario (III). The extracted texture can only be improved by eliminating the cast shadows as described above.

Looking at the untextured heads the differences can be seen, too (cf. Fig. 9.3 and 9.5). The images demonstrate that the shape of the resulting head gets more distinct and more precise, when using scenario (VI). At the results of scenario (VI) the facial features are much more pronounced than at scenario (I). To compare the results to the primary *MultiView-Fit* (scenario (III)), in Fig. 9.3 not only the results of the scenarios (I) and (VI) are shown but also the one of scenario (III). The difference between the head in the middle (scenario (III)) and at the right (scenario (VI)) is clearly visible, here most conspicuously at the shape of the chin. Fig. 9.5 shows that at some examples correspondence-related artifacts arise (a peak at the nose of sample (d) and (f) or a perturbation at the chin of sample (g)) but remarkable wrinkles can be identified still more distinctly (so the nasolabial fold of sample (c), also shown in Fig. 9.2).

A quantitative analysis would be difficult because it is hard to evaluate the precision of shape or texture to ground truth, even if a 3D scan is available: Coarse deformations affect the error measure, but still the result may look good. This is due to ambiguities in the reconstruction problem. We therefore restricted ourselves to a qualitative analysis on a small number of examples.

The tests show that the new approach is definitely suited for the general improvement of the textures, the better visibility of the wrinkles, and against the diffusion of the hair color. But regarding the visibility of the moles and flecks no obvious improvement can be seen compared to the already existing *MultiView-Fit*. The untextured heads also show the improvement and it is obvious that scenario (VI) delivered the most precise result because the facial features are much more pronounced.

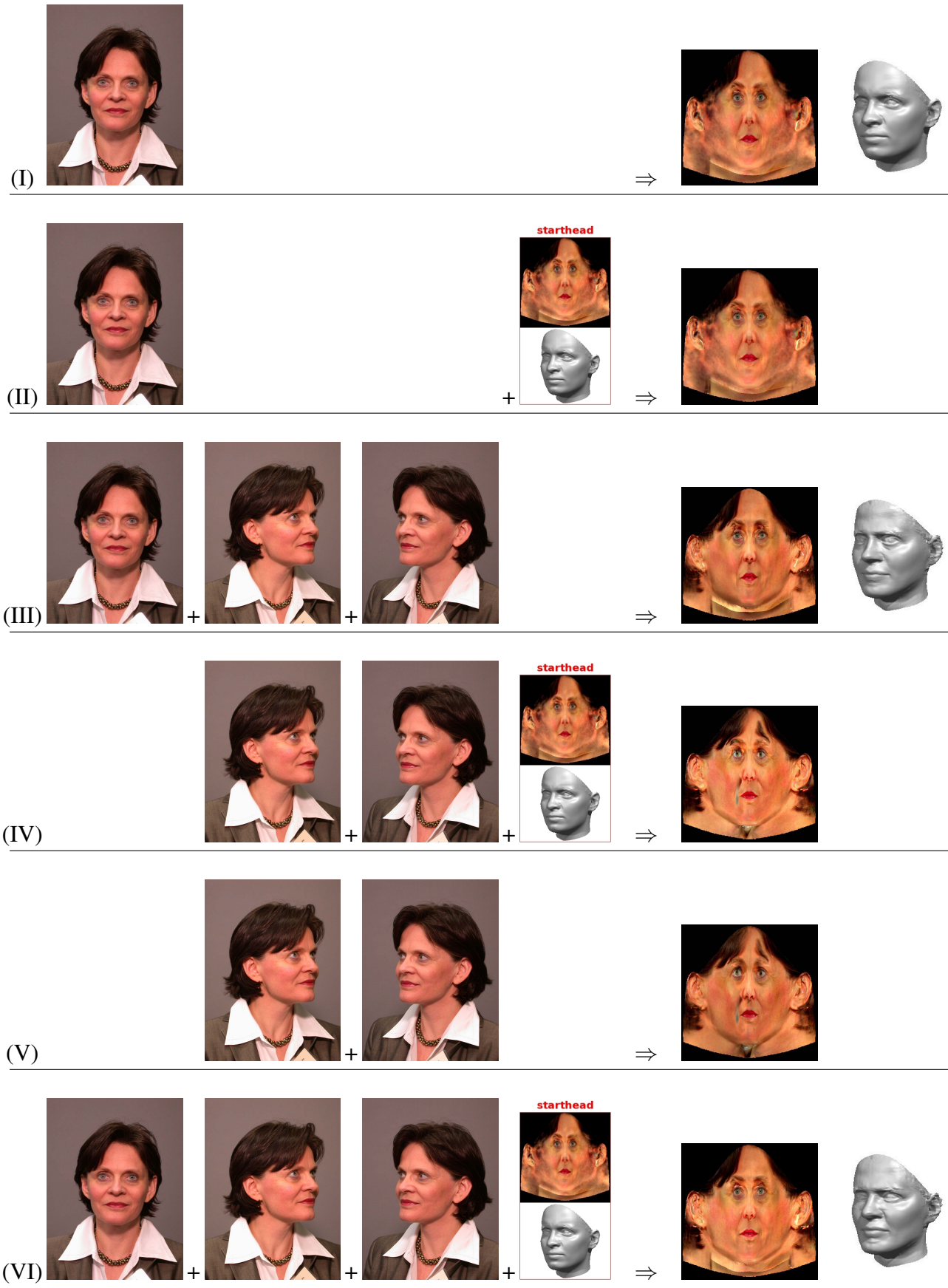
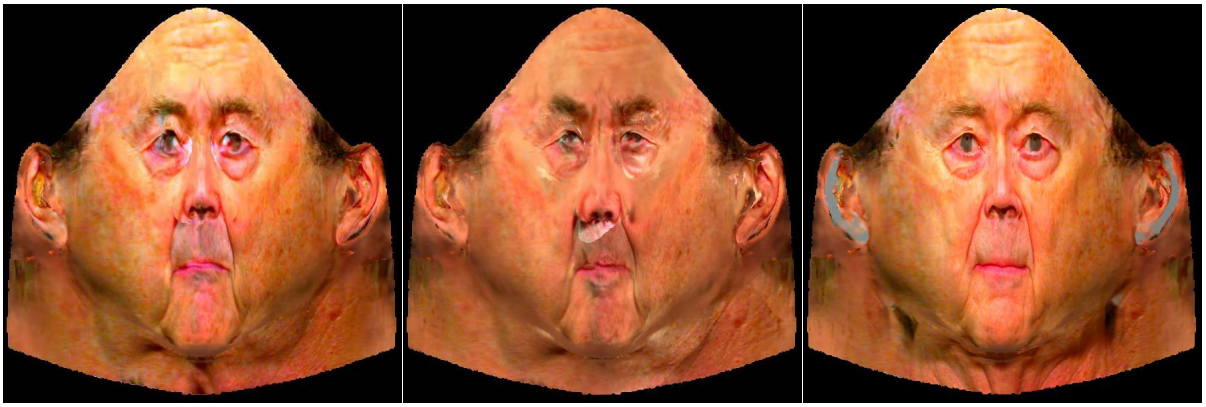


