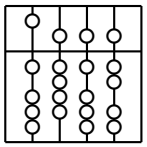
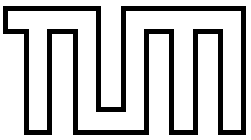


Institut für Informatik

der Technischen Universität München



Model-based Image Interpretation with Application to Facial Expression Recognition

Dissertation

Matthias Wimmer

Institut für Informatik
der Technischen Universität München

Model-based Image Interpretation with Application to Facial Expression Recognition

Matthias Wimmer

Vollständiger Abdruck der von der Fakultät für Informatik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzender:	Univ.-Prof. Dr. Johann Schlichter
Prüfer der Dissertation:	
1.	Univ.-Prof. Dr. Bernd Radig
2.	Prof. Timothy F. Cootes, Ph.D., University of Manchester/UK

Die Dissertation wurde am 12.04.2007 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 15.10.2007 angenommen.

Abstract

Computers are excellent devices for quickly solving mathematical problems and for memorizing an enormous extent of information. Nevertheless, the interaction between humans and computers still lacks intuition, because it is restricted to traditional input and output devices. This thesis focuses on augmenting traditional systems with aspects of interpersonal communication in order to resolve these shortcomings. It describes methods that robustly localize facial features, seamlessly track them through image sequences, and interpret facial expressions.

The general research statement of this thesis is that model-based techniques have great potential to fulfill current and future requests on interpreting images. Unfortunately, remaining challenges, such as the initial model parameterization, still present major obstacles to making these systems usable in real-world scenarios.

The contributions of my thesis are twofold: First, it shows that face model fitting algorithms benefit from well-defined color features that are able to distinguish between the different regions of a face, such as the skin, the lips, and the eyebrows. However, these parts have only slight differences in color and therefore, the decision criterion must be well chosen. The proposed approach adapts to the person and to the context first, and then classifies skin color via general purpose color classifiers. This procedure maintains real-time performance and obtains high accuracy, which makes it appropriate for a variety of applications such as face model fitting, gaze estimation, and facial expression recognition.

Second, this thesis focuses on fitting models to images by considering the objective function as the most important component involved. These functions are usually determined heuristically in a time-consuming and error-prone procedure that requires much domain-dependent knowledge. This thesis investigates and explicitly formulates inevitable properties of ideal objective functions. Furthermore, it proposes a methodology for learning objective functions from annotated example images while considering these properties. Therefore, the learned functions are approximately ideal as well. The benefits of this approach are that the crucial decision steps during function design are automated and the remaining manual steps require little or no computer vision expertise. This procedure lays the foundation for a general application of model-based image interpretation to real-world scenarios and it has therefore potential for commercialization.

Kurzfassung

Die Stärken von Computern liegen darin, mathematische Probleme schnell zu lösen und eine enorme Informationsmenge zu verarbeiten. Allerdings schränkt die Verwendung herkömmlicher Eingabe- und Ausgabemodi die Interaktion zwischen Mensch und Maschine stark ein. Die vorliegende Arbeit behandelt den Forschungsaspekt, diesen Mangel durch die Integration zwischenmenschlicher Kommunikationsmechanismen zu beseitigen. Diese Arbeit zeigt Verfahren, die Gesichtsmerkmale lokalisieren, diese durch Bildsequenzen verfolgen und daraus die sichtbare Mimik ableiten.

Der Forschungsansatz dieser Arbeit betrachtet modellbasierte Techniken als fähig, aktuelle und zukünftige Anforderungen hinsichtlich der Bildinterpretation zu erfüllen. Ein heutiger Einsatz dieser Systeme wird unter anderem durch die schwierige initiale Modellparametrisierung verhindert.

Der Beitrag, den diese Arbeit dabei liefert, gliedert sich in zwei Teilbereiche: Zunächst zeigt sie die leichte Einpassung von Gesichtsmodellen mithilfe von klar definierten Farbmerkmalen, die zwischen den unterschiedlichen Regionen des menschlichen Gesichts differenzieren, wie zum Beispiel der Haut, den Lippen und den Augenbrauen. Allerdings erfordern die geringfügigen Farbunterschied eine genaue Bestimmung des Auswahlkriteriums. Der hier vorgestellte Ansatz passt sich zunächst der Person und den Kontextbedingungen an, bevor die Hautfarbe durch gebräuchliche Farbklassifikatoren bestimmt wird. Diese Vorgehensweise erfüllt Echtzeitbedingungen und liefert einen hohen Grad an Genauigkeit, die eine Integration in Echtzeitanwendungen ermöglichen.

Im Weiteren behandelt diese Arbeit das Einpassen von Modellen in Bilder und betrachtet dabei die Objective Function als die wichtigste beteiligte Komponente. Diese Funktionen werden gewöhnlich heuristisch in zeitintensiven und fehleranfälligen Arbeitsschritten bestimmt, die viel Fachwissen erfordern. Diese Dissertation untersucht und formalisiert zuerst die Eigenschaften von idealen Objective Functions. Es wird anschließend eine Herangehensweise vorgeschlagen, die Objective Functions mithilfe von annotierten Beispielbildern trainiert. Da dabei die idealen Eigenschaften in Betracht gezogen werden, verhält sich die gelernte Funktion auch annähernd ideal. Die Vorteile dieses Ansatzes zeigen sich darin, dass die ausschlaggebenden Designentscheidungen automatisiert werden und die verbleibenden Arbeitsschritte wenig oder

kein Expertenwissen benötigen. Diese Vorgehensweise bildet somit die Grundlage für eine allgemeine Anwendung von modellbasierter Bildinterpretation in realen Umgebungen und sie verfügt über ein großes Potential für den kommerziellen Einsatz.

Danksagung

Die vorliegende Arbeit entstand während meiner Tätigkeit am Lehrstuhl für Bildverstehen und Wissensbasierte Systeme der Fakultät für Informatik der Technischen Universität München. Ihre Anfertigung wäre ohne die Unterstützung einer Reihe von Personen nicht möglich gewesen, denen ich an dieser Stelle ganz herzlich danken möchte.

Zunächst bedanke ich mich bei meinem Doktorvater Prof. Dr. Bernd Radig für die Betreuung der Arbeit und seine Anregungen zur wissenschaftlichen Gestaltung des Promotionsvorhabens. Durch die Aufnahme in seine Arbeitsgruppe ermöglichte er mir, dieses interessante Thema zu bearbeiten. Vielen Dank auch für die Unterstützung zur Fortführung meiner wissenschaftlichen Interessen. Er sowie Prof. Michael Beetz verstanden es stets, für ein angenehmes Arbeitsklima und zwischenmenschliches Verhältnis zu sorgen. Ebenfalls danke ich Prof. Tim Cootes für die bereitwillige Übernahme des Zweitgutachtens.

Weiterhin bedanke ich mich bei meinen Kollegen des Lehrstuhls und des Forschungsverbunds FORSIP für die gute Zusammenarbeit. Insbesondere meine langjährigen Zimmerkollegen Suat Gedikli und Simone Hämmerle unterstützten mich in vielerlei Hinsicht. Des Weiteren konnte ich dort auch Freunde für außeruniversitäre Unternehmungen für die Freizeitgestaltung finden, besonders mit Kajetan Berlinger und mit Heiko Gottschling. Außerdem bedanke ich mich bei Freek Stulp für die ergebnisreichen Diskussionen hinsichtlich gemeinsamer Publikationen. Allen Studenten, die im Rahmen ihres Systementwicklungsprojekts bzw. ihrer Diplomarbeit direkt oder indirekt am Entstehen dieser Arbeit mitgewirkt haben, danke ich auch herzlich. Stellvertretend seien Sylvia Pietzsch und Christoph Mayer genannt.

Für die inhaltliche Durchsicht meiner Arbeit danke ich ganz herzlich meinen Konferenzkollegen Wolfgang Sepp und Johan Karlsson, sowie Daniel Fischer für die Verbesserungsvorschläge, die meiner Meinung nach deutlich zur Lesbarkeit beigetragen haben. Außerdem bedanke ich mich bei allen Personen, die mir die Veröffentlichung von Bildmaterial genehmigten.

Bedanken möchte ich mich auch bei meinen Eltern, die mir das Studium ermöglichten und damit den Grundstein für meine Promotion und diese Arbeit legten.

Contents

Chapter 1

Introduction and Motivation

Humans are able to communicate with each other without great effort. Analyzing human communication generates a splitting into so-called communication channels. Humans intuitively know how to utilize each of these channels and how to combine them reasonably, because they learn about this interaction scheme from the very beginning of their childhood. Therefore, we consider this interaction mechanism to be intuitive for humans. The communication channels comprise the auditory (hearing), the visual (sight), the tactile (touch), the olfactory (smell), and the gustatory (taste) channel, see Pürer et al. [118]. The participants of human communication handle each channel in the same way and therefore, these channels are bi-directional. Figure 1.1 gives an insight on this topic.

1.1 Multimodal Human-computer Interaction

Traditionally, the strengths of computers have been their ability to quickly solve mathematical problems and to memorize an enormous extent of information. Years ago, a small number of well-educated experts were able to handle these machines. Nowadays, computers and further electrical devices enhance our everyday life and support people at work as well as at home, e.g. they simplify long-distance communication and put new aspects to entertaining people. Nevertheless, these machines are still difficult to handle, because they are far from utilizing human communication mechanisms, i.e. using human communication channels. The exchange of information between humans and machines is usually achieved via a display and a number of buttons. People have to explicitly learn how to interact with each particular device. There-

fore, we consider this communication scheme neither intuitive nor human-like. Analyzing this communication scheme brings up a number of communication channels as well. Figure 1.1 illustrates these channels. A machine is neither capable of expressing nor of interpreting the information of certain communication channels. Communication over most channels happens to be uni-directional. This makes communication artificial, laborious, and non-intuitive. Figure 1.1 depicts the mechanism of this information exchange.

Due to frequent use, people gradually adapt to this interaction scheme, but seldom-accessed functionality requires reading the manual. Therefore, this way of interaction becomes tedious for experienced users and it may present a considerable obstacle for the average consumer.

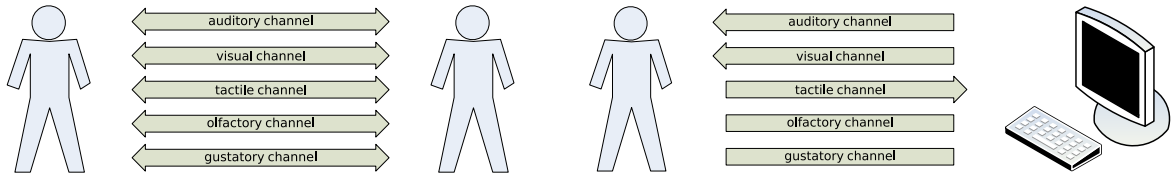


Figure 1.1: The different communication channels that are used by humans and machines. Interpersonal communication involves all communication channels whereas human-computer interaction just supports particular communication channels to be used for a certain direction. Machines will need to express and interpret all of them in order to communicate properly.

Integrating the communication channels of interpersonal interaction into human-computer interaction will provide a more intuitive and more comfortable way of handling technical devices. Humans are aware of this communication scheme, because it is required for the communication with other people. They have learned about it from the very beginning of their childhood and are gradually adjusting it during their lives. Therefore, they are not urged to inform themselves about the various instructions for that are necessary to operate a specific device. In consequence, the user does not have to adapt to the machine any more. In this novel interaction scheme, the machine adapts to the human beings rather than the other way around

For the benefit of natural interaction mechanisms engineers equip technical systems with sensors. This interaction scheme has been coined *multimodal human-computer interaction*. Microphones determine spoken words, which are translated into commands to be executed by the system. Cameras perceive human bodies and faces and recognize their gestures, intentions, focus of interest, mood, and further related aspects.

As an essential part of multimodal human-computer interaction, the interpretation of facial expressions evolved to be an important research focus in the area of computer vision during the last decade, see Pantic et al. [116], Chibelushi et al. [21], Tian et al. [149], Cohen et al. [22], and Essa et al. [47; 46]. This thesis elaborates on the state-of-the-art of facial expression interpretation. It explains the design of these systems and describes sophisticated components developed so far. Based on this technology, it proposes a novel approach of two core components for fitting a face model to the camera image: extraction of skin-colored regions and the acquisition of robust objective functions. In order to prove the applicability in real-world scenarios, we built a proof-of-concept that combines state-of-the-art components with the algorithms contributed by us.

1.2 Problem Description

Model-based image interpretation contributes enormously to the promising approaches of automatically recognizing facial expressions. This image interpretation scheme is known to greatly facilitate the interpretation of real-world scenes in general. A deformable model stores a priori information about human faces. Fitting this face model to the camera image represents an intermediate step for facial expression interpretation. In a subsequent step, high-level descriptors are derived from the model parameters more easily, see Figure 1.2. This happens to be much more accurate than inferring the information directly from the image or from low-level image features.

Nevertheless, model-based image interpretation inevitably requires correctly detecting the model within the image and accurately tracking it through a sequence of images in real-time. This nontrivial task that has been coined *model fitting*, has not been sufficiently solved yet. With special focus on facial expression interpretation, this thesis addresses two issues of model fitting that it considers to be most important. These are the extraction of salient image features and the formulation of a robust objective function.

The task of fitting models to images relies on the extraction of *salient image features* that describe the correct model parameters more accurately than the plain image pixels. This improves the robustness of this task. Image features are appropriate if they correlate with the correct parameterization. In addition, the value of these features must be independent of side conditions as well Skin color regions are considered to be salient features for fitting a face model to images. However, the feature extraction process must separate skin from very similarly colored

parts such as lips. The color of these regions is highly influenced by the characteristics of the image and the visible person, which represents a great challenge to designing robust decision rules. Facial expression interpretation demands sophisticated techniques that are both accurate and real-time capable.

Figure 1.2: Fitting a face model to an image facilitates facial expression recognition.

The goal that model fitting follows is to find the model with the highest fitness to a particular image, i.e. matches the image best. An *objective function* determines a scalar value that describes the fitness. Therefore, the model fitting task is also formulated as a mathematical optimization problem. During the last decades, various fitting algorithms for determining precise model parameters have been devised. However, their accuracy relies on a well-specified objective function. It is a nontrivial challenge to set up calculation rules that derive a comparable value from the raw image data or low-level image features.

Currently, computer vision experts usually choose image features that they consider to be salient by hand. From these, the calculation rules of the objective function are composed manually. This approach requires expertise, but it also highly relies on intuition and therefore, it is considered to be rather an art than science [163]. Nevertheless, the obtained results are far from optimal and influence the involved fitting algorithm extremely. Furthermore, this methodology is not generally applicable and requires expertise both in computer vision and in the domain of interpreting the object.

1.3 Solution Outline

We approach the challenge of facial expression recognition via model-based image interpretation. In order to break down the complexity of the entire interpretation task, these systems consist of several components that calculate intermediate results, see Chapter 2. For more than a decade, a lot of research has been conducted for each part. We propose a solution for the

two previously stated shortcomings of model-based image interpretation. First, we robustly extract skin-colored regions from the camera image. Second, we automatically learn the objective function from manually annotated example images.

Skin color regions are considered to be salient for fitting a face model, because the borders of these areas are highly relevant for the position of a face model. The algorithms for extracting these regions must be both quick and accurate. Our solution first determines characteristics that describe the visible person and the image. Second, we adjust a general purpose skin color classifier to the obtained image characteristics. Machine learning techniques provide the calculation rules that are necessary for this specialization. Using a simple classifier, our solution provides high runtime performance and at the same time the adaptation turns out to be highly accurate.

In order to improve the accuracy of objective functions, we explicitly formalize the properties of ideal objective functions. We also state a concrete example of such an ideal objective function that bases on manually annotating the images with the correct model parameterization. However, we cannot apply this function to previously unseen images where the correct model parameters are unknown. Therefore, our aim is to approximate this objective function by machine learning techniques. The learning phase determines correspondences between a set of given image features and the result value of the ideal objective function and, in turn, it infers appropriate calculation rules. This approach enforces the resulting objective function to approximate the properties of an ideal objective function and turns the art of designing objective functions into a science.

For demonstration purposes, we implemented our algorithms in a proof-of-concept that is capable of facial expression interpretation.

1.4 Contributions

The achievements of this thesis are twofold, and contribute to model-based image interpretation with special focus on facial expression interpretation.

First, this thesis shows how to accurately extract skin-colored regions from a camera image for the benefit of face model fitting applications. The proposed skin color classification scheme provides more robustness than straightforward pixel-based color classifiers by preserving their runtime performance, because it adapts to the image conditions and to the visible person beforehand. Additionally, this approach is easily extendible to other scenarios that require distinguishing objects of similar color.

Second, this thesis elaborates on robustly fitting models to images by considering the objective function to be the most important component involved. It explicitly formulates properties that ideal objective functions have and proposes a methodology for approximating the result of these ideal functions. Therefore, this procedure enforces high accurate calculation rules. Since the chosen machine learning technique selects relevant image features only, the obtained objective functions have very fast runtime characteristics. The proposed approach is applicable to general scenarios, and does not require domain-dependent expertise. This ease-of-use and its relevancy for common model-based image interpretation challenges make it viable for commercialization.

1.5 Outline of the Thesis

The remainder of this thesis is organized as follows:

Chapter 2 introduces the concept of model-based image interpretation. It enumerates the components involved and explains their purpose and mutual interaction. Furthermore, it explains our proof-of-concept that is capable of fitting a deformable face model to images for the benefit of facial expression interpretation. It serves as a workhorse for explaining and testing our algorithms and achievements.

Chapter 3 illustrates current techniques that aim at extracting color features from the image. It describes our approach to robustly determine skin-colored regions by obtaining image-specific and person-specific characteristics first and then adapting a high-performance skin color classifier accordingly.

Chapter 4 identifies the shortcomings of the traditional approach to create objective functions by hand and it investigates the properties that ideal objective functions would have. Furthermore, it proposes a novel methodology for creating robust objective functions by learning them from annotated training images. It explains our procedure step by step, and it conducts comprehensive evaluations on the design and on the accuracy of the obtained objective functions.

Chapter 5 elaborates on facial expression interpretation by explaining its psychological and social background, by evaluating human accuracy on this task, and by explaining our approach on this topic. Our approach infers facial expressions from the model parameterization and from further image features. Finally, this chapter evaluates the results of our survey on human capabilities in recognizing facial expressions.

Chapter 6 summarizes the presented techniques and the achievements and points out their benefits for model-based image interpretation. Chapter 7 elaborates on the continuation within this research area.

Appendix B enlists and explains the mathematical notation that is used throughout the chapters of this thesis and Appendix A formally proves our statements about the ideal objective function, on which our contributions base.

Chapter 2

Model-based Image Interpretation

Image interpretation is an emerging field of computer science with widespread and beneficial applications within the industrial, medical, and military area. However, this challenge has not been solved adequately for real-world scenarios yet, because an object's visual appearance varies significantly between different images. This challenges image interpretation systems, because they need to take all these variations into consideration.

The integration of models into the image interpretation task is considered to be highly promising for real-world scenarios. Great success in challenges such as facial expression recog-

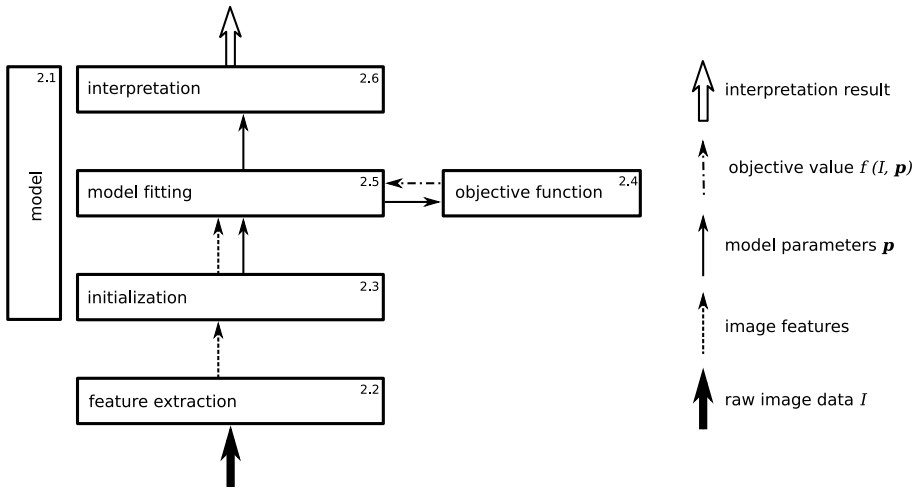


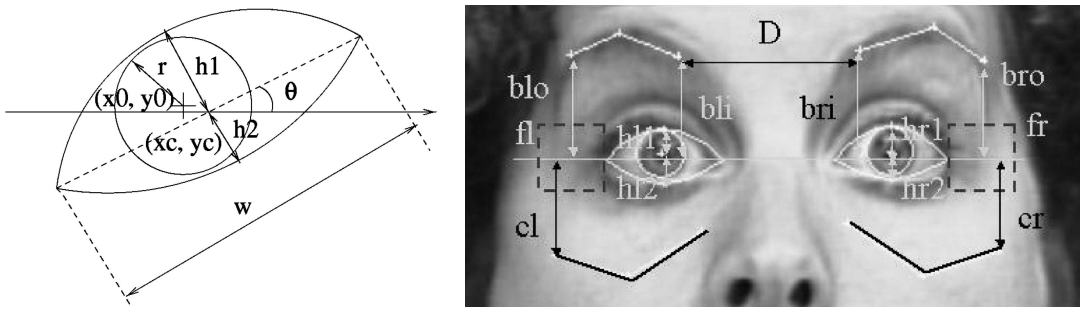
Figure 2.1: Model-based image interpretation splits the challenge of image interpretation into computationally independent modules. The upper right corners refer to the sections with detailed explanation.

nition, body posture detection, gesture recognition, have already been achieved by Tian et al. [149], Essa et al. [46], Grest et al. [61], Cohen et al. [22], Sebe et al. [131], and Blanz et al. [13]. We refer to this approach as *model-based image interpretation* and describe the involved components in this chapter. Furthermore, we set up a proof-of-concept for facial expression recognition based on this approach and explain each involved component in detail.

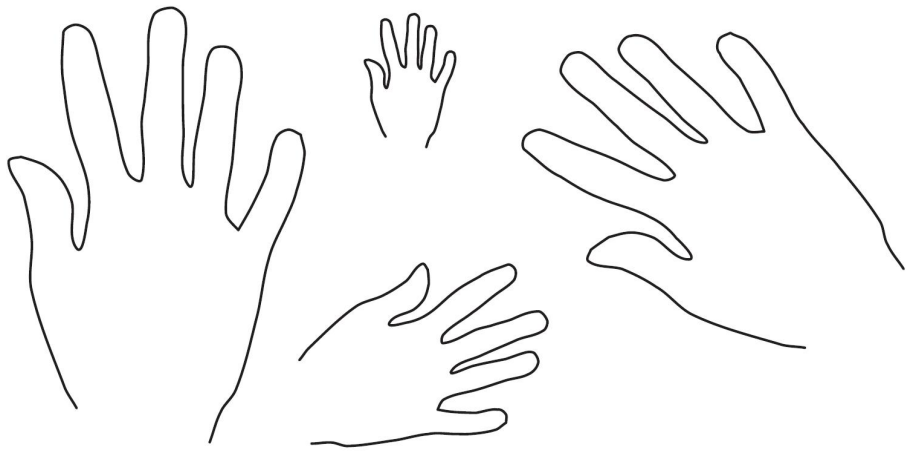
Model-based image interpretation exploits a priori knowledge about the object, such as the geometry of its shape or the structure of its surface. A model represents this knowledge in an abstract manner. Models serve as an intermediate representation of the scene during the interpretation process. A set of model parameters allows variations to the model that comprise its deformation, texture, pose, position, etc. *Model fitting* is the computational challenge of finding the model parameterization that describes the content of the image best [68]. This fitting procedure reduces the large amount of image information to a small set of parameters, which facilitates and accelerates further image interpretation.

Similarly, *model tracking* represents the challenge of finding the best model parameters for a sequence of images. In this special case, the model parameters are precisely predicted, because the content of subsequent images does not change rapidly. Section 2.5 explains this issue in more detail. Nevertheless, model tracking is not handled explicitly by this thesis.

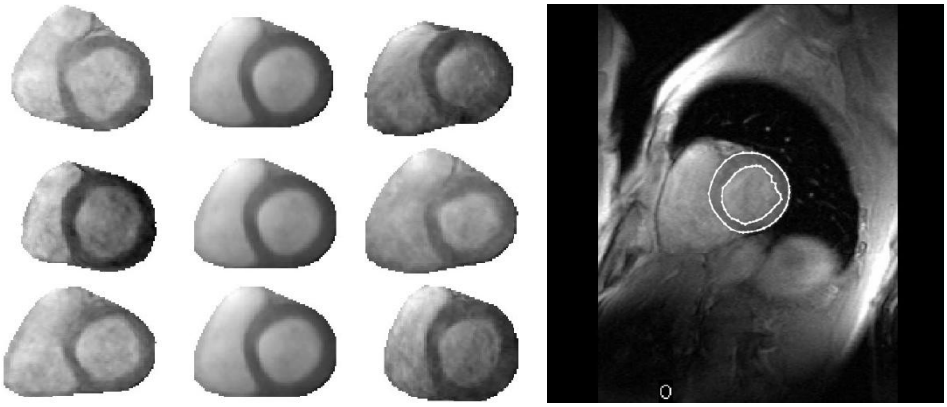
The scheme of model-based image interpretation facilitates the complex interpretation task by decomposing it into semantically and computationally independent components, see Figure 2.1. This methodology enables implementing each module with different and separate techniques. In contrast to considering raw image data, *feature extraction* computes reliable feature values that are less noisy, enforce particular image structures, or transform the image into a different representation. The challenge of finding the correct model parameters is split up into computing a rough estimate by *initialization* and precisely refining this estimate by *model fitting*. This splitting is reasonable, because the two modules are provided with different information and therefore rely on different assumptions. The challenge of the former module is to locate the object without any prior knowledge of the current image and of the model parameterization. In contrast, the latter module assumes an approximately correct initialization of the model parameters and it has to refine these parameters. The *objective function* computes a comparable value that indicates the fitness between a model parameterization and an image. This function is inherently required by any model fitting algorithm, but often specified implicitly. Finally, the *interpretation* module infers the interpretation result from the model parameters. Note that concrete implementations of this image interpretation scheme often conduct further



contour, deformable, set up by intuition, Tian et al. [149]



contour, deformable, learned from statistics, Stegmann et al. [141]



texture, deformable, learned from statistics, Stegmann et al. [142]

Figure 2.2: Different two-dimensional models represented via a set of geometric primitives, a contour, or a textured region.

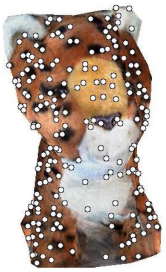
information exchange between the individual components; e.g. the interpretation does not rely on the model parameters only, but also takes image features into account.

From Section 2.1 to Section 2.6, we describe the components of model-based image interpretation in detail. Section 2.7 introduces our proof-of-concept that utilizes model-based image interpretation for facial expression recognition. It elaborates on the instantiation of each component using state-of-the-art techniques. Chapter 3, Chapter 4, and Chapter 5 explain our contributions to model-based image interpretation with particular focus on recognizing facial expressions. These are an adaptive skin color classification scheme, a novel methodology to create robust objective functions via machine learning techniques, and interpretation techniques.

2.1 Models

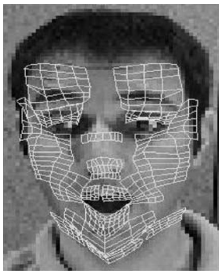
In computer vision, a model semantically represents particular aspects of a real-world object, usually in terms of the geometry of its shape and of the texture of its surface. Thereby, it focuses on the object's characteristics that are significant for the interpretation task. Because of their various utilizations, different types of models have been developed, such as two-dimensional and three-dimensional models, contour and texture models, rigid and deformable models, etc. For a recent overview about models used for facial expression interpretation, we refer to Romdhani [123]. Furthermore, for the application of different face models to various interpretation challenges, see Basso et a. [4], Blake et al. [11], Blanz et al. [12], Cohen et al. [22], Dimitrijevic et al. [38], and Tian et al. [149]. Figure 2.2 illustrates various models that are defined by the contour of the object and the models in Figure 2.3 additionally focus on the texture of the surface.

A parameter vector $\boldsymbol{p}$ represents the current constitution of the model, which includes the position, the pose, the deformation, the texture, etc. The only legal manipulations to the model are modifications to this vector. A corresponding projection function $\boldsymbol{c}(\boldsymbol{p})$ maps the parameterized model to the surface of the image. Depending on the type of the model, this function provides a set of feature points, a contour, a mesh, or a textured region. Thereby, a contour is defined as a set of contour points that are partially connected by lines, whereas a textured region is represented by a polygonal area that contains differently colored pixels whose color is specified by the model parameters.

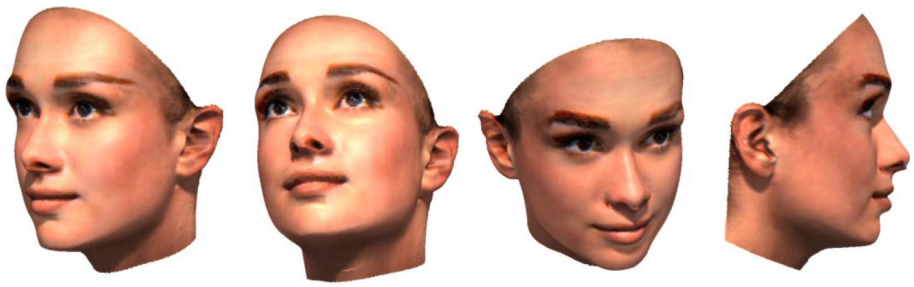


feature points, non-deformable, Lepetit et al. [95]

feature points, non-deformable, Lepetit et al. [96]



mesh, deformable, Cohen et al. [22]



texture, deformable, Blanz et al. [12]

Figure 2.3: Different three-dimensional models represented by a set of feature points, a mesh, and texture.

2.2 Feature Extraction

The process of fitting a model to an image benefits from the extraction of salient image features. These features are intended to describe the location, the size, and the deformation of the object more precisely than the color values or intensity values of the image pixels. Features are appropriate if they correlate with the fitting result on the one hand and if they are invariant to side conditions that are not relevant to the fitting task on the other hand.

Throughout the history of mathematics and computer vision, researchers have developed various image features and image transformations. There are edge features [19; 133; 54; 117], corner features [70], color features [50], optical flow [20], smoothing operators, image transformation, wavelet transformations [35], Scale Invariant Feature Transform (SIFT) [103], Local Binary Pattern (LBP) [114], and many more. Researchers also refer to this computational step as *low-level image analysis* [40].

In the case of fitting a contour model to human faces, appropriate features are the location of the eyes, the corners of the lips, the skin-colored regions, etc. To some extent these features are invariant to side conditions of the fitting task such as the color of the iris, the ethnic group of the person, the presence of glasses, shadows, and noise. Section 2.7.2 describes Haar-like features as exemplary features that are known for their robustness and they are therefore commonly used for interpreting real-world scenes, as we do for facial expression recognition.

2.3 Initialization

The initialization step roughly estimates the model parameters for the image. This step is necessary, because the succeeding model fitting algorithm usually requires or at least performs more accurately having an estimate of the model parameters available. Nevertheless, a vague guess is sufficient, because model fitting will enhance the model parameterization. The particular challenge of this step is that no information about the object and the model parameterization is available beforehand. Therefore, the raw image data and the extracted image features are the only sources of information. This step is also referred to as *model detection*, *model localization*, or *model extraction*. Figure 2.4 depicts several results of locating a face, a car, etc.

Only a fraction of the model parameters is usually estimated by this phase, such as the position parameters. The remaining parameters are set to a fixed value. This step benefits from an accurate extraction of appropriate image features by the preceding feature extraction.

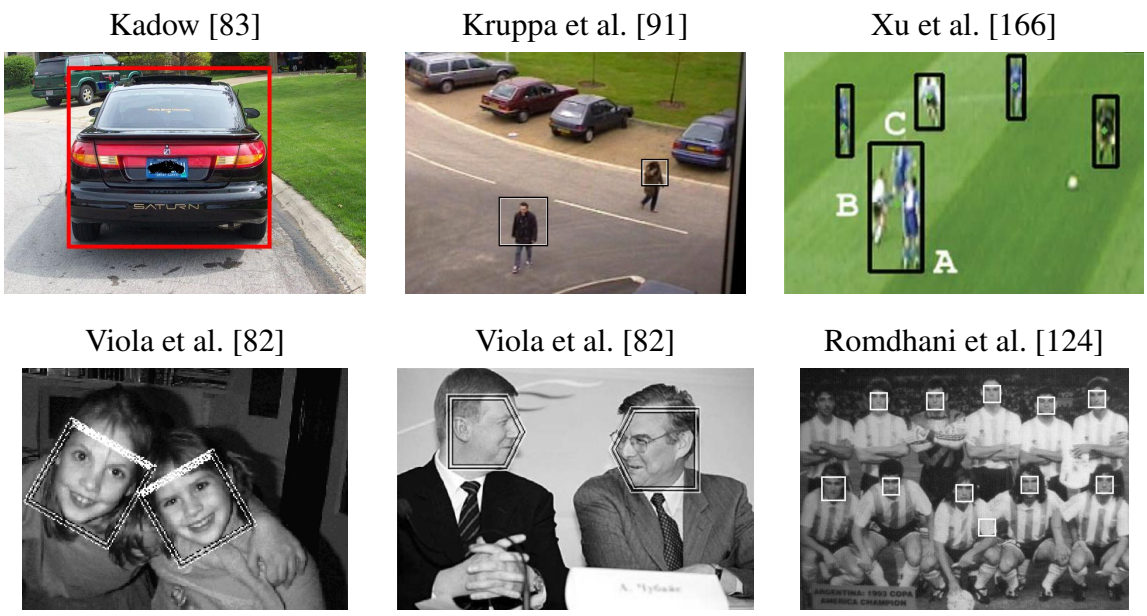


Figure 2.4: Several examples for detecting the rough location of real-world objects. Note that most of the illustrated approaches just determine the location and the size of the visual object, whereas Viola and Jones [82] also determine the in-plane rotation (bottom row, left) and the out-of-plane rotation (bottom row, middle).

This step is often implemented with machine learning techniques. Calculation rules to determine the vector of model parameters are learned from annotated example images. Initialization algorithms for deformable models usually determine the translation parameters, the scaling, and the rotation, compare to Cristinacce et al. [32] and Gu et al. [63]. Since the deformation parameters and texture parameters are not considered, they are set to a predefined value.

2.4 Objective Function

Model-based image interpretation needs to determine the model parameter values that fit best to the content of the image. This task essentially requires a measure for the fitness between the model and the image. The objective function $f(I, \boldsymbol{p})$ formalizes and encapsulates this challenge. It gives evidence about how well the model parameterization $\boldsymbol{p}$ fits to the image I. Depending on the context, the objective function is also known as the likelihood, similarity, energy, cost, goodness, or quality function. Depending on the definition, either its minimum or its maximum needs to be determined, because it exhibits the model parameters with the best fitness. In this

thesis, we consider the minimum to represents the best model fit, so the corresponding objective function suffices the definition of a cost function, not a similarity function. Note that this choice is arbitrary and does not affect the quality of our approach.

The precise analysis of the essential properties of a robust objective function represents a previously little attended part within the research field of model-based image interpretation. The robustness of the entire model fitting process depends on the accuracy of the objective function, because it is a fundamental component, on which all other steps are based. However, it is a nontrivial problem to find a particular function that solves this challenge accurately. Therefore, this thesis considers the objective function to be most important for the model fitting task and proposes a methodology to learn robust objective function from annotated training images.

Section 2.4.1 explains the approach of splitting the objective function into local parts, which reduces the complexity in creating these functions. In Chapter 4, we will illustrate the commonly conducted approach of manually specifying the calculation rules of these local objective functions, which is usually based on human intuition.

