Graz University of Technology

Institute for Computer Graphics and Vision

Master's Thesis

Segmentation of Face Images

Martin Hirzer

Graz, Austria, December 2008

Thesis supervisor

Univ. Prof. DI Dr. Horst Bischof

Instructor

DI Dr. Martin Urschler

Abstract

Since the introduction of electronic personal documents in recent years the analysis of passport photographs has become an important field of research. Such photographs have to fulfill a set of minimal quality requirements defined by the International Civil Aviation Organization (ICAO). As some of the specified requirements are related to certain image regions only, these regions must be located in advance. In this work an unsupervised segmentation method for color face images is presented. The developed tool is intended to be part of an automatic passport photograph inspection framework. Our focus is on knowledge based image segmentation. Hence we developed a total variation model that allows us to incorporate prior knowledge about typical passport photographs into the segmentation process. Since uniformity of the image background is one quality requirement defined by ICAO, our tool also contains a background classifier, which decides whether the background region is uniform or not. This enables the inspection framework to reject photographs with a non-uniform background at an early stage. We have conducted several experiments on face images from two different datasets in order to evaluate the performance of our algorithm. The obtained results demonstrate that our method is fairly robust and outperforms other methods targeted at the same problem, in particular an expert system and an AdaBoost classifier.

Keywords: Segmentation, Face Images, Prior Knowledge, Total Variation, Background Classification, ICAO

Acknowledgments

At this point I would like to thank my family for giving me the opportunity to choose an educational career according to my liking, and for always supporting me during my studies. I am very grateful to all members of the Institute of Computer Graphics and Vision. Special thanks go to Prof. Horst Bischof for supervising my master's thesis, and to Dr. Martin Urschler for his continuous assistance during my work, his advices and the proof-reading of my thesis. Further on I would like to thank Dr. Thomas Pock, who had always time to answer my questions.

I am also very thankful to DI Josef Alois Birchbauer from Siemens PSE Graz, Biometrics Center, for his suggestions regarding my work. Finally I would like to thank my fellow students for the successful collaboration in many lectures, and my friends who always supported me.

This master's thesis was developed in cooperation with Siemens PSE Graz, Biometrics Center.

Contents

List of Figures

Chapter 1

Introduction

Contents

1.1 Motivation and Problem Definition

The goal of the project was to develop a fully unsupervised segmentation tool for color passport photographs, which assigns each image pixel to one of the following classes: face, hair, shoulder or background. This tool is planned to be part of an automatic passport photograph inspection framework that checks whether a passport photograph meets the minimal quality requirements defined by the International Civil Aviation Organization [28]. These requirements include conditions that apply to the whole photograph (e.g. brightness, contrast) as well as conditions that are only related to certain regions in the photograph (e.g. hair must not cover face, background must be uniform). In order to be able to examine such region-specific conditions, the individual image regions must be located in advance. This is the main function of our method. However, since one criterion for valid passport photographs is a uniform background, a background classifier is also included in the tool. The classifier decides whether the background region is uniform or not, and therefore allows the inspection framework to

reject photographs with a non-uniform background at an early stage.

The only constraint on the input images is that they must be normalized so that a person's eyes lie on predefined positions within the image, as shown in Figure 1.1. This is achieved by a process called canonization: the original image is scaled, rotated and translated in order to place the eyes correctly [49]. As a result of the transformation necessary to obtain a canonical image a so called padding frame often arises in the transformed image. This is the case if the original image is not large enough to fill the entire canonical image. Usually these missing parts are set to a predefined color and therefore are easy to segment. However, as face images with arbitrary backgrounds can contain any color, we do not identify the padding frame by color matching. Instead it must be specified by the user. Due to the fact that the padding frame has already been determined by the canonization stage this is not a problem at all in this work.