Figure 9.2: **Tested scenarios shown on sample (c)**

(I) (II) (III)



(IV) (V) (VI)

(VI*)

(I) (III) (VI)

Figure 9.3: **Resulting textures and heads of the different scenarios (sample (a)):** Textures in the first two rows are the results of scenario (I) to (VI). The texture in the third row is also the result of scenario (VI), but cast shadows have been eliminated (in the following we refer to this with (VI*)). The untextured heads in the last row are the results of scenario (I), (III) and (VI).

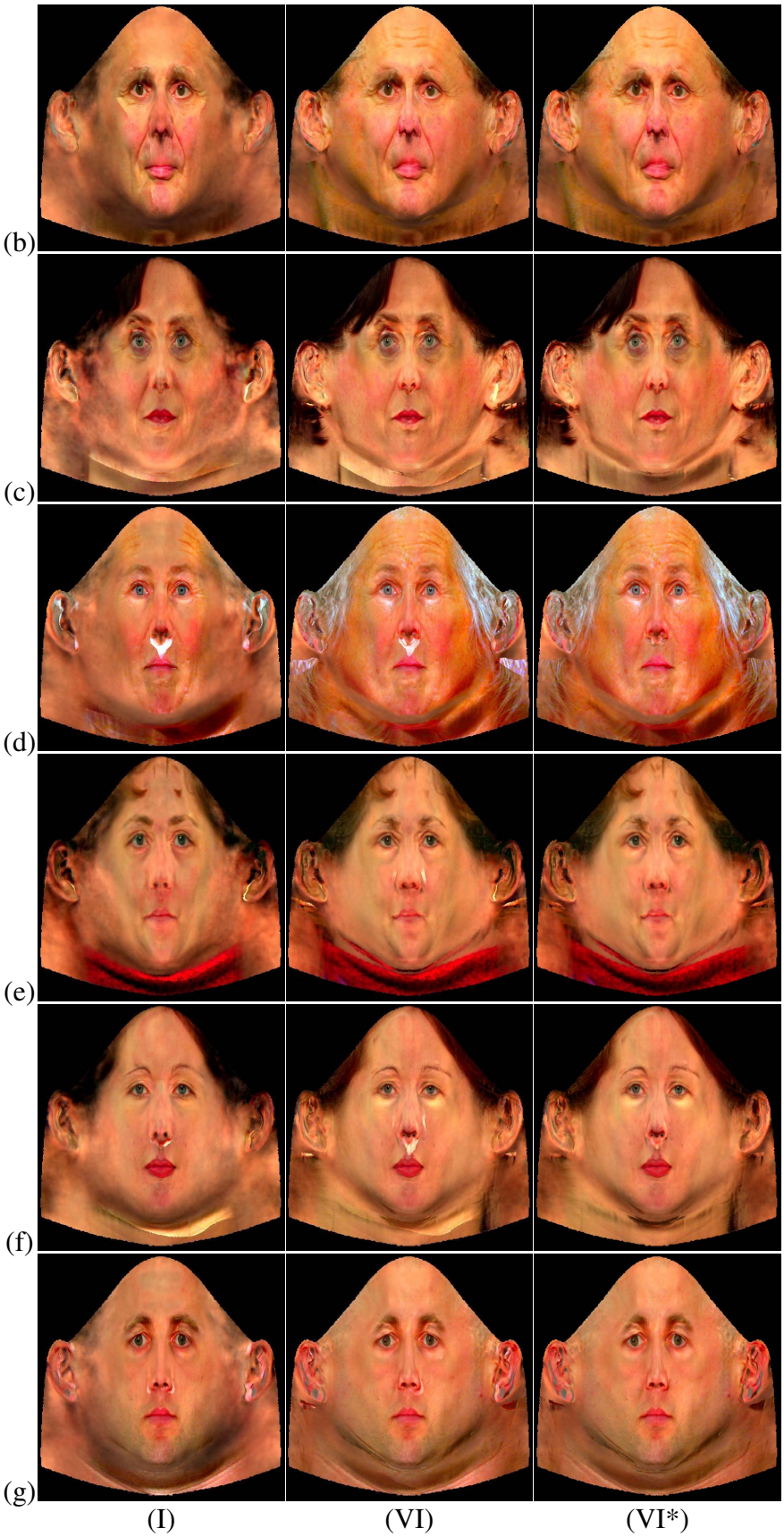


Figure 9.4: **Resulting textures of samples (b) to (g):** belonging to scenarios (I), (VI), and (VI*). The corresponding results of sample (a) can be found in Fig. 9.3.