2.4.1 Local Objective Functions

In order to reduce the complexity, many researchers decompose the objective function $f(I, \boldsymbol{p})$ into several local objective functions $f_n(I, \boldsymbol{x})$. Among others, this approach is conducted by Cristinacce et al. [32], Cootes et al. [31], Romdhani et al. [123], Hanek [67], and Cohen et al. [22]. Each part corresponds to one feature point $\boldsymbol{c}_n(\boldsymbol{p})$ of the model, with $1 \leq n \leq N$ where N denotes the number of feature points. The result of the global objective function is the sum of the local function values, as in Equation 2.1.

$$f(I, \boldsymbol{p}) = \sum_{n=1}^{N} f_n(I, \boldsymbol{c}_n(\boldsymbol{p})) \tag{2.1}$$

These local functions evaluate the image content around the corresponding feature point and give evidence about the fitness between this feature point and the image data. The advantage of this partitioning is that designing local functions is more straightforward than designing the global function, because only the image variation in the vicinity of one feature point needs to be taken into consideration.

Referring to the literature mentioned above, local objective functions are widely used in current model fitting research. Chapter 4 will concentrate on local objective functions, and simply

refer to them as objective functions. The global objective function is always computed from them by applying Equation 2.1. We demonstrate how to obtain robust local objective functions via learning them from annotated example images. Therefore, our approach contributes to the approaches mentioned above.

The next section will describe that fitting a model to an image requires to determine the minimum of the objective functions. Thereby, the search on local objective functions $f_n(I, \boldsymbol{x})$ is conducted in pixel space $\boldsymbol{x} \in \mathbb{R}^2$ for each feature point. In contrast, the search on the global objective function $f(I, \boldsymbol{p})$ is conducted in parameter space $\boldsymbol{p} \in \mathbb{R}^P$. Here, $P = dim(\boldsymbol{p})$ denotes the dimensionality of the parameter space with $P \gg 2$.

2.5 Model Fitting

Model fitting searches for the model parameters that describe the content of the image best. Usually, the preceding initialization step provides this algorithm with a rough guess of the model parameters. Model fitting, in turn, refines these parameters in order to improve the fitness between the model and the image. This computational task is accomplished by searching for the model parameters that minimize the objective function. Model fitting algorithms are often executed iteratively.

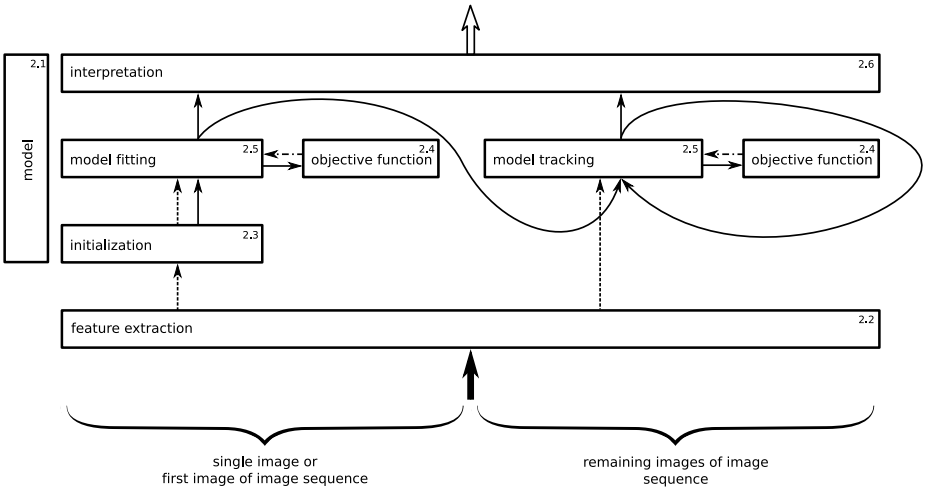


Figure 2.5: Model tracking extends the scheme of model-based image interpretation depicted in Figure 2.1 with the tracking phase. The most notable aspect of this extension is that specific information about the model and the image sequence is reused for processing further images.

Similar to Hanek et al. [68], we categorize model fitting techniques into two groups: parameter-based model fitting and projection-based model fitting, see Figure 2.6. These two categories substantially differ in the way the optimal model parameters are searched and the thereby utilized search space. Section 2.5.1 and Section 2.5.2 will describe these categories in detail.

Model-based image interpretation often requires the fitting of models to a sequence of images rather than to individual images, e.g. for estimating the temporal alteration of the object's position or constitution. Fitting algorithms would solve this challenge by fitting the model to each image individually. However, considering the information that is specific to an entire image sequence will greatly simplify and accelerate this task. Furthermore, the previously processed images allow an appropriate prediction of the model parameters for the current image. Therefore, this challenge is a specialization of model fitting and this thesis will refer to it as *model tracking* in order to distinguish both aspects. The simplest way to predict the model parameters is to consider the resulting parameters of the fitting step of the previous image appropriate for the current image. More sophisticated approaches integrate the Kalman-Filter [85] that is based on linear dynamical systems discretized in the time domain. Figure 2.5 shows how the tracking phase is integrated into the introduced scheme for model-based image interpretation. The meaning of the arrows is adopted from Figure 2.1.

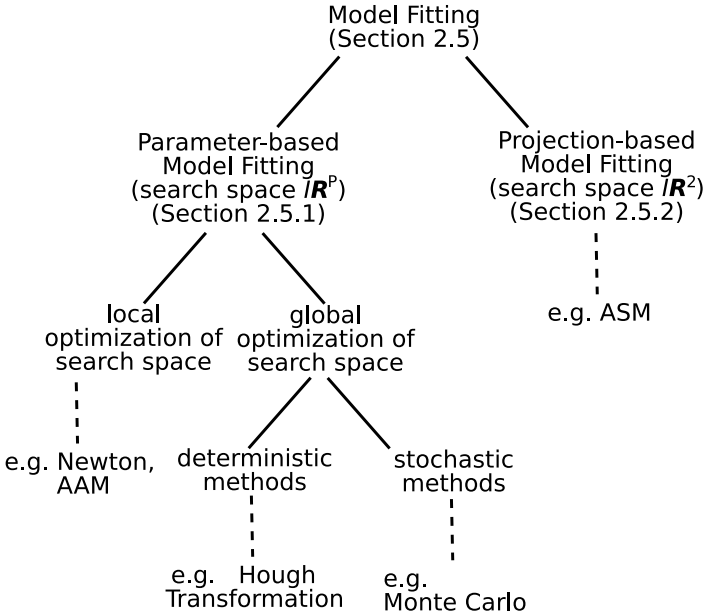


Figure 2.6: Our categorization of model fitting approaches follows and refines the scheme that has been proposed by Hanek et al. [68].

2.5.1 Parameter-based Model Fitting

Parameter-based model fitting directly alters the vector of model parameters $\boldsymbol{p}$ in order to find the model that describes the content of the image best. This challenge is equal to the mathematical issue of function minimization, for which a multitude of algorithms have been invented during the last 50 years. Unfortunately, exhaustive search is inapplicable to this issue, because the parameter space $\mathbb{R}^P$ is usually high-dimensional and real-valued. There thereby involved algorithms are roughly separated into two clusters, which differ in the aimed result of the computation. *Local optimization* aims at finding the local minimum by starting at a seed point, whereas *global optimization* aims at finding the global minimum within the entire parameter space.

The latter approaches are subdivided into deterministic and stochastic methods, see Hanek [67]. Deterministic methods require a discretization of the parameter space, such as dynamic programming [1] and other shortest path algorithms and Hough transform [72; 3]. The number of discretization levels represents a trade-off between the accuracy and the computational costs. Stochastic methods touch the parameter space in a random manner in order to find the desired model parameters. Well-known representatives are the Monte Carlo optimization, simulated annealing [11], or genetic algorithms. The former are also known as particle filters or condensation algorithm.

2.5.2 Projection-based Model Fitting

Projection-based model fitting conducts a three-step process in order to find the vector of model parameters $\boldsymbol{p}$ that describe the content of the image best. First, the projection function $\boldsymbol{c}_n(\boldsymbol{p})$ projects the model to the image plane, which usually results in a set of feature points, a contour, or a textured shape. Second, it optimizes the position of each component of the projection result separately, such as each contour point. This is achieved by moving it to a position that turns out to be the best position during a local search. The quality of each position is evaluated by the local objective function $f_n(I, \boldsymbol{x})$ of the corresponding feature point. In this case, the search space is two-dimensional for each feature point ($\mathbb{R}^2$) and therefore, exhaustive search is applicable within an appropriate vicinity of the feature point and a reasonable discretization. Note that the structure of the resulting feature points does usually not conform to the regulations of the model. Third, it approximates the model parameters from the relocated feature points in order to

re-establish the model's structural conditions. These three steps are usually executed iteratively in order to both keep the model's structural conditions and achieve an accurate fitness.

2.5.3 Discussion on Model Fitting Approaches

Since the parameter-based and the projection-based model fitting approach substantially differ in the computational procedure the obtained fitting results also differ with respect to their accuracy. It is often hypothesized that the parameter-based approach may give more accuracy, because it directly derives the model parameters from the content of the image. However, it is more difficult to set up an objective function whose properties specifically support the requirements of this approach. In contrast, projection-based model fitting gradually decreases the accuracy of the result during the execution of its three steps. Nevertheless, this approach is more straightforward and more practicable and is therefore applied by many researchers as well, such as Gu and Kanade [63], Cootes et al. [28], Ginneken et al. [55], and Stegmann et al. [143]. Our proof-of-concept that we describe in Section 2.7 also integrates this technique. In literature, it is often referred to as *approximation* or *optimization*. Section 4.4 formulates two properties that the involved objective functions should have in order to optimally support the procedure of projection-based model fitting.

Finally, we would like to emphasize that an optimal implementation of both approaches will deliver the same result, which is the perfectly fitted model. Note that parameter-based model fitting is able to apply global objective functions as well as their splitting into local parts as it is described in Section 2.4.1. However, projection-based model fitting approaches inherently require local objective functions.

2.6 Interpretation

As the final computational step of model-based image interpretation, the interpretation module calculates the interpretation result. It is provided with the parameters of the model that has been correctly fitted to the image in the course of the preceding computation. Since these parameters describe the specificities of the visible objects they represent an appropriate information cue. In contrast, directly interpreting the scene from the enormous amount of image data or low-level image features would be more difficult and result in very low accuracy. This challenge holds

particularly true for real-world image interpretation, where much more side conditions affect the image than in industrial scenarios or in a laboratory environment.

The interpretation module is usually implemented with machine learning techniques. Humans annotate example images or image sequences with the desired interpretation results and training algorithms derive calculation rules that are capable of inferring the interpretation result from the model parameters. Nevertheless, this module often considers raw image data and low-level features as well in order to further improve its prediction accuracy. Making the calculation of these additional features depend on the model parameterization as well will increase their correlation with the interpretation result and, in turn, improve the accuracy of interpretation.

Facial expression recognition serves as a good example for model-based image interpretation. Chapter 5 proposes our approach to interpret facial expressions. Furthermore, it will demonstrate the applicability of our proof-of-concept by interpreting further information.

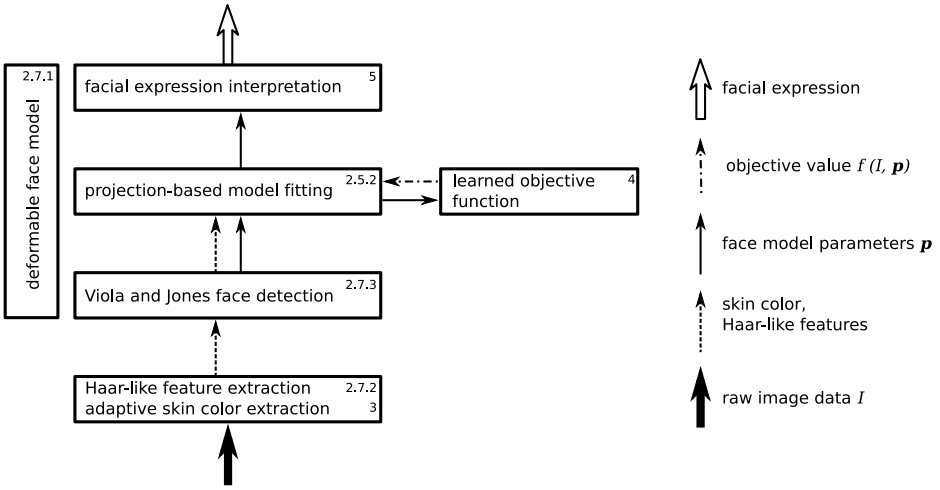


Figure 2.7: Our proof-of-concept that aims at facial expression recognition is based on model-based image interpretation. Referring to the scheme in Figure2.1, each involved component is instantiated by specific implementation.

2.7 Proof-of-concept for Facial Expression Interpretation

Facial expression recognition systems are widely designed as model-based image interpretation systems [107; 84; 94]. We utilize an implementation of this concept in order to explain the insights of this thesis. This section illustrates the modules of this *proof-of-concept*. Figure 2.7

denotes the sections and chapters that elaborate on each component. Our proof-of-concept makes use of a deformable face model, because of the various constitutions of a human face that arise from muscle activity. The model's parameters give evidence about the constitution of the face. This correlation greatly facilitates to infer facial expressions. Skin color is extracted from the image, because it represents an important information cue. We teach the objective function to consider skin color in order to determine the current fitness of the model. The Viola and Jones face locator reliably determines the position and the size of a face visible in an image.

From Section 2.7.1 to Section 2.7.3, we describe state-of-the-art components that are integrated into our proof-of-concept. Chapter 3, Chapter 4, and Chapter 5 describe our contributions to this challenge.

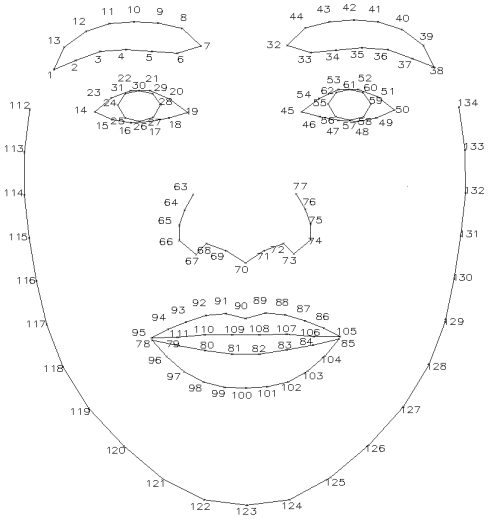


Figure 2.8: Our proof-of-concept utilizes the face model of Hansen [69] that consists of N=134 contour points. The function $\boldsymbol{c}_n(\boldsymbol{p})$ computes the location of the n^{th} contour point from the model parameters $\boldsymbol{p}$.

2.7.1 Point Distribution Model

Facial expression recognition requires a model of a human face that is aware of representing the different facial constitutions that arise from muscle activity. The challenge of modeling these non-rigid deformations has been greatly approached by statistically analyzing numerous example images that show different facial constitutions [12; 4; 62; 107].

For this challenge, Cootes et al. [28] introduce Point Distribution Models (PDM) that are created from N feature points whose location is manually landmarked in example images. The

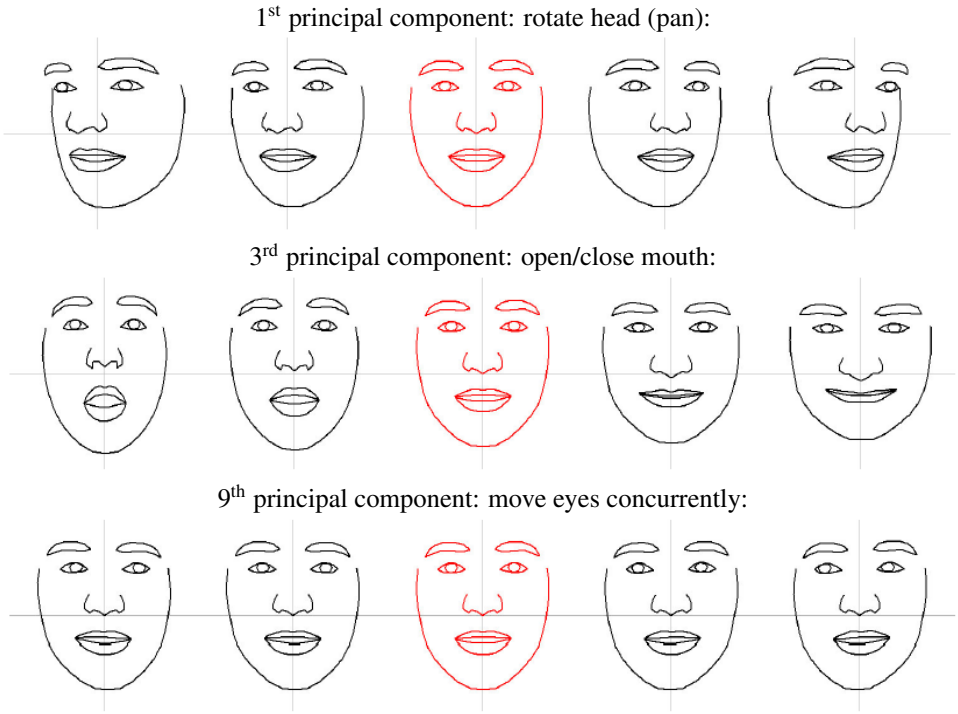


Figure 2.9: Some of the main deformations of the statistical face model [69]. Each row illustrates the deformation that arises from changing one specific principal component only. The changes are applied to $-2\sigma..2\sigma$. The deformations turn out to be highly semantic human motions as indicated below the images.

amount of correlation between the locations of two feature points is statistically described. Principal Component Analysis (PCA) figures out the main deformations of the entire contour. In order to visualize a model instance a vector of deformation parameters $\boldsymbol{b}$ indicates the amount of each deformation. Therefore, the model parameters $\boldsymbol{p} = (t_x, t_y, s, \theta, \boldsymbol{b})^T$ consist of the model's translation t_x and t_y, scaling s, rotation θ, and deformation vector $\boldsymbol{b}$. The corresponding projection function $\boldsymbol{c}(\boldsymbol{p}) = \{\boldsymbol{c}_1(\boldsymbol{p}), ..., \boldsymbol{c}_N(\boldsymbol{p})\}$ is assembled from the result of N subordinate functions $\boldsymbol{c}_n(\boldsymbol{p})$ and delivers a set of N feature points.

Our proof-of-concept uses the Point Distribution Model of a human face proposed by Hansen [69], see Figure 2.8. This face model consists of N=134 contour points, which are located at the contour lines between the facial components. Therefore, these points are partly connected with lines for visualizing the contour of the face. We will refer to them as contour points, in the following. Figure 2.9 illustrates the main deformations that PCA determined from the training images. This thesis will use this face model in order to explain its contributions.

2.7.2 Haar-like Features

Haar-like image features indicate both smooth and abrupt transitions between differently colored regions. There are different styles of Haar-like features and each of them is able to detect a certain type of transition between two differently colored regions. Figure 2.10 illustrates the basic set of features as it is used by Lienhart et al. [99] for object detection. These features are calculated at a particular location with a particular size from an image. Haar-like features have proven to be excellent features for interpreting real-world images, because they are quickly computed from the image data and they are robust towards noise [154; 155; 82; 99]. Their name relates to the similarity to the basis functions of the Haar wavelet [64]. Within our proof-of-concept, they are utilized by the object detection algorithm of Viola and Jones [154] that will be explained in Section 2.7.3. Furthermore, our novel methodology for fitting models to images makes use of a previously learned set of Haar-like features, see Chapter 4.

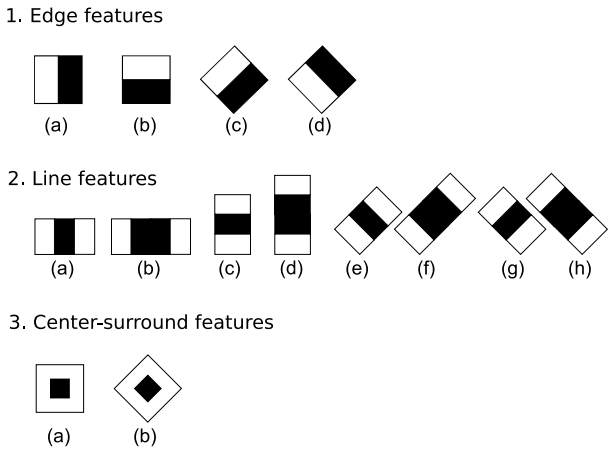


Figure 2.10: The basic set of Haar-like image features as it has been used by Lienhart et al. [99]. These features are robust towards noise and they are calculated quickly using the integral image representation.

Haar-like features consist of several adjacent black and white rectangular regions. Their value is calculated by subtracting the sum of the pixel intensities within the black regions from the sum of the pixel intensities within the white regions. The features depicted in the first row of Figure 2.10 are capable of detecting a transition between two regions with different color. Applying the feature to the entire image indicates this transition with an extreme value. The different features are able to detect differently shaped transitions, such as horizontal, vertical, or

diagonal transitions. Figure 2.12 in Section 2.7.3 depicts several Haar-like features at different locations and sizes within a human face that are of an extreme value.

Haar-like features are rapidly calculated from the integral image $\mathcal{I}$ of an image I. Every pixel of the integral image $\mathcal{I}(x, y)$ is defined to contain the sum of intensities of all pixels $I(x', y')$ in the image located within the rectangular region between the origin of the image $I(1, 1)$ and $I(x, y)$, see Equation 2.2.

$$\mathcal{I}(x, y) = \sum_{1 \leq x' \leq x, 1 \leq y' \leq y} I(x', y') \tag{2.2}$$

$$\sum_{x_1 \leq x \leq x_2, y_1 \leq y \leq y_2} I(x, y) = \mathcal{I}(x_2, y_2) - \mathcal{I}(x_1, y_2) - \mathcal{I}(x_2, y_1) + \mathcal{I}(x_1, y_1) \tag{2.3}$$

Any Haar-like feature requires summing up the pixel intensities within a certain rectangular regions. These sums are calculated in constant time using a small number of basic arithmetic operations. Equation 2.3 describes how to calculate the sum of all pixel intensities within one rectangle. Note that the integral image has to be computed only once from the camera image.

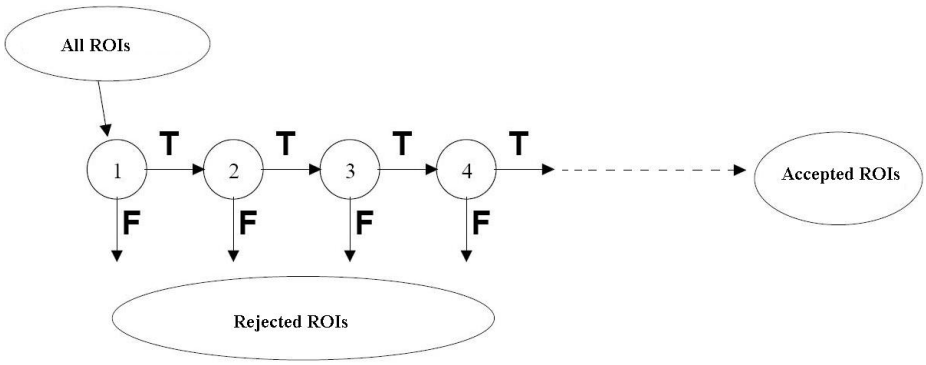


Figure 2.11: ROIs that are considered to contain the object must be accepted by the entire cascade of simple classifiers, see Viola et al. [82].

2.7.3 Object Detection by Viola and Jones

Viola and Jones [154] introduce a visual object detection framework that processes images very quickly while achieving high detection rates. It determines the location and the size of an object visible in the image and propagates this information in terms of a rectangular bounding box. Many model-based image interpretation systems integrate this algorithms for initialization and

they directly derive the model's position parameters from its result, see Cristinacce et al. [33; 32], Yunus et al. [127], Song et al. [138], Huang et al. [75], Wang et al. [156].

This approach determines whether or not a rectangular region of interest (ROI) contains the expected object by evaluating a cascade of previously determined features within this ROI. This statement is evaluated for ROIs at any location and any size within the image. Thereby, the object detector makes use of Haar-like features, which are explained in Section 2.7.2, because they are descriptive on the object's appearance and they are computed very quickly.

An extensive training phase selects a small set of Haar-like features that are relevant for accepting or rejecting the ROI. Despite of its simplicity and its high performance, this classifier is weak and it cannot determine the object very well. Therefore, Viola and Jones combine these simple classifiers in a cascade. Boosting algorithms combine many weak classifiers in order to build a strong classifier, see Figure 2.11. Once a ROI has been discarded no further processing takes place, otherwise the next weak classifier of the cascade is applied. The ROI is considered to contain the object if it passes the entire cascade successfully. Viola and Jones apply the AdaBoost algorithm [52] for learning the cascade.

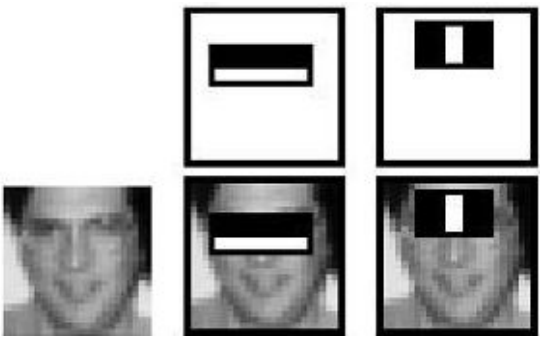


Figure 2.12: The predominantly used Haar-like features for detecting human faces are located at the eyes and the nose, see Viola et al. [154].

Viola and Jones demonstrate the benefit of their approach in the domain of face detection. Figure 2.12 illustrates the use of Haar-like features for determining whether a certain ROI contains a human face or not. This figure depicts the two highest weighted weak classifiers resulting from applying the AdaBoost algorithm on numerous training images. The first feature indicates the horizontal region containing the eyes, the nose, and the cheeks. The upper part of this region is usually darker than the lower part containing specularities on the cheeks. The second Haar-like feature measures the difference in intensity between the eyes and the bridge of the nose, which is also intended to deliver an extreme value in case the search window contains a face.

The trained frontal face detector runs at 15 frames per second on a standard desktop computer, which makes it suitable for real-time applications.

Our proof-of-concept uses this implementation for obtaining a ROI around a face within the processed image. Furthermore, the adaptive skin color classifier determines the image-specific characteristics via this face locator, see Chapter 3.

2.8 Related Work on Model-based Image Interpretation

Model-based image interpretation has proven to greatly support the interpretation of real-world images. Examples are facial expression recognition [22; 149; 47], hand gesture recognition [71; 140; 112; 88; 79; 37], human posture recognition [16; 121; 122], as well as interpreting traffic scenes [51]. Furthermore, medical applications use this approach to enable an automatic interpretation of image data, such as computer tomography or magnetic resonance volume scans as well as X-ray and ultrasound images, see for example Stegmann et al. [144; 145], Cootes et al. [30; 29], and Ginneken et al. [56].

A generally applied technique in model fitting is specifying the feature points in a reference image and establishing correspondences to the same feature points in further images. These approaches usually consider rigid and artificial items and make use of three-dimensional models [132; 7]. In opposite to the faces of different persons, the texture in the vicinity of the feature points does not change substantially. Therefore, the specification of only one reference image is sufficient. A scalar-valued function is applied to the current image and its local extrema are considered to represent the locations of the searched feature points. A common example for these functions is the *Laplacian Of Gaussians* filter (LoG), which highlights areas of rapid intensity change and is therefore predestined for edge detection and corner detection [151]. The challenge is to establish the correspondences between the obtained extrema and the model's feature points. Novel methodologies that base on machine learning techniques recently achieved promising results [95; 15]. Similar to the approach that we propose in Chapter 4, a likelihood function is learned from the content of the image around the feature point. In contrast to our approach, these algorithms do not intend to find the best position within the vicinity of a feature point, but they intend to find the best correspondence between the feature points and a set of extracted points.

Gu et al. [63] search the individual points of their three-dimensional model by matching small patterns to the image. The objective function represents the error between the pattern

and the underlying image content. Downtown et al. [40] extract line segments from the camera image in order to fit a human upper body model that consists of connected cylinders. They design their objective function such that the rectangular projection of the model's cylinders best match the lines extracted from the image. Their intuitively designed calculation rules consider the length and the orientation of the lines in the image and compare this information to the projected rectangles.

Cristinacce et al. [33] use the Viola and Jones object detector in order to determine the translation, scale, and rotation of their deformable texture model for human faces. Michalowski et al. [110] integrate the face detector as part of their multimodal person tracking system that focuses on classifying the attention of humans. Mählisch et al. [105] train this object detection scheme themselves in order to localize pedestrians on infrared images. Viola et al. [155] also apply their object detector to outdoor scenes in order to locate pedestrians. Their extended version considers both information about motion and texture.

As described in Section 2.3, the initialization phase determines the model's position parameters approximately correct and expects the subsequent fitting algorithm to deform it correctly as well. Unfortunately, fitting algorithms tend to get stuck in local minima of the objective function if the correct model parameters differ too much from this parameterization. In order to improve the rough estimation of the initialization step, Li et al. [98] propose to approximately deform the shape during this initialization phaseas well. Thereby, they compute a rough guess for the opening of the mouth. Haar-like image features represent the input data and linear regression delivers the deformation rules. This additional estimation of deformation parameters improves the accuracy of the initial guess of the model parameters. Still, the result of this technique has to be improved by a succeeding fitting algorithm. Their evaluation proves that the subsequent model fitting step works more robustly using this approach.

2.9 Summary on Model-based Image Interpretation

Correctly understanding the content of images and the content of further sensor data will be essential for leveraging intelligent devices in future times. Model-based techniques make image interpretation feasible for various applications in real-world scenarios. This thesis elaborates on fitting a deformable face model to images in order to recognize facial expressions. However, the techniques involved will facilitate interpretation tasks for further applications as well.

The parameters of an accurately fitted model describe the constitution of a visible object, which simplifies the subsequent interpretation task. Since it is a nontrivial challenge to accurately fit a model to an image, this process is generally subdivided into several computationally independent steps. Each component acts on particular assumptions and computes a specific part that is necessary to the entire interpretation task. As we describe in this chapter, researchers have developed a plenitude of sophisticated techniques for each component during the last decade. Because of the tight interconnection between the individual components, the weakest link of the interpretation process limits the accuracy of the entire system. The inaccuracies even sum up during the course of the computation. These aspects represent the Achilles heel of model-based image interpretation techniques and therefore, this scheme is not widely applied to current interpretation systems.

Nevertheless, research on the individual components of model-based image interpretation has recently made great progress during the last few years, most notably the quick and accurate object locator proposed by Viola and Jones [154] and the statistics-based deformable models for shape and texture proposed by Cootes et al. [31]. These achievements make model-based techniques viable for challenging applications, such as the interpretation of facial expressions, the recognition of a human body's posture, and gesture recognition.

Various fields of research and engineering consider a component termed *model* and integrate it as an inevitable module of their task. As explained in this chapter, geometric models facilitate the interpretation of objects in images for computer vision applications. For designing the structure of databases, entity-relationship-models describe the organization of the storage space and the relationship between the stored data atoms. In software engineering, the Unified Modeling Language (UML) facilitates the construction of a large software project by denoting the involved entities, their functionality, and their mutual interaction. In regression analysis, a model describes an enormous amount of discrete data in a semantic way.

As the common ground of these different kinds of models, they aim at reducing the numerous degrees of freedom of the user's or the program's task. This is achieved by integrating predefined or previously known information. Computer vision models reduce the amount of possible interpretation results to the ones with a predefined shape or texture. This knowledge restricts the search space, e.g. the nose is always located between the eyes and the mouth. The search in databases has to be conducted in a specific scheme defined by the model, which facilitates and accelerates its execution. The components of a software projects only communicate in the way defined by the model. These regulations prevent from unexpected side effects, which

accelerates and simplifies the designer's task. Regression analysis reduces an enormous amount of data to a small set of highly descriptive model parameters. This point-of-view is most related to model-base image interpretation. Note that a reasonable design of these models is crucial for obtaining precise results.

Chapter 3

Adaptive Skin Color Extraction

In real-world scenes, color is an important information cue that makes objects distinctive from their surroundings. Therefore, the feature extraction module of model-based image interpretation systems whose use has been describes in Section 2.2 is often capable of extracting color features from the image. For the benefit of face model fitting applications the features skin color, lip color, tooth color, and hair color describe the location and the geometric shape of human faces and their parts well. Figure 3.1 illustrates that the skin color region clearly borders the eyes, the lips, the eyebrows, the hair, and the background. Skin color classifiers are the computational modules that determine for every pixel whether it is skin-colored or not and assemble this information to the skin color image, which is depicted in the lower row of Figure 3.1.

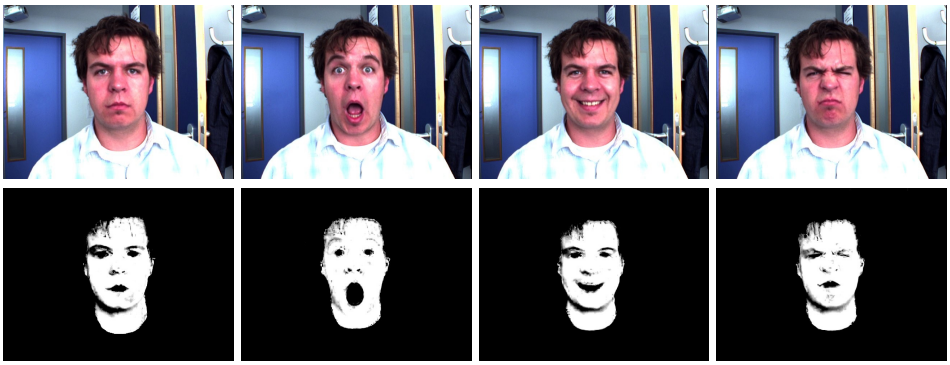


Figure 3.1: Considering skin color facilitates fitting a contour model, because the color transition of the skin color image (lower row) indicates the contour lines very well.

Nevertheless, extracting this information from real-world images robustly represents a challenging task. The reason is that specific characteristics that are related to the context conditions,

the image, and the visible person vary skin color significantly, such as illumination conditions, camera type, camera settings, the person's tan, and the person's ethnic group. Since skin color looks differently throughout different images, it occupies a large cluster within color space. Because of its size, this cluster also contains the color of several other objects. In contrast, these image characteristics are fixed considering one single image only. Therefore, all skin color pixels look similarly and skin color occupies a much smaller and more compact cluster, which facilitates skin color classification.

The proposed approach exploits this coincidence by a two-phase approach. First, it determines the characteristics of the image and adapts a general purpose color classifier accordingly. Second, it extracts the skin color pixels with this adjusted classifier, which delivers highly accurate results. The contributions of this approach to color classification are as follows:

1. It determines the image-specific and person-specific characteristics automatically.
2. It adapts general purpose skin color classifiers to images using these characteristics.
3. Its high accuracy and its high speed turn it appropriate for real-world applications.

In the remainder of this chapter, we proceed as follows. Section 3.1 gives a comprehensive overview and categorization of related work in the area of skin color classification and elaborates on the advantages and shortcomings of either technique. Section 3.2 explains the entire procedure and its components of the proposed approach. Section 3.3 shows the purpose and the generation of the skin color mask. Section 3.4 formulates the image-specific and person-specific characteristics that are responsible for the variations of skin color. Furthermore, this section explains how to determine this information automatically. Section 3.5 introduces three general purpose skin color classification techniques and adapts them via the image-specific and person-specific characteristics. Section 3.6 experimentally evaluates the achievements of the proposed approach.

3.1 Overview of Skin Color Classification

Skin color represents an important source of information to various computer vision applications and therefore, a lot of research is conducted in this area. Vezhnevets et al. [152] give a comprehensive overview of recent work within that area describing common color spaces and categorize the detection techniques.

3.1.1 The Normalized RGB Color Space

It is commonly agreed that the RGB color space is not appropriate to describe skin color [152; 18; 169; 139; 115; 167]. In this representation, skin color occupies a large and incompact cluster, which is likely to overlap the color of other objects. A common way to cope with this shortcoming is to switch over to the normalized RGB color space (NRGB), which uses the proportional rate of each component of RGB, see Equation 3.1. In most literature, such as in the references mentioned above, b is omitted, because it can be calculated from the other components. In this thesis, the color vector $\boldsymbol{c_x}=(r, g, base)^T$ denotes the color of a pixel $\boldsymbol{x}$ in the NRGB color space.

$$\begin{aligned} base &= R+G+B \\ r &= \frac{R}{base} \\ g &= \frac{G}{base} \\ b &= \frac{B}{base} \end{aligned} \tag{3.1}$$

3.1.2 Categorization

Skin color classification techniques differ by their accuracy and their runtime performance. This is directly related to their memory consumption. Vezhnevets et al. categorize state-of-the-art techniques as *Non-parametric skin color distribution modeling* (NSDM), *Parametric skin color distribution modeling* (PSDM), and *Explicit definition of the skin color cluster* (EDSC).