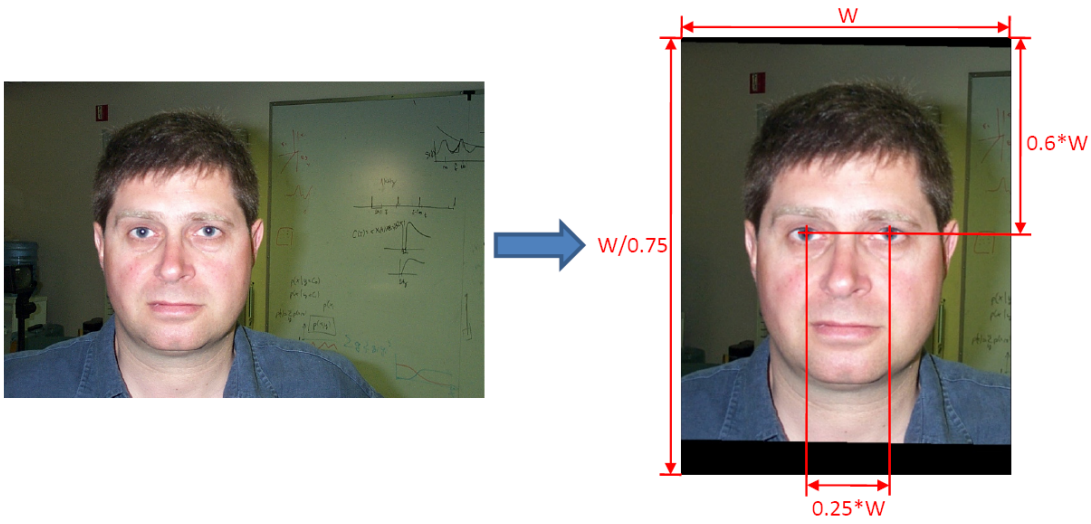


Figure 1.1: Canonization step. The original image is transformed in order to place the eyes on predefined positions. The black stripes in the right image denote the padding frame. The image is taken from [5].

From the above description of the input image format it is clear that the face region is the only region that is surely present in the image. All other regions may or may not appear in the image, depending on the image content and the canonization. Photographs of bald people, for example, do not contain a hair region. The shoulder region can also be missing, because it either has already been missing in the original image or it has got lost because of the transformation applied in the canonization stage. In photographs of people with voluminous hair the background region may be

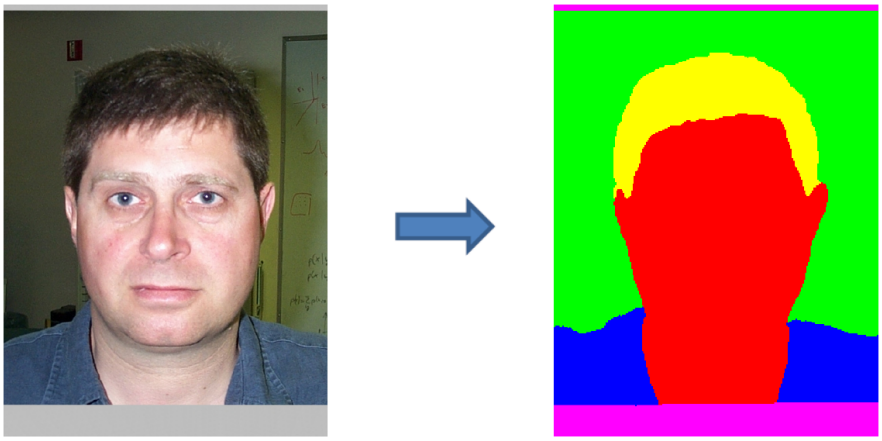


Figure 1.2: Main project goal. A canonical input image is partitioned into face, hair, shoulder and background region.

completely occluded by hair so that it is not visible in the image. Because of these reasons no assumptions on the number of regions actually present in the image can be made beforehand. Instead this information must be obtained during the segmentation process. In Figure 1.2 the main project goal is summarized.