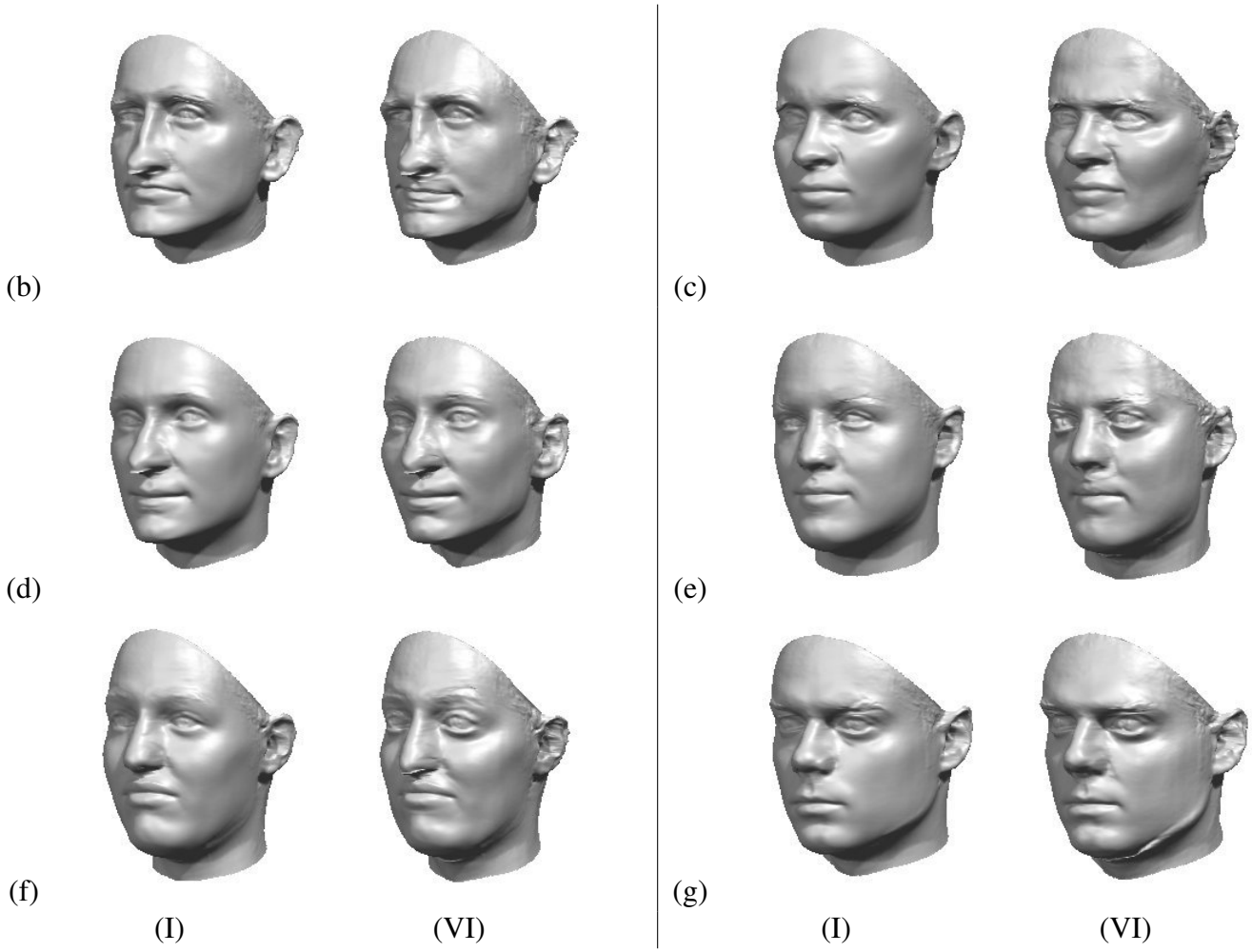


Figure 9.5: **Resulting heads of samples (b) to (g):** of scenario (I) and (VI) each. The corresponding results of sample (a) can be found in Fig. 9.3. The images demonstrate that the shape of the resulting head gets more distinct and more precise, when using scenario (VI).

Chapter 10

Conclusions and Future Work

This work shows the wide range of challenges, aiming on a robust automated face reconstruction and face recognition. We have presented contributions to processing steps preceding and during the fitting process of the Morphable Model of faces. The initial goal was reached to make the adaption process more robust and to allow a fully automatic reconstruction of a high-resolution model. Several aspects of the fitting algorithm have been studied in depth, weak points have been revealed and solutions have been provided.

Chapter 5 has addressed steps prior to the core fitting process. With the aid of the model, an automatic initialization has been facilitated. By combining Support Vector Machines and the 3D Morphable Model, we have addressed the problem of fully automated 3D shape reconstruction from raw video frames or other images. We presented a new 3D-based confidence measure for the plausibility of feature point configurations, using the Morphable Model. The system has proved to be robust with respect to a variety of imaging conditions, such as pose and lighting. The results and the rating scores by human participants demonstrate that the system produces a high percentage of photo-realistic reconstructions which makes it useful for a wide range of applications. For the future, it would be easily conceivable to use other detectors than SVM to provide different feature combinations. The new 3D-based confidence measure is independent of the specific choice of a feature selection method.

In the subsequent chapters, challenges within the adaption process have been successfully addressed. One of them was to cope with a weak initialization, and another was to improve the shape fitting by adapting to the contour more precisely.