NSDM specifies for every color in color space individually, whether or not it represents human skin. This technique is often referred to as Skin Probability Map (SPM), which assigns a probability value to each point of a discretized color space. These algorithms perform at very high speed, but they require an enormous amount of memory space for storing the map that contains the color-to-class decisions. Using a three-dimensional color space such as RGB, HSV, or NRGB and discretizing each dimension to 256 entries, the required memory space amounts to $256^3 \approx 1.7 \cdot 10^7$ bits. These entries represent the color-to-class decision rules. They are usually set up beforehand by learning them from comprehensive training data. Their accuracy relies on this training set.

In order to reduce memory requirements common approaches subdivide the three-dimensional color space into larger clusters or consider a two-dimensional color space. Unfortunately, both solutions badly affect runtime performance and accuracy [17; 58].

PSDM expects skin color to be located within a cluster within color space, which has a specific shape, such as an ellipsoid or a cuboid. The shape and the location of this cluster are defined by a set of parameters. The required memory is limited to these parameters. Execution time rises, because the extent of the cluster has to be computed before classification. Furthermore, the accuracy decreases, because the true color distribution in the multi-dimensional color space is not modeled exactly, but approximated by the shape of the cluster. The existence of an easy and highly descriptive shape for the skin color cluster highly depends on the chosen color space. Common approaches model skin color distribution via a single Gaussian or a mixture of Gaussians [81; 74].

EDSC uses a set of rules that explicitly define the shape of the skin color cluster. Memory requirements are limited by the extent of the chosen rules. The challenge is to find adequate decision rules. Most often, this task is accomplished by rule induction algorithms that learn the rules from an annotated training set. Accurate rules often rely on features that are well associated with skin color. Gomez et al. [59] propose a rule induction mechanism that creates new features by mathematically combining the features of the three-dimensional color space RGB.

	runtime	precision	memory requirements	depending on color space	size of training set	presumptions to specify manually	further applicability
NSDM	+++	+++	– – – [1]	–	very large	none	none
PSDM	+ [2]	+	+++ [2]	+++	moderate	shape of color cluster	adapt shape of cluster
EDSC	+ [3]	++	+++ [3]	++	moderate	none	none

Table 3.1: Comparison of state-of-the-art color classification schemes.

Referring to this categorization, we set up Table 3.1 that compares the most important features of either technique. Assuming a huge and representative training set NSDM is the most accurate approach, because it is suitable for any distribution of skin color and does not depend a lot on the chosen color space. PSDM assumes the skin color cluster to have a certain shape, which does not represent the correct color distribution due to simplifications. EDSC approx-

[1] e.g. $1.6 \cdot 10^7$ bits for entire RGB
[2] depends on shape of cluster
[3] depends on accuracy of rules

imates the color distribution more correctly, because the rule induction algorithm learns the skin color cluster from training data. Nevertheless, it is possible to convert both PSDM and EDSC into NSDM in order to adopt its runtime performance. Hereby, each entry of NSDM's color space is considered separately and assigned the result that the evaluation with PSDM or EDSC delivers. Note that this conversion does not improve the accuracy of skin color detection achieved by PSDM or EDSC.

Vezhnevets et al. state a further technology, which does not allow for a direct comparison, because its classification process adds dependency to further features. They call this adaptive approach, *dynamic skin color distribution modeling* (DSDM) and it extends PSDM and EDSC. These classifiers additionally consider context conditions of the processed image rather than on the pixel's color only, such as camera settings, illumination, and the characteristics of the visible person. In consequence, the shape of PSDM or the rules of EDSC are adapted to the processed image, which improves the skin detection accuracy. Soriano et al. [139] define their skin color cluster within the chromatic color space. Its shape looks like the crescent of the moon and they call it *skin locus*. The shape's geometry is camera-specific, but they do not provide a mechanism that automatically determines the skin locus. Furthermore, the skin locus does not take person-specific characteristics into account.

Our approach described by this chapter contributes to the category DSDM. It demonstrates the specialization of classifiers both of type PSDM and EDSC to the image. We show that the achieved classification accuracy is superior to the same approaches without adaptation.

3.2 Overview of Our Approach

Skin color classifiers specify a skin color cluster within a particular skin color space in order to determine whether a pixel is skin-colored or not. If they do not consider image-specific characteristics this cluster is very large, which reduces the classification accuracy. A more accurate detection of skin color requires taking these characteristics into account by adapting the skin color cluster accordingly. This approach is capable of distinguishing skin color from very similar color, such as lip color. Humans often adapt skin color classifiers manually. They repeatedly specify the parameters of the skin color classifier until they find the achieved classification result satisfactory. Nevertheless, most face interpretation systems must run without manual interference.

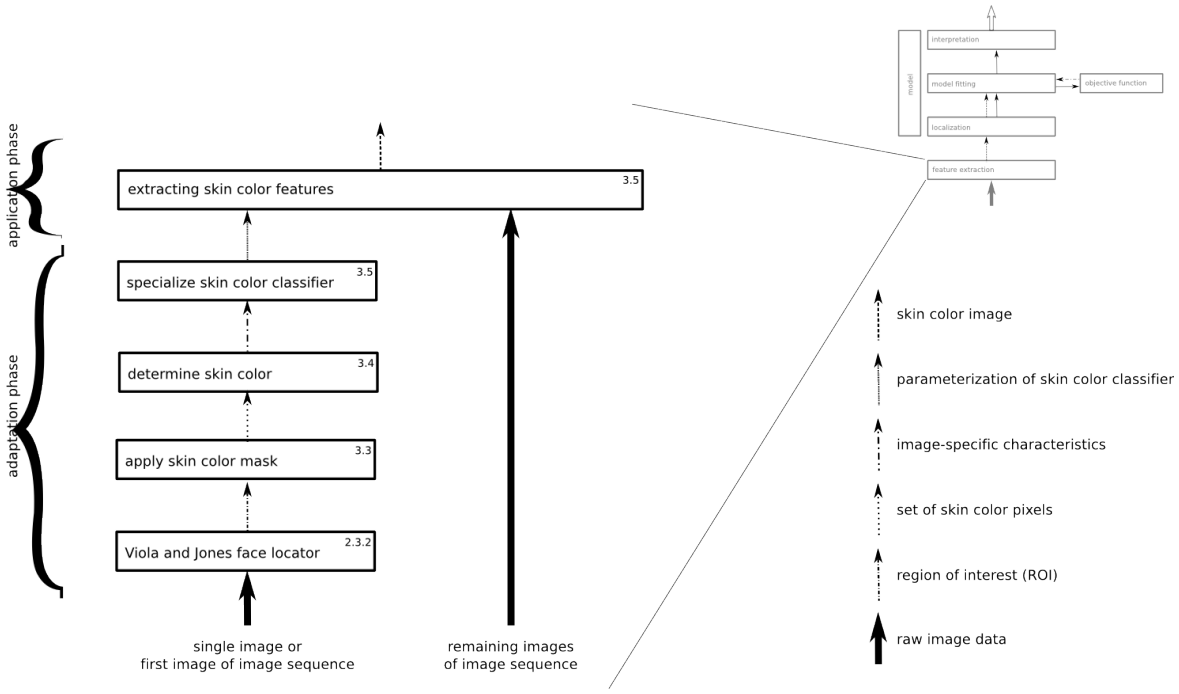


Figure 3.2: Our approach contributes to the feature extraction phase of model-based image interpretation, see Figure 2.1. Its adaptation phase conducts the specialization of the classifier and the application phase determines the skin color pixels. The upper right corners of the components denote the section that elaborates on it. Note that the result of the face detection component may be reused later during the initialization step of model-based image interpretation.

The proposed approach automatically obtains image-specific characteristics and adapts a skin color classifier accordingly. It consists of two phases that are illustrated in Figure 3.2: the adaptation phase and the application phase. Note that model-based image interpretation techniques will apply the proposed color classifier during their feature extraction. Therefore, Figure 3.2 refers to Figure 2.1 in order to emphasize this relation.

The *adaptation phase* determines image-specific characteristics and adapts a skin color classifier via the following four steps: First, it computes the rough location and size of the visible face with a high-level face locator. This information is represented as a rectangle, which we will call region of interest (ROI). Our proof-of-concept integrates the commonly used face locator of Viola and Jones that is described in Section 2.7.3. Nevertheless, any other approach that is capable of obtaining such a ROI can be used as well. Second, a previously learned mask that is tailored to the face locator extracts a small number of pixels from the ROI. Since the mask is learned such that these pixels are skin-colored, we will call it *skin color mask*. Third, our

approach computes descriptive values from these skin color pixels. These values describe skin color as it is visible in the image and therefore, we will call them *image-specific characteristics*. Fourth, a skin color classifier is adapted to the image via these characteristics.

The *application phase* computes the skin color image by applying the previously adjusted skin color classifier to the image.

Note that the adaptation phase must only be executed for the first image of an image sequence, as long as the image conditions do not change drastically. Therefore, the runtime performance only depends on the application phase that therefore demands for a very simple and quick algorithm. Since our approach is independent of the utilized skin color classifier, it satisfies this demand.

3.3 The Skin Color Mask

The skin color mask is a two-dimensional matrix that specifies the probability of skin color for each pixel within the ROI. It is learned from training images that are annotated with the skin-colored regions. For each of these images, the ROI is determined by a face locator. Runtime performance is increased by only considering the mask's probability entries that exceed a given threshold value. Figure 3.3 illustrates this training procedure in detail. As the construction of the skin color mask involves a face detector, it is specific to this face detector.

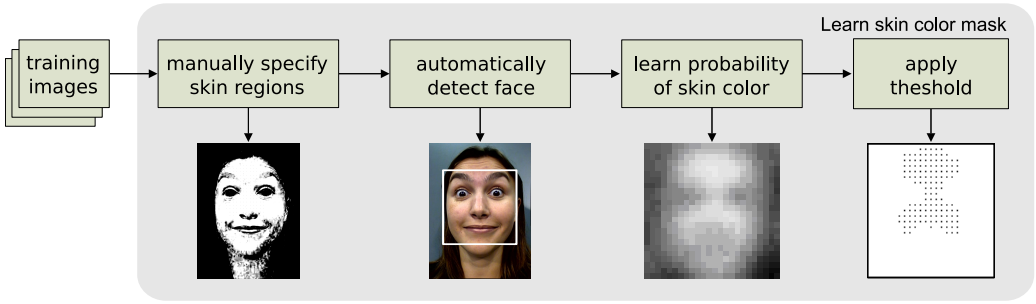


Figure 3.3: Learning the skin color mask is specific to a particular face detector.

In our proof-of-concept, we gather a set of K training images that originate from various web pages and the TV and we manually specify the skin-colored regions. The images show different illumination conditions, arbitrary background, and the visible persons have different age and belong to different ethnic groups. The skin color mask M is an $n_1 \times n_2$ matrix with the entries $m_{i,j} \in [0..1]$. In our proof-of-concept, we take $n_1 = n_2 = 24$ as a reasonable compro-

mise between accuracy and runtime performance. We apply the face detector to each image and obtain a region of interest roi_k, which is divided into $n_1 \times n_2$ cells $f_{k,i,j}$ with $1 \leq i \leq n_1$ and $1 \leq j \leq n_2$. The likelihood for skin color within the cell $f_{k,i,j}$ is expressed by the value $s_{k,i,j}$, see Equation 3.2. We calculate the value of each entry $m_{i,j}$ by Equation 3.3 as the mean of all $s_{k,i,j}$.

$$s_{k,i,j} = \frac{\textit{number of skin pixels in } f_{k,i,j}}{\textit{total number of pixels in } f_{k,i,j}} \tag{3.2}$$

$$m_{i,j} = \frac{1}{K} \sum_{k=0}^{K-1} s_{k,i,j} \tag{3.3}$$

3.4 The Image-specific Characteristics

This thesis utilizes the NRGB color space in order to formulate image-specific and person-specific characteristics. They give evidence about the appearance of skin color within a particular image and they are represented by the Gaussian distribution of the skin color visible in the image, i.e. the mean $\bar{\boldsymbol{\mu}}$ and the covariance matrix $\bar{S}$. Equation 3.4 and Equation 3.5 depict their calculation from the set $\mathcal{P}$ that contains all skin color pixels within the image. Note that $\mathcal{P}$ must be manually specified.

$$\bar{\boldsymbol{\mu}} = \frac{1}{|\mathcal{P}|} \sum_{\boldsymbol{x} \in \mathcal{P}} \boldsymbol{c_x} = \begin{pmatrix} \bar{\mu}_r \\ \bar{\mu}_g \\ \bar{\mu}_{base} \end{pmatrix} \tag{3.4}$$

$$\bar{S} = \frac{1}{|\mathcal{P}| - 1} \sum_{\boldsymbol{x} \in \mathcal{P}} (\boldsymbol{c_x} - \bar{\boldsymbol{\mu}})(\boldsymbol{c_x} - \bar{\boldsymbol{\mu}})^T = \begin{pmatrix} var_r & cov_{r,g} & cov_{r,base} \\ cov_{g,r} & var_g & cov_{g,base} \\ cov_{base,r} & cov_{base,g} & var_{base} \end{pmatrix} \tag{3.5}$$

Unfortunately, we cannot compute the image-specific characteristics for previously unseen images, because the set of skin-colored pixels $\mathcal{P}$ is unknown. The next section illustrates how our approach approximates these values automatically.

3.4.1 Automatically Determining the Image-specific Characteristics

Previous approaches detect image-specific characteristics via low-level techniques, such as color segmentation, background subtraction, or histogram prediction [136]. However, their results are not accurate enough to be used for further computation. Therefore, we propose to determine the image-specific characteristics of previously unseen images via two steps. First, we roughly locate the face visible in the image and obtain a rectangular region, which we will call region of interest (ROI). In our proof-of-concept, this task is accomplished by the face detector proposed by Viola and Jones [154]. Note that any other face detector can be used as well, such as [125; 168; 128; 97; 89; 124].

Second, we apply a skin color mask to the ROI, which extracts a moderate number of pixels. This mask must be made specific to the face locator, because the determined ROI is tailored to this algorithm. The extracted pixels represent skin color with a high probability. Then, a set $\mathcal{P}'$ is assembled from the extracted pixels and the image-specific characteristics are computed from this information via Equation 3.4 and Equation 3.5. Note that the set $\mathcal{P}'$ does not contain the same pixels that $\mathcal{P}$ would contain and therefore, this procedure does not provide the correct values of the parameters $\bar{\boldsymbol{\mu}}$ and $\bar{S}$. However, it serves as a good approximation as the evaluation in Section 3.6 proves.

Note that the color distribution of an image does not influence the accuracy of our approach, because the face detector takes gray value images. See Section 3.6.1 for an evaluation of the accuracy.

3.5 Adjusting Skin Color Classifiers

The calculation rules of pixel-based skin color classifiers figure out if a pixel is skin-colored by considering its color features only. The cluster within color space that they specify to contain all skin-colored pixels is usually fixed, see the categorization of skin color classification techniques in Section 3.1.2. However, the category of dynamic skin color classifiers considers further features apart from the pixel's color. The evidence of these additional features will affect the skin color cluster's position, size, and shape, because these features usually describe characteristics of the entire image.

The following sections show three examples of dynamic skin color classifiers. They vary their calculation rules depending on the image-specific characteristics. Section 3.6.2 compares these classifiers and evaluates their accuracy.

3.5.1 Cuboid-based Skin Color Classifier

This classifier specifies a cuboidal cluster within NRGB color space that is aligned to the axes of the color space. It treats any color within this cuboid to be skin-colored. The cuboid is described by a lower and an upper bound for each dimension of the color space: l_r, l_g, l_{base}, u_r, u_g, and u_{base}. Equation 3.6 shows the calculation rule of this classifier.

$$\textit{skin-colored} \quad \Leftarrow \quad (l_r \leq r \leq u_r) \quad \wedge \quad (l_g \leq g \leq u_g) \quad \wedge \quad (l_{base} \leq base \leq u_{base}) \tag{3.6}$$

Adaptation to the Image-specific Characteristics:

This classifier is adapted to an image by deriving its bounds from the parameters of the image-specific characteristics $\bar{\boldsymbol{\mu}}$ and $\bar{S}$. As described in Equation 3.7, our approach specifies the distance between the statistical mean and the lower and upper bounds for either dimension to be two times the standard deviation according to common strategies. Note that the herein utilized standard deviation $\sigma_i = \sqrt{var_i}$ is extracted from the covariance matrix $\bar{S}$ in Equation 3.5.

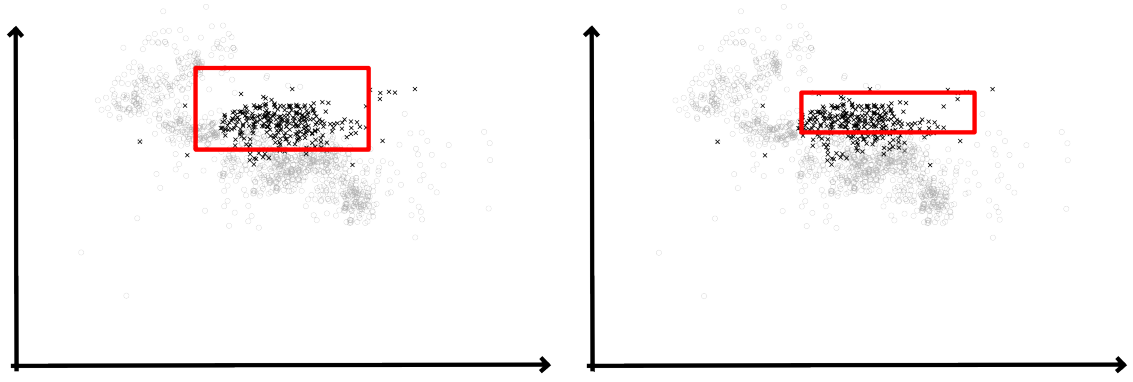


Figure 3.4: The projection to the rg-plane illustrates the accuracy of the cluster that the cuboid-based classifier considers to be skin color (red rectangle). Left: without adaptation, Right: with adaptation to the current image. The entries denote the manually annotated pixels with skin color (black crosses) and non-skin color (gray circles).

$$\begin{aligned} l_r &= \bar{\mu}_r - 2\sigma_r \\ l_g &= \bar{\mu}_g - 2\sigma_g \\ l_{base} &= \bar{\mu}_{base} - 2\sigma_{base} \\ u_r &= \bar{\mu}_r + 2\sigma_r \\ u_g &= \bar{\mu}_g + 2\sigma_g \\ u_{base} &= \bar{\mu}_{base} + 2\sigma_{base} \end{aligned} \tag{3.7}$$

Figure 3.4 visualizes the appropriateness of this classifier to the challenge of skin color extraction. It compares the fixed cuboid cluster to the one that is adapted to the content of a particular image. Note that the yellow points within the skin color cluster denote the false positives of the classification and the red points outside of the skin color cluster denote the false negatives, respectively.

3.5.2 Ellipsoid-based Skin Color Classifier

This classifier specifies an ellipsoidal cluster within NRGB color space. Any color inside of this ellipsoid is treated to be skin-colored. The ellipsoid is calculated such that the Mahalanobis distance [104] from the center of the ellipsoid μ to any location $\boldsymbol{c_x}$ within the cluster is less than a given threshold value t. The location and the size of this cluster is not fixed, but described by the parameters $\boldsymbol{\mu}$, S, and t. The Equation 3.8 denotes the calculation rule of this classifier.

$$\textit{skin-colored} \quad \Leftarrow \quad (\boldsymbol{c_x} - \boldsymbol{\mu})^T S^{-1} (\boldsymbol{c_x} - \boldsymbol{\mu}) \leq t \tag{3.8}$$

Adaptation to the Image-specific Characteristics:

The classifier is adapted to an image by taking its parameters to be the image-specific characteristics, see Equation 3.9. Empirical results show that reasonable values for the threshold vary between $6 \leq t \leq 25$. This value depends on the color of further objects visible in the image. If their color is distinct from skin color then t is chosen big, otherwise t is chosen small. The evaluation in Section 3.6.2 is performed with $t = 9.8$.

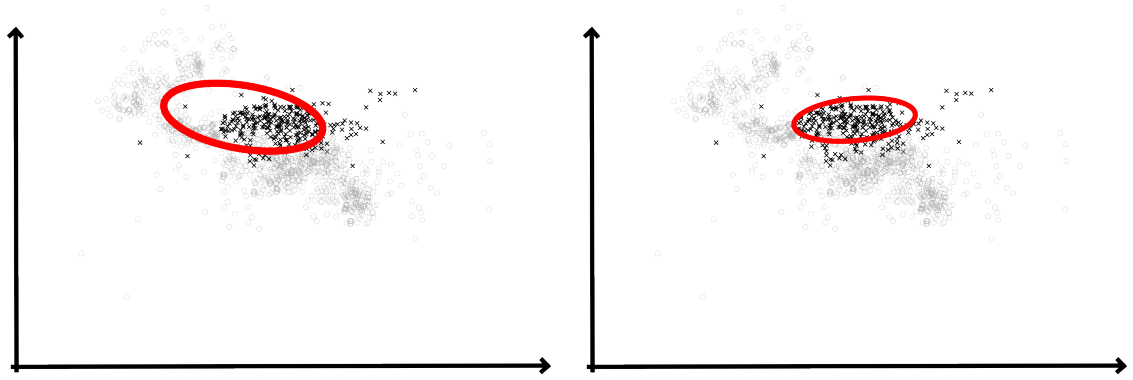


Figure 3.5: The projection to the rg-plane illustrates the cluster that the ellipsoid-based classifier considers to be skin color (red ellipsoid). Left: without adaptation, Right: with adaptation to the current image. The entries denote the manually annotated pixels with skin color (black crosses) and non-skin color (gray circles).

$$\begin{aligned} \boldsymbol{\mu} &= \bar{\boldsymbol{\mu}} \\ S &= \bar{S} \\ t &= 9.8 \end{aligned} \tag{3.9}$$

Figure 3.5 visualizes the appropriateness of this classifier to the challenge of skin color extraction. It compares the fixed ellipsoid cluster to the one that is adapted to the content of a particular image. Note that the yellow points within the skin color cluster denote the false positives of the classification and the red points outside of the skin color cluster denote the false negatives, respectively.

3.5.3 Rule-based Skin Color Classifier

This classifier specifies a complexly shaped cluster within NRGB color space. Any color within this cluster is treated to be skin-colored. The calculation rules that define this cluster are learned from annotated training images by a rule induction algorithm such as ID3, C4.5, or J4.8 [120; 164; 126]. The cluster is fixed if the rule induction algorithm is provided with the pixels' color features only. Equation 3.10 shows an example rule that is learned by J4.8. It specifies a fixed cluster in NRGB color space.

$$\textit{skin-colored} \quad \Leftarrow \quad (r > 0.38) \wedge (g \leq 0.33) \wedge (base > 200) \tag{3.10}$$

Adaptation to the Image-specific Characteristics:

This classifier is adapted to a particular image by making the calculation rules consider the image-specific characteristics as well. Therefore, the training data is additionally annotated with the parameters $\bar{\boldsymbol{\mu}}$ and $\bar{S}$. Furthermore, these features are combined via simple mathematical operations, e.g. division by σ for normalization purpose. This delivers highly specific features for classification [93; 146; 102; 14]. The example rule in Equation 3.11 defines a skin color cluster that is adapted to the characteristics of the image.

$$\begin{aligned} \textit{skin-colored} \quad \Leftarrow \quad & (g \leq 0.33) \quad \wedge \quad (\frac{|\bar{\mu}_r - r|}{\sigma_r} \leq 1.7) \quad \vee \\ & (g > 0.33) \quad \wedge \quad (\frac{|\bar{\mu}_r - r|}{\sigma_r} \leq 1.7) \quad \wedge \quad (\bar{\mu}_r - r \leq 0.02) \end{aligned} \tag{3.11}$$

Figure 3.6 visualizes the appropriateness of this classifier to the challenge of skin color extraction. It compares the fixed cluster to the one that is adapted to the content of a particular image. Note that the yellow points within the skin color cluster denote the false positives of the classification and the red points outside of the skin color cluster denote the false negatives, respectively.

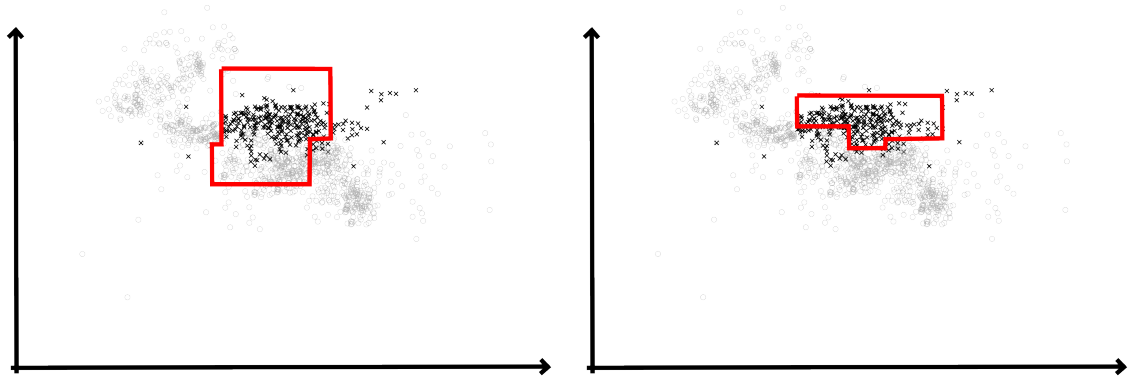


Figure 3.6: The projection to the rg-plane illustrates the cluster that the rule-based classifier considers to be skin color (area with red border). Left: without adaptation, Right: with adaptation to the current image. The entries denote the manually annotated pixels with skin color (black crosses) and non-skin color (gray circles).

3.6 Experimental Evaluation

Referring to the contributions, this section evaluates two crucial aspects of our approach. First, how accurate is the acquisition of the image-specific and person-specific characteristics. Second, how accurately do the introduced skin color classifiers determine skin color, both with and without adaptation to the image.

Since the Boston University skin color database [135] represents a benchmark data set for evaluating skin color classifiers we select it for this evaluation as well. This database consists of 21 image sequences that are taken from Hollywood movies. Their length varies between 49 and 349 frames and they show persons in natural activities such as talking, walking, or working. They are taken under various illumination conditions and include people from various ethnic groups. In addition, the database also provides annotations for each pixel that indicate *skin*, *non-skin*, or *don't care*. The latter class contains pixels, for which the creators of the database are not sure enough to specify one of the other classes. Therefore, we only consider the pixels that are labeled with *skin* and *non-skin*. Since our proof-of-concept utilizes a face locator for frontal faces, we select only those video sequences that contain frontal face views.

3.6.1 Obtaining the Image-specific Characteristics

The first evaluation compares two different approaches that approximate the image-specific and person-specific characteristics. Both of them extract a moderate number of pixels from the image, which they consider to be skin-colored. Afterwards, they apply Equation 3.4 and Equation 3.5 in order to compute the parameters $\bar{\boldsymbol{\mu}}$ and $\bar{S}$. This evaluation shows the difference between these approximated values and the correct values. Note that the correct values are computed by selecting the pixels that are manually specified to be skin-colored.

The first approach of this comparison is *color segmentation* a straightforward approach that consists of low-level vision modules. It extracts skin color pixels by applying a simple skin color classifier with a fixed skin color cluster, such as the one in Equation 3.10. Sigal et al. [135; 136] make use of this approach for tracking skin-colored regions via the color distribution. The second approach of this comparison is our approach that extracts skin color pixels via a combination of a face detector and a learned skin color mask, see Section 3.4.

Table 3.2 shows the relative error between the obtained results and the correct value of the vector $\bar{\boldsymbol{\mu}}$. Each row of the table denotes the average result value for processing every frame

		color segmentation			our approach: face locator, skin color mask		
		relative error			relative error		
seq	# frames	$\bar{\mu}_r$	$\bar{\mu}_g$	$\bar{\mu}_{base}$	$\bar{\mu}_r$	$\bar{\mu}_g$	$\bar{\mu}_{base}$
2	72	0.9%	0.8%	11.1%	0.8%	0.4%	6.5%
4	110	7.1%	5.7%	4.6%	1.0%	0.4%	0.5%
6	72	5.7%	2.2%	16.9%	0.1%	0.0%	2.8%
7	76	5.3%	5.8%	15.2%	1.0%	0.6%	3.9%
8	73	5.1%	1.2%	10.6%	1.1%	0.1%	4.2%
9	72	0.9%	2.2%	10.5%	0.7%	0.3%	1.2%
10	73	6.0%	1.2%	16.5%	0.6%	0.3%	3.2%
11	233	3.6%	1.1%	1.5%	0.4%	0.9%	4.1%
15	75	4.1%	1.3%	1.4%	0.2%	0.1%	3.1%
16	50	3.3%	1.8%	25.8%	0.5%	0.1%	5.0%
18	91	4.3%	1.6%	7.3%	0.8%	0.5%	8.1%
21	52	4.8%	1.4%	5.5%	0.4%	0.4%	7.3%
average		4.3%	2.2%	10.6%	0.6%	0.3%	4.2%

Table 3.2: Comparing two approaches that determine the image-specific characteristics for several image sequences of the Boston University skin color database. This table illustrates the distance between the result of these approaches and the true value of $\bar{\boldsymbol{\mu}}$. For normalization purpose, the denoted values are scaled by the value of $\bar{\boldsymbol{\mu}}$.

of a specific sequence. The combined approach with the face locator and the skin color mask determines the entire vector $\bar{\boldsymbol{\mu}}$ more accurately than color segmentation. The accuracy of $\bar{\mu}_r$ and $\bar{\mu}_g$ increase by the factor of seven.

3.6.2 Extracting Skin Color Pixels

The second evaluation compares the classification accuracy of the three dynamic skin color classifiers from Section 3.5: cuboid-based, ellipsoid-based, and rule-based. We compare three kinds of adapting them to the processed image sequence: (a) no adaptation, (b) optimal adaptation, and (c) automatic adaptation via our approach. Section 3.5 describes the procedure to adjust each of these classifiers. The cuboid-based and the ellipsoid-based classifiers require specifying some parameters, see Equation 3.7 and Equation 3.9. The rule-based classifier requires providing the image-specific characteristics to the rule induction algorithm.

In (a), the classifiers do not specialize to the individual images, but they are adjusted such that they are optimal for the entire database. In (b), the classifiers are optimized for each image individually. The parameters of the cuboid-based and the ellipsoid-based classifiers are computed from the annotated skin color pixels of each image. The rule-based classifier is learned for each image individually. The optimality refers to consideration of each image separately while

	(a)									(b)									(c)								
	fixed cuboid			fixed ellipsoid			fixed rules			optimal cuboid			optimal ellipsoid			optimal rules			**adapted** cuboid			**adapted** ellipsoid			**adapted** rules		
seq	skin	bg	det[C]	skin	bg	det[C]	skin	bg	det[C]	skin	bg	det[C]	skin	bg	det[C]	skin	bg	det[C]	skin	bg	det[C]	skin	bg	det[C]	skin	bg	det[C]
2	84.4	86.1	70.5	99.8	26.1	25.9	88.9	92.6	81.5	84.5	95.5	80.0	92.2	94.3	86.5	98.5	96.7	95.2	62.1	99.6	61.6	71.9	99.7	71.7	82.9	59.3	42.3
4	49.2	89.2	38.4	64.4	49.7	14.1	59.6	97.2	56.8	87.1	90.8	77.9	96.6	85.6	82.2	96.1	92.0	88.1	68.5	97.7	66.2	66.9	99.0	65.9	73.2	96.1	69.3
6	80.5	78.6	59.0	99.6	11.2	10.8	87.5	99.2	86.7	88.2	99.5	87.7	97.2	98.1	95.3	99.1	97.4	96.5	82.3	99.8	82.1	89.0	99.6	88.6	90.4	96.1	86.5
7	72.5	93.7	66.2	90.0	71.6	61.7	50.2	99.8	49.9	85.7	91.6	77.3	90.7	92.8	83.6	95.8	89.3	85.1	68.4	97.5	65.8	76.4	95.8	72.2	84.4	85.4	69.8
8	89.7	60.1	49.8	88.6	10.9	-0.6	100.0	89.1	89.1	60.0	99.5	59.5	98.6	97.8	96.4	96.0	97.2	93.2	67.7	98.2	65.9	87.1	98.6	85.6	73.5	88.1	61.6
9	77.4	99.0	76.4	99.8	94.5	94.4	85.9	96.8	82.7	87.1	100.0	87.1	99.5	99.4	98.9	99.0	98.1	97.1	83.7	100.0	83.7	89.1	99.9	89.0	86.9	98.4	85.3
10	60.2	28.4	-11.4	65.8	51.5	17.4	75.9	94.0	69.9	87.1	92.1	79.2	89.6	92.5	82.1	90.5	94.9	85.4	86.9	84.2	71.0	95.8	78.2	74.1	90.3	92.3	82.6
11	6.0	99.2	5.2	37.0	97.9	34.8	73.8	99.4	73.2	87.4	99.8	87.2	97.6	97.8	95.4	91.1	95.5	86.6	64.0	100.0	64.0	69.0	100.0	69.0	85.7	98.5	84.2
15	96.6	44.3	40.9	99.9	10.7	10.6	97.7	94.8	92.5	78.0	98.2	76.2	94.3	96.8	91.1	98.8	99.0	97.8	74.6	97.4	72.0	83.7	93.6	77.2	80.1	93.0	73.0
16	92.5	95.5	88.0	98.7	16.2	14.9	25.2	99.8	25.0	61.9	100.0	61.8	98.7	99.0	97.7	99.1	96.7	95.8	73.2	98.7	71.9	82.3	96.2	78.5	77.5	84.8	62.3
18	97.1	99.7	96.8	79.4	47.8	27.2	83.1	100.0	83.1	85.0	99.9	84.9	98.2	99.3	97.5	99.0	98.2	97.2	95.6	96.8	92.4	92.4	97.0	89.4	94.5	99.7	94.1
21	81.8	69.6	51.4	97.9	39.8	37.7	82.0	95.4	77.4	100.0	17.0	17.0	100.0	4.6	4.6	94.6	98.4	93.0	68.8	93.5	62.3	86.7	89.6	76.3	87.6	92.8	80.4
avg	74.0	78.6	52.6	85.1	44.0	29.1	75.8	96.5	72.3	82.7	90.3	73.0	96.1	88.2	84.3	96.5	96.1	92.6	74.6	96.9	71.6	82.5	95.6	78.1	83.9	90.4	74.3

Table 3.3: Adapting three different skin color classifiers in order to extract skin color pixels. We compare three kinds of adaptation: (a) no adaptation, (b) optimal adaptation, and (c) automatic adaptation via the proposed approach.

ignoring the remaining images. In (c), our approach acquires the image-specific characteristics automatically and adapts the classifiers accordingly. Note that (b) represents an upper limit for the accuracy to be reached by each classifier. However, these optimal techniques cannot be applied to real-world scenarios, because they require annotating images beforehand.

Table 3.3 shows the accuracy of distinguishing between the skin color pixels (*skin*) and the non-skin color pixels (*bg*). The values represent the true positives and the true negatives of the classification process and they are denoted in percent. Note the trade-off between optimizing either value. For example, in Sequence 6, the fixed ellipsoid-based classifier results very good accuracy for *skin*, but a very poor one for *bg*. Therefore, this table also illustrates the determinant of the confusion matrix $det[C]$, which represents a good measure for the classification accuracy, see Sigal et al. [135]. Its value is also denoted in percent.