1.2 ICAO Machine Readable Travel Documents

The International Civil Aviation Organization was formed in 1944 by 52 nations as an international institution for safe and economic air traffic. In particular the organization has established the following objectives for the period 2005 to 2010:

- Enhancing safety in global civil aviation
- Enhancing security in global civil aviation
- Environmental protection, i.e. minimizing the adverse effect of global civil aviation on the environment
- Enhancing efficiency of aviation operations
- Maintaining continuity of aviation operations
- Strengthening law in international civil aviation

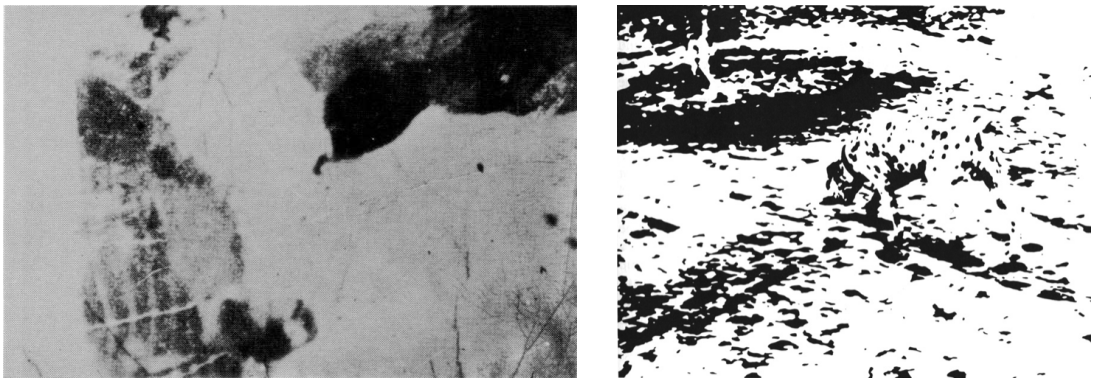
One division within the the broad field of functions of ICAO are machine readable travel documents (MRTDs). For centuries now passports have served as an instrument for the identification of people, mostly in the context of traveling and tourism. Furthermore passports can represent a certain diplomatic protection in foreign jurisdictions. There has been a variety of different forms of passports throughout history, ranging from handwritten recommendations on parchment in the early years, to standardized travel documents including pictures and biometric data. In early years, where traveling and tourism were very infrequent, the effort of checking a person's travel documents was negligible. However, in modern times globalization and mass tourism dramatically increased the administrative workload necessary for control procedures. As a result a new form of travel documents that shifts this workload from human controllers to the computer is needed. Therefore these documents have to be machine readable. In a typical scenario a human controller is equipped with a machine reader that scans passports and automatically extracts the necessary information. This information is then used to verify the authenticity of the passport, or to check a person against a watch list.

Clearly, one prerequisite for machine readable travel documents is standardization. This includes the document format as well as the stored data. Since a modern passport contains a photograph of its owner, ICAO also defined a number of conditions that passport photographs have to fulfill [29]. As already mentioned in the previous section, some of these requirements apply to the whole photograph (e.g. brightness, contrast, image format), while others are only targeted at specific image regions (e.g. hair must not cover face, background must be uniform, person must not smile). To be able to examine region-specific conditions the location of the individual image regions must be known. For example, background uniformity can only be checked after the background region has been extracted from the whole image. Thus we have to divide the image into a set of disjoint regions that correspond to objects visible in the image, a task that is known as image segmentation in computer vision.

1.3 Knowledge Based Image Segmentation

Image segmentation is the partitioning of an image into a set of disjoint regions in such a way that these regions comply with objects present in the image. While this is a rather simple task for humans, it is a very hard one for computers. The human visual

system automatically divides the field of view into semantic regions. In order to do so it uses a lot of different cues, like object appearance, edges, texture, proximity, similarity and symmetry. Taking a look at Figure 1.3, what does one see? One might not be able to identify the depicted scenes at first glance, but after a while one will probably recognize familiar objects and be able to categorize the scenes.