We have presented a new approach for using feature detectors in 3DMM fitting in this thesis. The algorithm involves adaptive features, which is crucial to leverage the advantages of 3DMMs, and it is based on a new type of cost function that forms a tradeoff between feature similarity and some more global criteria such as geometric configuration, correct reproduction of color values and high prior probability. The evaluation is based on a scenario where the 3DMM fitting is initialized by a set of potentially unreliable feature detectors, and the algorithm iteratively refines the feature positions. The results show that the proposed algorithm improves recognition rates significantly. The other part of our evaluation is focused on the contributions of different design

options in our algorithm, and it demonstrates that both the adaptiveness and the new type of cost function increase the robustness. Our proposed cost function using the SAFL approach is a new contribution on a conceptual level, combining optimization and feature detection in a novel way. We do feel that this will stimulate discussions and further developments. Apart from this, there is space for further improvements. The matching decelerates the adaption process, highly depending on the size of the area containing the head. It could be made faster by reducing the image resolution or using a more efficient matching algorithm. Another point to think about is how to effectively hamper outliers. Here, the algorithm could be combined with the 3D-based confidence measure, presented in this thesis, too.

Improving the contour fitting of the 3DMM, different approaches have been introduced. Using directional constraints and considering a certain number of contour points every 2000 iterations, the model is forced to adapt to the detected contour. The effectiveness of this, on the supposition that the real contour line is known, has been shown. Not knowing the real contour line, the application of a special template matching approach has been demonstrated to be the most efficient solution. Prospectively also an occlusion mask can be considered for the contour fitting. If there are occlusions in the search direction it is highly probable that the real contour line is covered. In this case the contour search to this direction has to be abandoned.

This thesis also contains a study about different kinds of automated occlusion detections done. The goal is to identify occlusions and exclude them from the fitting process to let the model fit to the accurate parts of the image only. The most applicable detection algorithms are based on a *first guess* occlusion mask followed by a specific growing process. Their usefulness has been shown on both peripheral occlusions (like hair and clothes) and central occlusion (glasses, specular highlights, and objects in front of the face).

Finally, the benefit of reconstructing one model from multiple images has been demonstrated. It has been shown that the use of appropriate start conditions improves the reconstruction, too. Reconstructing a head out of a frontal view yield to this appropriate *start head*. Using a combination of this *start head* and three different views as input images, results in a more precise reconstruction of the shape of the head. After fitting, the original texture can be extracted from the input images to map it onto the reconstructed shape. The extraction result has been improved, too: no diffusion of colors occurs in the marginal areas of the face, and finer structures get visible.

The contributions presented in this dissertation may open up further developments. They all focus on specific weak points. In the future, they can be combined into one system. For this reason it has to be analyzed in which way these individual solutions influence each other and what is the best way of combination. Future investigations should also concentrate on the improvement of the fitting under difficult conditions. One possible approach would be a study about the behavior of the fitting for low image quality. With regard to the possibility of a wider field of application these robust and reliable models will become more and more important in the future.

Publications

Conferences

Pia Breuer, Christian Eckes and Stefan Müller. Hand Gesture Recognition with a novel IR Time-of-Flight Range Camera – A pilot study. In *Proceedings Mirage 2007, Computer Vision / Computer Graphics Collaboration Techniques and Applications*, 2007, INRIA Rocquencourt, France, pp. 247-260

Pia Breuer, Kwang-In Kim, Wolf Kienzle, Bernhard Schölkopf and Volker Blanz. Automatic 3D Face Reconstruction from Single Images or Video. In *Proceedings of the 8th International Conference on Automatic Face and Gesture Recognition*, 2008.

Pia Breuer and Volker Blanz. Self-Adapting Feature Layers. In *Proceedings of the 11th European Conference on Computer Vision*, 2010.

Journals

Christian Eckes, Konstantin Biatov, Frank Hülsken, Joachim Köhler, Pia Breuer, Pedro Branco and L. Miguel Encarnacao. Towards Sociable Virtual Humans: Multimodal Recognition of Human Input and Behavior. *The International Journal of Virtual Reality*, Volume 6, Number 4, pp. 21-30, December 2007.

Techn. Reports

Pia Breuer, Kwang-In Kim, Wolf Kienzle, Volker Blanz and Bernhard Schölkopf. Automatic 3D Face Reconstruction from Single Images or Video. *Max-Planck-Institut für Biologische Kybernetik – Technical Report No. 160*

Bibliography

[ACM09] Edoardo Ardizzone, Marco La Cascia, and Marco Morana. Probabilistic corner detection for facial feature extraction. In *Proceedings CIAP09*, 2009.

[AGR97] Joseph J. Atick, Paul A. Griffin, and A. Norman Redlich. Statistical approach to shape from shading: Reconstruction of 3D face surfaces from single 2D images. *Neural Computation*, 8:1321–1340, 1997.

[AKV08] Brian Amberg, Reinhard Knothe, and Thomas Vetter. Expression invariant face recognition with a morphable model. In *Proceedings of the 8th International Conference on Automatic Face and Gesture Recognition*, 2008.

[ARF+07] Brian Amberg, Sami Romdhani, Andrew Fitzgibbon, Andrew Blake, and Thomas Vetter. Reconstructing high quality face-surfaces using model based stereo. In *Proceedings of the 11th International Conference on Computer Vision*, 2007.

[AZ05] Ognjen Arandjelovic and Andrew Zisserman. Automatic face recognition for film character retrieval in feature-length films. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, pages 860–867, 2005.

[BB08] Wilhelm Burger and Mark J. Burge. *Digital Image Processing*. Springer, New York, NY, 2008. www.imagingbook.com.