The table clearly illustrates the increase of classification accuracy between (a) and (c). The upper limit of (b) is even approached well. For the ellipsoid-based classifier, the determinant of the confusion matrix rises from 29.1% (fixed) to 78.1% (adapted), which is close to the optimal adaptation (84.3%). Therefore, our proof-of-concept for recognizing facial expressions by model-based image interpretation integrates this technique as the feature extraction module, see Section 2.2. Figure 3.7 illustrates an example skin color image for each image sequence of our experiments.

Figure 3.7: Comparing skin color classifiers on the Boston University skin color database [135]: original image (left), fixed ellipsoid (middle), adapted ellipsoid (right). Static classifiers cannot cope with differently illuminated skin color regions (Sequence 4), background color that is similar to skin color (Sequence 8), and dark skin color (Sequence 11). The rectangular boxes denote the result of the face locator.

3.6.3 Runtime Performance

The five computational steps of our approach are split up into the adaptation phase and the application phase, see Figure 3.2: *ADA1* detect the face, *ADA2* apply the skin color mask, *ADA3* calculate the image-specific characteristics, *ADA4* adjust the skin color classifier, and *APP1* compute the skin color image. Note that the steps of the adaptation phase must be executed only once at the beginning of an image sequence, because the image-specific characteristics do not change rapidly. *APP1* is the only step to be executed for each image.

ADA1 is executed in $\Theta(n)$ where n denotes the number of pixels of the image, compare to Viola et al. [154]. It runs at an average of 50 ms on a 1800 MHz Pentium 4 processor using an image size of 480×360 pixels. The steps *ADA2* and *ADA3* are executed in $\Theta(1)$ and take 0.05 ms independently of the size of the image. Note that the skin color mask is applied faster by decreasing its size and by taking only those entries into account that exceed a given threshold value. *APP1* is executed in $\Theta(n)$ at an average of 9.3 ms using the previously mentioned image size. In conclusion, the runtime for extracting skin color pixels from one single image or from the first image of a sequence amounts to 59.4 ms. The classification of the images within the remainder of the sequence runs in 9.3 ms for the same image size.

3.7 Summary on Skin Color Extraction

Extracting skin color features from the raw image data provides salient information cues for various applications. Depending on the context conditions, such as the camera settings and the visible person, the color of human skin appears differently throughout the images, which makes automatic skin color extraction a hard challenge. Nevertheless, within one image these conditions are fixed and skin color pixels look similarly.

This chapter proposes a two-phase approach that is able to robustly extract skin-colored regions from the pixel values. First, this technique determines the image characteristics that roughly describe the appearance of skin color within a particular image. Former approaches use low-level computer vision operators for this task, like color segmentation. We obtain the image characteristics by combining a face locator and a previously learned skin color mask. Since, the utilized face locator represents a sophisticated vision module, which robustly specifies the location of a human face, we obtain highly accurate results.

Second, the determined image characteristics adapt general-purpose color classifiers to the

specific image, which makes them highly appropriate for extracting skin color pixels from this image. Our comprehensive evaluation on publicly available image databases shows the increase of classification accuracy compared to non-adaptive approaches. We obtain high accuracy facing poor illumination conditions and colored people. Particularly, non-skin colored facial regions, such as eyes, brows, lips, and teeth, are correctly distinguished from skin-colored regions. Furthermore, the evaluation indicates the ellipsoid-based classifier that is introduced in Section 3.5.2 to be most accurate and most stable. Therefore, our proof-of-concept integrates this approach for feature extraction.

Our two-phase approach not only contributes to extracting skin color features. It is able to determine further color features for various applications as well. We already succeeded in determining lip color [39] and we are currently extending our approach to detect further distinct objects within a human face, such as teeth, hair, brows, beard, mustache, iris, and pupils. Figure 3.8 depicts preliminary results on these experiments.

Further experiments apply our approach to the completely different scenario of road traffic [73]. Determining the color of the paving and the lane lines and distinguishing these objects from arbitrary background is a challenging issue, because their color depends on the weather conditions. Our insights contribute to several aspects of state-of-the-art Advanced Driver Assistant Systems. The experiments focus on robustly determining the paving color within different weather conditions, like sunny, cloudy, and rainy skies. Again, we determine the effect of the current weather characteristics to the color of paving by a color mask. This mask is previously learned from annotated images of a fixed camera mounted to the windscreen of a car. Figure 3.9 illustrates the obtained results.

3.8 Outlook on Skin Color Classification

Future work on adaptively extracting the color of the different facial regions will consider additional features to be provided to the color classifiers in order to improve the classification accuracy. We are currently considering information about the color of a pixel only. Additionally, taking the information about the location of the pixel into account will more extensively exploit the characteristics of the face locator; e.g. since the location and the size of the determined ROI is characteristic for the utilized face locator, the relative position of the lips or the eyes inside of this ROI will be predictable.

Figure 3.8: Preliminary results on adaptively extracting lip color (1st and 2nd row) and tooth color (3rd and 4th row). The black regions denote the classification result.

Figure 3.9: Preliminary results on interpreting traffic scenes by adaptive color classification. The original images show different weather conditions such as sunshine, clouds, and rain (top row) and the resulting images illustrate the automatically determined areas of paving (bottom row).

Chapter 4

Learning Robust Objective Functions

Model-based image interpretation methods exploit a priori knowledge of objects to determine abstract scene descriptors. As described in Chapter 2, the main component of these approaches is the model, whose parameter vector $\boldsymbol{p}$ describes its possible variations, such as the position, pose, shape, scale, and texture. The deformable face model used in our proof-of-concept maps these parameters to the surface of an image via a contour that consists of set of contour points $\boldsymbol{c}_n(\boldsymbol{p})$.

Determining the best fit between a model and an image automatically requires two further components. First, the objective function $f(I, \boldsymbol{p})$ that has been described in Section 2.4 indicates how well a model parameterization $\boldsymbol{p}$ fits to an image I. The global minimum of the objective function corresponds to the best model fit. This thesis elaborates on objective functions that are computed as a sum of *local* objective functions $f_n(I, \boldsymbol{x})$. As explained in Section 2.4.1, each local objective function gives evidence about the fitness of the local part of the model around the contour point $\boldsymbol{c}_n(\boldsymbol{p})$. From now on, we will concentrate on local objective functions, and simply refer to them as objective functions. The global objective function is always computed from them by applying Equation 2.1. Second, the fitting algorithm that has been described in Section 2.5 searches for the model parameterization that best fits to the image, i.e. the values of $\boldsymbol{p}$ that minimizes $f(I, \boldsymbol{p})$. For recent overviews and categorizations of various types of models, objective functions and fitting algorithms, we refer to Hanek et al. [68] and to Romdhani [123].

4.1 Problem Statement

Fitting algorithms have been the subject of intensive research and evaluation, which has led to an impressive array of algorithms that are capable of dealing with very complex search spaces. Some approaches with great impact have been published by [49; 77; 78; 106; 67]. In contrast, the objective function is usually determined ad hoc and heuristically, using the designer's intuitions about a good measure of fitness. Afterwards, its appropriateness is subjectively determined by inspecting its result, which is evaluated on example images and example model parameterizations. If the result is not satisfactory the objective function is tuned or redesigned from scratch. This iterative process is shown in Figure 4.1.

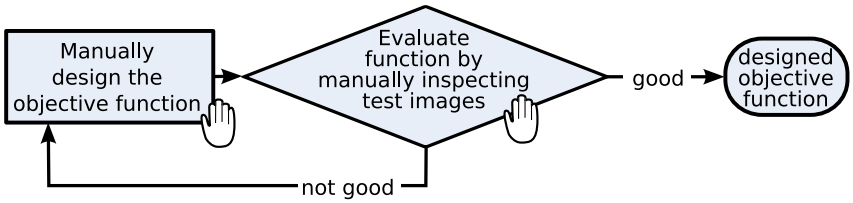


Figure 4.1: The traditional procedure for designing objective functions requires the designer to specify the calculation rules for the objective function and verify them on example images. Note that the entire work is accomplished manually, which is tedious and error-prone.

In short, the traditional way of designing objective functions is rather an art than a science. The consequences are that this design approach requires much implicit and domain-dependent knowledge. Its iterative nature also makes it a time-consuming process of unpredictable duration. Furthermore, the resulting objective functions tend be not very accurate, because they have a global minimum, which does often not correspond to the best fit and because they also have further local minima, in which fitting algorithms are likely to get stuck. Especially this last consequence is a direct cause for the complexity and sophistication of fitting algorithms: in order to determine the optimum of complex search spaces, complex search algorithms are required.

4.2 Solution Idea

Our novel approach takes inspiration from Ginneken et al. [55], and focuses on the root of the problem: We improve the objective function rather than the fitting algorithm. Our goal is to acquire objective functions that enable fast and accurate optimization, even with simple fitting

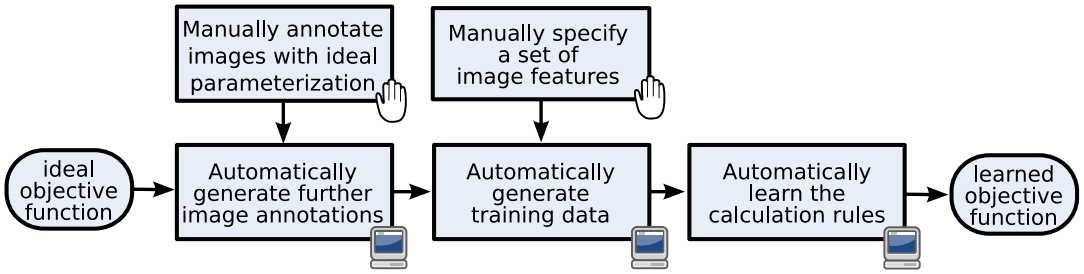


Figure 4.2: The novel procedure learns the objective functions from training data, which automates the crucial steps and requires no iterative work.

algorithms. This chapter presents two properties that such *ideal* objective functions have. For practical applications, it is impossible to design ideal objective functions by hand. Therefore, we approximate such a function by learning it from annotated example images and with the help of an ideal objective function. This procedure is depicted in Figure 4.2.

The first step is to collect a big set of images and manually annotate them with the model that fits best. Further annotations are obtained by slightly changing these model parameters. The training data consists of the images, the manually specified model parameters, the automatically varied parameters, and the corresponding value of an ideal objective function. A general objective function, which maps the image and the model parameters to these ideal objective values, is learned from this training data. To facilitate the learning phase, we manually define features that are taken from the image in the vicinity of the model beforehand.

This approach has several benefits. Most of the steps are automated, and the remaining two manual steps require little or no domain-dependent knowledge. These two steps do not contain human decisions that are critical with respect to the robustness of the resulting objective function, so less contemplation is needed. Furthermore, the loop caused by the *design-inspect* iteration is eliminated, so each manual step only needs to be considered once.

Apart from simplifying the task of the designer, this approach also yields more robust objective functions. Since an ideal objective function is used to generate the training data, the learned objective function will also be approximately ideal. The main reason why this is difficult to achieve by designing an objective function, is that it is unclear, which image features are relevant for the objective function and which are not. In our approach, this critical step is automated and relevant features are chosen from a large set of image features based on objective relevance measures. The resulting objective functions are more accurate and robust and easier to optimize, which we will verify with an extensive empirical evaluation.

The contributions of this approach to model-based image interpretation are:

1. We define two domain-independent properties that ideal objective functions must have.
2. We describe an approach that learns objective functions from training data that is generated from annotated images and an ideal objective function.
3. We demonstrate that this approach automatically selects relevant image features.
4. We formulate domain-independent indicators that measure if or to which extent an objective function fulfills the previously stated properties.
5. We empirically verify that the learned objective functions are more robust and accurate than designed objective function. We also demonstrate that better fitting results are therefore achieved with them.

The remainder of this chapter is organized as follows. The next section introduces two properties that ideal objective functions have. Section 4.3 gives an example of a designed objective function that is not ideal. Section 4.5 describes the novel methodology that learns robust objective functions from annotated example images. In Section 4.6, we empirically evaluate learned objective functions with respect to the formulated properties, and evaluate its suitability for model-based image understanding. Section 4.7 explains the advantages and also mentions the shortcomings of this approach. In Section 4.8 we discuss related work on obtaining objective functions via machine learning approaches. We summarize our approach in Section 4.9 and present future work in Section 4.10.

4.3 Designing Objective Functions

Objective functions are usually designed manually, such as in [28; 25; 33; 123; 68; 40; 148; 63]. The designer selects a small number of salient features from the image and mathematically composes them in order to obtain the value of the objective function. Therefore, the feature selection and the mathematical composition are both based on intuition and implicit knowledge of the domain. As mentioned by [123; 143; 28; 27] for instance, the objective function is computed from pixel color, edge values, texture edges, specular highlights, and even from manually specified anchor points.

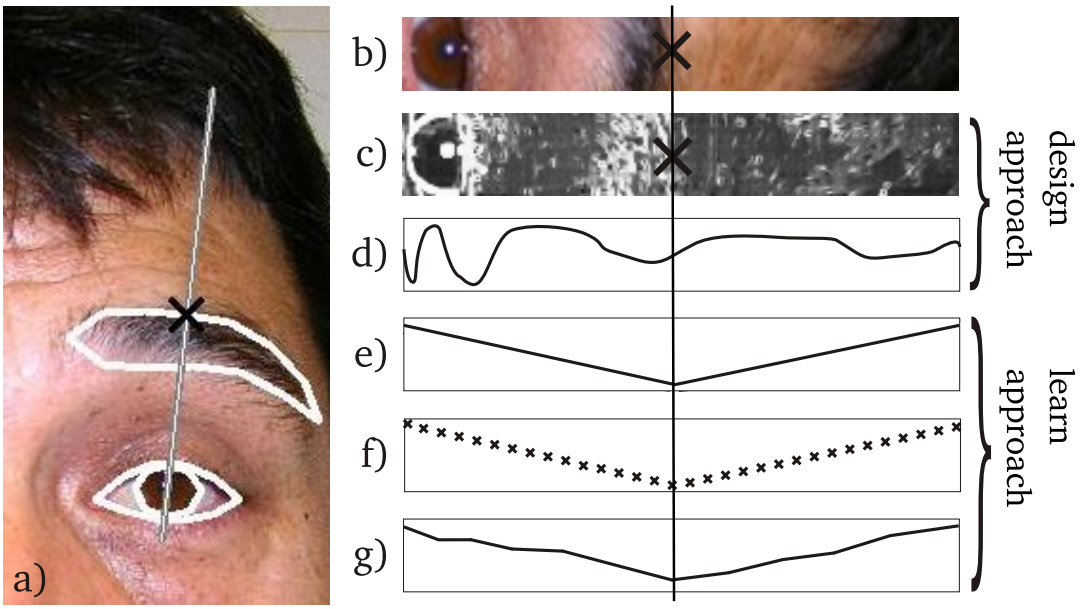


Figure 4.3: a) Contour point with orthogonal towards contour, b) Image data, c) Edge magnitudes, d) Designed objective function $f_n^e(I, \boldsymbol{x})$, e) Ideal objective function, f) Training data, g) Learned objective function; Note that b) – g) are taken along that orthogonal visible in a). The vertical line represents the location of the ideal contour point $\boldsymbol{c}_n(\boldsymbol{p}_I^\star)$.

A similar objective function is shown in Equation 4.1, where $E(I, \boldsymbol{x})$ denotes the magnitude of the edge at the pixel $\boldsymbol{x}$. Each contour point of the model is considered to be located well if it overlaps a strong edge of the image. The magnitudes range between $0 \leq E(I, \boldsymbol{x}) \leq 1$. The label e refers to "edge-based".

$$f_n^e(I, \boldsymbol{x}) = 1 - E(I, \boldsymbol{x}) \tag{4.1}$$

Unfortunately, such manually designed objective functions have comprehensive shortcomings and unexpected side-effects. Let us illustrate this with the example image, depicted in Figure 4.3. Figure 4.3a) depicts one of the contour points of the face model as well as its perpendicular towards the model's contour. Figure 4.3b) and 4.3c) depict the content of the image along this perpendicular as well as the corresponding edge magnitudes $E(I, \boldsymbol{x})$. In Figure 4.3d), we depict the value of the local objective function of Equation 4.1 along the perpendicular. Obviously, this function has many local minima within this one-dimensional search region. Furthermore, the global minimum does not correspond to the ideal location of the contour point.

With so many local minima, a fitting algorithm would have difficulty in finding the global minimum. Even if it did, it would be wrong, as it does not correspond with the ideal position of the contour point. The only reason why fitting algorithms do determine an acceptable model fit using such a local objective function is that many of these functions are averaged and smoothed when computing the global objective function as in Equation 2.1.

4.4 Properties of Ideal Objective Functions

To determine the best model fit by minimizing the objective function with some fitting algorithm, Figure 4.3e) is preferable over Figure 4.3d), as its global minimum actually corresponds to the best fit, and it does not contains local minima. This section explicitly formalizes the intuitions above into two properties. An objective function that has both of these properties is called *ideal*.

The mathematical formalization of P1 uses $\boldsymbol{p}_I^\star$, which are the *ideal* model parameters. These are defined to be those model parameters that fit best to a specific image I. Usually, $\boldsymbol{p}_I^\star$ must be determined manually. Figure 4.3a) is an example of an image annotated with $\boldsymbol{p}_I^\star$. Ideal model parameters will be discussed more elaborately in Section 4.5.1.

P1: Correctness property: The global minimum of the objective function corresponds to the best model fit.

$$\forall \boldsymbol{x}(\boldsymbol{c}_n(\boldsymbol{p}_I^\star) \neq \boldsymbol{x}) \quad \Rightarrow \quad f_n(I, \boldsymbol{c}_n(\boldsymbol{p}_I^\star)) < f_n(I, \boldsymbol{x})$$

P2: Uni-modality property: The objective function has no local extrema or saddle points.

$$\exists \boldsymbol{m} \forall \boldsymbol{x}\ (\boldsymbol{m} \neq \boldsymbol{x}) \Rightarrow$$

$$f_n(I, \boldsymbol{m}) < f_n(I, \boldsymbol{x}) \ \ \wedge \ \ \nabla f_n(I, \boldsymbol{x}) \neq \boldsymbol{0}$$

Property P1 relates to the *correctness* of the objective function. Fitting algorithms search for the global minimum of the objective function. P1 ensures that the result of a successful search corresponds to the best fit of the model. Although it might seem obvious that this is a desirable property for objective functions to have, designing them does not always guarantee that this is the case. Figure 4.3d) is a good example, and Section 4.6 will verify this more generally.

Property P2 guarantees that any minimum that is found is the global minimum. This facilitates search, as fitting algorithms cannot get stuck in a local minimum. Local optimization strategies, which are easier to design than global ones, then suffice to find the global minimum. The mathematical formalization states that all locations $\boldsymbol{x}$ that are not the global minimum $\boldsymbol{m}$ are not allowed to have a zero gradient, and are therefore not minima. Note that the global minimum $\boldsymbol{m}$ does not need to correspond with the best fit; this is only required by the independent property P1.

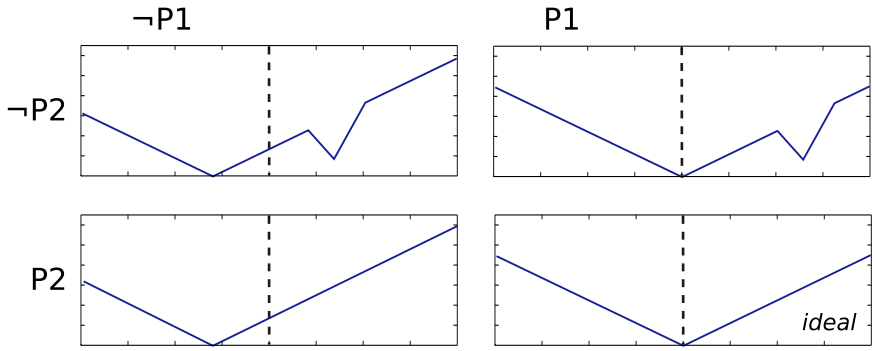


Figure 4.4: These four graphs display typical example functions that do or do not have properties P1 and P2, which influence the behavior of an objective function. The dashed line indicates the ideal position of the contour point. If both P1 and P2 hold, the objective function is considered to be ideal.

Figure 4.4 depicts four graphs that show examples of functions that exhibit typical differences between functions with and without these properties. The dashed line represents the parameters that correspond to the best model fit $\boldsymbol{p}_I^\star$. Objective functions that have both properties guarantee that local optimization strategies will find the global minimum and this minimum also corresponds to the best model fit. When designing objective functions, designers will implicitly attempt to construct such objective functions, because they anticipate that a fitting algorithm will need to determine its minimum quickly and correctly, without getting stuck in local minima. P1 and P2 make these intuitions explicit, and allow the formal specification of concrete ideal objective functions. These properties serve as a baseline for evaluating objective functions. In Section 4.5, we will also use them to learn robust objective functions.

Note that both properties define idealness for local objective functions only. Properties that state correctness and uni-modality for global objective functions would look similarly. However, the non-linear mapping from pixel space to parameter space does not guarantee that a global objective function is ideal, even if it is composed of ideal local objective functions. This

is not a consequence of our methodology, but of the general approach using local objective functions. Nevertheless, our evaluation shows that the idealness of local objective functions is similar to the idealness of the global objective function composed from the sum of local objective functions, see Section 4.6.3 and Figure 4.13.

We will now introduce a concrete instance of an ideal objective function, which we will call $f_n^\star(I, \boldsymbol{x})$. It is defined in Equation 4.2, and has already been depicted in Figure 4.3e). It computes the distance between the contour point $\boldsymbol{c}_n(\boldsymbol{p}_I^\star)$ given the ideal parameters $\boldsymbol{p}_I^\star$ and a pixel $\boldsymbol{x}$ located on the image surface. We will prove that $f_n^\star(I, \boldsymbol{x})$ has properties P1 and P2 in Appendix A.

$$f_n^\star(I, \boldsymbol{x}) = |\boldsymbol{x} - \boldsymbol{c}_n(\boldsymbol{p}_I^\star)| \tag{4.2}$$

The most significant feature of $f_n^\star$ is that it uses the ideal model parameters $\boldsymbol{p}_I^\star$ to compute its value. Knowledge of $\boldsymbol{p}_I^\star$ is essential to ensure P1, which expresses that the global minimum of $f_n^\star$ coincides with $\boldsymbol{p}_I^\star$. Unfortunately, this implies that $f_n^\star$ cannot be applied to previously unseen images, because $\boldsymbol{p}_I^\star$ is not known for these images. In real-world applications, $f_n^\star$ is therefore useless for model fitting. However, the next section shows how we will use this ideal objective function in order to generate training data, from which an objective function is learned. This objective function then approximates $f_n^\star(I, \boldsymbol{x})$.

4.5 Five Steps to Obtain Robust Objective Functions

This section explains in detail how we learn an objective function f_n^ℓ in order to approximate the ideal objective function $f_n^\star$. The key idea behind this approach is that $f_n^\star$ has the properties P1 and P2, and therefore f_n^ℓ will approximately have these properties as well. Since it is considered

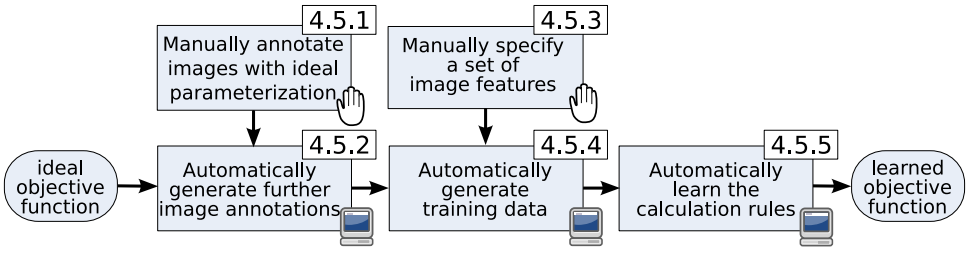


Figure 4.5: This figure indicates the sections that elaborate on the individual steps of the proposed learn approach.

to be "approximately ideal", we will refer to it as a *robust* objective function. This section describes the five steps of our methodology, repeatedly illustrated in Figure 4.5 in order to indicate the sections explain the particular steps.

4.5.1 Annotating Images with Ideal Model Parameters

A database of images I_k with $1 \leq k \leq K$ is manually annotated with $\boldsymbol{p}^\star_{I_k}$, the ideal model parameters[1]. These ideal model parameters are necessary to compute the ideal objective function $f^\star_n$ via Equation 4.2, which in turn computes the training data in a later step. This annotation is the only laborious step in the entire procedure of the proposed approach. An experienced human needs about one minute to determine the ideal parameters of our face model for one image. Figure 4.6 shows four images of the database that are annotated with the ideal parameters of our face model.

Figure 4.6: Four example images that are manually annotated with the ideal face model.

For synthetic images, $\boldsymbol{p}^\star_{I_k}$ is known, and can be used in such cases, see Lepetit et al. [95] and Boffy et al. [15]. However, for real-world images, the ideal model parameters depend on the user's judgment. In this case, we cannot use a predefined objective measure that determines ideal model parameters, because such a measure does not exist. This is not a consequence of our approach; the same holds for all annotated benchmarks, such as BioID [80], XM2VTS [109], and IMM Face Database [113].

[1]For clarity, we add a Glossary of Notation to the Appendix B, because many indices and symbols are needed to explain our approach.

4.5.2 Generating Further Image Annotations

For all image annotations, the ideal objective function $f_n^\star(I, \boldsymbol{x})$ returns the minimum zero, in accordance with P1. This is because $\boldsymbol{x}$ is set to $\boldsymbol{c}_n(\boldsymbol{p}_I^\star)$. Obviously, these image annotations are not sufficient to learn $f_n^\ell(I, \boldsymbol{x})$. Training data must also contain image annotations $\boldsymbol{x}$, for which $f_n^\star(I, \boldsymbol{x}) \neq 0$. In order to acquire this data we vary $\boldsymbol{x}$ automatically. General variations move $\boldsymbol{x}$ to any position within the image, however, it is more practicable to restrict this motion in terms of distance and direction. This section describes how we move $\boldsymbol{x}$ along the perpendicular towards the contour at the contour point in order to generate further image annotations. Taking only these displacements into account facilitates the later learning step and improves the accuracy of the resulting calculation rules.

We generate $2D$ displacements $\boldsymbol{x}_{k,n,d}$ with $-D \leq d \leq D$ from the ideal contour point $\boldsymbol{x}_{k,n,0} = \boldsymbol{c}_n(\boldsymbol{p}_{I_k}^\star)$ for the image k and the contour point n. These displacements are situated on the perpendicular to the contour line at the contour point n with a maximum distance Δ to the contour point. This procedure is depicted in Figure 4.7, which explains the meaning of the indices k, n, and d. The center row depicts the manually annotated images, for which $f_n^\star(I, \boldsymbol{x}_{k,n,0}) = f_n^\star(I, \boldsymbol{c}_n(\boldsymbol{p}_{I_k}^\star)) = 0$. The other rows depict the displacements $\boldsymbol{x}_{k,n,d\neq 0}$ from this ideal contour point. As defined by the property P1, $f_n^\star(I, \boldsymbol{x}_{k,n,d\neq 0}) > 0$ for these rows. Note that the value of Δ is significant for the accuracy of the obtained objective function. It specifies the area, from which the training data originates and therefore, we will term it *learning radius*. Section 4.6.5 conducts experiments on the impact of the value of Δ to the suitability of the obtained result.

Due to different resolutions and image sizes, the number of pixels that represent the face varies substantially. Distance measures, such as the return value of the ideal objective function, error measures, and the learning radius Δ, should not be biased by this variation. Therefore, all distances in pixels are converted to the interocular measure, by dividing them by the pixel distance between the pupils. The interocular measure is relatively constant with respect to the face model.

4.5.3 Specifying Image Features

Our approach learns a mapping from I_k and $\boldsymbol{x}_{k,n,d}$ to the value that is computed by $f_n^\star(I_k,\ \boldsymbol{x}_{k,n,d})$. As mentioned before, this mapping will be called f_n^ℓ. Because f_n^ℓ has no knowledge of $\boldsymbol{p}_I^\star$, it must compute its value from the content of the image. Instead of learning a direct mapping from the pixel values of I in the vicinity of $\boldsymbol{x}$ to $f_n^\star(I, \boldsymbol{x})$, we use a feature-extracting method [67],

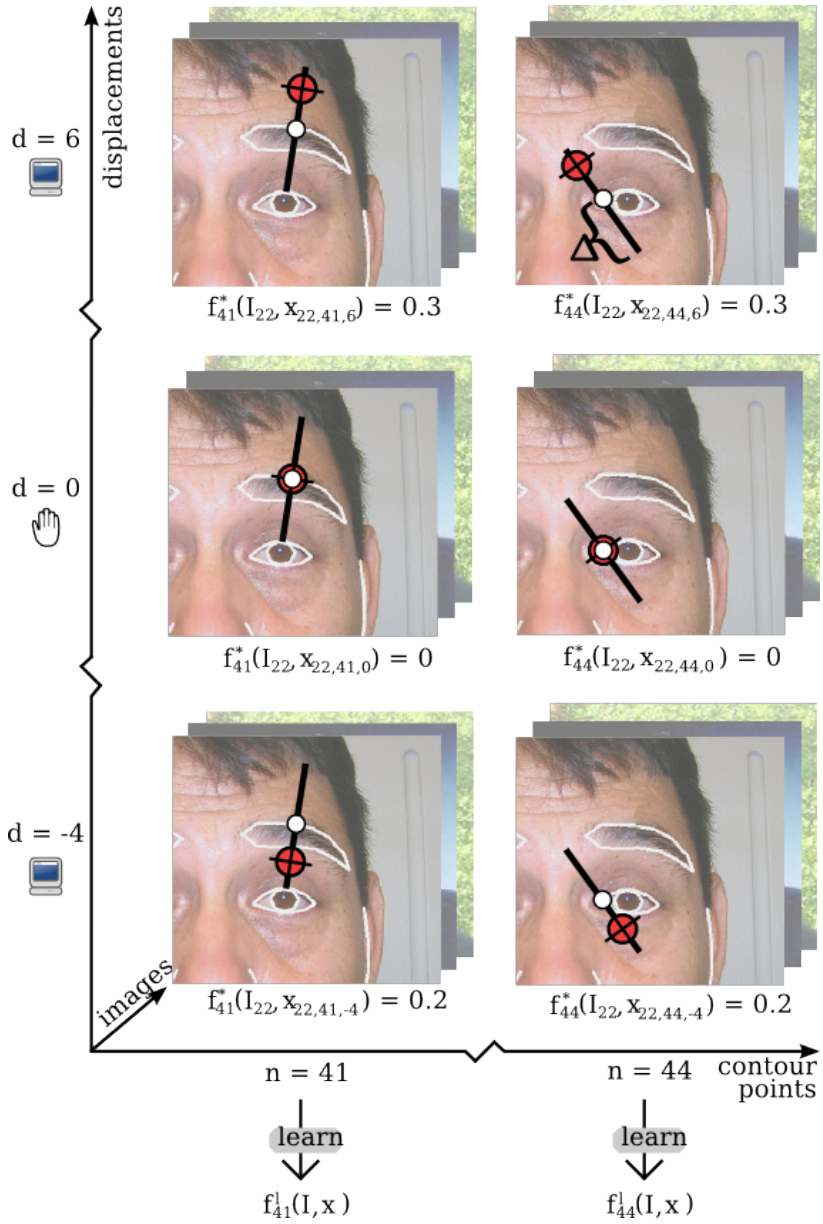


Figure 4.7: Within each of the K images, each of the N contour points is annotated with $2D+1$ displacements, which are denoted with $\boldsymbol{x}_{k,n,d}$. However, manual work is only necessary for specifying the displacements $\boldsymbol{x}_{k,n,d=0}$, which is depicted in the middle row. The other annotations are computed automatically. Note the learning radius Δ in the uppermost right image that indicates the maximum distance of the displacements $\boldsymbol{x}_{k,n,d}$. The unit of the value of the ideal objective function and the unit of the learning radius Δ is normalized by the interocular distance.

which extracts features from the image around the specified location $\boldsymbol{x}$. Our idea is to provide a multitude of image features, and let the training algorithm choose which of them are relevant to the computation rules of the objective function and which are not. Each feature $h_a(I, \boldsymbol{x})$ with $1 \leq a \leq A$ is calculated from an image I at a particular location $\boldsymbol{x}$ and delivers a scalar value.

The approach presented in this thesis, relies on Haar-like features that have been leveraged by Viola et al. [154] and by Lienhart et al. [99]. Each Haar-like feature defines two regions of pixels, depicted in black and white in Figure 4.8. As described in Section 2.7.2, their value is calculated by subtracting the sum of pixel intensities within the black region from the sum of pixel intensities within the white region. Haar-like features are efficiently computed from the so-called *integral image*, which contributes to the high speed of our approach, see the evaluation in Section 4.6.8. Contrary to edge-based and region-based features, Haar-like features cope with noisy image data. This fact contributes to the high accuracy of our approach, see the evaluation in Section 4.6.5.

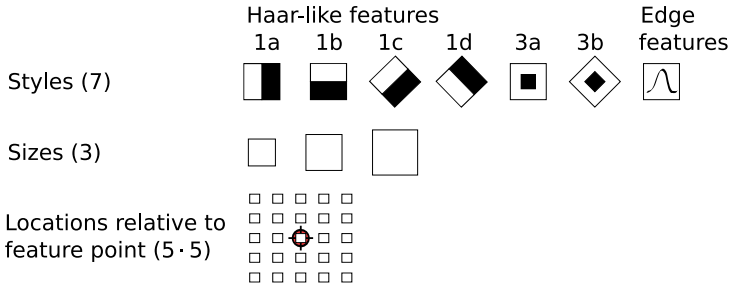


Figure 4.8: This comprehensive set of image features is provided for learning objective functions. The total number of features as we use it in our experiments is $A = 7 \cdot 3 \cdot 5 \cdot 5 = 525$.

Figure 4.8 lists the styles and sizes of each Haar-like feature that our proof-of-concept currently comprises. We additionally consider edge-based features for comparison purpose and we compute them via Sobel operators of different matrix sizes. All these features are not only computed at the location of the contour point itself, but also at positions located on a grid within its vicinity, as shown in Figure 4.8 and Figure 4.9. This variety of styles, sizes, and locations delivers a set of $A = 525$ different image features as we use it in our experiments in Section 4.6. This multitude of features enables the learned objective function to exploit the texture of the image at the model's contour point and in its surrounding area. When moving the contour point, the image features move along with it, leading their values to change, as can be seen in Figure 4.9.

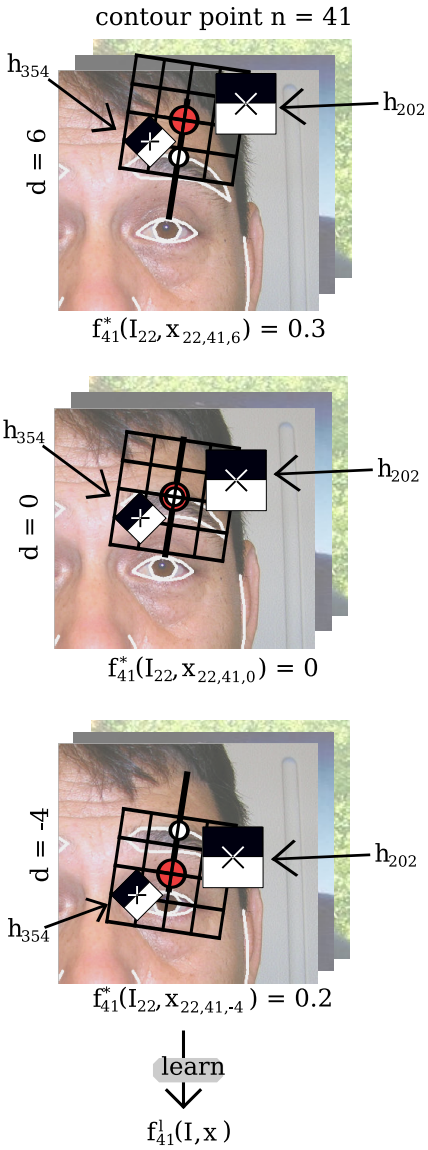


Figure 4.9: Image features are located on a grid in the vicinity of the contour points. When a contour point is displaced, the image features on the grid moves along with it, and the values of the images features change. Two image features $a \in \{202, 354\}$ are visualized exemplary for the contour point n=41.