(a) **(b)**

Figure 1.3: Two examples of difficult image segmentation

In Figure 1.3a one can see a cow. And Figure 1.3b shows a Dalmatian placed pretty much in the image center, walking towards the shadow of a tree located in the upper left corner. But how is it possible for the human visual system to recognize objects in such distorted images? The key is prior knowledge. People who have never seen a Dalmatian, a tree or a cow before will most likely not be able to see these objects in the images. On the other hand people who are aware of these objects will most probably have no problem in identifying them.

We see that introduction of prior knowledge can significantly simplify image analysis tasks, or even make them possible in the first place. Using only simple edge or region based segmentation techniques for our complex task of segmenting color passport photographs is likely to fail. It is obvious that incorporating prior knowledge will help us considerably in solving the task, particularly as we know what to expect in typical passport images.

1.4 Outline of the Master's Thesis

In Chapter 2 we first give an overview of several classical segmentation methods. We start with threshold based techniques (Section 2.2) and describe the basic principles as well as more sophisticated algorithms, like optimal thresholding in Section 2.2.1. Next edge based methods are reviewed in Section 2.3, including edge detectors (Section 2.3.3), the snake model (Section 2.3.4) and geodesic active contours (Section 2.3.5). Region based segmentation approaches are examined in Section 2.4. In addition to basic splitting and merging methods (Sections 2.4.1 to 2.4.3) we also outline the ROI-SEG algorithm, an advanced method that achieves image segmentation by combining subsegmentation results (Section 2.4.4). In Section 2.5 we focus on total variation models, in particular the ROF model (Section 2.5.1), the TV-L^1 model (Section 2.5.2) and the weighted total variation (Section 2.5.3), on which our algorithm is based. Chapter 2 concludes with a description of two methods that aim for solving the same problem as we intend to, namely an expert system (Section 2.6.1) and an AdaBoost classifier (Section 2.6.2).

In Chapter 3 our segmentation algorithm is presented in detail. After a short overview (Section 3.1) the individual steps are explained. These are initialization (Section 3.2), region growing (Section 3.3), program flow and definition of start regions (Section 3.4) and post-processing (Section 3.5). Finally we describe the background classifier that is also included in our segmentation tool in Section 3.6.

Chapter 4 is devoted to our experiments and their results. After outlining the used dataset (Section 4.2) and defining the error metrics (Section 4.3), we compare our algorithm to an algorithm based on an expert system and another technique based on an AdaBoost classifier in Sections 4.4 and 4.5. After all Section 4.6 shows some qualitative results of our method.

Finally in Chapter 5 we make some concluding remarks regarding our segmentation algorithm and give an outlook of possible future improvements of our method.

Chapter 2

Related Work

Contents

2.1 Image Segmentation

Image segmentation is a fundamental problem in computer vision and has been a research topic for many years. It is one of the most important techniques for the analysis of image data. The goal is to divide an image into regions that have a strong correlation with objects of the real world contained in the image. In [48] Sonka et al. differentiate between two kinds of segmentation, namely complete and partial segmentation. A complete segmentation of an image R consists of a finite set of disjoint regions $R_1, ..., R_N$, which correspond uniquely to image objects:

$$R = \bigcup_{i=1}^{N} R_i, \qquad R_i \cap R_j = 0 \quad \forall\, i \neq j \tag{2.1}$$

To achieve a complete segmentation, higher level processing that incorporates knowledge about the scene is usually necessary. The exception are tasks where contrasted objects located on a uniform background have to be segmented, like assembly parts, blood cells or printed characters. Such problems can be solved with a simple thresholding approach, no knowledge is needed. In a partial segmentation, on the other hand, the extracted regions do not correspond directly to image objects, but instead are cues to aid higher level processing. A typical example for partial segmentation is the detection of parts of object boundaries, which then can be grouped by a higher level process in order to obtain regions complying with objects. Hence the common approach for a complete segmentation of images containing complex scenes is a partial segmentation, followed by one or more higher level processes.