[BB10] Pia Breuer and Volker Blanz. Self-adapting feature layers. In *Proceedings of the 11th European Conference on Computer Vision*, 2010.

[BHB00] Christoph Bregler, Aaron Hertzmann, and Henning Biermann. Recovering non-rigid 3D shape from image streams. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, pages 2690–2696, 2000.

[BKK+08] Pia Breuer, Kwang-In Kim, Wolf Kienzle, Bernhard Schölkopf, and Volker Blanz. Automatic 3D face reconstruction from single images or video. In *Proceedings of the 8th International Conference on Automatic Face and Gesture Recognition*, 2008.

[Bla00] Volker Blanz. *Automatische Rekonstruktion der dreidimensionalen Form von Gesichtern aus einem Einzelbild*. PhD thesis, Tübingen, Germany, 2000.

[BLGT06] Manuele Bicego, Andrea Lagorio, Enrico Grosso, and Massimo Tistarelli. On the use of SIFT features for face authentification. In *Proceedings IEEE CVPRW06*, 2006.

[BM92] Paul J. Besl and Neil D. McKay. A method for registration of 3–D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

[BMVS04] Volker Blanz, Albert Mehl, Thomas Vetter, and Hans-Peter Seidel. A statistical method for robust 3D surface reconstruction from sparse data. In Yiannis Aloimonos and Gabriel Taubin, editors, *2nd International Symposium on 3D Data Processing, Visualization, and Transmission, 3DPVT 2004*, pages 293–300, Thessaloniki, Greece, 2004. IEEE.

[Boo89] Fred L. Bookstein. Principal warps: Thin-plate splines and the decomposition of deformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1989.

[Bra01] Matthew Brand. Morphable 3D models from video. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, pages 456–463, 2001.

[BS97] Chris J. C. Burges and Bernhard Schölkopf. Improving the accuracy and speed of support vector machines. In *Advances in Neural Information Processing Systems*, volume 9, pages 375–381, 1997.

[BV99] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3D faces. In *Computer Graphics Proceedings SIGGRAPH'99*, pages 187–194, 1999.

[BV03] Volker Blanz and Thomas Vetter. Face recognition based on fitting a 3D morphable model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(9):1063– 1074, 2003.

[CAS] Oya Celiktutan, Hatice Cinar Akakin, and Bulent Sankur. Multi-attribute robust facial feature localization. In *Proceedings of the 8th International Conference on Automatic Face and Gesture Recognition*.

[CC03] Amit K. Roy Chowdhury and Rama Chellappa. Face reconstruction from monocular video using uncertainty analysis and a generic model. *Computer Vision and Image Understanding*, 91(1-2):188–213, 2003.

[CC06] David Cristinacce and Timothy F. Cootes. Feature detection and tracking with constrained local models. In *Proceedings IEEE British Machine Vision Conference*, 2006.

[CCS04] David Cristinacce, Timothy F. Cootes, and Ian Scott. A multi-stage approach to facial feature detection. In *Proceedings IEEE British Machine Vision Conference*, pages 277–286, 2004.

[CET98] Timothy F. Cootes, Gareth J. Edwards, and Christopher J. Taylor. Active appearance models. In Burkhardt and Neumann, editors, *Computer Vision – ECCV'98 Vol. II*, Freiburg, Germany, 1998. Springer, Lecture Notes in Computer Science 1407.

[CFM+92] Michael Cohen, Horacio Franco, Nelson Morgan, David Rumelhart, and Victor Abrash. Hybrid neural network / hidden markov model continuous speech recognition. In *Proceedings of the International Conference on Spoken Language Processing*, pages 915–918, 1992.

[CTCG92] Timothy F. Cootes, Christopher J. Taylor, David H. Cooper, and Jim Graham. Training models of shape from sets of examples. In *Proceedings IEEE British Machine Vision Conference*, 1992.

[CTCG95] Timothy F. Cootes, Christopher J. Taylor, David H. Cooper, and Jim Graham. Active shape models - their training and application. *Computer Vision and Image Understanding*, 61:38–59, 1995.

[DB04] Roman Dovgard and Ronen Basri. Statistical symmetric shape from shading for 3d structure recovery of faces. In T. Pajdla and J. Matas, editors, *Proceedings of the 8th European Conference on Computer Vision*. Springer-Verlag LNCS 3022, 2004.

[DHS01] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification*. John Wiley & Sons, New York, 2nd edition, 2001.

[DIF04] Miodrag Dimitrijevic, Slobodan Ilic, and Pascal Fua. Accurate face models from uncalibrated and ill-lit video sequences. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, June 2004.

[EL08] Victor Erukhimov and Kuang-Chih Lee. A bottom-up framework for robust facial feature detection. In *Proceedings of the 8th International Conference on Automatic Face and Gesture Recognition*, 2008.

[EZ06] Mark R. Everingham and Andrew Zisserman. Regression and classification approaches to eye localization in face images. In *Proceedings of the 7th International Conference on Automatic Face and Gesture Recognition*, pages 441–446, 2006.

[FB81] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[FIHO06] Nobuo Funabiki, Megumi Isofai, Teruo Higashino, and Masashi Oda. An eye-contour algorithm from face image using deformable templates. *Memoirs of the Faculty of Engineering, Okayama University*, 2006.

[FK07] Pawel Forczmanski and Georgy Kukharev. Comparative analysis of simple facial features extractors. *RealTimeIP*, 1(4):239–255, 2007.

[FPS08] Nathan Faggian, Andrew Paplinski, and Jamie Sherrah. 3D morphable model fitting from multiple views. In *Proceedings of the 8th International Conference on Automatic Face and Gesture Recognition*, 2008.