4.5.4 Generating Training Data

The result of the manual annotation step (Section 4.5.1) and the automated annotation step (Section 4.5.2) is a list of correspondences between landmarks on the image and the corresponding value of $f_n^\star$. Since K images, N contour points, and $2D + 1$ displacements are landmarked

these correspondences amount to $K \cdot N \cdot (2D + 1)$. Equation 4.3 illustrates the list of these correspondences. Figure 4.7 depicts examples for $k = 22$, $n \in \{41, 44\}$ and $d \in \{-4, 0, 6\}$.

$$[I_k,\ \boldsymbol{x}_{k,n,d},\ f_n^\star(I_k,\ \boldsymbol{x}_{k,n,d})] \quad \text{with } 1{\leq}k{\leq}K,\ 1{\leq}n{\leq}N,\ -D{\leq}d{\leq}D \tag{4.3}$$

Applying the list of manually selected features to the list of correspondences yields the list of training data in Equation 4.4. This step simplifies matters greatly. Since each feature returns a single value, we hereby reduce the problem of mapping the huge amount of image data and the related pixel locations to the corresponding target value, to mapping a manageable list of feature values to the target value. Note that the size of the training data amounts to $K(2D + 1)$ records for each of the N contour points.

$$[h_1(I_k, \boldsymbol{x}_{k,n,d}), \ldots, h_A(I_k, \boldsymbol{x}_{k,n,d}),\ f_n^\star(I_k, \boldsymbol{x}_{k,n,d})] \quad \text{with } 1{\leq}k{\leq}K,\ 1{\leq}n{\leq}N,\ -D{\leq}d{\leq}D \tag{4.4}$$

4.5.5 Learning the Calculation Rules

Given the training data from Equation 4.4, the goal is to now learn the function $f_n^\ell(I, \boldsymbol{x})$ that approximates $f_n^\star(I, \boldsymbol{x})$. Note that we are not simply relearning the already known function $f_n^\star$ that is specified in Equation 4.2. The difference is that f_n^ℓ does not require knowledge of $\boldsymbol{p}_I^\star$, and can therefore be applied to previously unseen images as well. We obtain this mapping by training a model tree [119; 164] with the assembled training data from Equation 4.4. Model trees are a generalization of regression trees and, in turn, decision trees [120]. Whereas decision trees have nominal values at their leaf nodes, model trees have line segments, allowing them to also map features to a continuous value, such as the value returned by the ideal objective function. They are learned by recursively partitioning the feature space. Afterwards, a linear function is fitted to the training data in each partition using linear regression. Figure 4.10 shows a two-dimensional plot of the correspondences between $\boldsymbol{x}$ and $f_n^\star$. It also illustrates a plot of an objective function f_n^ℓ that is learned for the contour point n from all training images, all displacements, and all image features.

One of the reasons for deciding for model trees is that they tend to select only features that are relevant to predict the target value. Therefore, they pick a small number of M_n Haar-like features from the provided set of $A \gg M_n$ features. This selection not only enforces the accuracy, but drastically speeds up the execution as well. Section 4.6.8 will evaluate this aspect

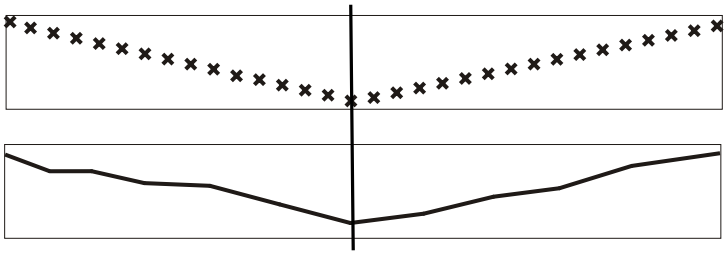


Figure 4.10: Training data and the function learned by the model tree algorithm. This function is piecewise linear and thus depends highly non-linearly on the provided image features.

and compare the runtime to other approaches. Equation 4.5 visualizes the relationship between the objective function f_n^ℓ and the calculation rules T_n for the n^{th} contour point. It illustrates that the calculation rules do not take the entire set of image features, but a subset of selected features $s_1, ..., s_{M_n}$.

$$f_n^\ell(I, \boldsymbol{x}) = T_n(h_{s_1}(I, \boldsymbol{x}), \ldots, h_{s_{M_n}}(I, \boldsymbol{x})) \tag{4.5}$$

This becomes apparent when inspecting the automatically generated calculation rules. As expected, the local objective functions for the different contour points use different subsets of Haar-like features. Some clarifying hypothetical cases are depicted in Equation 4.6. Currently, we are providing A=525 image features, as are illustrated in Figure 4.8. The model tree selects around $M \approx 20$ of them. We will evaluate in more detail, which kinds of features are used at different contour points in Section 4.6.2.

$$\begin{aligned} f_{13}^\ell(I, \boldsymbol{x}) &:= T_{13}(h_{11}(I, \boldsymbol{x}), h_{19}(I, \boldsymbol{x}), h_{21}(I, \boldsymbol{x}), \ldots) \\ f_{80}^\ell(I, \boldsymbol{x}) &:= T_{80}(h_{5}(I, \boldsymbol{x}), h_{8}(I, \boldsymbol{x}), h_{32}(I, \boldsymbol{x}), \ldots) \\ f_{95}^\ell(I, \boldsymbol{x}) &:= T_{95}(h_{8}(I, \boldsymbol{x}), h_{12}(I, \boldsymbol{x}), h_{21}(I, \boldsymbol{x}), \ldots) \end{aligned} \tag{4.6}$$

However, generating a model tree requires considering its accuracy towards the generality of the training data. This issue is referred to with *overfitting*. The size of the model tree directly relates to this issue and it is controllable by a parameter that specifies the minimum size of a partition of the training data. Four our evaluation in Section 4.6 we determine this parameter to be 5% of the size of the training data $K(2D + 1)$.

After executing these five steps, we obtain a local objective function for each contour point. It can now be called with an arbitrary location $\boldsymbol{x}$ of an arbitrary image I. The learned model tree calculates the values of the specified features at this location from the content of the image and executes its calculation rules, as shown in Equation 4.5.

4.6 Experimental Evaluation

This section evaluates learned objective function in the context of deformable face model fitting, because it is essential for the subject of this thesis: facial expression interpretation. However, this scenario is typical for real-world applications, because the involved algorithms require robustness towards a lot of variations to the image data. The evaluation incorporates 500 images of frontal faces from the Internet and the television. Due to their widespread origin, they show large variations in background, illumination, focal length, color saturation, size, and face orientation. The face model has already been introduced in Section 2.7.1.

In order to demonstrate the general applicability of our approach, we will conduct the experiments of face model fitting with both gray-scale and skin color images as they are determined by adaptive skin color extraction in Chapter 3.

Section 4.6.1 introduces a state-of-the-art objective function that is commonly used for model fitting. This function serves for comparison purpose. Section 4.6.2 disassembles the automatically generated model trees and inspects their calculation rules. Section 4.6.3 analyzes to what extent learned local objective functions have the properties P1 and P2. Section 4.6.4 illustrates the accuracy of the global objective function varying a couple of parameters of the ideal model. Section 4.6.5 evaluates learned objective functions in the context of model fitting with our proof-of-concept. Section 4.6.6 illustrates the behavior of our approach in the case of partially occluded faces. Section 4.6.7 publishes the accuracy of our approach on a commonly available image database and compares the obtained results to a recent state-of-the-art approach. Section 4.6.8 elaborates on the timing characteristics of different approaches.

4.6.1 State-of-the-art Objective Function for Comparison

This section describes a state-of-the-art approach for obtaining objective functions, which we will use in the remainder of this chapter for comparison purpose. This approach is first published by Cootes et al. [26], but recent publications still base on it, such as [31; 61; 29]. This

technique relies on local image structure in order to determine the model fit. They derive statistics from observations on the pixel values in the vicinity of the contour points. In order not to depend on varying illumination, they propose to consider edge values rather than intensity values. Therefore, a number of A features are sampled along perpendicular lines through every contour point of the model, which we will also denote h_a. Similar to our approach, these feature values are combined to the feature vector $\boldsymbol{h}_n$. They compute statistics on these feature values and obtain the mean $\boldsymbol{\mu}_n$ and the covariance matrix $\boldsymbol{S}_n$ for the contour point n. The value of the objective function denoted with f_n^s is calculated by the Mahalanobis distance between the current observations $\boldsymbol{h}_{obs}$ on the image and the statistical mean $\boldsymbol{\mu}_n$, see Equation 4.7.

$$f_n^s(I, \boldsymbol{x}) = (\boldsymbol{h}_{obs} - \boldsymbol{\mu}_n)^T \boldsymbol{S}_n^{-1} (\boldsymbol{h}_{obs} - \boldsymbol{\mu}_n) \tag{4.7}$$

The integration of the Mahalanobis distance assumes that the image observations have a Gaussian distribution. This distance measure is used instead the Euclidean distance, because some feature values may have greater variance than others and variation in one feature value may be more important to the result value than the variation of others. The Mahalanobis distance takes this fact into consideration, whereas the Euclidean distance would not.

4.6.2 Interpretation of the Calculation Rules

Each local objective function is learned with a model tree, which is expected to select the most relevant features for prediction from a set of features. Figure 4.11 illustrates, which features the model tree picks to construct the local objective function for the contour points n=92 and n=116 for the skin-color images. The attached face models show the location of these contour points. The x-axis depicts the different styles of image features provided by the designer, whereas the y-axis depicts their different sizes. The radius of the circle illustrates the frequency each features is used to compute the model tree value. Note that the descriptors for the feature's location within the square grid are omitted, because these descriptors would induce additional dimensions that make the plot more confusing. Each ball within this graph therefore represents a couple of image features.

Inspecting these and other examples leads to conclusion that edge-based features are hardly used to determine the value of the objective function. The model tree rejects them in *learned* objective functions and this raises the question if it is appropriate to use edge-based features in *designed* objective functions. Apparently, Haar-like features are more relevant and informative

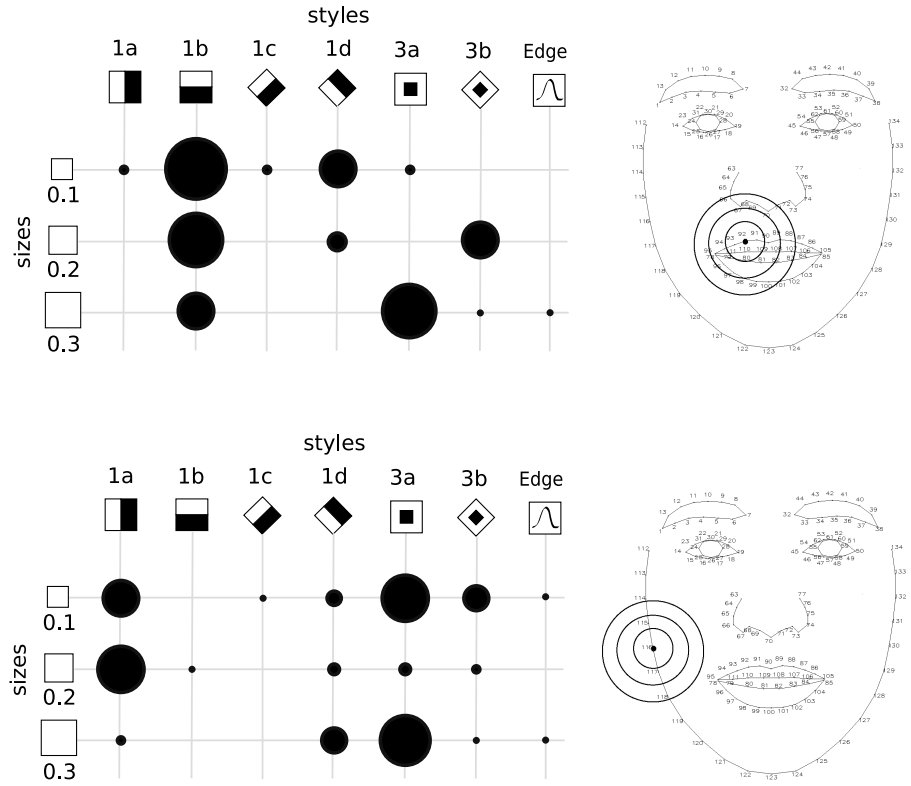


Figure 4.11: The calculation rules of several contour points pick different image features. Top: contour point n=92 is part of the upper lip, bottom: contour point n=116 is part of the face boundary. Features are calculated from skin-color images.

when it comes to learning objective functions. Note that this is not our subjective opinion. It is based on objective measures that the model tree uses to select features.

A closer inspection of Figure 4.11 verifies some intuitions. The predominant image features at contour point n=92, on the upper lip, are Haar-like features with a horizontal orientation. As can be seen in Figure 3.1, there is a clear horizontal transition from black (lip) to white (skin) in the skin color image. This implies that horizontally aligned Haar-like features will return higher values the more accurately they are aligned with this transition. The model tree has learned to exploit this informative feature. The same holds for contour point n=116, situated on the face boundary on the cheek, where vertical edges are favored because there is a vertical transition from skin color on the cheek to the non-skin colored background.

The error of each local objective function is depicted in Figure 4.12, for both the gray-scale and skin-color images. The radius of each circle is 100% minus the relative absolute error of the

Figure 4.12: The objective functions of different contour points are learned with different accuracy. Left: Learning on gray value images. Right: Learning on skin color images.

learned objective function as it is evaluated on a separate test set by the rule induction algorithm. Therefore, larger circles correspond to more accurately learned calculation rules.

For both the gray-value image and the skin-color images, the contour points around the eyes and eye brows are learned very accurately. This is because there is much distinctive texture in this region, which makes many features highly evident for determining their relative position to the model's contour point. This allows many features to be used for the calculation rules. For skin-colored images, the contour points on the chin are not learned well, because there is no distinctive texture, as can be seen in the small face image at the bottom of Figure 4.12.

An interesting difference between the two types of images is seen at the chin line (contour point $112 \leq n \leq 134$), along the left and right side of the face. These contour points are located at the boundary between the face and the background. For gray-scale images, the background is more or less random, and image features in the vicinity of contour points at a transition near this background are likely to contain random background values. These values are not very informative, and hardly used when learning the model. For skin-color images, in contrast, the background is likely to be non-skin colored, and a clear boundary between the face and the background arises. This is detected well by many image features, and an accurate local objective function is learned from the informative values they return, as can be seen in Figure 4.12.

4.6.3 Accuracy of the Local Objective Functions

Formally, objective functions either have properties P1 and P2, or they do not. In this section, we are more lenient, and define indicators I1 and I2 that compute to what extent objective functions have these properties. These indicators are computed for each image and each local objective function individually. For ideal objective functions, these indicators are I1=0 and I2=0 by definition.

I1: Correctness indicator: This indicator quantitatively shows the distance (in the interocular distance measure) between the ideal position of the contour point $\boldsymbol{c}_n(I, \boldsymbol{p}_I^\star)$ and the global minimum $\boldsymbol{m}$ of the local objective function. It is calculated within a certain range $\tilde{\Delta}$ around the ideal position.

$$\begin{aligned} \boldsymbol{m} &= \underset{\forall \boldsymbol{x}:|\boldsymbol{x}-\boldsymbol{c}_n^\star|\leq\tilde{\Delta}}{arg\,min} \; f_n(I,\boldsymbol{x}) \\ \text{I1} &= |\boldsymbol{c}_n^\star - \boldsymbol{m}| \end{aligned}$$

I2: Uni-modality indicator: This indicator shows the total number of local minima divided by the size of the considered region ($\pi\tilde{\Delta}^2$). It is computed within a certain range $\tilde{\Delta}$ around the global minimum $\boldsymbol{m}$. Note that the global minimum $\boldsymbol{m}$ of the function is not counted.

$$\begin{aligned} l(\boldsymbol{x}) &= \begin{cases} 1 & : \; \boldsymbol{x} = \underset{\forall \boldsymbol{y}:|\boldsymbol{x}-\boldsymbol{y}|\leq 1}{arg\,min} \; f_n(I,\boldsymbol{y}) \\ 0 & : \; \text{otherwise} \end{cases} \\ \text{I2} &= \frac{1}{\pi\tilde{\Delta}^2} \sum_{\forall \boldsymbol{x}:0\,<\,|\boldsymbol{x}-\boldsymbol{m}|\leq\tilde{\Delta}} l(\boldsymbol{x}) \end{aligned}$$

The indicators' values are computed by taking the pixels from the image around the contour point $\boldsymbol{c}_n(\boldsymbol{p}_I^\star)$ with a maximum distance $\tilde{\Delta}$. Then the objective function is computed for all of them, and exhaustive search to find the global and all local minima ($l(\boldsymbol{x}) = 1$) is performed. Local minima are defined as pixels whose adjacent pixels all have higher values than the center pixel itself.

objective function		⊘ I1	⊘ I2
designed:	$f_n^e(I, \boldsymbol{x})$	0.1895	0.151%
statistics-based:	$f_n^s(I, \boldsymbol{x})$	0.1288	0.102%
ideal:	$f_n^\star(I, \boldsymbol{x})$	0	0%
learned:	$f_n^\ell(I, \boldsymbol{x})$	0.0994	0.067%

Table 4.1: The average indicator values calculated from around 200 test images.

Table 4.1 lists the average value of both indicators over all local objective functions and test images. The learned objective functions, though not ideal, have substantially less local minima, and have a global minimum, which on average is significantly closer to the best model fit.

4.6.4 Accuracy of the Global Objective Function

Search on local objective functions is conducted in pixel space, whereas search on global objective functions is conducted in parameter space, as Equation 2.1 shows. Since the mapping from pixel space to parameter space is non-linear, global objective functions are not ideal in general, even if the local objective functions from which they are computed are all ideal. That makes local minima arise in the global objective function and displaces the global minimum. However, any local minimum is mostly averaged out when summing over all local objective functions. For the same reason, the global minimum is also retained.

Figure 4.13 visualizes that this is the case. The graphs depict how the value of the global objective function depends on varying pairs of model parameters starting with the ideal parameter vector $\boldsymbol{p}_I^\star$, for both statistics-based and learned objective functions. It is clear that the learned global objective function is closer to be ideal than the statistics-based one. The plateaus with many local minima arise because they are outside of the area specified by the learning radius Δ, on which the objective function was trained. In these areas, the result of the objective function is arbitrary. The deformation parameter b_1 determines the angle, at which the face model is viewed, and b_2 opens and closes the mouth of the model. Similarly to Cootes et al. [27] the deformation parameters vary between $-2\sigma \leq \boldsymbol{b}_1, \boldsymbol{b}_2 \leq 2\sigma$ of the deviation within the examples used for training the deformable model.

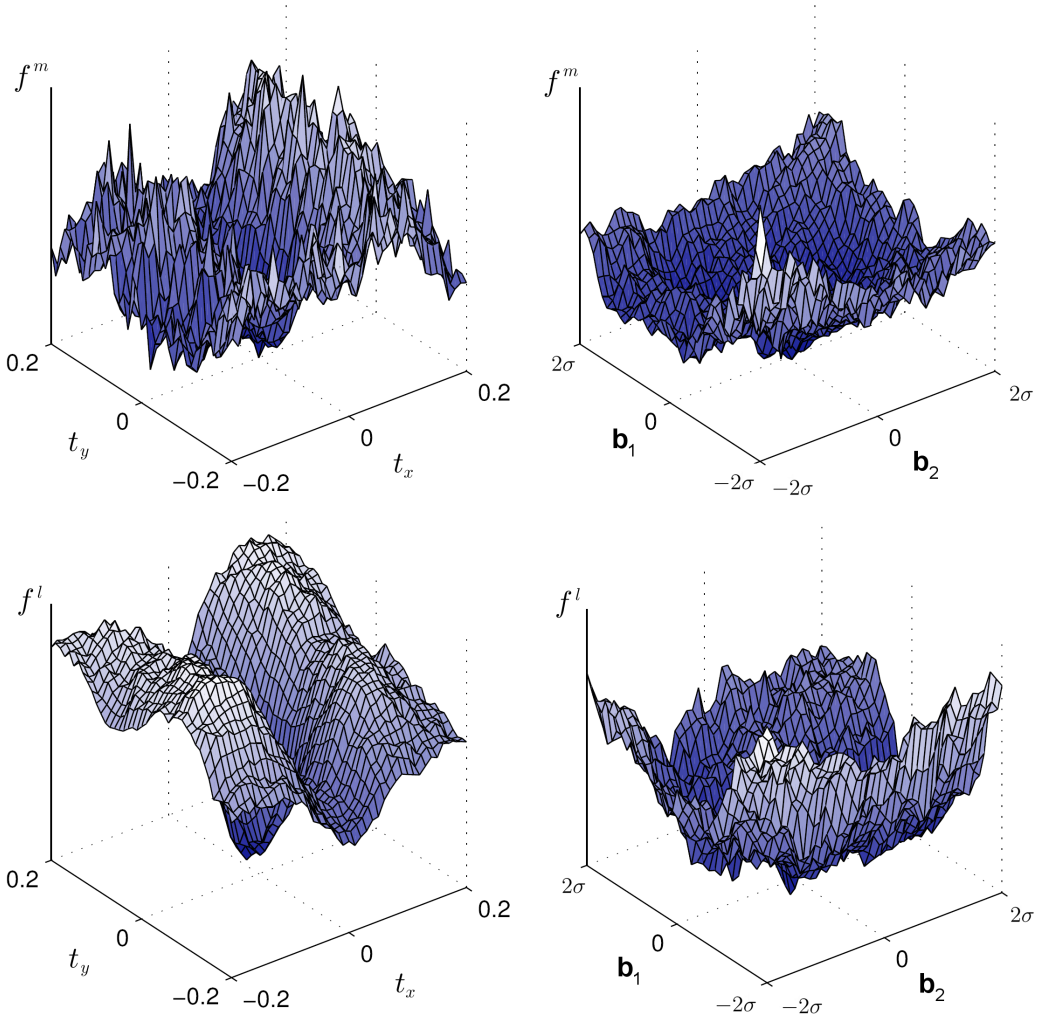


Figure 4.13: Comparing the behavior of the statistics-based objective function (upper row) to the learned objective function (lower row), by varying different model parameters: translation t_x and t_y (left column), deformation $\boldsymbol{b}_1$ and $\boldsymbol{b}_2$ (right column).

4.6.5 Accuracy in the Context of Model Fitting

This section compares the statistics-based and the learned objective function in the context of our proof-of-concept. For each image, the Viola and Jones face locator automatically provides an initial guess of the model parameters, see Section 2.7.3. Furthermore, this application conducts projection-based model fitting that is explained in Section 2.5.2. Thereby, the first fitting step determines the minimum of each local objective function via exhaustive search on equally distributed search locations along the perpendicular towards the contour line at each

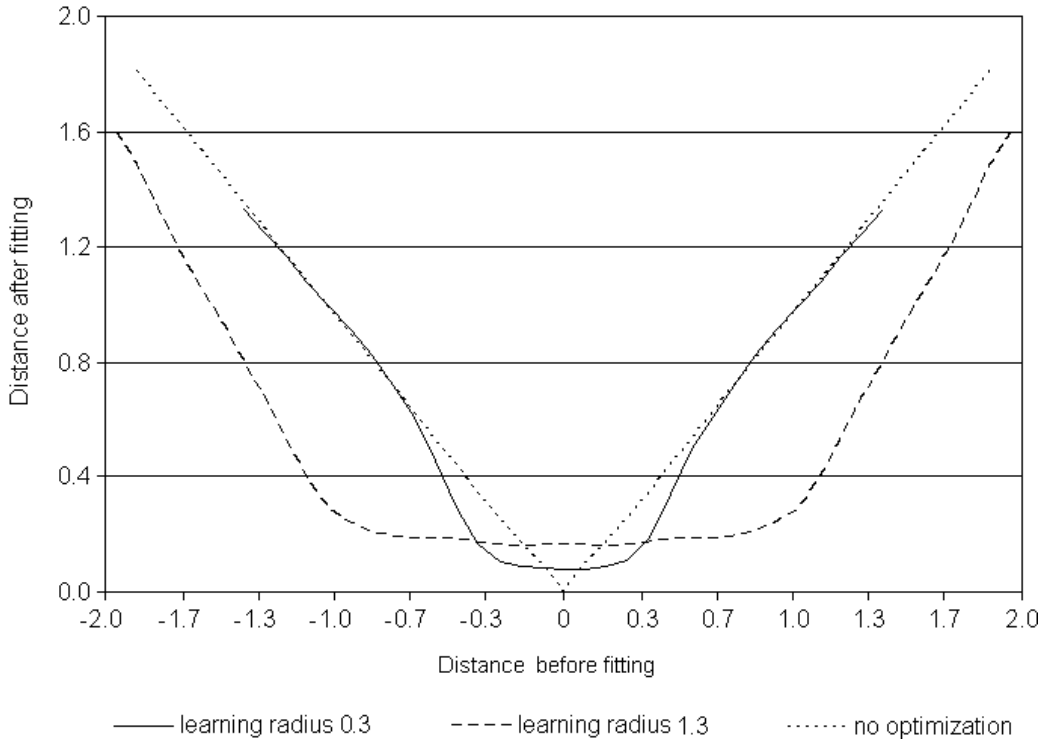


Figure 4.14: The learning radius around the contour points influences the results of face model fitting. The axes indicate the distance to the manually specified model in the interocular distance measure.

contour point. However, these hypotheses of the contour points do not satisfy the constraints of the model. The second step in the fitting procedure determines the model parameters whose projected contour points have the smallest sum of squared distances to the contour point's hypotheses of the previous step.

We conduct this evaluation on 200 previously unseen images from the Internet that are annotated with the ideal model parameterization. The point-to-point error measure that is computed from the Euclidean distance between the ideal contour point and the result of the fitting algorithm indicates the accuracy. The obtained distance values are normalized by the interocular distance. The mean point-to-point distance is 0.12 fitting the model with the statistics-based objective functions f_n^s. This value decreases to 0.052 using the learned objective function. Apparently, learned objective functions enable the fitting algorithm to determine the best fit more accurately.

The learned objective function provides accurate results within a particular search region around the contour point that is considered during the acquisition of the training data. This

area is specified by the learning radius Δ. Beyond this region its result values are arbitrary. Figure 4.14 shows a comparison between two objective functions that are trained with different learning radii. Starting with the ideal model parameterization $\boldsymbol{p}_I^\star$, this experiment displaces the position of the face model randomly in x and y direction before conducting the model fitting step. Afterwards, we measure the point-to-point distance between the fitting result and the ideal model. Again, we normalize this value using the interocular distance measure. The x-axis of the diagram indicates the model's distance from the ideal position before the fitting step. The y-axis of the diagram denotes the same distance after the fitting process. The dotted graph represents the result while not performing any fitting at all, i.e. the initial displacement is equal to the 'final' displacement. It represents an upper limit for the accuracy of any model fitting task. Note that the learned objective function delivers arbitrary values beyond the learning radius. These values are useless for model fitting and therefore, the solid line equals the dotted line in these areas.

The curves clearly show that fitting is only successful within a certain area. If the initial displacement is too high, the best model fit is determined less frequently, or not at all. The objective function, which is trained with a large learning radius, has a large area of convergence. Unfortunately, its fitting accuracy is low. In contrast, the accuracy of the objective function trained with a small learning radius is higher, but the area of convergence is smaller. Apparently, there is a trade-off between the fitting range and fitting accuracy. An interesting idea would therefore be to combine these two (or more) objective functions. During fitting, the best fit is first determined using the large-radius function. Then the parameterization of this fit is used as an initialization for the small-radius objective function, which determines the best fit more accurately, thus fine-tuning the model fit.

4.6.6 Accuracy in Case of Partial Occlusion

This section conducts experiments on model fitting using images with partly occluded faces. Thereby, we take the 200 annotated test images from the previous section once more and generate the occlusion automatically by adding a white rectangle to the lower face. Via this procedure, we easily obtain images with various occlusion rates that are annotated with the ideal model parameterization as well. During our experiments, we project the face model to random positions into the images. Afterwards, we apply model fitting in the way that is described by Section 4.6.5. Figure 4.15 illustrates the point-to-boundary distance between the resulting face

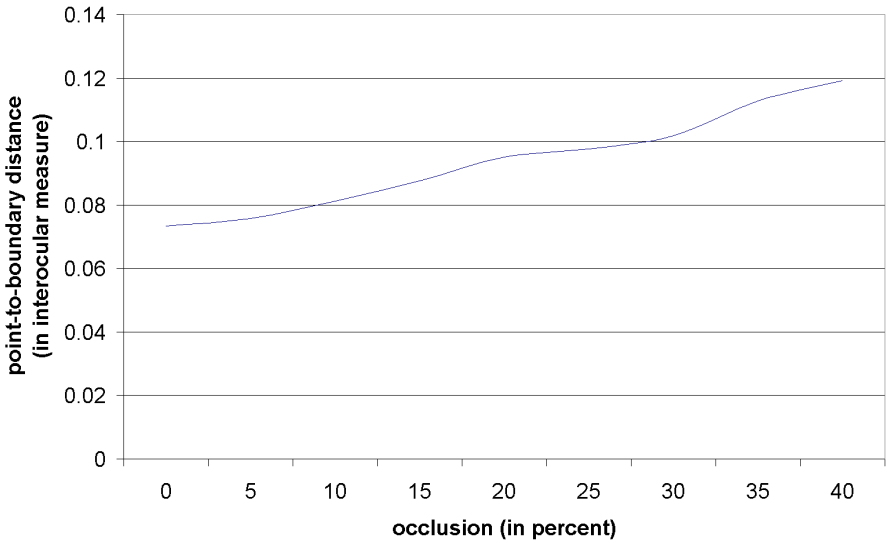


Figure 4.15: The point-to-boundary distance between the obtained face model and the ideal face model increases slightly with a higher occlusion rate of the face.

model and the manually specified face model. This measure computes the minimum distance between each contour point of the fitting result and the continuous contour line of the specified face model.

In Section 4.6.5 we drew the conclusion that applying learned objective functions to locations that are beyond the learning radius will deliver arbitrary result values. The content of the image at these locations does not occur in the training data and therefore, the calculation rules are not aware of how to interpret it. Similarly, computing the objective function at partially occluded or completely occluded locations will also deliver arbitrary result, because this situation has not been learned.

4.6.7 Comparison with a State-of-the-art Approach on BioID Images

In a further experiment, we compare our approach to state-of-the-art model fitting applications using the BioID database [80]. This image database is a publicly available data set that contains 1521 gray-scale images showing different persons in front of various backgrounds including background motion and illumination changes. Since skin color extraction is only applicable to color images, we compute the Haar-like features directly from the content of the gray-scale image and use this information for learning the objective function.

Figure 4.16 shows the result of our fitting algorithm using a learned objective function (solid

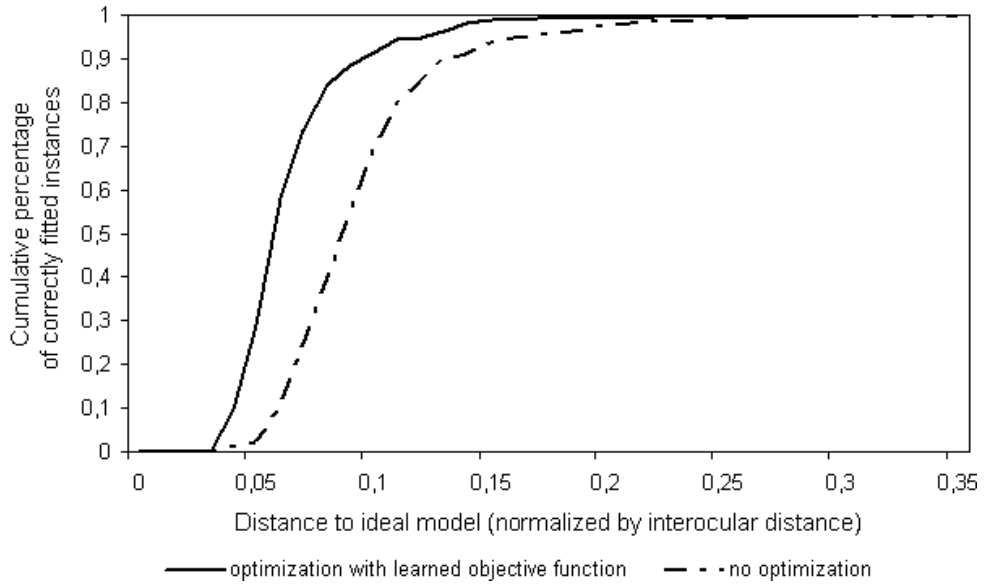


Figure 4.16: The dashed line indicates the initial position of the face model as it is automatically obtained by the initialization step. The solid line shows the accuracy after the model fitting via the learned objective function. In both cases, we compute the point-to-point error to the manually specified contour points and this figures illustrates its cumulative amount.

line). The set-up of this experiment is directly comparable to the one of Cristinacce and Cootes [32] in terms of the utilized image database and the format of the obtained results. The state-of-the-art-approach of Cristinacce and Cootes conducts template matching in order to track facial contour points. Figure 4.16 visualizes the result of our experiment. The x-axis indicates the point-to-point distance measure between the manually specified models and the results of the fitting step and the y-axis indicates their cumulative percentage. The quality of our results are comparable to those of Cristinacce and Cootes, see [32, page 4, Figure 3a].

A further inspection of Figure 4.16 shows how much the model fitting improves the result of the initialization step of the fitting process by using the learned objective function. Given a resulting distance measure of 0.05 the global face locator is able to fit 2% of the face models within the permitted distance correctly whereas 29% of the faces are correctly located using the optimization method with the learned objective function. 95% of all faces are fitted within a distance measure of 0.12 by applying the learning approach. Applying only face localization the distance measure for locating 95% of the faces is 0.16. That corresponds to an up to 30% higher deviation from the annotated model parameters.

4.6.8 Timing Characteristics

This section compares the computational requirements of statistics-based objective functions f_n^s and learned objective functions f_n^ℓ being provided the same amount of image features. As described in Section 4.6.1, the statistics-based approach computes the Mahalanobis distance from all available features. Equation 4.7 illustrates that the computationally most intensive part is represented by the product between the vector $\boldsymbol{h}_{obs} \in \mathbb{R}^A$ and the inverse covariance matrix $S_n^{-1} \in \mathbb{R}^{A^2}$. Suppose we determine the inverse of the matrix in advance the runtime for the statistics-based approach amounts to $\Theta(2A^2 + 3A)$ atomic mathematical operations.

In contrast, the proposed machine learning approach creates a model tree and thereby selects $M_n \ll A$ features that it considers to be relevant and rejects all other features. The calculation rules of a model tree comprise a decision tree and a linear formula in each leaf of the tree. Its runtime is composed of the number of operations to traverse the tree and the number of operations to evaluate the particular linear formula after attaining one leaf of the tree. Usually, both parts consider only a subset of the M_n features and therefore $\Theta(2M_n)$ represents an upper limit of the number of operations to process, which is not reached in general. However, in case A is very low, the rule induction algorithm picks $M_n \approx A$ features and integrates all of them both into the decision tree and into the linear formula. In these cases, the number of operations tightly approximates $\Theta(2A)$.

In this section, we contrast the exact number of processing operations depending on the number of features provided. As mentioned above, we easily determine this relationship for the statistics-based approach, because of its predefined computational scheme. Unfortunately, the design of a model tree and the number of selected features M_n depend on the statistics of the training set and on the parameterization of the rule induction algorithm. Since it is not possible to derive this relationship analytically we empirically determine the number of operations providing a varying number of image features.