Sonka et al. also divide segmentation methods into three classes according to the dominant features they use. The first class encompasses methods that use global knowledge about an image, usually in form of a histogram of image features. The second class contains edge based approaches. These methods detect edges in an image and try to group them meaningfully into edge chains that correspond to object borders. Region based segmentation methods form the third group. Their goal is to find regions that comply with image objects directly. Note that the second and third group solve a dual problem. Each closed boundary represents a region, and each region can be described by its closed boundary. Due to their different nature edge based and region based algorithms give different results in most cases. Thus one can combine their segmentation results in a single description structure, like a region adjacency graph. In such a graph nodes represent regions, and graph arcs represent adjacency relations according to detected region borders.

2.2 Thresholding

Gray value thresholding is the simplest and oldest segmentation technique. It is only suitable for very simple problems, where objects are characterized by a rather constant reflectivity or light absorption of their surfaces and differ clearly from the background. In its simplest form thresholding transforms an input image f into a binary image g as

stated in the following formula, where T is the gray value threshold:

$$\begin{aligned} g(i,j) &= 1 \quad \text{for} \quad f(i,j) \geqslant T \\ g(i,j) &= 0 \quad \text{else} \end{aligned} \tag{2.2}$$

As one can imagine, choosing an appropriate value for T is crucial for successful image segmentation. However, a single threshold for the whole image (global threshold) will fail in many cases. Only very simple problems, for example segmentation of dark objects on a bright background under controlled illumination conditions, can be solved with such an approach. But if gray value variations within objects or background occur, a more advanced technique, like adaptive thresholding, is required. Adaptive thresholding uses a threshold value that varies over the image as a function of local image characteristics. This can be realized, for instance, by dividing the image into subimages and defining a local threshold in each subimage. If a subimage does not provide enough data for reliable threshold determination, its threshold is interpolated from the thresholds of neighboring subimages. Finally each subimage can be processed using its own threshold value.

There exist many modifications to basic thresholding as defined by Equation (2.2). One option is to use a set D of gray values instead of a single threshold, so that the image is segmented into regions of pixels with gray values from the set and background otherwise:

$$\begin{aligned} g(i,j) &= 1 \quad \text{for} \quad f(i,j) \in D \\ g(i,j) &= 0 \quad \text{else} \end{aligned} \tag{2.3}$$

This segmentation method is called band thresholding. Additionally to segmenting images this thresholding definition can be used as border detector. Given an image of dark objects on bright background one just has to search for pixels with gray values that are darker than the background, but brighter than the objects. Defining D in this way will result in detection of object borders.

Of course band thresholding is not restricted to a single set D of gray values. One can define several disjoint sets of gray values $D_1, D_2, ..., D_N$, and assign pixels a certain number according to the set that contains their gray value, e.g. $1, 2, ..., N$. As a result the segmented image is no longer binary, but rather consists of a limited set of gray values.

2.2.1 Threshold Detection Methods

All thresholding methods mentioned so far have one thing in common. They rely on a suitable choice of the threshold value(s). Choosing an appropriate value can be simplified greatly if some a priori knowledge about the segmentation result is available. An example for this is the segmentation of letters on a printed text sheet. If we know that the letters cover $1/p$ of the sheet area, we can use the image histogram to determine a threshold value T such that $1/p$ of all image pixels have gray values less than T, and the remaining pixels have gray values larger than T. Unfortunately, we usually do not have such concrete prior knowledge, and more complex threshold detection methods are needed.

Normally these advanced methods try to derive a suitable threshold value from the analysis of the image histogram. For example, the histogram of an image of dark objects on a bright background is bi-modal (see Figure 2.1). It shows two peaks, one at darker gray values originating from the objects, and one at brighter gray values caused by the background. To separate objects from background, it makes intuitive sense to define the threshold as the gray value that has a minimum histogram value between the two peaks. For multi-modal histograms several thresholds are necessary, one at every minimum between two maxima. However, locating peaks in an image histogram is not always that easy as in case of the example shown in Figure 2.1. Often it is very difficult or even impossible to decide whether a local maximum in the histogram is significant or not (see [42] for details). Furthermore one has to bear in mind that even a histogram with distinctive peaks and well chosen thresholds does not guarantee a correct segmentation result, because an image histogram does not provide any information about the local distribution of the gray values. For example, an image consisting of only two equally large regions, one being white and the other being black, has almost the same histogram as a salt-and-pepper noise image. Hence it is imperative to check segmentation results that are derived solely from histogram analysis, using no other image properties.