[FPZ03] Rob Fergus, Pietro Perona, and Andrew Zisserman. Object class recognition by unsupervised scale-invariant learning. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, 2003.

[FS95] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of online learning and an application to boosting. In *Computational Learning Theorie: Eurocolt'95*, 1995.

[Fua99] Pascal Fua. Using model-driven bundle-adjustment to model heads from raw video sequences. In *Proceedings of the 7th International Conference on Computer Vision*, Corfu, Greece, September 1999.

[FV08] C. Fernandez and Maria A. Vicente. Face recognition using multiple interest point detectors and SIFT descriptors. In *Proceedings of the 8th International Conference on Automatic Face and Gesture Recognition*, 2008.

[Gav99] Dariu M. Gavrila. Traffic sign recognition revisited. In *Proceedings of the 21st DAGM Symposium für Mustererkennung*, Springer, 1999.

[GILF08] Nemanja Grujić, Slobodan Ilić, Vincent Lepetit, and Pascal Fua. 3D facial pose estimation by image retrieval. In *Proceedings of the 8th International Conference on Automatic Face and Gesture Recognition*, 2008.

[GK06] Lie Gu and Takeo Kanade. 3D alignment of face in a single image. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

[GSBB03] Meirav Galun, Eitan Sharon, Ronen Basri, and Achi Brandt. Texture segmentation by multiscale aggregation of filter response and shape elements. In *Proceedings of the 9th International Conference on Computer Vision*, 2003.

[HB06a] Tal Hassner and Ronen Basri. Example based 3D reconstruciton from single 2D images. In *Proceedings IEEE CVPRW06*, 2006.

[HB06b] Bernd Heisele and Volker Blanz. Morphable models for training a component-based face recognition system. In W. Zhao and R. Chellappa, editors, *Face Processing - Advanced Modeling and Methods*, pages 439–462. 2006.

[HBH02] Jennifer Huang, Volker Blanz, and Bernd Heisele. Face recognition using component-based SVM classification and morphable models. In S.-W. Lee and A. Verri, editors, *Pattern Recognition with Support Vector Machines, First International Workshop, SVM 2002*, Niagara Falls, Canada, 2002. Springer-Verlag LNCS 2388.

[HHB03] Jennifer Huang, Bernd Heisele, and Volker Blanz. Component-based face recognition with 3D morphable models. In *Proceedings of the 4th International Conference on Audio- and Video-Based Biometric Person Authenticitation*, Surrey, UK, 2003.

[HL04] Scott Helmer and David G. Lowe. Object class recognition with many local features, 2004.

[HLM07] Yuchi Huang, Qingshan Liu, and Dimitris Metaxas. A component based deformable model for generalized face alignment. In *Proceedings of the 11th International Conference on Computer Vision*, 2007.

[HSP+02] Bernd Heisele, Thomas Serre, Massimiliano Pontil, Thomas Vetter, and Tomaso Poggio. Categorization by learning and combining object parts. In *NIPS*, 2002.

[ISO04] ISO/IEC. Iso/iec jtc 1/sc 37 n 506. Biometric Data Interchange Formats Part 5: Face Image Data, March 22, 2004.

[JM08] Hongjun Jia and Aleix M. Martinez. Face recognition with occlusions in the training and testing sets. In *Proceedings of the 8th International Conference on Automatic Face and Gesture Recognition*, 2008.

[KB06] Ira Kemelmacher and Ronen Basri. Molding face shapes by example. In *Proceedings of the 9th European Conference on Computer Vision*, 2006.

[KBFS05] Wolf Kienzle, Gökhan Bakir, Matthias Franz, and Bernhard Schölkopf. Face detection - efficient and rank deficient. In *Advances in Neural Information Processing Systems*, pages 673–680, 2005.

[KD08] Dong-Hoon Kim and Rozenn Dahyot. Face components detection using SURF descriptors and SVMs. In *Proceedings IMVIP08*, 2008.

[KKV07] Michael Keller, Reinhard Knothe, and Thomas Vetter. 3D reconstruction of human faces from occluding contours. In *Proceedings Mirage 2007, Computer Vision / Computer Graphics Collaboration Techniques and Applications*, 2007.

[KSYY08] Tatsuo Kozakaya, Tomoyuki Shibata, Mayumi Yuasa, and Osamu Yamaguchi. Facial feature localization using weighted vector concentration approach. In *Proceedings of the 8th International Conference on Automatic Face and Gesture Recognition*, 2008.

[LASG08] Kuang-Chih Lee, Dragomir Anguelov, Baris Sumengen, and Salih Burak Gokturk. Markov random field models for hair and face segmentation. In *Proceedings of the 8th International Conference on Automatic Face and Gesture Recognition*, 2008.

[Liu07] Xiaoming Liu. Generic face alignment using boosted appearance model. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[LKP03] Rainer Lienhart, Alexander Kuranov, and Vadim Pisarevsky. Empirical analysis of detection cascades of boosted classifiers for rapid object detection. In *DAGM'03, 25th Pattern Recognition Symposium*, 2003.

[LlQjW01] Ying Li, Xiang lin Qi, and Yun jiu Wang. Eye detection by using fuzzy template matching and feature-parameter-based judgement. *Pattern Recognition Letters*, 2001.

[Low99] David G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the 7th International Conference on Computer Vision*, 1999.

[Low04] David G. Lowe. Distinctive image features from scale-invariant keypoints. 2004.

[LPD+06] Georg Langs, Phillipp Peloschek, René Donner, Michael Reiter, and Horst Bischof. Active feature models. In *Proceedings International Conference on Pattern Recognition*, 2006.