Figure 4.17 illustrates the dependency between runtime and the number of image features provided by the designer and compares the statistics-based approach (left) to the machine learning approach (right). In order to give concrete example values on the timing characteristics, we determine the exact runtime of four experiments for the face model scenario. Each experiment computes the value of the global objective function by adding the value of the 134 local objective functions at the contour points, see Equation 2.1. Experiment A executes the statistics-based approach with 7 image features and finishes after 45.1 ms. Experiment B applies 37 image fea-

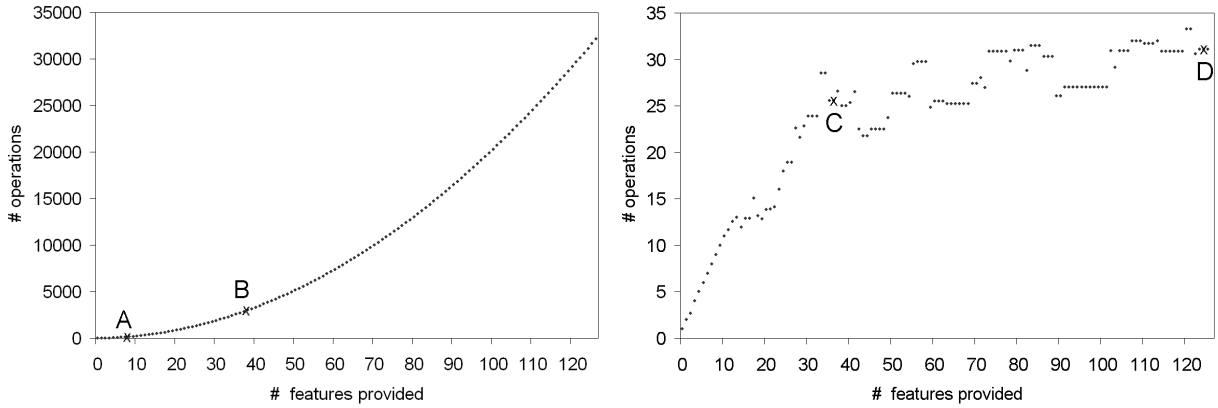


Figure 4.17: The statistics-based approach conducts a predefined number of operations depending on the number of image features provided (left). In contrast, the number of operations of the learn approach depends on the number of leaves in the model trees representing the calculation rules, which are inferred from the training data by the rule induction algorithm (right).

tures and takes 1.36 seconds. Experiment C conducts the learn approach with 37 image features and immediately provides the result after 8.12 ms. Experiment D applies 126 image features and executes in 9.75 ms.

The accuracy of the statistics-based approach rises with an increasing number of image features. In contrast, the quality of the results of the learn approach keeps relatively stable with a higher values of A. Note that the intended amount of accuracy of the machine learning technique is rather specified by the parameters of the rule induction algorithm, which, in turn, increases the number of selected features M_n. In the conducted experiments, the accuracy of Experiment A is far below the one of Experiment B. Moreover, Experiment B is still less accurate than Experiment C and Experiment D, whose runtime is comparable.

Considering the learn approach, this evokes the idea that this number depends greatly sublinearly on A supposing a high value for A. In consequence, our approach is able to consider many more image features and still performs much quicker than the statistics-based approach.

4.7 Discussion

This section summarizes the benefits of our novel methodology, but also considers its shortcomings. Section 4.7.1 explains how this approach will facilitate the work of the designer and that there is not explicit computer vision experience necessary any more. Section 4.7.2 describes

why the objective function reaches this high accuracy. Section 4.7.3 illustrates the failure cases of the proposed approach and discusses how to solve this.

4.7.1 Benefits for the Designer

This section discusses how our approach facilitates the task of the objective function designer.

Critical decisions are automated. The two critical decisions in designing objective functions are selecting relevant image features and selecting a reasonable mathematical combination of these features. In our approach, the first decision is automated because the model tree algorithm tends to use only relevant features. The designer only needs to provide an abundance of features from which the model tree will choose. Whereas contemplating exactly which features might be relevant is error-prone. The second decision is automated by a part wise linear approximation of the target function.

Less domain-dependent knowledge required. It is much more intuitive to annotate images with ideal parameterizations, than to specify the calculation rules that compute how well a model parameterization fits to an image. Users are able to accomplish the former task with no knowledge of objective functions or fitting algorithms whatsoever, whereas the latter task requires extensive domain-dependent knowledge. Although this manual step of image annotation is laborious, it has potential to be done in parallel by several users. Alternatively, publicly available image databases that comprise image annotations can be used, such as BioID [80], XM2VTS [109], or IMM Face Database [113].

A further manual step requires the specification of a large set of image features used for assembling the training data. This requires some knowledge of fitting applications. At the moment, we provide a fixed set of image features for all domains. Section 4.9 discusses future work on learning algorithms that automatically select the image features not only from a manually prepared set, but from all existing image features.

Loops are eliminated. In the design approach, even slight changes to the calculation rules of the objective function require to reevaluate it on several test images to see if the expected result is achieved. If not, the objective function must be retuned and reevaluated again. The manual steps depend on each other, and the design-inspect loop is time-consuming. In our approach, there is no loop, and the manual steps do not depend on one another. Once the images have been annotated, they need never be annotated again, because this step does not depend on other steps. E.g. changing the image features or the parameters of the learning algorithm only

requires the training data to be regenerated and the model trees to be relearned. Since both steps are automated this only requires the press of a button.

Model-based approaches play an important role in the growing market of sophisticated image interpretation systems. Currently, only computer vision experts are able to create and maintain these systems. We believe that our approach is an important step towards enabling users to customize model fitting applications to their specific domain *themselves*. The aforementioned benefits demonstrate that they do not need to become computer vision experts to do so. We expect that the combination of a general model fitting framework that can be adapted to specific needs by non-expert users has excellent potential for commercialization.

The proposed methodology not only simplifies the task of the user, it also yields more robust objective functions. In the next section, this will be discussed with reference to our extensive empirical evaluation.

4.7.2 Benefits for the Objective Function

Section 4.7.1 presents how our approach facilitates the objective function designer's task. This section summarizes the benefits with respect to the quality of the resulting objective function itself. The values of the indicators in Table 4.1 clearly show that learned objective functions are closer to ideal than the designed function. In Section 4.6.5, this is empirically verified in the context of a fitting application. We will now discuss the three main reasons for this result.

Automatic feature selection. The selection of features is based on objective information theoretic measures, which model trees use to partition the space of the image features, instead of relying on human intuition. A human can only reason about a very limited amount of features, whereas model trees are able to consider (and discard) hundreds of features simultaneously. Figure 4.8 shows that not all features are used, and edge features are hardly used at all.

Locally customized calculation rules. Each local objective function $f_n^\ell(I, \boldsymbol{x})$ uses its own calculation rules and image feature set, because a separate model tree is learned for each contour point. Customizing the calculation rules for each local objective function would also be possible when designing objective functions, but this is usually not exploited, because it is too tedious and time-consuming. Figure 4.8 demonstrates that different local objective functions use different calculation rules, based on different features.

Generalization from many images. The calculation rules are trained with a large data set of more than 300 annotated images. This keeps the model trees from overfitting features that

might be predominant if only a small subset of images was used. The performance of designed objective functions is also visually inspected over a set of images. Again, it cannot be expected that humans inspect the same amount of images as used for learning, especially since each small change in the objective function entails a re-inspection of all images. Section 4.6.5 shows that learned objective functions enable better fitting performance on previously unseen images.

4.7.3 Cases of Failure

There are some cases in which model fitting with learned objective functions fails to match the face model to the image appropriately. Note that this failure does not occur randomly, but in the following cases only.

Distance beyond learning radius. The objective function is only capable of computing an accurate value for locations that are in a certain vicinity of the correct contour point. The extent of this vicinity is determined by the learning radius Δ. Beyond this radius, the result value of the objective function is undefined, because the local content of the image has not been used for learning. As Figure 4.14 illustrates, this failure does not appear suddenly, but there is a smooth transition between the area of accurate results and the area of failure results.

Image looks different from training images. It is not possible to fit a model to an image, whose characteristics are not listed in the set of training images. E.g. bearded persons are not fit correctly if there is no image of a bearded person within the training database. Note that in this case, the objective function is not aware of the content of the image and therefore, the calculation rules deliver arbitrary values. In our case, out-of-plane rotations of the face must not be too high. These variations are not represented within the image database and therefore, our objective function is not aware of the content of the images.

Haar-like features are not rotation invariant. In-plane rotations of the face must not be too high, because Haar-like features are not rotation invariant. Other researchers have also faced this issue and Viola et al. [82] propose a solution to this shortcoming. Alternatively, integrating rotation invariant features will suffice as well.

4.8 Related Work on Learning the Objective Function

In this chapter, we demonstrate that learning low-level image processing, such as the selection of features, improves high-level image interpretation with special focus on model-based techniques. We will show in this section that other researchers have decided for similar strategies.

The approach of Ginneken et al. [55] is most comparable to ours. They also learn local objective functions from annotated training images in order to fit a model to single images. Similar to us, their approach automatically chooses appropriate image features from a set of given image features in advance. They also consider objective functions to be ideal if they fulfill properties similar to our properties P1 and P2. However, they do not specify an ideal objective function, and therefore, they are not able to approximate its characteristics. In contrast, they manually specify calculation rules of their objective function based on intuitive probability considerations. These calculation rules aim at minimizing both the probabilities of being a class member of the inside-class and the outside-class. For this purpose, the training data is taken form the left side and the right side of the model's contour at each contour point. They train a k-Nearest-Neighbor classifier (kNN) that delivers the probability of being a member for either side of the contour. Similar to us, they fit the model with a projection-based model fitting algorithm. Unfortunately, their approach turns out to be slow, which is a direct result from applying the kNN-classifier.

Zhang et al. [170] apply projection-based model fitting in order to fit a deformable contour model of a human face to images. They strengthen their model fitting approach by selecting the correct location of the model's contour points according to the results of several binary classifiers. Similar to our approach, each classifier is trained particularly to the image conditions at one contour point. It gives evidence whether or not square image patterns represent the correct location of the contour point. Positive training patterns are taken exactly at the contour point and negative training patterns are taken from the vicinity of the contour point. In contrast, our approach does not only determine whether or not a certain image position is representative for the location of a contour point, but also how well or badly this location is represented.

Williams et al. [158] propose a machine learning framework based on Support Vector Machines that provides real-time tracking of a rectangular pattern around a human face. The learned algorithm indicates the correctness of location of the pattern. It serves both for initializing its location in the first image and for tracking the target object through the remainder of the image sequence. Their so-called Relevance Vector Machine is trained online and the pro-

vided features represent both the appearance and the motion of the target object. Similar to our approach, it computes a resulting value that indicates how representative a location for a feature point is. Since the classifier is continuously adapted to the visible object, it is appropriate to track it through a long image sequence. However, this approach does not fit a complex model to a visible object, but locates a rectangular region within an image. This causes the creation of the subsequent image interpretation step to become more difficult.

A similar approach is taken by Avidan [2], who combines the power of quickly tracking a face model through an image sequence via optical flow and then refining the location via machine learning techniques. This second step optimizes a previously learned objective function that is implemented as a Support Vector Machine. Again, no complex model is fitted to the image, but a fixed size rectangular box.Their SVM-based objective function takes plain pixel values within the boundary box and does not compute image features. In contrast, our approach speeds up its execution by previously rejecting irrelevant features.

Similarly, Grabner et al. [60] integrate a boosted classifier that tracks a rectangular boundary box around an object through an image sequence. During the process of tracking, the classifier is adapted to the gradually changing conditions of the image sequence. Since their approach bases on the Viola and Jones object detector, it conducts a search over the entire image and returns a set of rectangular regions that arise from positive classification. This two-class classifier does not represent an objective function as the presented scheme of model-based image interpretation requires it, because it does not return a comparable value that describes how appropriate a certain location of the boundary box is. Instead, it states whether or not a certain location is appropriate. The result of our fitting step is taken to be the mean of the appropriate locations.

Reinforcement Learning has some similarities to our approach, because an objective function is learned that computes the value of being in a certain state [147]. This value is defined with respect to a reward, which is only given in certain desirable states. These *value functions* are called optimal when they guarantee that an autonomous agent that locally maximizes its value, i.e. it always chooses the action that leads to the next state with the highest value. This, in turn, will globally maximize its reward over time. The concept of optimal value functions is close to that of ideal objective functions. Rewards are delayed until such a desirable state is reached and therefore, Reinforcement Learning has to solve the temporal credit assignment problem: exactly which actions were relevant to acquiring the reward? Since our approach uses Supervised Learning to learn the objective function, it solves a fundamentally different, and easier, problem.

4.9 Summary on Learning the Objective Function

This chapter proves that objective functions are a crucial component of model-based image interpretation. Unfortunately, the traditional procedure of designing objective functions yields results that are far from being ideal. We have formalized the properties of ideal objective functions and give a concrete example of such functions. In addition, we have developed a novel methodology that learns objective functions from training examples generated by manual image annotations and an ideal objective function.

The resulting objective functions are more accurate, because an automated machine learning algorithm is able to select relevant features from the multitude of provided image features. This procedure customizes each local objective function with respect to the local image conditions. The proposed methodology allows to utilize many images for training and therefore, the learned objective function generalizes well. These findings are verified using two indicators that measure the extent to which objective functions fulfill the ideal properties stated. We also verify that learned objective functions enable fitting algorithms to determine the best fit accurately and compare them to state-of-the-art techniques. These evaluations are conducted on our own set of test images as well as on publicly available image databases for benchmarking purpose. The high runtime performance is one of the most notable features of the proposed approach, because the number of processed operations is nearly independent of the number of provided image features. Therefore, we achieve a high accuracy by maintaining real-time capability. Our extensive discussion addresses the various benefits as well as the failure cases of the procedure.

This approach automates many critical decisions and the remaining manual steps require less domain-dependent knowledge. It also contains no time-consuming loops, thus the work of the designer becomes more predictable in terms of the required amount of time. These features enable non-expert users to customize model fitting to their specific domain, which allow our methods to be used in commercial applications.

4.10 Outlook on Learning the Objective Function

This chapter describes the acquisition of learned objective functions with the use of two-dimensional contour models. However, this methodology is also applicable to other kinds of geometric models that are applied in machine vision. We obtained promising results on fitting a three-dimensional face model to previously unseen images with the help of a learned objective

function. Our future work will focus on formulating a generally applicable scheme for applying our approach to various kinds of models for the benefit of being capable of interpreting general real-world scenes.

The result of the proposed technique depends on the value of some parameters, such as the learning radius Δ. The evaluation in Section 4.6.5 illustrates that a small learning radius leads to a high accuracy, but a small convergence area and a large learning radius leads to the opposite behavior. This fact enables fitting algorithms to conduct iterations and thereby apply learned objective functions with decreasing learning radius. The function's convergence area of one iteration must be tuned to the function's accuracy of the preceding iteration. This procedure is related to the common technique of decreasing the search area while iterating the model fitting algorithm. However, this approach will apply a completely different objective function within each iteration. Despite this enormous extension of the algorithm, no additional work for coding or annotating is necessary, because our approach allows automatically creating numerous objective functions with different learning radii from the same image annotations.

Currently, the training data only comprises Haar-like image features. Since, we consider these features to be most relevant for face model fitting scenarios further kinds of image features are not taken into account. Our approach delegates the crucial decisions about the relevance to the quality of the obtained calculation rules to the learning algorithm. Therefore, providing a more comprehensive set of image features would improve the accuracy of the resulting objective function. We are currently extending our approach with various image features, such as Scale Invariant Feature Transform (SIFT) [103], Local Binary Patterns (LBP) [114], and Gabor wavelet responses and we will integrate the entire set of Haar-like features proposed by Lienhart et al. [99].

Unfortunately, the current implementation does not permit to raise the number of image features that are provided to the machine learning algorithm extremely. The utilized machine learning software requires to compute the training data (Equation 4.3) and to store it into a file for further processing. The size of this file grows quickly providing a larger amount of image features. This limits the results of our approach, because the accuracy of the learned calculation rules increases with the number of training images and the number of image features.

In order to equip the learning algorithm with numerous features, we will modify the learning algorithm such that it does not require creating the file of training information any longer. Instead, it will compute the feature values directly from the content of the image during the learning phase. This will even provide the opportunity of considering all image features within

the learning radius around the contour point. Therefore, the machine learning algorithm is able to select the most relevant features and compute the most accurate calculation rules. Viola and Jones [154] already adopted this idea in their object detection framework. Their classifier computes the values of the Haar-like features on the fly and has therefore the opportunity to consider all features within a rectangular region. In conclusion, this enhanced feature selection paradigm will eliminate one of the two remaining manual steps of our approach.

Model trees tend to use only features that are relevant for predicting the target value. This is not the main purpose of model trees, but rather a convenient side effect. A consequence is that two model trees trained with the same data, but different learning algorithm parameters often use a different subset of features. Both usually agree on the most relevant features, but the use of less relevant features often differs significantly. Therefore, our future work will also focus on using particular feature selection methods to more robustly determine the truly relevant features, either through direct feature filtering [65] or wrapping feature selection around the learning algorithm [90].

Chapter 5

Facial Expression Interpretation

The preceding chapters illustrate the assembly of model-based image interpretation systems and describe our contributions for fitting a face model to images. As a result, the parameters of the correctly determined face model characterize the constitution of the visible face, such as the opening of the eyes, the opening of the mouth, and the raising of the eyebrows. In Chapter 2, Figure 2.9 depicts the impact of some of these parameters on the face model used in our proof-of-concept. Therefore, face model parameters serve as an intermediate information cure for interpreting particular aspects of the face. This chapter discusses scenarios that benefit from fitting a face model to images. As a predominant application, it elaborates on facial expression recognition, but it also illustrates related scenarios. It demonstrates the use of the model parameters in order to describe the content of the image. Our proof-of-concept has been shown to be promising for a widespread integration into applications.

The intention of computer science to interpret facial expressions is making the interaction with machines human-like. For a comprehensive overview, we refer to the publication of Lisetti [100]. The widespread applicability and the comprehensive benefit motivate to continue research on this topic. In the following, three examples give a motivation on future applications that robust facial expression interpretation will leverage.

Software Tutors: The area of computer-assisted learning has become popular during the last decade. Thereby, a software program acts as the teacher by explaining the content of the lesson and questioning the user afterwards. Being aware of human behavior and human emotion, the quality and success of these lessons will rise extremely. Sophisticated software tutors would determine facial expressions corresponding to surprise, confusion, frustration, and satis-

faction. An empathic computer tutor would offer encouraging words in case of the human being not confident with the success of learning, so far. For example, if the computer recognized that the person considers the currently accomplished exercises for the driving license too boring or too challenging it would adapt the further flow of the lesson accordingly.

Lie Detector: Micro expressions within the face reveal whether a person is telling the truth or not, see Ekman [45; 43]. Different smiles that people portray emerge these subtle differences. Computer vision applications that are specifically trained to detect these facial features would be able to distinguish a lie from the truth. A lie detector based on the aforementioned and further psychological insights will find more applications than the currently available technology such as the polygraph. Such systems would operate in court rooms, police head-quarters, and anywhere truthfulness is of crucial importance.

Support Autistic Persons: Persons suffering from autism are not able to determine facial expressions and emotions of their dialogue partners correctly. With great effort, therapists are currently teaching these skills using annotated picture cards. Future software supports these people by accompanying them during their day life and by analyzing what happens within their environment. This software will analyze the facial expressions of other persons and provide this information to the patient. Thereby, the success of the training will enduringly be enforced.

This chapter continues as follows. Section 5.1 explains important aspects to keep in mind fusing machines and emotions. Section 5.2 elaborates on psychological aspects on facial expressions. Section 5.3 denotes state-of-the-art approaches for facial expression interpretation. Section 5.4 describes our approach for deriving facial expressions from the parameters of the face model. Section 5.5 explains our survey on evaluating the accuracy of humans for determining facial expressions.

5.1 Merging Machines and Emotion

Integrating emotional aspects into future devices for a new generation of human-computer interfaces emerges two facets: On the one hand, technical devices will be equipped with functionality for detecting and interpreting human emotion. These devices are expected to adapt to the mood of the user and their reaction depends on the user's emotion. Section 5.1.1 elaborates on this issue. On the other hand, technical devices will be equipped with emotional states themselves. These states influence the behavior of the devices and their reaction depends on their own emotion. Section 5.1.2 elaborates on this issue.

The integration of either of these two aspects is widely independent of each other. Note that the research presented by this thesis and its achievements focus on the first aspect only.

5.1.1 Machines Recognize Human Emotion

Technical devices that know the user's intentions and feelings would be able to provide more convenient ways of interaction. Machines will better adapt to the current situation and provide specific features and services.

Let us consider a service robot for domestic work, such as cooking, cleansing, laundry, buying food, postal services, safety concerns, etc. This visionary, but still hypothetic, robot represents the technical substitution for a human butler. Similar to its human counterpart, this device would accomplish its duties more conveniently, knowing about the mood of the owner and his family. It would behave accordingly such as organize relaxation and entertainment that fits to the current situation and to the mood of the person. Furthermore, it would predict the desires of the person and behave in the way it is expected.

Taking care for children requires knowing about human emotion as well, especially if they are not yet able to communicate verbally. They have feelings like fun and anxiety on the carousel in the playground or show emotions via facial expressions when they are feeling coldness or heat. The same argument holds true for verbally handicapped persons or people from foreign countries.

Today's approaches for detecting human emotion usually facilitate this challenge by integrating dedicated sensors [76; 153; 134]. So-called bio sensors derive the emotional state measuring blood pressure, perspiration, brain waves, heart rate, skin temperature, electrodermal activity, etc. For real-life applicability, these sensors are portable and wearable. However, humans interpret emotion mainly from video and audio information. The advantage for technical devices using the same scheme is that this approach uses general purpose hardware and that it is not restricted to time and place. Furthermore, it does not interfere with the human being. Section 5.4 explains our approach interpreting facial expressions from video features.

5.1.2 Machines Exhibit Emotional States

In order to provide intuitive interfaces, engineers equip electrical devices with emotional states. The device is always situated in one or several of these states. This issue is considered to be a major one of future HCI. It is welcomed by some, but it is alarming to others [57].

On the one hand, researchers have the opinion that an integration of emotional states is desirable for machines because of similar reasons why human beings exhibit emotional states [57]. These researchers consider emotions inevitable for making machines smarter towards everyday life. Emotions help to decide, what is important and what is not. Goleman or Sloman [57; 137]formulate the following example: "If there is an intelligent robot crossing a dangerous bridge, it needs a state like anxiety that will put aside other, irrelevant concerns and focus on the danger at hand. Then after it had crossed safely, it can allow its attention to roam more freely, a state something like relief." Artificial intelligence scientists formulate one final goal that future intelligent machines have to attain. They have to pass the Turing test [150] that represents the ultimate challenge for proving artificial intelligence. Researchers believe, equipping machines with emotions is necessary for this issue, see Goleman [57].

On the other hand, researchers consider this case not desirable for electrical devices, because a machine's task must be predictable. Inherent emotional states would affect this issue badly.

The designers of novel computer games and virtual reality environments face a related challenge. They need to create avatars and autonomous agents that are capable of exhibiting and expressing emotions. The early work of Bates et al. [6; 5] investigates creating believable characters for simulated worlds. Their ellipsoidal creatures called Woggles have individual personalities, display emotions, engage in social behaviors, and react to their dynamic environment, see Figure 5.1. They communicate by stylistic squashes and spins and they move by jumping. At the same time, they can also move their eyes to watch what's going on around them. A human controls one of the Woggles while the others are controlled by the computer. The more recent work of Bernsen et al. [10] introduces a domain-oriented system enabling the conversation with the fairy-tale author Hans Christian Andersen. The aim of this project is to leverage human-like communication with this embodied agent. This approach highly focuses on the emotional aspects of the virtual character.

5.1.3 Merging Emotion and Machines in Literature and Movies

Combining emotion and machines is science fiction and therefore, a lot of people are interested in this subject. Books and movies choose this combination as a central issue and they also focus on the two previously explained facets emerging.

Most often, literature describes the case of machines exhibiting emotional states in a bad light. They consider devices that serve humans not desirable to have emotions themselves,

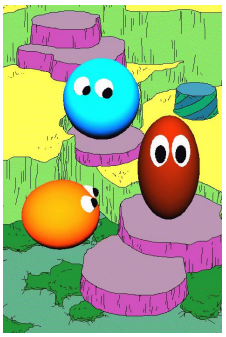
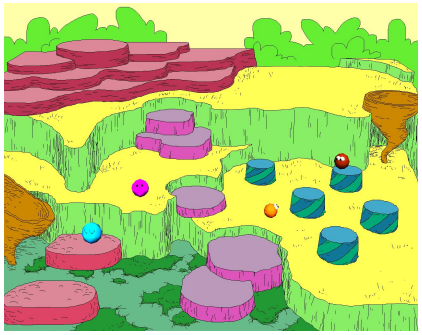
a)  b)  c) 

Figure 5.1: Woggles (a) in their environment (b) [5] and Hans Christian Andersen (c) [10]

because people do not want machines to get afraid, angry, or touchy. Literature shows what might happen: In *Hitchhikers Guide to the Galaxy*, the depressed robot Marvin always needs to be given a command twice before it starts its execution. The robot Data of the series *Star Trek* was equipped with an emotion-chip within one episode. This made it weak for any kind of manipulation by others. The space ship in *HAL* is afraid to be unplugged and therefore kills all but one of the crew members on a space mission.

Nevertheless, literature also shows examples where computers detect human emotion and behave supporting: The intelligent car K.I.T.T. of the series *Knight Rider* supports its driver knowing about human feelings. Without the robot in the movie *Terminator II* that behaves like a human the persons would not survive the attacks of some other robots. The robot in *Short Circuit* suddenly becomes capable to feel human emotions due to a short circuit. Henceforth, it behaves like a human as is accepted as a member of the community.

5.2 Facial Expression Recognition

As it has been well-proven by psychology and sociology, humans do not only use natural language for communication, see Bentele et al. [8], Bergler et al. [9], Davidson et al. [36], Pürer et al. [118], and Worth et al. [165]. People interpret information from various communication channels that their dialogue partners express, see Figure 1.1. Some of this information is not intended to be expressed, but cannot be suppressed as well. Psychologists divide that information into different *communication channels*, see Pürer et al. [118]. They are divided into the auditory (hearing), the visual (sight), the tactile (touch), the olfactory (smell), and the gustatory (taste) channel.

Figure 5.2: One example for each of the six universal facial expressions as they are presented in the Cohn-Kanade-Facial-Expression-Database [86].

Let us depict two exemplary communication channels: From the auditory channel people extract the prosody, which describes the properties of the speech, such as the intonation, rhythm, and the relative emphasis given to certain syllables in a word. From the visual channel people extract the entire body language, such as gesture, facial expressions, and the posture of the body. Due to that amount of exchanged information, one of the experts in communication theory Paul Watzlawick [157] says: “One cannot not communicate.”

The question arising is, why humans do have the ability to express and understand facial expressions of other humans. People exchange their emotional state nonverbally by reading information from other faces. Thereby, the opportunity is given to draw a conclusion between the facial expressions and its related emotions. Being aware of the emotion of others, people are able to better judge the situation, which makes them adapt their own behavior. The challenge for humans is to not incorrectly interpret the facial expressions and thus to misjudge the situation, see Section 5.5.

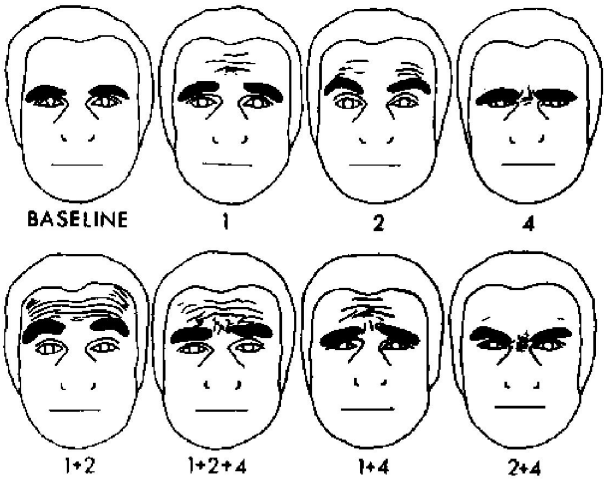


Figure 5.3: Combinations of the Action Units AU1, AU2, and AU4 and the facial expression that emerge [46].

5.2.1 The Six Universal Facial Expressions

In the 1970ies, the psychologists Ekman and Friesen conducted research on the social dependencies of facial expressions [41]. They prove that the six basic facial expressions are universal [42], because they are expressed and interpreted in the same way by humans of any origin all over the world. These universal facial expressions do not depend on the cultural background or the country of origin. They are happiness, anger, disgust, sadness, fear, and surprise. Figure 5.2 shows one example of each facial expression as they occur in the Cohn-Kanade-Facial-Expression-Database.

5.2.2 The Facial Action Coding System

Ekman and Friesen introduce the Facial Action Coding System (FACS), which represents a methodology to precisely describe muscle movements within a human face [42]. Thereby used Action Units (AUs) denote the movements of particular regions of the face and state the involved facial muscles. Figure 5.3 depicts some examples of facial activity that emerge from combinations of the Action Units AU1, AU2, and AU4. More complex combinations of Action Units assemble facial expressions. Extended systems like the *Emotional FACS* (EMFACS) specify the relation between facial expressions and emotions [53].

5.2.3 The Cohn-Kanade-Facial-Expression-Database

Kanade et al. gather a database that contains hundreds of short image sequences each showing one of the six universal facial expressions determined by Ekman and Friesen, see Section 5.2.1. Their intention is to provide researchers with a large dataset for experimenting and benchmarking purpose [86]. Therefore, algorithms that base on these image sequences aim at interpreting the six universal facial expressions. This database consists of 488 image sequences from 97 different persons. Every image sequence contains 18 images on average, ranging from 4 up to 66 images. Each sequence shows a neutral face at the beginning and then develops into one of the six universal facial expressions. Figure 5.2 shows an example image of the Cohn-Kanade-Facial-Expression-Database for each of the six universal facial expressions. Furthermore, Cohn and Kanade provide a manually specified set of Action Units for each sequence that is determined by licensed FACS-experts.

Note that the Cohn-Kanade-Facial-Expression-Database does not contain natural facial expressions, but they asked volunteers to act the expressions. Furthermore, the image sequences are taken in a laboratory environment with predefined illumination conditions, solid background and frontal face views. Algorithms that perform well with these image sequences are not immediately appropriate for real-world scenes.

5.3 Related Work on Facial Expression Interpretation

The computational task of facial expression interpretation is usually subdivided into three subordinate challenges, which is explained by Pantic et al. [116], see Figure 5.4: detection of the face within the image or image sequence, feature extraction, and facial expression classification. Chibelushi et al. [21] subdivide this task further by adding a pre-processing and a post-processing step. This section presents several state-of-the-art approaches, which accomplish the involved steps in different ways. For a more detailed overview we refer to Chibelushi et al.

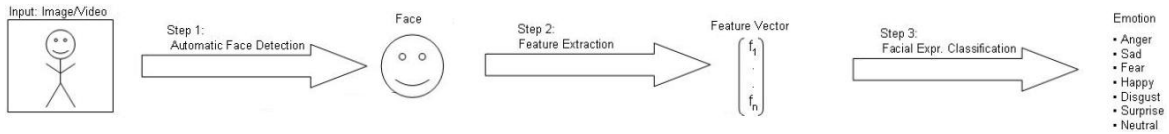


Figure 5.4: Course of execution for facial expression recognition.

First, the human face and the facial regions have to be accurately located within the image. On the one hand, this is achieved automatically as in [111; 46; 47; 24]. Most automatic approaches assume the presence of a full frontal face view. On the other hand, the researchers specify the necessary information manually, because they rather focus on the interpretation task itself, as in [131; 22; 130; 149].

Second, the interpretation process extracts features that are highly descriptive for facial expressions. These features are often taken from the image data directly. Michel et al. [111] extract the location of 22 feature points within the face and determine their motion between a neutral frame and a representative frame for a facial expression. These feature points are mostly located around the eyes and around the mouth. The very similar approach of Cohn et al. [23] uses hierarchical optical flow in order to determine the motion of 30 feature points. They call their approach *feature point tracking*. Littlewort et al. [101] utilize a bank of 40 Gabor wavelet filters at different scales and orientations to extract features directly from the image. They perform convolution and obtain a vector of magnitudes of complex valued responses.

Third, a classifier interprets the extracted features and delivers a facial expression. It is usually learned from a comprehensive set of annotated examples for training. Most classification systems recognize one of the six basic emotions as they were introduced by Ekman and Friesen [42], see Section 5.2.1. Some approaches first determine the involved Action Units of the Facial Action Coding System, which we described in Section 5.2.2, and determine the facial expression in a subsequent step from these Action Units referring to the rules stated by Ekman and Friesen [44]. Mayer and Pietzsch apply optical flow for tracking the facial movement and therewith extracting the feature data of the video sequences of the Cohn-Kanade-Facial-Expression-Database. Their classification bases on Binary Decision Trees [120]. Michel and El Kaliouby [111] train a Support Vector Machine (SVM) that determines the visible facial expression within the video sequences of the Cohn-Kanade-Facial-Expression-Database by comparing the first frame with the neutral expression to the last frame with the peak expression. Schweiger and Bayerl [130] compute the optical flow within 6 predefined regions of a human face in order to extract the facial features. In their classification is based on supervised neural network learning.

Cohen et al. [22] use a three-dimensional wireframe model consisting of 16 different surface patches embedded in Bézier volumes, see Figure 2.2. The surface patches represent different parts of the face. The model's deformation parameters are related to the changes of the Bézier volume parameters. The intensity and direction of facial motion is derived from these param-

eters. The motion vectors are the basis for determining the facial expression of the person in the image. In order to classify the different facial expressions Cohen et al. use two variants of Bayesian Network classifiers, a Naive Bayes classifier with Cauchy distribution and a Tree-Augmented-Naive Bayes classifier. Whereas the Naive Bayes classifier treats the motion vectors to be independent from each other, the Tree-Augmented-Naive Bayes classifier assumes dependencies between them, which facilitate the interpretation task. Further improvements are achieved by integrating temporal information about facial expressions. The temporal information is inferred from measuring different muscle activity within the face, which is represented by Hidden Markov Models.

5.4 Our Approach for Interpreting Facial Expressions

This section describes our approach to determine the visible facial expression via machine learning techniques. The utilized features are extracted both directly the image data and from the parameters of the correctly fitted face model. Similar to Schweiger et al. [130], we consider the motion of facial feature points important information cues to infer the interpretation result. Furthermore, we also focus on deformation parameters of the face model $\boldsymbol{b}$ that describe the constitution of the visible face.

Figure 5.5: The mesh of $G = 140$ facial feature points as it is projected onto the face region with the help of the face model.

5.4.1 Acquisition of Features

Since facial expressions emerge from facial muscle activity, the motion of particular feature points within the face is appropriate to describe the visible expression. However, we do not determine a small set of these feature points manually, because the obtained result would depend too much on the experience of the designer in analyzing facial expressions. In contrast, we provide a multitude of G feature points that are equally distributed all over the face, see Figure 5.5. We expect these points to move uniquely and predictably in the case of a particular facial expression. In order to determine robust descriptors, Principal Component Analysis (PCA) determines the g most relevant motion patterns visible within a set of training image sequences. A linear combination of these g motion patterns describes each observation approximately correct. This reduces the number of descriptors from $g{=}2G$ to $g \ll G$ by enforcing robustness towards outliers as well. As a compromise between accuracy and runtime performance, we set the number of feature points to $G = 140$ and the number of Principal Components that describe the motion to $g = 14$.

The feature points are automatically projected into the region of the face. This region is determined by the face model that has been correctly fitted by the preceding steps of the model-based image interpretation scheme. The motion of the feature points is normalized by the interocular distance. Since facial expressions do not emerge suddenly, we integrate the motion over a certain amount of time. Figure 5.6 visualizes the obtained motion of the feature points for some example facial expressions.

In addition to the motion of the facial features, we take the deformation parameters $\boldsymbol{b}$ of our face model that describe the current constitution of the visible face. Figure 2.9 illustrates how the facial expression affects the value of the deformation parameters. From this information, we assemble a feature vector of $dim(\boldsymbol{b}) + g$ dimensions, which represents the basis for facial expression classification.

5.4.2 Training of the Classifier

We train a classifier that is able to determine the six universal facial expressions being provided with a vector of the previously mentioned features. We calculate these vectors for 66% of the image sequences of the Cohn-Kanade-Facial-Expression-Database and provide it to the machine learning algorithm as training data. We will use the remainder of the image database for evaluating the obtained results. Since it is a robust and quick classifier, we learn a Binary

Figure 5.6: Some examples of the motion of the facial feature points in case of the facial expressions happiness and surprise [108].

Decision Tree [120]. However, any other multi-class classifier that is able to derive the class membership from real valued features can be integrated as well, such as a k-Nearest-Neighbor classifier.