[LPMM05] Jinho Lee, Hanspeter Pfister, Baback Moghaddam, and Raghu Machiraju. Estimation of 3D faces and illumination from single photographs using a bilinear illumination model. In *Rendering Techniques*, pages 73–82, 2005.

[LVB+93] Martin Lades, Jan C. Vorbrüggen, Joachim Buhmann, Jörg Lange, Christoph v.d. Malsburg, Rolf P. Würtz, and Wolfgang Konen. Distortion invariant object recognition in the dynamical link architecture. *IEEE Transactions on Computer*, (42), 1993.

[MN08] Stepehn Milborrow and Fred Nicolls. Locating facial features with an extended active shape model. In *Proceedings of the 10th European Conference on Computer Vision*, 2008.

[MS05] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2005.

[MT06] Yu Meng and Bernard Tiddeman. Implementing the scale invariant feature transform (SIFT) method. Technical report, Department of Computer Science, University of St. Andrews, May 2006.

[MWM08] Louis-Philippe Morency, Jacob Whitehill, and Javier Movellan. Generalized adaptive view-based appearance model: Integrated framework for monocular head pose estimation. In *Proceedings of the 8th International Conference on Automatic Face and Gesture Recognition*, 2008.

[NlT08] Minh Hoai Nguyen and Fernando De la Torre. Learning image alingment without local minima for face detection and tracking. In *Proceedings of the 8th International Conference on Automatic Face and Gesture Recognition*, 2008.

[NPlTF08] Minh Hoai Nguyen, Joan Perez, and Fernando De la Torre Frade. Facial feature detection with optimal pixel reduction svms. In *Proceedings of the 8th International Conference on Automatic Face and Gesture Recognition*, 2008.

[OFG97] E. Osuna, R. Freund, and F. Girosi. Training support vector machines: An application to face detection. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, pages 130–136, 1997.

[OKK+05] Jeong-Su Oh, Dong-Wook Kim, Jin-Tae Kim, Yong-In Yoon, and Jong-Soo Choi. Facial component detection for efficient facial characteristic point extraction. In *Proceedings ICIAR05*, 2005.

[Ope06] OpenCV library. http:/sourceforge.net/projects/opencvlibrary/, July 2006.

[PCRK05] Kun Peng, Liming Chen, Su Ruan, and Georgy Kukharev. A robust algorithm for eye detection on gray intensity face without spectacles. *Journal of Computer Science and Technology*, 2005.

[PFS+05] P. Jonathon Phillips, Patrick J. Flynn, Todd Scruggs, Kevin W. Bowyer, Jin Chang, Kevin Hoffman, Joe Marques, Jaesik Min, and William Worek. Overview of the face recognition grand challenge. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

[PGM+03] P. Jonathon Phillips, Patrick Grother, Ross J. Michaels, Duane M. Blackburn, Elham Tabassi, and Mike Bone. Face recognition vendor test 2002: Evaluation report. NISTIR 6965, National Institute of Standards and Technology, 2003.

[PJ06] Unsang Park and Anil K. Jain. 3D face reconstruction from stereo video. In *Proceedings of the 3rd Canadian Conference on Computer and Robot Vision*, 2006.

[PKA+09] Pascal Paysan, Reinhard Knothe, Brian Amberg, Sami Romdhani, and Thomas Vetter. A 3D face model for pose and illumination invariant face recognition. In *Sixth IEEE International Conference on Advanced Video and Signal Based Surveillance*, 2009.

[Pla99] John C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in Large-Margin Classiers*, pages 61–74. 1999.

[PTVF92] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C*. Cambridge University Press, Cambridge, 1992.

[PV07] Jean-Sébastien Pierrard and Thomas Vetter. Skin detail analysis for face recognition. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[PWHR98] P. Jonathon Phillips, Harry Wechsler, Jeffrey Huang, and Patrick J. Rauss. The FERET database and evaluation procedure for face recognition algorithms. *Image and Vision Computing*, 16(5):295–306, 1998.

[Rei97] M.J.T. Reinders. Eye tracking by template matching using an automatic codebook generation scheme. In *Proceedings Third annual conference of the Advanced School for Computing and Imaging*, 1997.

[RKB04] Carsten Rother, Valdimir Kolmogorov, and Andrew Blake. "GrabCut": interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics*, 23(3):309–314, 2004.

[RKX+08] Krishnan Ramnath, Seth Koterna, Jing Xiao, Changbo Hu, Iain Matthews, Simon Baker, Jeffrey Cohn, and Takeo Kanade. Multi-view AAM fitiing and construction. *International Journal of Computer Vision*, 2008.

[RPV06] Sami Romdhani, Jean-Sébastien Pierrard, and Thomas Vetter. 3D morphable face model, a unified approach for analysis and synthesis of images. In W. Zhao and R. Chellappa, editors, *Face Processing - Advanced Modeling and Methods*, pages 127–158. 2006.

[RTBA07] Antonio Rama, Francesc Tarrés, J.M. Braup, and Ramón Alujas. Face detection using fuzzy integral. In *Proceedings ICIP07*, 2007.

[RTN07] Antonio Rama, Francesc Tarrés, and Jacek Naruniec. A robust non-linear face detector. In *Proceedings SIGMAP07*, 2007.

[RTSB04] Sami Romdhani, Philip Torr, Bernhard Schölkopf, and Andrew Blake. Efficient face detection by a cascaded support-vector machine expansion. In *Proceedings of the Royal Society of London*, 2004.

[RV03] Sami Romdhani and Thomas Vetter. Efficient, robust and accurate fitting of a 3D morphable model. In *Proceedings of the 9th International Conference on Computer Vision*, 2003.

[RV05] Sami Romdhani and Thomas Vetter. Estimating 3D shape and texture using pixel intensity, edges, specular highlights, texture constraints and a prior. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

[RV07] Sami Romdhani and Thomas Vetter. 3D probabilistic feature point model for object detection and recognition. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[SBB01a] Eitan Sharon, Achi Brandt, and Ronen Basri. Segmentation and boundary detection using multiscale intensity measurements. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, 2001.

[SBB01b] Terence Sim, Simon Baker, and Maan Bsat. The CMU pose, illumination, and expression (PIE) database of human faces. Technical Report CMU-RI-TR-01-02, The Robotics Institute, Carnegie Mellon University, January 2001.

[SBB02] Terence Sim, Simon Baker, and Maan Bsat. The CMU pose, illumination, and expression (PIE) database. In *Proceedings of the 5th International Conference on Automatic Face and Gesture Recognition*, pages 53–58, 2002.

[SGH95] M. Schenkel, I. Guyon, and D. Henderson. On-line cursive script recognition using time-delay neural networks and hidden markov models. *Machine Vision and Applications*, 8(4):215–223, 1995.

[SH05] William A. P. Smith and Edwin R. Hancock. Recovering facial shape using a statistical surface normal model. In *Proceedings ICIP*, 2005.

[SH08] M. Saquib Sarfraz and Olaf Hellwich. Statistical appearance models for automatic pose invariant face recognition. In *Proceedings of the 8th International Conference on Automatic Face and Gesture Recognition*, 2008.

[SK00] Henry Schneiderman and Takeo Kanade. A statistical method for 3D object detection applied to faces and cars. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, 2000.

[SLC08] Jason M. Saraghi, Simon Lucey, and Jeffrey F. Cohn. Deformable face fitting with soft correspondence constraints. In *Proceedings of the 8th International Conference on Automatic Face and Gesture Recognition*, 2008.

[SMB+99] Bernhard Schölkopf, Sebastian Mika, Chris J. C. Burges, Philipp Knirsch, Klaus-Robert Müller, Gunnar Rätsch, and Alex J. Smola. Input space vs. feature space in kernel-based methods. *IEEE Transaction on Neural Networks*, 10(5):1000–1017, 1999.

[STC04] J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

[TÖ5] Klaus D. Tönnies. *Grundlagen der Bildverarbeitung*. Pearson Studium, München, 2005.

[TWTP] Feng Tang, Jin Wang, Hai Tao, and Qunsheng Peng. Probabilistic hierarchical face model for feature localization. In *Proceedings WACV07*.

[Vap00] Vladimir N. Vapnik. *The Nature of Statistical Learning Theory - Second Edition*. Springer-Verlag, New York, 2000.

[VJ01] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, 2001.

[VJ04] Paul Viola and Michael Jones. Robust real-time face detection. *International Journal of Computer Vision*, 75(2):137–154, 2004.

[Wan07] Xin Wang. Laplacian operator-based edge detectors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2007.

[WBMR08] Jianxin Wu, S. Charles Bribaker, Matthew D. Mullin, and James M. Rehg. Fast asymmetric learning for cascade face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2008.

[WGJW05] Peng Wang, Matthew B. Green, Qiang Ji, and James Wayman. Automatic eye detection and its validation. In *Proceedings IEEE Workshop on Face Recognition Grand Challenge Experiments (with CVPR)*, 2005.

[WHHB04] Benjamin Weyrauch, Jennifer Huang, Bernd Heisele, and Volker Blanz. Component-based face recognition with 3D morphable models. In *Proceedings IEEE Workshop on Face Processing in Video, FPIV04*, 2004.

[WLS+02] Chenyu Wu, Ce Liu, Heung-Yeung Shum, Ying-Qing Xu, and Zhengyou Zhang. Automatic eyeglass removel from face images. In *Proceedings of the 5th Asian Conference on Computer Vision*, 2002.

[WMR08] Matthias Wimmer, Christoph Mayer, and Bernd Radig. Robustly classifying facial components using a set of adjusted pixel features. In *Proceedings of the 8th International Conference on Automatic Face and Gesture Recognition*, 2008.

[XBMK04] Jing Xiao, Simon Baker, Iain Matthews, and Takeo Kanade. Real-time combined 2D+3D active appearance models. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, June 2004.

[XWT+05] Le Xin, Qiang Wang, Jianhua Tao, Xiaoou Tang, Tieniu Tan, and Harry Shum. Automatic 3D face modeling from video. In *Proceedings of the 10th International Conference on Computer Vision*, 2005.

[XY04] Yi Xiao and Hong Yan. Extraction of glasses in human face images. In *Proceedings of the International Conference on Biometrics*, 2004.

[YKA02] M.-H. Yang, D.J. Kriegman, and N. Ahuja. Detecting faces in images: a survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(1):34–58, 2002.

[ZC99] Wen-Yi Zhao and Rama Chellappa. Robust face recognition using symmetric shape-from-shading, university of maryland, CARTR-919, 1999.

[ZC00] Wen-Yi Zhao and Rama Chellappa. Illumination-insensitive face recognition using symmetric shape-from-shading. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, pages 286–293, 2000.

[ZdW04] Fei Zuo and P.H.N. de With. Fast facial feature extraction using a deformable shape model with haar-wavelet based local texture attributes. In *Proceedings ICIP04*, 2004.

[ZdW05] Fei Zuo and Peter H.N. de With. Facial feature extraction using a cascade of model-based algorithms. In *Proceedings AVSBS05*, 2005.

[ZLA+04] Zhengyou Zhang, Zicheng Liu, Dennis Adler, Michael F. Cohen, Erik Hanson, and Ying Shan. Robust and rapid generation of animated faces from video images: A model-based modeling approach. *International Journal of Computer Vision*, 58(2):93–119, 2004.