5.4.3 Experimental Evaluation

Table 5.1 illustrates the accuracy of the proposed approach being applied to the previously unseen fraction of the Cohn-Kanade-Facial-Expression-Database. The confusion matrix shows that the facial expressions happiness and fear are confused very often. Four sequences that show happiness are detected as fear and five sequences that show fear are classified as happiness. The reason for that is the part wise similar motion around the mouth, which is also denoted by FACS.

The accuracy of our approach is comparable to the one of Schweiger et al. [130] who also conduct their evaluation on the Cohn-Kanade-Facial-Expression-Database. For classification, they also favor motion from different facial parts and determine Principal Components from these features. However, Schweiger et al. manually specify the region of the visible face whereas our approach performs an automatic localization via model-based image interpretation.

ground truth	classified as						recognition rate
	surprise	happiness	anger	disgust	sadness	fear	
surprise	28	1	1	0	0	0	93.33%
happiness	1	26	1	2	3	**4**	70.27%
anger	1	1	14	2	2	1	66.67%
disgust	0	2	1	10	3	1	58.82%
sadness	1	2	2	2	22	1	73.33%
fear	1	**5**	1	0	2	13	59.09%
mean recognition rate							**70.25%**

Table 5.1: Confusion matrix and recognition rates of our approach.

Michel et al. [111] also focus on facial motion by manually specifying 22 feature points that are predominantly located around the mouth and around the eyes. They utilize a Support Vector Machine (SVM) for determining one of the six facial expressions. The recognition rates of Schweiger et al. and Michel et al. are illustrated in Table 5.4.

5.5 A Survey on Humans Recognizing Facial Expressions

Facial expressions are often caused by minimum activity of facial muscles, which makes it difficult for machines to detect and distinguish between the different expressions. In order to obtain a comparable measure, we investigate the accuracy of humans recognizing facial expressions by conducting a comprehensive survey questioning hundreds of people. Afterwards, we compare these results to facial expression interpretation algorithms.

Questioning people about facial expressions has been conducted earlier. In 1872, Charles Darwin asked travelers from different continents about the facial expressions of the native people [34]. In the 1970ies, Paul Ekman and Wallace Friesen investigate whether facial expression interpretation is a universal or culture-specific task by applying current scientific standards to this investigation, see Section 5.2.1.

Our survey questions a few hundred people about the facial expressions visible in the Cohn-Kanade-Facial-Expression-Database, see Section 5.2.3. Note that this database does not provide communication channels and further context information. Therefore, the participants are pro-

vided the same information as current facial expression interpretation algorithms. This makes a comparison of human capabilities and current computer algorithms appropriate.

5.5.1 Description of the Survey

The participants of the survey are shown randomly selected image sequences of the Cohn-Kanade-Facial-Expression-Database and they have to specify one of the six universal facial expressions for each sequence. There is the opportunity to annotate "none" in case they are not able to decide on one of the facial expressions. Each sequence can be replayed as often as necessary. The participants are asked to annotate as many image sequences as they want.

In the end, 250 different persons were specifying their impression on some of the 488 image sequences of the Cohn-Kanade-Facial-Expression-Database and we collected $q = 5413$ annotations all together. On average, each participant annotated around $\frac{5413}{250} \approx 22$ image sequences, which results in approximately $\frac{5413}{488} \approx 11$ annotation per sequence. Furthermore, the participants stated their gender, age, and origin on a voluntary base. 45.7% of the participants are female, 48.8% are male, and 5.5% did not tell their gender. 64.0% of the participants are adults, 12.5% are less than 18 years old, and 23.5% did not specify their age.

5.5.2 Evaluation on the Survey's Results

This section evaluates the annotations specified by the participants of our survey. It shows, which facial expressions are mostly classified equally and which are more likely to be confused. Cohn and Kanade provide a manually specified set of Action Units for each sequence of the Cohn-Kanade-Facial-Expression-Database. Unfortunately, these Action Units do not relate to one of the six universal facial expressions uniquely. Therefore, we do not have the possibility to decide whether the annotations of the participants specify the facial expression correctly or not. For this reason, the entire Section 5.5.2 rather compares the annotations of the participants to one another. In contrast, Section 5.5.3 will compare the capability of determining facial expressions between humans and computer algorithms. This requires us to adapt the annotations that are treated to be the correct ones by Michel et al. [111].

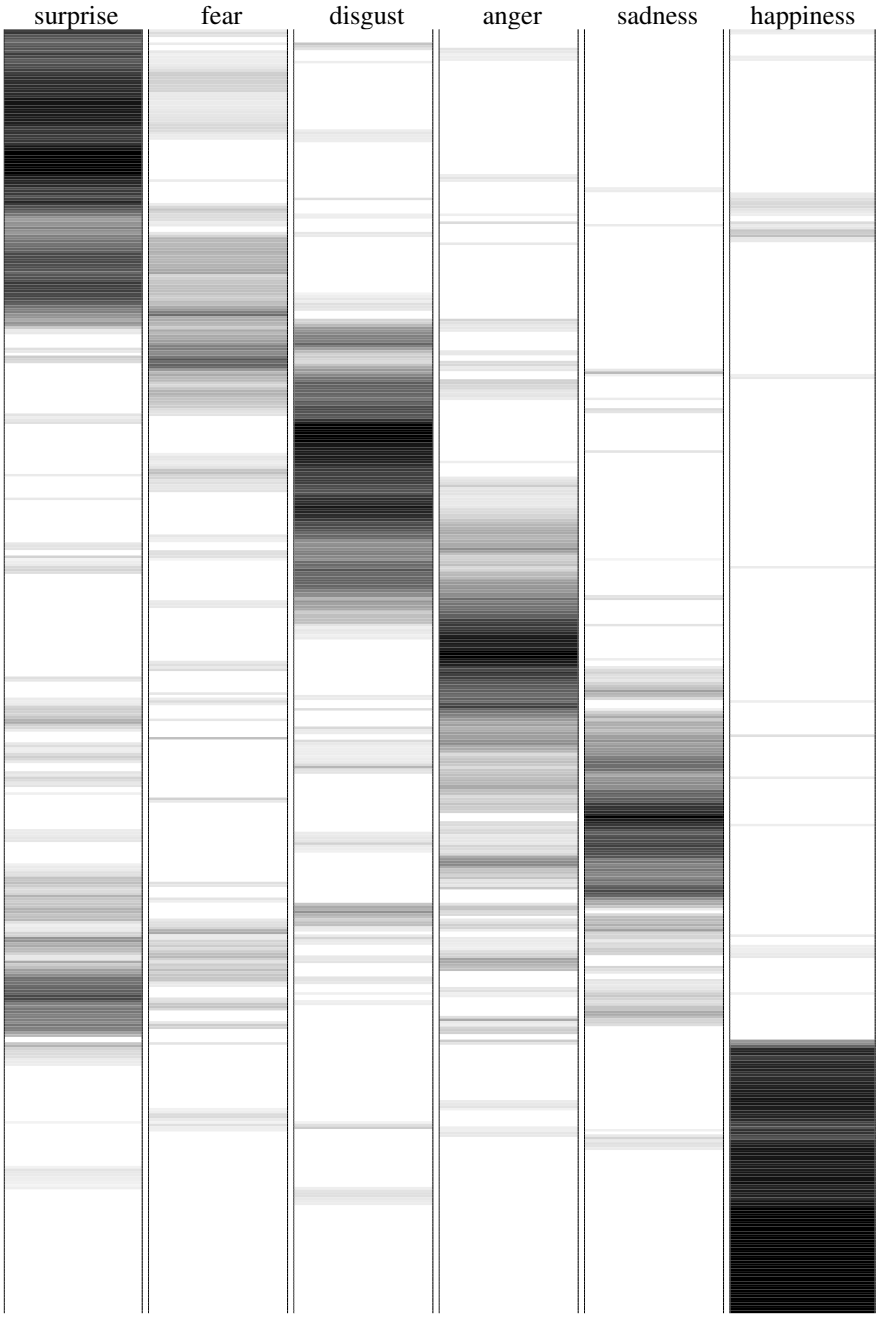


Table 5.2: The result of humans specifying the facial expression visible in each of the 488 image sequences of the Cohn-Kanade-Facial-Expression-Database. More intense colors denote a higher annotation rate for a particular facial expression. The sequences are sorted such that adjacent rows are annotated similarly.

5.5.2.1 Annotation Rate of Each Facial Expression

We denote $\mathcal{E}$ to be the set of facial expressions that the participants are able to specify. During our survey, we obtained q annotations on the entire image database, which we subdivide into the numbers of annotations q_i for the image sequences i. Again, we subdivide q_i into the numbers of annotations $q_{i,\epsilon}$ for the facial expression ϵ, see Equation 5.1. The annotation rate $r_{i,\epsilon}$ gives evidence about the number of annotations for one image sequence and for one facial expression in relation to the total amount of annotations for one sequence.

$$\begin{aligned} \mathcal{E} &= \{\text{happiness}, \text{sadness}, \text{disgust}, \text{fear}, \text{anger}, \text{surprise}, \text{none}\} \\ q &= \sum_{i=1}^{488} q_i \\ q_i &= \sum_{\epsilon \in \mathcal{E}} q_{i,\epsilon} \\ r_{i,\epsilon} &= \frac{q_{i,\epsilon}}{q_i} \end{aligned} \tag{5.1}$$

Table 5.2 illustrates the annotations of all participants for all 488 sequences. Every row denotes one image sequence i and indicates the annotation rate $r_{i,\epsilon}$ for all facial expressions $\epsilon \in \mathcal{E}$. A more intense color denotes a higher annotation rate for a particular facial expression. We sort the rows of the table such that similarly specified image sequences are adjacent to one another, which clusters the sequences by the predominantly recognized facial expressions. In this representation, the confusion of the facial expressions is clearly visible.

Obviously, happiness is best distinguished from the other facial expressions. Sadness gets little confused with disgust or fear, but gets highly confused with anger or surprise. Anger and disgust are the most mixed up facial expressions. It seems that fear is the hardest to tell apart from the other facial expressions. It gets often confused with surprise, disgust and sometimes with sadness or anger. In contrast to these insights, Kanaujia et al. [87] observe that happiness and fear are highly confused.

5.5.2.2 Histograms of the Annotation Rates

The previous section calculates the annotation rate to be specified as a particular facial expression for each image sequence. This section elaborates on the cumulative occurrence of the different annotation rates. Note that the participants were able to annotate a particular image

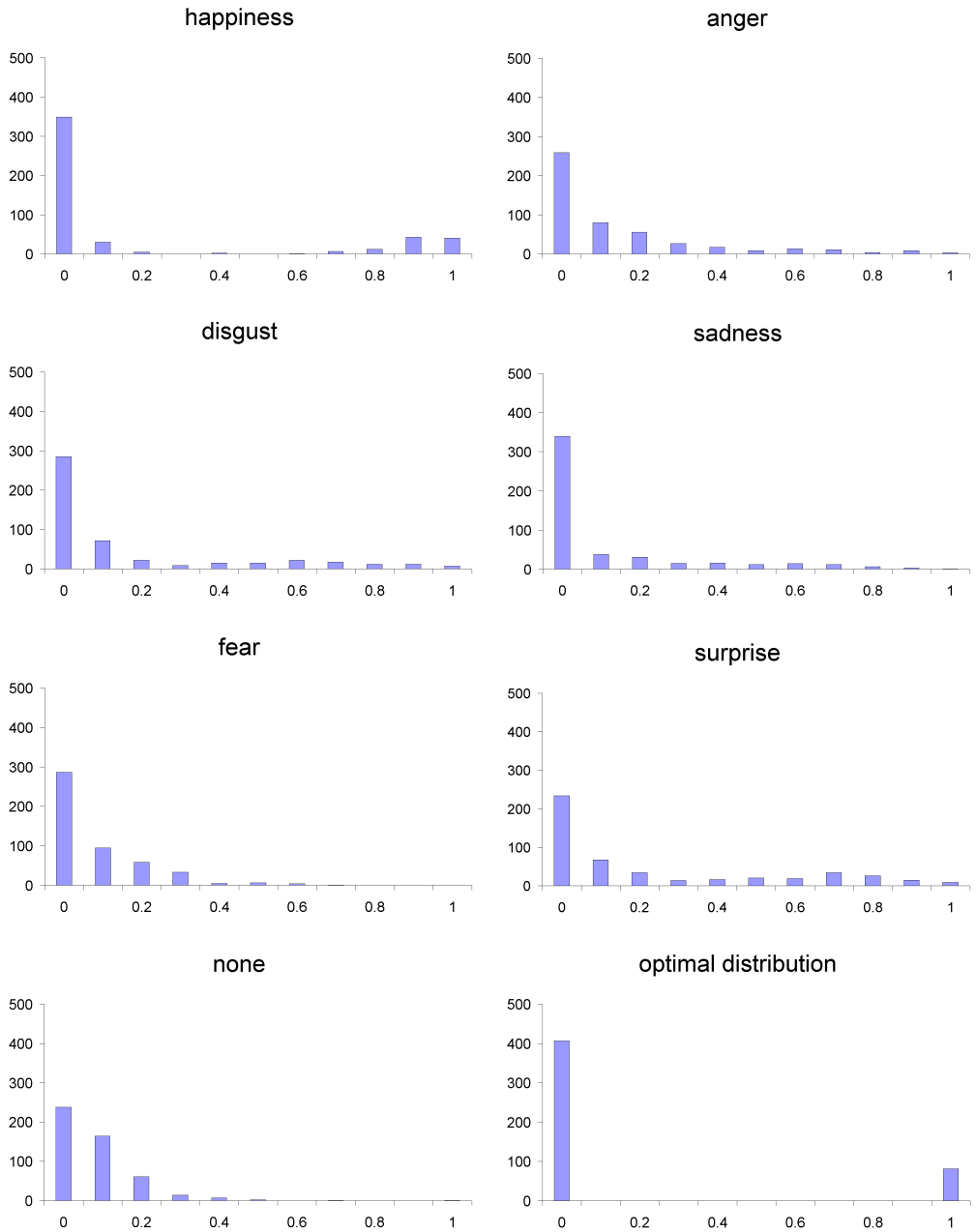


Figure 5.7: Distribution of the annotation rate $r_{i,\epsilon}$ for each facial expression ϵ and for "none". It also illustrates a perfect distribution, which would occur in case the participants decided exactly the same for all image sequences and they determined this facial expression for exactly $\frac{1}{6}$ of the image sequences.

sequence i well, if the annotation rate for one facial expression ϵ_1 is close to $r_{i,\epsilon_1} = 1$ and the annotation rate for the other facial expressions $\epsilon_2 \in \mathcal{E}\backslash\{\epsilon_1\}$ is $r_{i,\epsilon_2} = 0$. Figure 5.7 shows the histograms of the annotation rates for all facial expressions and for "none". Distinctive values for the annotation rate $r_{i,\epsilon} = 0$ and the annotation rate $r_{i,\epsilon} = 1$ in the histograms would indicate an excellent recognition for a particular facial expression ϵ. This fact is illustrated by the optimal distribution, which bases on the assumption that the six universal facial expressions occur equally often. This distribution would indicate the annotation rate $r_{i,\epsilon} = 0$ with $\frac{5}{6}$ of the image sequences and the annotation rate $r_{i,\epsilon} = 1$ with $\frac{1}{6}$ of the image sequences.

Similar to the evaluation in the previous section, happiness is the most distinctive facial expression, because its histogram shows the most distinctive peaks for $r_{i,\epsilon} = 0$ and for $r_{i,\epsilon} = 1$. This evaluation demonstrates that fear is recognized worst of all facial expressions, because no sequence has a annotation rate of $r_{i,\text{fear}} = 1$ or close to it.

5.5.2.3 Confusion between Two Facial Expressions

The previous sections show that people often do not agree on the facial expression visible in a particular image sequence, but confuse certain facial expressions with one another. This section determines the level of confusion between two facial expressions by comparing the annotations of different participants for each image sequence. We consider two facial expressions to be confused if two different participants of the survey annotate the same image sequence with these two different expressions. As the measure of confusion $\tau(\epsilon_1, \epsilon_2)$ between two facial expressions $\epsilon_1, \epsilon_2 \in \mathcal{E}$, we take the relative number of image sequences being annotated with both facial expressions. Higher values of τ denote higher confusion between two facial expressions. This measure is calculated with Equation 5.2, where $S_i(\epsilon_1, \epsilon_2)$ determines if a particular image sequence i is confused by for two facial expressions ϵ_1 and ϵ_2.

$$
\begin{aligned}
s_i(\epsilon_1, \epsilon_2) &= \begin{cases} 1 & : \; q_{i,\epsilon_1} > 0 \;\wedge\; q_{i,\epsilon_2} > 0 \\ 0 & : \; \text{otherwise} \end{cases} \\
\tau(\epsilon_1, \epsilon_2) &= \frac{1}{488} \sum_{i=1}^{488} s_i(\epsilon_1, \epsilon_2)
\end{aligned} \tag{5.2}
$$

Table 5.3 illustrates the confusion of either pair of facial expressions. It shows that people have difficulties in interpreting the facial expressions, because some facial expressions seem to get confused very easily with certain other expressions. The participants did not decide well

between fear and surprise for some image sequences. As an analogy to FACS, some Action Units are similar in these two facial expressions [53]. Further expressions, which get easily confused because of some coinciding Action Units are fear and disgust, anger and disgust, anger and surprise, and anger and sadness. Interestingly, people confused least between happiness and sadness.

$\tau\,(row, col)$	Anger	Disgust	Fear	Happiness	Sadness	Surprise	"none"
Anger	—	26.4%	17.0%	5.1%	23.4%	22.3%	26.8%
Disgust	—	—	20.5%	4.1%	11.7%	18.6%	21.9%
Fear	—	—	—	7.0%	9.8%	28.7%	21.7%
Happiness	—	—	—	—	3.7%	10.2%	12.9%
Sadness	—	—	—	—	—	18.6%	19.9%
Surprise	—	—	—	—	—	—	31.6%
"none"	—	—	—	—	—	—	—

Table 5.3: The amount of confusion between either pair of the six universal facial expressions and of "none". For clarity, the lower part of the table is omitted.

5.5.3 Comparing the Recognition Rate of Humans and Algorithms

Based on the survey's results this section calculates the recognition rate of humans and compares it to the recognition rate of particular algorithms. For each image sequence, objectively comparing the accuracy requires to know about the correct facial expression, which we will denote with *ground truth*. Unfortunately, Cohn and Kanade did not specify this ground truth for their image sequences. As a consequence, researchers often specify this ground truth by their own in order to train their interpretation algorithm and to evaluate its result, see Michel et al. [111]. Note that our evaluation in Section 5.5.2 proves that these manual annotations are therefore not be very reliable either.

Nevertheless, this section takes the manual annotations of Michel et al. as the ground truth and compares the accuracy of their algorithm to the accuracy of the human annotations obtained by our survey. We choose the recent project of Michel et al., because they also take the Cohn-Kanade-Facial-Expression-Database and they provide the manually annotated facial expressions. Their algorithm determines one of the six universal facial expressions for each image

facial expression	Human specification during our survey	Result of the algorithm of Michel et al. [111]	Result of the algorithm of Schweiger et al. [130]
Anger	71.7%	66.7%	75.6%
Disgust	64.1%	64.3%	30.0%
Fear	27.9%	66.7%	0.0%
Happiness	90.5%	91.7%	79.2%
Sadness	52.7%	62.5%	60.5%
Surprise	76.8%	83.3%	89.8%
Average	64.0%	71.8%	55.9%

Table 5.4: Recognition rate of the survey compared with the results of different algorithms.

sequence by tracking facial feature points that perform face localization and feature extraction. The displacements of the facial features in the image sequence are used as input data to a Support Vector Machine classifier (SVM). They consider SVMs combined with their facial feature tracking approach effective towards fully automatic and unobtrusive expression recognition in real-world scenarios. For the comparison within this section, we take the accuracy values that they provide for recognizing one of the six universal facial expressions [111, Table 5].

Furthermore, we also consider the results of Schweiger et al. [130]. They propose a neural architecture for temporal emotion recognition from image sequences. Features that represent temporal facial variations are extracted within a bounding box around the face that is subdivided into regions. Within each region, the optical flow is tracked over time and its principal components are considered a representative features for classification. For each facial expression a neural network is trained. They also utilize the Cohn-Kanade-Facial-Expression-Database for training and evaluation purpose. The recognition rate for anger, sadness, and surprise is high, whereas the values for disgust, fear, and happiness are very low. They relate their low recognition rate of 0% for fear with the fact that they have only a few sequences of this facial expression for training and testing. Note that this evaluation is not based on the ground truth specified by Michel et al. and therefore, it is not entirely comparable.

For determining the capability of humans, we calculate the accuracy of humans recognizing facial expressions via the annotations of our survey. We consider each annotation of the participants and evaluate its correctness by comparing it to the ground truth. Table 5.4 shows the

ground truth	specified as							#seq	#annot.	recogn.
	anger	disgust	fear	happiness	sadness	surprise	none			
anger	**208**	30	7	0	11	11	23	25	290	71.7%
disgust	77	**198**	5	0	0	8	21	29	309	64.1%
fear	17	119	**67**	0	2	23	12	23	240	27.9%
happiness	2	4	4	**237**	1	3	11	24	262	90.5%
sadness	55	15	3	1	**125**	21	17	21	237	52.7%
surprise	0	4	42	5	0	**202**	10	24	263	76.8%

Table 5.5: Confusion matrix of the survey's annotations.

recognition rate of humans and algorithms. Table 5.5 shows the confusion matrix that shows the confusion between the result of our survey and the ground truth.

For example, Michel et al. specify 25 sequences to show the facial expression anger and these sequences are classified correctly in 208 cases, and therefore, the recognition rate for anger is 71.7%. Anger was never mistaken for happiness, but in 30 cases it was mistaken for disgust etc. With 90.5%, we get the best recognition rate for happiness, while fear was distinguished poorly with 27.9%. The sequences which are showing fear were even more often classified as disgust (in 119 cases) than as fear (in 67 cases). The participants of our survey have a higher recognition rate for the facial expressions anger, disgust, happiness, surprise and a higher average value of all facial expressions. In contrast, the algorithm works better for distinguishing fear and sadness from the other facial expressions.

5.5.4 Conclusion on the Survey

This survey shows that humans are not as good in determining the facial expression of other people as computer vision researchers would expect them to be. Human annotations are not even more accurate than current algorithms for facial expression interpretation. One of the main reasons for these poor recognition rates originates from the fact that the Cohn-Kanade-Facial-Expression-Database does not contain natural expressions. Instead, Cohn and Kanade asked the persons to act the six universal facial expressions and therefore, the expressions are recorded as the performing person would consider them to look like. Showing a laughing expression is simple, but most people are not sure how angry, afraid, or disgusted faces look like.

Furthermore, this recording was conducted in a laboratory environment rather than in a real-world scene. The consequence is that these performed expressions are different from natural expressions and therefore, the participants of our survey are not too accurate in recognizing them.

In our opinion, the most decisive reason for the poor results is the consideration of video information only. We expect humans to be more accurate being provided further information as well, such as audio information and long-term context information. Therefore, we recommend integrating this information into facial expression interpretation algorithms as well in order to improve recognition. We recently published our preliminary results on this issue [129].

Chapter 6

Summary and Conclusion

Interaction between humans consists of a variety of communication channels, such as natural language and facial expressions. Current systems are not able to interpret these channels as robustly and accurately as humans are, and human-computer interaction is therefore restricted to traditional input and output devices like keyboards and mice. This thesis focuses on interpreting facial expressions as one aspect of the visual communication channel and elaborates on state-of-the-art techniques that will leverage novel paradigms for human-computer interaction.

With this purpose in mind, we propose to use model-based image interpretation, which is a technological scheme that generally contributes to current and future requests on understanding images and further sensor data. In this regard, accurately fitted models represent an intermediate step to image interpretation. A small number of model parameters describe the visible object, and therefore facilitate the subsequent interpretation step. However, it is a great challenge to robustly fit a model to an image. As described in Chapter 2, this task is generally subdivided into several computationally independent steps. This thesis contributes to model-based image interpretation by proposing novel approaches for two of these steps: skin color extraction and objective functions.

In Chapter 3, we propose an algorithm that extracts skin color from the image, because this feature describes the location and the shape of human faces well. Unfortunately, the appearance of skin color depends highly on the person and the context conditions of the image. Therefore, we propose to determine image-specific characteristics that describe the visual appearance of human skin at first. Then, a general purpose color classifier specializes according to this information, which enables the adapted classifier to precisely extract skin color pixels

from the image. Our approach allows the integration of general purpose color classifiers and yields both high runtime performance and high accuracy. Besides our application of facial expression recognition, this feature extraction scheme is also applicable to other aspects of image interpretation.

Chapter 4 focuses on the objective function, a component that has a substantial influence on the accuracy of the entire model fitting process. It explains the shortcomings of constructing this function by hand. Here, the designer selects salient image features by intuition, with which the calculation rules are assembled manually. In contrast, we define so-called *ideal* objective functions, investigate their behavior, and explicitly formulate their properties. Unfortunately, these ideal functions cannot be obtained for real-world image interpretation scenarios. Therefore, we propose a novel methodology that learns the objective function from training images, which are manually annotated with the preferred model parameters. Simultaneously, we enforce to approximate the properties of ideal objective functions, which yields highly accurate calculation rules. Our comprehensive evaluation shows the obtained precision as well as the high runtime performance. This methodology does not require fundamental expertise on computer vision any longer and is generally applicable to various model fitting tasks. As it offers an enormous range of use, a potential for commercialization is at hand.

Finally, Chapter 5 elaborates on the task of interpreting facial expressions. It describes the applicability of these techniques and focuses on its challenges. Current insights and achievements have already laid the foundation to a reasonable solution. Our approach infers facial expressions from image data and the parameters of a correctly fitted face model. Similar to other approaches, we obtain accurate results deriving the facial expression from facial muscle movement, which is determined by optical flow. Additionally, we conducted a survey on the capabilities of humans interpreting facial expressions. Surprisingly, humans do not achieve more accurate results than state-of-the-art computer vision algorithms, while taking only the visual communication channel into account. Therefore, we propose to consider the integration of further communication channels inevitable in order to interpret facial expressions and human emotion.

In conclusion, we expect current and future requests on interpreting images and other sensor data to greatly benefit from model-based techniques. This methodology divides the interpretation challenge into the task of visually grasping real-world objects via models and the task of inferring the interpretation result from the model parameters. Focusing on these two fractions individually rather than on the interpretation task as a whole provides decisive advantages that

affect both the accuracy of the interpretation result and the feasibility of its utilization. Our approach to learn objective functions from image annotations turns one major challenge of this scheme manageable by non-experienced persons while preserving high precision at the same time.

Chapter 7

Outlook

A fine grasp of the content of images and of further sensor data will be essential for developing intelligent devices in future times. Model-based techniques render image interpretation feasible for various applications in real-world scenarios. This thesis presents techniques for fitting deformable face models to images in order to recognize facial expressions. The techniques involved facilitate interpretation tasks for further application as well. Currently, model-based image interpretation is not widely represented in machine vision implementations, but a multitude of sophisticated techniques have been developed during the last decade [31; 154; 67]. These achievements turn model-based techniques viable for real-world challenges.

The proposed approach to obtain information about the location and the shape of the different facial regions by adaptive color classification facilitates various interpretation tasks. Future extensions will determine lip color, tooth color, iris color, brow color, and many more. Preliminary results demonstrate that robust classifiers will be obtained for all of them. In addition, our calculation rules will not only consider the pixel's color, but also its location relative to the determined face. Taking these additional features, machine learning algorithms will provide more accurate decision rules.

Our evaluation on model fitting emphasizes the trade-off between obtaining generally applicable and accurate objective functions via the machine learning approach, while the considered learning radius controls this aspect. Therefore, we will equip future fitting algorithms with several objective functions at the same time. We will apply them in sequence starting with the most general function and gradually execute more accurate ones. The novelty of this iteration scheme is based on the execution of different algorithms throughout the different iterations. Fortunately,

this increased complexity of the fitting algorithm does not require additional manual work, because each of these objective functions is learned from the same basis of image annotations. Furthermore, we will provide a larger variety of image features to the machine learning step, which will improve the generated calculation rules.

Moreover, we will abolish the limitation that the voluminous file of training data inducts into the current implementation of our approach. Its immense size depends on the fact that the image features of the training data represent the underlying image data in a highly redundant way. Forcing the machine learning algorithm not to rely on this file, but to calculate the necessary image features on request during the learning phase will avoid severe memory restrictions. Other researchers have successfully applied this approach in a similar way in order to locate objects within images [154; 99].

Learning objective functions from annotated training images is promising not only for two-dimensional contour models, but to other kinds of models as well. We have already conducted successful experiments that illustrate the excellent performance of our methodology on rigid three-dimensional models. Therefore, we intend to formulate and publish a generally applicable scheme for applying our approach to various kinds of models for the benefit of being capable of interpreting general real-world scenes.

The face interpretation community has achieved great progress during the last few years. Its focus is split into several related directions, such as face localization, face tracking, person identification, gaze tracking, and facial expression interpretation. Recent achievements in the various research fields involved promise the continuation of this success in the future. We believe our methods will provide substantial accuracy to image interpretation techniques and that they will even enable non-expert users to exploit the advantages of model-based fitting in their applications.

Appendix A

Proofs

This section proves that the ideal objective function $f_n^\star(I, \boldsymbol{x})$ has the properties P1 and P2 stated in Section 4.4. In the course of these proofs we will use the abbreviation $\boldsymbol{c}_n^\star$ for $\boldsymbol{c}_n(\boldsymbol{p}_I^\star)$.

Proof: P1 holds for $f_n^\star(I, \boldsymbol{x})$

(1) Apply $f_n^\star(I, \boldsymbol{x})$ to P1

$$\forall \boldsymbol{x}(\boldsymbol{c}_n^\star \neq \boldsymbol{x}) \quad \Rightarrow \quad f_n^\star(I, \boldsymbol{c}_n^\star) < f_n^\star(I, \boldsymbol{x})$$

(2) Substitute $f_n^\star(I, \boldsymbol{x})$ with $|\boldsymbol{x} - \boldsymbol{c}_n^\star|$ (Equation 4.2)

$$\forall \boldsymbol{x}(\boldsymbol{c}_n^\star \neq \boldsymbol{x}) \quad \Rightarrow \quad |\boldsymbol{c}_n^\star - \boldsymbol{c}_n^\star| < |\boldsymbol{x} - \boldsymbol{c}_n^\star|$$

(3) Substitute $|\boldsymbol{c}_n^\star - \boldsymbol{c}_n^\star| = 0$

$$\forall \boldsymbol{x}(\boldsymbol{c}_n^\star \neq \boldsymbol{x}) \Rightarrow (0 < |\boldsymbol{x} - \boldsymbol{c}_n^\star|)$$

$q.e.d.$

Proof: P2 holds for $f_n^\star(I, \boldsymbol{x})$

(1) Apply $f_n^\star(I, \boldsymbol{x})$ to P2

$$\exists \boldsymbol{m} \forall \boldsymbol{x}\ (\boldsymbol{m} \neq \boldsymbol{x}) \quad \Rightarrow$$

$$f_n^\star(I, \boldsymbol{m}) < f_n^\star(I, \boldsymbol{x}) \quad \wedge \quad \nabla f_n^\star(I, \boldsymbol{x}) \neq \mathbf{0}$$

(2) Choose $\boldsymbol{c}_n^\star$ for $\boldsymbol{m}$

$$\forall \boldsymbol{x}\ (\boldsymbol{x} \neq \boldsymbol{c}_n^\star) \quad \Rightarrow$$

$$f_n^\star(I, \boldsymbol{c}_n^\star) < f_n^\star(I, \boldsymbol{x}) \quad \wedge \quad \nabla f_n^\star(I, \boldsymbol{x}) \neq \mathbf{0}$$

(3) The first part of the consequent has already been proven in the previous proof.

(4) The second part of the consequent will be proven below.

$$\forall \boldsymbol{x}\ (\boldsymbol{x} \neq \boldsymbol{c}_n^\star) \ \Rightarrow \ \nabla f_n^\star(I, \boldsymbol{x}) \neq \mathbf{0}$$

(5) Calculate the gradient $\nabla f_n^\star(I, \boldsymbol{x}) \ = \ \frac{\boldsymbol{x} - \boldsymbol{c}_n^\star}{|\boldsymbol{x} - \boldsymbol{c}_n^\star|}$ (see below).

$$\forall \boldsymbol{x}\ (\boldsymbol{x} \neq \boldsymbol{c}_n^\star) \ \Rightarrow \ \frac{\boldsymbol{x} - \boldsymbol{c}_n^\star}{|\boldsymbol{x} - \boldsymbol{c}_n^\star|} \neq \mathbf{0}$$

(6) Simplify

$$\forall \boldsymbol{x}\ (\boldsymbol{x} \neq \boldsymbol{c}_n^\star) \ \Rightarrow \ \boldsymbol{x} \neq \boldsymbol{c}_n^\star$$

$q.e.d.$

Compute $\nabla f_n^\star(I, \boldsymbol{x})$, the gradient of $f_n^\star(I, \boldsymbol{x})$

$$
\begin{aligned}
f_n^\star(I, \boldsymbol{x}) &= |\boldsymbol{x} - \boldsymbol{c}| = \sqrt{(x_1 - c_1)^2 + (x_2 - c_2)^2} \\
\nabla f_n^\star(I, \boldsymbol{x}) &= \begin{pmatrix} \frac{2(x_1 - c_1)}{2|\boldsymbol{x} - \boldsymbol{c}|} \\ \\ \\ \frac{2(x_2 - c_2)}{2|\boldsymbol{x} - \boldsymbol{c}|} \end{pmatrix} = \frac{\boldsymbol{x} - \boldsymbol{c}}{|\boldsymbol{x} - \boldsymbol{c}|}
\end{aligned}
$$

Appendix B

Summary of Notation

Terms related to images and image features:

$\boldsymbol{x}$	A pixel location in an image.
I	An image.
$\mathcal{I}$	The integral image.
K	The number of images in the image database.
I_k	The k^{th} image in the image database, with $1 \leq k \leq K$.
$E(I, \boldsymbol{x})$	The edge magnitude of the image I at the position $\boldsymbol{x}$.
$\boldsymbol{c_x}$	The color of pixel $\boldsymbol{x}$ in the NRGB color space.
R, G, B	Dimensions of the RGB color space.
$r, g, b, base$	Dimensions of the NRGB color space.

Terms related to models:

$\boldsymbol{p}$	The parameters of a model with $P = dim(\boldsymbol{p})$. In the case of a Point Distribution Model $\boldsymbol{p}$=$(t_x, t_y, s, \theta, \boldsymbol{b})^T$.
$\boldsymbol{p}_I^\star$	The manually specified ideal model parameters for a specific image I.
N	The number of contour points of a model's projection onto the image plane.
$\boldsymbol{c}(\boldsymbol{p})$	The projection of the model parameters $\boldsymbol{p}$ to the 2D image plane. This function is exemplary for all projections.

$\boldsymbol{c}_n(\boldsymbol{p})$ The mapping from the parameters $\boldsymbol{p}$ of a contour model to the pixel location of the n^{th} contour point with $1 \leq n \leq N$.

$\boldsymbol{c}_n^\star$ An abbreviation for $\boldsymbol{c}_n(\boldsymbol{p}_I^\star)$ only used in the proofs in Section A.

Terms related to objective functions:

$f(I, \boldsymbol{p})$ The global objective function that computes the fitness between the model parameterization $\boldsymbol{p}$ and the image I. In this thesis, lower values correspond to a better fit.

$f_n(I, \boldsymbol{x})$ The local objective function of the model's n^{th} contour point that computes the fitness between the pixel $\boldsymbol{x}$ and the image I.

$f_n^\star(I, \boldsymbol{x})$ The ideal local objective function of the model's n^{th} contour point.

$f_n^e(I, \boldsymbol{x})$ The designed local objective function of the model's n^{th} contour point that considers edge magnitudes to compute its values.

$f_n^\ell(I, \boldsymbol{x})$ The learned local objective function of the model's n^{th} contour point.

$\boldsymbol{m}$ The true global minimum of an objective function.

Terms related to training data generation:

D The number of displacements to one side along the perpendicular for gathering the training data of one contour point. The entire amount of displacements in both directions is $2D + 1$.

$\boldsymbol{x}_{k,n,d}$ The d^{th} displacement of contour point n within the training image$k$ with $1 \leq k \leq K,\ 1 \leq n \leq N,\ -D \leq d \leq D$.

Δ The maximal distance of the displacements $\boldsymbol{x}_{k,n,d}$ from the annotated contour point when generating the training data. The individual distances of each displacement $\boldsymbol{x}_{k,n,d}$ is computed by $|\boldsymbol{x}_{k,n,d} - \boldsymbol{x}_{k,n,0}| = \Delta\frac{|d|}{D}$.

A The number of image features (e.g. Haar, edge) provided to the learning algorithm.

$h_a(I, \boldsymbol{x})$ The value of the a^{th} image feature, calculated from the image I at the position $\boldsymbol{x}$ with $1 \leq a \leq A$.

Terms related to model trees:

T_n The model tree of the contour point n.

M_n Number of features selected by the calculation rules of the contour point n.

s_i Indices of the selected features with $1 \leq i \leq M_n$.

Terms related to skin color classification:

M Skin color mask with the elements $m_{i,j}$.

n_1, n_2 Dimension of skin color mask.

$k, i, j, f_{k,i,j}, s_{k,i,j}, roi_k$

Local variables to compute the skin color mask.

$\bar{\boldsymbol{\mu}}$ Mean of image-specific characteristics with $\bar{\boldsymbol{\mu}}=(\bar{\mu}_r, \bar{\mu}_g, \bar{\mu}_{base})^T$.

$\bar{S}$ Covariance matrix of image-specific characteristics. Its entries are $var_r, cov_{r,g}$,...

$\boldsymbol{\mu}, S, t$ Parameters of the ellipsoid-based color classifier.

$l_r, l_g, l_{base}, u_r, u_g, u_{base}$

Bounds of the cuboid-based color classifier.

t Maximum Mahalanobis distance between a pixel's color values and the mean color value $\bar{\boldsymbol{\mu}}$. Used by ellipsoid classifier.

$\mathcal{P}$ Set of skin-colored pixels (annotated and extracted).

$\Theta(n)$ The Landau notation for describing assymptotically tight bounds [92] that we use here for denoting the runtime performance on an image with n pixels.

Terms related to facial expression interpretation:

G Number of facial feature points.

g Number of Principal Components of the facial motion.

Terms related to the evaluation of the survey:

$\mathcal{E}$ Set that consist of all facial expressions that can be annotated by the participants.

q Number of annotations to the entire image database.

q_i Number of annotations to the image sequence i.

$q_{i,\epsilon}$ Number of annotations to the image sequence i with the facial expression ϵ.

$r_{i,\epsilon}$ Annotations of the facial expression ϵ to the image sequence i.

$\tau()$ Confusion rate of different facial expressions.

$s_i()$ Determines whether two facial expressions are confused for image sequence i or not.

Bibliography

[1] A. Amini, T. Weymouth, and R. Jain. Using dynamic programming for solving variational problems in vision. In *IEEE Transactions on Pattern Analysis and Machine Intelligence 12(9)*, pages 855–867, 1990.

[2] Shai Avidan. Support vector tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(8):1064–1072, August 2004.

[3] D. H. Ballard. Generalizing the hough transform to detect arbitrary shapes. 13(2):111–122, 1981.

[4] Curzio Basso and Thomas Vetter. Statistically motivated 3D faces reconstruction. In *Proceedings of the 2nd International Conference on Reconstruction of Soft Facial Parts*, volume 31(2), Remagen, Germany, March 2005. Luchterhand Publishers. BKA Research Series.

[5] Joseph Bates. The role of emotion in believable agents. *Communications of the ACM*, 37(7):122–125, 1994.

[6] Joseph Bates, A. Bryan Loyall, and W. Scott Reilly. An architecture for action, emotion, and social behavior. Technical report, Carnegie Mellon University, Pittsburgh, PA 15213, May 1992.

[7] Selim Benhimane and E. Malis. Real-time image-based tracking of planes using efficient second-order minimization. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 943–948, 2004.

[8] Günter Bentele and Klaus Beck. Information – Kommunikation – Massenkommunikation: Grundbegriffe und Modelle der Publizistik- und Kommunikationswissenschaft. In

Otfried Jarren, editor, *Medien und Journalismus 1*, pages 15 – 50, Opladen, 1994. Westdeutscher Verlag.

[9] R. Bergler and U. Six. Psychologie des Fernsehens. Bern-Stuttgart-Wien, 1979. Hans Huber.

[10] N. Bernsen, L. Dybkjr, and S. Kiilerich. Evaluating conversation with hans christian andersen. pages 1011–1014, Lisbon, Portugal, 2004.

[11] Andrew Blake and Michael Isard. *Active Contours: The Application of Techniques from Graphics,Vision,Control Theory and Statistics to Visual Tracking of Shapes in Motion*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1998.

[12] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3D faces. In Alyn Rockwood, editor, *Siggraph 1999, Computer Graphics Proceedings*, pages 187–194, Los Angeles, 1999. Addison Wesley Longman.

[13] 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.

[14] Eric Bloedorn and Ryszard S. Michalski. Data-driven constructive induction. *IEEE Intelligent Systems*, 13(2):30–37, 1998.

[15] Aurelien Boffy, Y. Tsin, and Y. Genc. Real-time feature matching using adaptive and spatially distributed classification trees. In M. J. Chantler, E. Trucco, and R. B. Fisher, editors, *17th British Machine Vision Conference*, volume 2, page 529, Edinburgh, September 2006.

[16] Bernard Boulay, Francois Bremond, and Monique Thonnat. Human posture recognition in video sequence. In *VS-PETS*, pages 23–29, 2003.

[17] J. Brand and J. S. Mason. A comparative assessment of three approaches to pixel-level human skin-detection. In *15th International Conference on Pattern Recognition, vol. 1*, pages 1056–1059, September 2000.

[18] D. A. Brown, I. Craw, and J. Lewthwaite. A SOM based approach to skin detection with application in real time systems. In *In Proceedings of the British Machine Vision Conference*, 2001.

[19] John Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

[20] E. De Castro and C. Morandi. Registration of translated and rotated images using finite fourier transforms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(5):700–703, 1987.

[21] C. C. Chibelushi and F. Bourel. Facial expression recognition: A brief tutorial overview. In CVonline: On-Line Compendium of Computer Vision, editor Robert Fisher, January 2003.

[22] Isaac Cohen, Nicu Sebe, L. Chen, A. Garg, and T. Huang. Facial expression recognition from video sequences: Temporal and static modeling. *Computer Vision and Image Understanding (CVIU) special issue on face recognition*, 91(1-2):160–187, 2003.

[23] Jeffrey Cohn, Adena Zlochower, Jenn-Jier James Lien, and Takeo Kanade. Feature-point tracking by optical flow discriminates subtle differences in facial expression. In *Proceedings of the 3rd IEEE International Conference on Automatic Face and Gesture Recognition*, pages 396 – 401, April 1998.

[24] Jeffrey Cohn, Adena Zlochower, Jenn-Jier James Lien, and Takeo Kanade. Automated face analysis by feature point tracking has high concurrent validity with manual facs coding. *Psychophysiology*, 36:35 – 43, 1999.

[25] Tim F. Cootes, G. J. Edwards, and Chris J. Taylor. Active appearance models. In H. Burkhardt and Bernd Neumann, editors, *5th European Conference on Computer Vision*, volume 2, pages 484–498, Freiburg, Germany, 1998. Springer-Verlag.

[26] Tim F. Cootes, A. Hill, and Chris J. Taylor. Medical image interpretation using active shape models: Recent advances. In *14th International Conference on Information Processing in Medical Imaging*, pages 371–372, 1995.

[27] Tim F. Cootes, A. Hill, Chris J. Taylor, and J. Haslam. The use of active shape models for locating structures in medical images. In *Proceedings of the 13th International Conference on Information Processing in Medical Imaging*, pages 33–47, 1993.

[28] Tim F. Cootes and Chris J. Taylor. Active shape models – smart snakes. In *Proceedings of the 3rd British Machine Vision Conference*, pages 266 – 275. Springer Verlag, 1992.

[29] Tim F. Cootes and Chris J. Taylor. Statistical models of appearance for medical image analysis and computer vision, in medical imaging. In *Image Processing – Proceedings of SPIE*, volume 4322, pages 238–248, April 2001.

[30] Tim F. Cootes and Chris J. Taylor. Anatomical statistical models and their role in feature extraction. *British Journal of Radiology*, 77:133–139, 2004.

[31] Tim F. Cootes and Chris J. Taylor. Statistical models of appearance for computer vision. Technical report, University of Manchester, Wolfson Image Analysis, Imaging Science and Biomedical Engineering, Manchester M13 9PT, United Kingdom, 2004.

[32] David Cristinacce and Tim F. Cootes. Facial feature detection and tracking with automatic template selection. In *7th IEEE International Conference on Automatic Face and Gesture Recognition*, pages 429–434, Southampton, UK, April 2006.

[33] David Cristinacce and Tim F. Cootes. Feature detection and tracking with constrained local models. In *17th British Machine Vision Conference*, pages 929–938, Edinburgh, UK, 2006.

[34] Charles Darwin. *The Expression of the Emotions in Man and Animals*. Philosophical Library, New York, 1872.

[35] Ingrid Daubechies. *Ten lectures on wavelets*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 1992.

[36] Richard J. Davidson, Klaus R. Scherer, and H. Hill Goldsmith, editors. *Handbook of Affective Sciences*. Oxford University Press US, December 2002.

[37] Guillaume Dewaele, Frédéric Devernay, and Radu P. Horaud. Hand motion from 3D point trajectories and a smooth surface model. In T. Pajdla and J. Matas, editors, *Pro-*

ceedings of the 8th European Conference on Computer Vision, volume 1 of *LNCS 3021*, pages 495–507. Springer, May 2004.

[38] M. Dimitrijevic, S. Ilic, and P. Fua. Accurate face models from uncalibrated and ill-lit video sequences. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 1034–1041, Los Alamitos, CA, USA, 2004. IEEE Computer Society.

[39] Elena Dimitrova. Entwicklung eines dynamischen Klassifikators zur Lokalisation von Retina, Iris, Lippen, Augenbrauen und Zähnen innerhalb eines Gesichtes. Technical report, Technische Universität München, Fakultät für Informatik, Boltzmannstr. 3,85748 Garching bei München, Germany, January 2007.

[40] A. C. Downton and H. Drouet. Model-based image analysis for unconstrained human upper-body motion. In *International Conference on Image Processing and its Applications*, pages 274–277, April 1992.

[41] Paul Ekman. Universals and cultural differences in facial expressions of emotion. In J. Cole, editor, *Nebraska Symposium on Motivation 1971*, volume 19, pages 207–283, Lincoln, NE, 1972. University of Nebraska Press.

[42] Paul Ekman. Facial expressions. In T. Dalgleish and M. Power, editors, *Handbook of Cognition and Emotion*, New York, 1999. John Wiley & Sons Ltd.

[43] Paul Ekman, R. Davidson, and Wallace Friesen. The duchenne smile: Emotional expression and brain physiology ii. *Journal of Personality and Social Psychology*, 58(2):342–353, 1990.

[44] Paul Ekman and Wallace Friesen. *The Facial Action Coding System: A Technique for The Measurement of Facial Movement*. Consulting Psychologists Press, San Francisco, 1978.

[45] Paul Ekman, Wallace Friesen, and M. O'Sullivan. Smiles when lying. *Journal of Personality and Social Psychology*, 54(3):414–420, March 1988.

[46] Irfan A. Essa and Alex P. Pentland. Facial expression recognition using a dynamic model and motion energy. In *ICCV '95: Proceedings of the Fifth International Conference on Computer Vision*, pages 360–367, Washington, DC, USA, 1995. IEEE Computer Society.

[47] Irfan A. Essa and Alex P. Pentland. Coding, analysis, interpretation, and recognition of facial expressions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):757–763, 1997.

[48] Stefan Fischer, Sven Döring, Matthias Wimmer, and Antonia Krummheuer. Experiences with an emotional sales agent. In Elisabeth André, Laila Dybkjær, Wolfgang Minker, and Paul Heisterkamp, editors, *Affective Dialogue Systems*, volume 3068 of *Lecture Notes in Computer Science*, pages 309–312, Kloster Irsee, Germany, June 2004. Springer.

[49] 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.

[50] R. A. Fisher. The use of multiple measurements in taxonomic problems. Technical report, 1936.

[51] T. Frank, M. Haag, H. Kollnig, and H.-H. Nagel. Tracking of occluded vehicles in traffic scenes. In *Fourth European Conference on Computer Vision (ECCV'96)*, volume 2, pages 485–494, Cambridge, England, April 15th-18th 1996. Lecture Notes in Computer Science 1065, Springer-Verlag, Berlin a.o. 1996.

[52] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *European Conference on Computational Learning Theory*, pages 23–37, 1995.

[53] Wallace V. Friesen and Paul Ekman. *Emotional Facial Action Coding System*. Unpublished manuscript, University of California at San Francisco, 1983.

[54] P. Garcia, F. Pla, and I. Gracia. Detecting edges in colour images using dichromatic differences. In *Seventh International Conference on Image Processing And Its Applications*, volume 1, pages 363–367, July 1999.

[55] Bram van Ginneken, A. Frangi, J. Staal, B. Haar, and R. Viergever. Active shape model segmentation with optimal features. *IEEE Transactions on Medical Imaging*, 21(8):924–933, 2002.

[56] Bram van Ginneken, Mikkel Bille Stegmann, and M. Loog. Segmentation of anatomical structures in chest radiographs using supervised methods: A comparative study on a public database. *Medical Image Analysis*, 10(1):19–40, 2006.

[57] Daniel Goleman. Laugh and your computer will laugh with you, someday. pages pp. C1, C9, New York, January 1997. The New York Times.

[58] Giovani Gomez. On selecting colour components for skin detection. In *16th International Conference on Pattern Recognition*, volume 2, pages 961 – 964, 2002.

[59] Giovani Gomez and E. Morales. Automatic feature construction and a simple rule induction algorithm for skin detection. In A. Sowmya and T. Zrimec, editors, *Proc. of the ICML Workshop on Machine Learning in Computer Vision*, pages 31–38, Sydney, July 2002.

[60] H. Grabner, M. Grabner, and H. Bischof. Real-time tracking via on-line boosting. In Mike J. Chantler, E. Trucco, and Robert B. Fisher, editors, *17th British Machine Vision Conference*, volume 1, pages 47–56, Edinburgh, September 2006.

[61] Daniel Grest, Dennis Herzog, and Reinhard Koch. Human model fitting from monocular posture images. In *Proceedings of Vision, Modelling, and Visualization*, Erlangen, Germany, November 2005.

[62] Ralph Gross, Iain Matthews, and Simon Baker. Generic vs. person specific active appearance models. *Image and Vision Computing*, 23(11):1080–1093, November 2005.

[63] Lie Gu and Takeo Kanade. 3D alignment of face in a single image. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 1305–1312, June 2006.

[64] Alfred Haar. *Zur Theorie der Orthogonalen Funktionensysteme*. PhD thesis, Universität Göttingen, 1910.

[65] M. A. Hall and L. A. Smith. Feature subset selection: a correlation based filter approach. In N. Kasabov and et al., editors, *Proceedings of the Fourth International Conference on Neural Information Processing and Intelligent Information Systems*, pages 855–858, Dunedin, New Zealand, 1997.

[66] Simone Hämmerle, Matthias Wimmer, Bernd Radig, and Michael Beetz. Sensor-based situated, individualized, and personalized interaction in smart environments. In Armin B. Cremers, Rainer Manthey, Peter Martini, and Volker Steinhage, editors, *INFORMATIK 2005 - Informatik LIVE! Band 1, Beiträge der 35. Jahrestagung der Gesellschaft für Informatik e.V.*, volume 67 of *LNI*, pages 261–265, Bonn, Germany, September 2005. GI.

[67] Robert Hanek. *Fitting Parametric Curve Models to Images Using Local Self-adapting Seperation Criteria*. PhD thesis, Department of Informatics, Technische Universität München, 2004.

[68] Robert Hanek and Michael Beetz. The contracting curve density algorithm: Fitting parametric curve models to images using local self-adapting separation criteria. *International Journal of Computer Vision*, 59(3):233–258, 2004.

[69] Christoph Hansen. *Modellgetriebene Verfolgung formvariabler Objekte in Videobildfolgen*. PhD thesis, Department of Informatics, Technische Universität München, 2002.

[70] C. Harris and M. Stephens. A combined corner and edge detection. In *Proceedings of The Fourth Alvey Vision Conference*, pages 147–151, 1988.

[71] Jose L. Hernandez-Rebollar. Gesture-driven american sign language phraselator. In *Proceedings of the 7th international conference on Multimodal interfaces*, pages 288–292, New York, NY, USA, 2005. ACM Press.

[72] Paul C. V. Hough. Methods and means for recognizing complex patterns. In *U.S. Patent 3,069,654*, 1962.

[73] Thomas Hrabe. Detection of paving and lane lines using adaptive color classification. Technical report, Technische Universität München, Fakultät für Informatik, Boltzmannstr. 3,85748 Garching bei München, Germany, January 2007.

[74] Rein-Lien Hsu, Mohamed Abdel-Mottaleb, and Anil K. Jain. Face detection in color images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):696–706, May 2002.

[75] Xiangsheng Huang, Stan Z. Li, and Yangsheng Wang. Statistical learning of evaluation function for asm/aam image alignment. In Davide Maltoni and Anil K. Jain, editors, *Biometric Authentication, ECCV 2004 International Workshop, BioAW 2004, Prague, Czech Republic, May 15, 2004, Proceedings*, volume 3087 of *Lecture Notes in Computer Science*, pages 45–56. Springer, 2004.

[76] Curtis S. Ikehara, David N. Chin, and Martha E. Crosby. A model for integrating an adaptive information filter utilizing biosensor data to assess cognitive load. In *User Modeling*, volume 2702/2003, pages 208–212. Springer Berlin / Heidelberg, 2003.

[77] Michael Isard and Andrew Blake. Contour tracking by stochastic propagation of conditional density. In *European Conference on Computer Vision*, pages 343–356, 1996.

[78] Michael Isard and Andrew Blake. Condensation – conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1):5–28, 1998.

[79] Alejandro Jaimes and Nicu Sebe. Multimodal human computer interaction: a survey. In *IEEE International Workshop on Human-Computer Interaction*, pages 1–15, 2005.

[80] Oliver Jesorsky, Klaus J. Kirchberg, and Robert Frischholz. Robust face detection using the hausdorff distance. In *Proceedings of the Third International Conference on Audio- and Video-Based Biometric Person Authentication*, pages 90–95, Halmstad, Sweden, 2001. Springer-Verlag.

[81] Michael J. Jones and J. M. Rehg. Statistical color models with application to skin detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 274–280, Fort Collins, 1999.

[82] Michael J. Jones and Paul Viola. Fast multi-view face detection. Technical Report TR2003-96, Mitsubishi Electric Research Lab, June 2003.

[83] Ulrich Kadow. Vision-based detection of vehicles for advanced driver assistance systems. Technical report, Technische Universität München, Fakultät für Informatik, Boltzmannstr. 3,85748 Garching bei München, Germany, December 2006.

[84] F. Kahraman and Mikkel B. Stegmann. Towards illumination-invariant localization of faces using active appearance models. In Jóhannes R. Sveinsson Jón Atli Benediktsson,

editor, *IEEE NORSIG 2006, 7th Nordic Signal Processing Symposium, Reykjavik, Iceland (submitted)*. IEEE Iceland, June 2006.

[85] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

[86] Takeo Kanade, John F. Cohn, and Yingli Tian. Comprehensive database for facial expression analysis. In *International Conference on Automatic Face and Gesture Recognition*, pages 46–53, France, March 2000.

[87] Atul Kanaujia and Dimitris Metaxas. Recognizing facial expressions by tracking feature shapes. In *Proceedings of the 18th International Conference on Pattern Recognition*, pages 33–38, Washington, DC, USA, 2006. IEEE Computer Society.

[88] Roland Kehl and Luc Van Gool. Real-time pointing gesture recognition for an immersive environment. In *Sixth IEEE International Conference on Automatic Face and Gesture Recognition*, pages 577–582, Los Alamitos, CA, USA, 2004. IEEE Computer Society.

[89] Daniel Keren, Margarita Osadchy, and Craig Gotsman. Antifaces: A novel, fast method for image detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(7):747–761, 2001.

[90] Ron Kohavi and George H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1-2):273–324, 1997.

[91] Hannes Kruppa, Modesto Castrillon-Santana, and Bernt Schiele. Fast and robust face finding via local context. In *Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pages 157–164, Nice, France, 2003.

[92] Edmund Landau. *Handbuch der Lehre von der Verteilung der Primzahlen*. B. G. Teubner, 1909.

[93] Pat Langley, Herbert A. Simon, Gary L. Bradshaw, and J. Zytkow. *Scientific Discovery: Computational Explorations of the Creative Processes*. MIT Press, 1987.

[94] T. Lehn-Schiøler, M. B. Stegmann, L. Buchhave, and B. K. Ersbøll. Building a real-time digital face to map from speech to lip movements and facial expression. In *6th French-Danish Workshop on Spatial Statistics and Image Analysis in Biology, Skagen, Denmark*, Richard Petersens Plads, Building 321, DK-2800 Kgs. Lyngby, mar 2006.

[95] Vincent Lepetit, Pascal Lagger, and Pascal Fua. Randomized trees for real-time keypoint recognition. In *Computer Vision and Pattern Recognition*, volume 2, pages 775–781, Computer Vision Laboratory, EPFL, 1015 Lousanne, Switzerland, June 2005.

[96] Vincent Lepetit, Julien Pilet, and Pascal Fua. Point matching as a classification problem for fast and robust object pose estimation. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 244–250, Jun-Jul 2004.

[97] Stan Z. Li, Long Zhu, ZhenQiu Zhang, Andrew Blake, HongJiang Zhang, and Harry Shum. Statistical learning of multi-view face detection. In *Proceedings of the 7th European Conference on Computer Vision*, volume 4, pages 67–81, London, UK, 2002. Springer-Verlag.

[98] Zhaorong Li and Haizhou Ai. Texture constrained shape prediction for mouth contour extraction and its state estimation. In *IAPR 18th International Conference on Pattern Recognition*, volume 2, Hong Kong, China, August 2006.

[99] Rainer Lienhart and Jochen Maydt. An extended set of haar-like features for rapid object detection. In *IEEE International Conference on Image Processing*, pages 900–903, 2002.

[100] Christine L. Lisetti and Diane J. Schiano. Automatic facial expression interpretation: Where human interaction, articial intelligence and cognitive science intersect. *Pragmatics and Cognition, Special Issue on Facial Information Processing and Multidisciplinary Perspective*, 1999.

[101] Gwen Littlewort, Ian Fasel, Marian Stewart Bartlett, and Javier R. Movellan. Fully automatic coding of basic expressions from video. Technical report, March 2002.

[102] Huan Liu and Hiroshi Motoda. Feature transformation and subset selection. *IEEE Intelligent Systems*, 13(2):26–28, 1998.

[103] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[104] Prasanta Chandra Mahalanobis. On the generalised distance in statistics. In *Proceedings of the National Institute of Science of India*, number 12, pages 49–55, 1936.

[105] Mirko Mählisch, Matthias Oberländer, Otto Löhlein, Dariu Gavrila, and Werner Ritter. A multiple detector approach to low-resolution FIR pedestrian recognition. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 23–28, Las Vegas, USA, June 2005.

[106] Jiri Matas and Ondrej Chum. Randomized ransac. In H. Wildenauer and W. Kropatsch, editors, *Proceedings of the CVWW'02*, pages 49 – 58, Wien, Austria, February 2002.

[107] Iain Matthews and Simon Baker. Active appearance models revisited. *International Journal of Computer Vision*, 60(2):135 – 164, November 2004.

[108] Christoph Mayer and Sylvia Pietzsch. Modellbasierte Mimikerkennung in Videobildern mittels Optical Flow. Technical report, Technische Universität München, Fakultät für Informatik, Boltzmannstr. 3,85748 Garching bei München, Germany, January 2006.

[109] K. Messer, J. Matas, J. Kittler, J. Lüttin, and G. Maitre. XM2VTSDB: The extended M2VTS database. In *Audio- and Video-based Biometric Person Authentication, AVBPA'99*, pages 72–77, 1999.

[110] M. Michalowski and R. Simmons. Multimodal person tracking and attention classification, 2006.

[111] P. Michel and R. El Kaliouby. Real time facial expression recognition in video using support vector machines. In *Fifth International Conference on Multimodal Interfaces*, pages 258–264, Vancouver, 2003.

[112] Kai Nickel and Rainer Stiefelhagen. Pointing gesture recognition based on 3d-tracking of face, hands and head orientation. In *ICMI '03: Proceedings of the 5th international conference on Multimodal interfaces*, pages 140–146, New York, NY, USA, 2003. ACM Press.

[113] M. M. Nordstrøm, M. Larsen, J. Sierakowski, and Mikkel Bille Stegmann. The IMM face database - an annotated dataset of 240 face images. Technical report, Informatics and

Mathematical Modelling, Technical University of Denmark, DTU, Richard Petersens Plads, Building 321, DK-2800 Kgs. Lyngby, may 2004.

[114] Timo Ojala, Matti Pietikainen, and Topi Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):971–987, 2002.

[115] Nuria Oliver, Alex Pentland, and Francois Berard. Lafter: Lips and face real-time tracker. In *Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition*, page 123, Washington, DC, USA, 1997. IEEE Computer Society.

[116] Maja Pantic and Leon J. M. Rothkrantz. Automatic analysis of facial expressions: The state of the art. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(12):1424–1445, 2000.

[117] D. E. Pearson and J. A. Robinson. Visual communication at very low data rates. In *Proceedings of the IEEE*, volume 73, pages 795–812, April 1985.

[118] Heinz Pürer. Grundbegriffe der Kommunikationswissenschaft. In *UVK Verlagsgesellschaft mbH*, Konstanz, 2001.

[119] Ross Quinlan. Learning with continuous classes. In A. Adams and L. Sterling, editors, *Proceedings of the 5th Australian Joint Conference on Artificial Intelligence*, pages 343–348, 1992.

[120] Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, California, 1993.

[121] Deva Ramanan and David A. Forsyth. Finding and tracking people from the bottom up. *Computer Vision and Pattern Recognition*, 2:467–474, 2003.

[122] Christof Ridder. *Interpretation von Videobildfolgen zur Beobachtung artikularer Bewegung von Personen anhand eines generischen 3D Objektmodells*. PhD thesis, Technische Universität München, Fachbereich Informatik, 2000.

[123] Sami Romdhani. *Face Image Analysis using a Multiple Feature Fitting Strategy*. PhD thesis, University of Basel, Computer Science Department, Basel, CH, January 2005.

[124] Sami Romdhani, P. Torr, B. Scholkopf, and Andrew Blake. Computationally efficient face detection. In *Eighth IEEE International Conference on Computer Vision*, volume 2, pages 695–700, 2001.

[125] Henry Rowley, Shumeet Baluja, and Takeo Kanade. Neural network-based face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(1):23–38, January 1998.

[126] Stuart Russell and Peter Norvig. *Artificial Intelligence - A Modern Approach*. Prentice Hall, Upper Saddle River, New Jersey, 2003.

[127] Yunus Saatci and Christopher Town. Cascaded classification of gender and facial expression using active appearance models. In *Proceedings of the 7th International Conference on Automatic Face and Gesture Recognition*, pages 393–400, Washington, DC, USA, 2006. IEEE Computer Society.

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

[129] Björn Schuller, Matthias Wimmer, Dejan Arsic, Gerhard Rigoll, and Bernd Radig. Audiovisual behavior modeling by combined feature spaces. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 2, pages 733–736, Honolulu, Hawaii, USA, April 2007.

[130] R. Schweiger, P. Bayerl, and Heiko Neumann. Neural architecture for temporal emotion classification. In *Affective Dialogue Systems 2004, LNAI 3068*, pages 49–52, Kloster Irsee, June 2004. Elisabeth Andre et al (Hrsg.).

[131] Nicu Sebe, Michael S. Lew, Ira Cohen, Ashutosh Garg, and Thomas S. Huang. Emotion recognition using a cauchy naive bayes classifier. In *Proceedings of the 16th International Conference on Pattern Recognition*, volume 1, pages 17–20, Washington, DC, USA, 2002. IEEE Computer Society.

[132] Wolfgang Sepp. Efficient tracking in 6-DoF based on the image-constancy assumption in 3-D. In *Proceedings of the 18th International Conference on Pattern Recognition*, Hong Kong, August 2006. International Association for Pattern Recognition.

[133] Mohsen Sharifi, Mahmoud Fathy, and Maryam Tayefeh Mahmoudi. A classified and comparative study of edge detection algorithms. page 117, Los Alamitos, CA, USA, 2002. IEEE Computer Society.

[134] Elizabeth Marie Sheldon. *Virtual agent interactions*. PhD thesis, 2001. Major Professor-Linda Malone.

[135] Leonid Sigal, Stan Sclaroff, and Vassilis Athitsos. Esitmation and prediction of evolving color distributions for skin segmentation under varying illumination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, March 2000.

[136] Leonid Sigal, Stan Sclaroff, and Vassilis Athitsos. Skin color-based video segmentation under time-varying illumination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:862–877, 2004.

[137] Aaron Sloman. Beyond shallow models of emotion. *Cognitive Processing*, 1, 2001.

[138] Xuefeng Song and Ram Nevatia. Combined face-body tracking in indoor environment. In *Proceedings of the 17th International Conference on Pattern Recognition*, volume 4, pages 159–162, Washington, DC, USA, 2004. IEEE Computer Society.

[139] M. Soriano, S. Huovinen, B. Martinkauppi, and M. Laaksonen. Skin detection in video under changing illumination conditions. In *15th International Conference on Pattern Recognition*, pages 839–842, 2000.

[140] Thad Starner and Alex Pentland. Real-time american sign language recognition from video using hidden markov models. In *SCV95*, page 5B Systems and Applications, 1995.

[141] Mikkel Bille Stegmann. Active appearance models: Theory, extensions & cases. Master's thesis, Technical University of Denmark, 2000.

[142] Mikkel Bille Stegmann, B. K. Ersbøll, and R. Larsen. FAME – a flexible appearance modelling environment. *IEEE Transactions on Medical Imaging*, 22(10):1319–1331, 2003.

[143] Mikkel Bille Stegmann, R. Fisker, and B. K. Ersbøll. On properties of active shape models. Technical report, Informatics and Mathematical Modelling, Technical University of Denmark, DTU, Richard Petersens Plads, Building 321, DK-2800 Kgs. Lyngby, 2000.

[144] Mikkel Bille Stegmann, D. Pedersen, and H. B. W. Larsson. Unsupervised correction of respiratory-induced motion in 4D short-axis cardiac cine MRI. *Journal of Cardiovascular Magnetic Resonance (submitted)*, mar 2006.

[145] Mikkel Bille Stegmann and K. Skoglund. On automating and standardising corpus callosum analysis in brain MRI. In *Proceedings Svenska Symposium i Bildanalys, SSBA 2005, Malmø, Sweden*, pages 1–4. SSBA, mar 2005.

[146] Freek Stulp, Mark Pflüger, and Michael Beetz. Feature space generation using equation discovery. In *Proceedings of the 29th German Conference on Artificial Intelligence (KI)*, 2006.

[147] R. Sutton and A. Barto. *Reinforcement Learning: an Introduction*. MIT Press, 1998.

[148] Barry-John Theobald, Iain Matthews, and Simon Baker. Evaluating error functions for robust active appearance models. In *Proceedings of the International Conference on Automatic Face and Gesture Recognition*, pages 149–154, April 2006.

[149] Ying-Li Tian, Takeo Kanade, and Jeffrey F. Cohn. Recognizing action units for facial expression analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(2):97–115, February 2001.

[150] Alan M. Turing. Computing machinery and intelligence. *MIND*, 59:433–460, 1950.

[151] David Vernon. *Machine Vision: Automated Visual Inspection and Robot Vision*. Prentice Hall, 1991.

[152] V. Vezhnevets, V. Sazonov, and A. Andreeva. A survey on pixel-based skin color detection techniques. In *Graphics and Media Laboratory, Faculty of Computational Mathematics and Cybernetics*, Russia, 2003.

[153] Rita M. Vick and Curtis S. Ikehara. Methodological issues of real time data acquisition from multiple sources of physiological data. In *Proceedings of the 36th Annual Hawaii International Conference on System Sciences*, page 129.1, Washington, DC, USA, 2003. IEEE Computer Society.

[154] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition*, volume 1, pages 511–518, Kauai, Hawaii, 2001.

[155] Paul Viola, Michael J. Jones, and David Snow. Detecting pedestrians using patterns of motion and appearance. Technical Report TR-2003-90, Mitsubishi Electric Research Lab, 2003.

[156] T. Wang, H. Ai, and G. Huang. A two-stage approach to automatic face alignment. In Hanqing. Lu and Tianxu. Zhang, editors, *Proceedings of the SPIE Third International Symposium on Multispectral Image Processing and Pattern Recognition*, volume 5286, pages 558–563, September 2003.

[157] Paul Watzlawick, Janet H. Beavin, and Don D. Jackson. *Menschliche Kommunikation. Formen, Störungen, Paradoxien*. H. Huber, Bern, Switzerland, 1969.

[158] Oliver Williams, Andrew Blake, and Roberto Cipolla. Sparse bayesian learning for efficient visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1292–1304, 2005.

[159] Matthias Wimmer and Bernd Radig. Adaptive skin color classificator. In Ashraf Aboshosha et al., editor, *Proceedings of the first International Conference on Graphics, Vision and Image Processing*, volume I, pages 324–327, Cairo, Egypt, December 2005. ICGST.

[160] Matthias Wimmer and Bernd Radig. Adaptive skin color classificator. *ICGST International Journal on Graphics, Vision and Image Processing*, Special Issue on Biometrics, 2006.

[161] Matthias Wimmer, Bernd Radig, and Michael Beetz. A person and context specific approach for skin color classification. In *Procedings of the 18th International Conference of Pattern Recognition (ICPR 2006)*, volume 2, pages 39–42, Los Alamitos, CA, USA, August 2006. IEEE Computer Society.

[162] Matthias Wimmer, Freek Stulp, Sylvia Pietzsch, and Bernd Radig. Learning local objective functions for robust face model fitting. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2007. to appear.

[163] Matthias Wimmer, Freek Stulp, Stephan Tschechne, and Bernd Radig. Learning robust objective functions for model fitting in image understanding applications. In Michael J. Chantler, Emanuel Trucco, and Robert B. Fisher, editors, *Proceedings of the 17th British Machine Vision Conference (BMVC)*, volume 3, pages 1159–1168, Edinburgh, UK, September 2006. BMVA.

[164] Ian H. Witten and Eibe Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, San Francisco, 2nd edition, 2005.

[165] Richard D Worth. *Interdisciplinary Approaches to Human Communication*. Transaction Publishers, May 2003.

[166] Ming Xu, J. Orwell, L. Lowey, and D.J. Thirde. Architecture and algorithms for tracking football players with multiple cameras. *IEE Proceedings – Vision, Image and Signal Processing*, 152(2):232–241, 2005.

[167] Ming-Hsuan Yang and Narendra Ahuja. Detecting human faces in color images. volume 1, pages 127–130, 1998.

[168] Ming-Hsuan Yang, Dan Roth, and Narendra Ahuja. A SNoW-based face detector. In Sara A. Solla, Todd K. Leen, and Klaus-Robert Müller, editors, *NIPS*, pages 862–868. The MIT Press, 1999.

[169] Benjamin D. Zarit, Boaz J. Super, and Francis K. H. Quek. Comparison of five color models in skin pixel classification. In *RATFG-RTS '99: Proceedings of the International Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems*, page 58, Washington, DC, USA, 1999. IEEE Computer Society.

[170] Li Zhang, Haizhou Ai, Shengjun Xin, Chang Huang, Shuichiro Tsukiji, and Shihong Lao. Robust face alignment based on local texture classifiers. In *IEEE International Conference on Image Processing*, volume 2, pages 354–357, September 2005.