Despite this disadvantage many histogram based segmentation approaches have been developed. One option, for instance, is to suppress pixels that have a high gradient magnitude, i.e. border pixels, when constructing the image histogram. This results in deeper valleys between histogram peaks and simplifies the determination of suitable threshold values. Further examples of histogram transformation methods can be found in [53] and [54]. Another method is optimal thresholding, an approach that

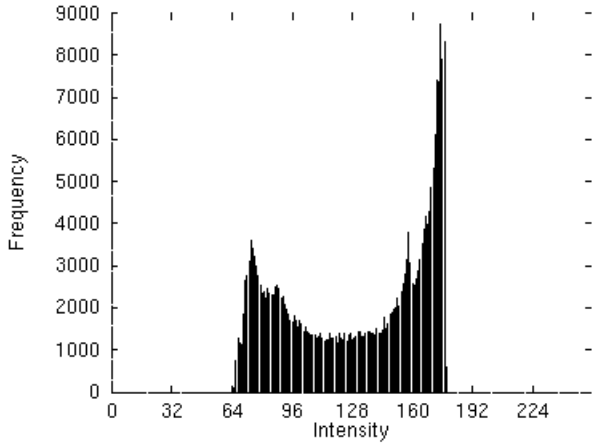


Figure 2.1: Bimodal histogram taken from [48]

tries to minimize the segmentation error. Here the histogram is approximated by a weighted sum of probability densities with normal distribution, as shown in Figure 2.2. Depending on the number of normal distributions used to represent the histogram, one or more optimal thresholds can be determined. They are defined as the gray values corresponding to the minimum probabilities between the maxima of the normal distributions. More threshold detection algorithms, like histogram concavity analysis, entropic methods and relaxation methods, are described in [46].

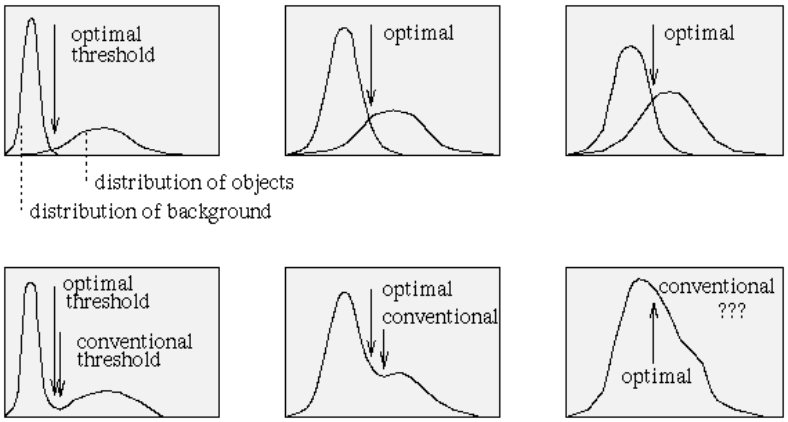


Figure 2.2: Histogram approximated by two normal distributions, one derived from the objects present in the image, and one derived from the background. The figure is taken from [48].

2.3 Edge Based Methods

Edge based segmentation methods use information about image edges in order to divide an image into regions corresponding to objects. They consist of two steps. First edges are detected by an edge detector. But the detector result alone is usually not sufficient to segment an image. Thus the found edges must be combined into edge chains that comply better with object borders in a second processing step. Note that the resulting edge chains represent a partial segmentation only in many cases, and further processing is necessary to extract full regions from edge chains.

2.3.1 Border Detection

Edge based segmentation algorithms encounter two main problems. The first one is the detection of edges where no real object border exists, primarily caused by image noise and object textures. The second case are missing edges at object border locations, resulting from noise and occlusions.

To overcome these problems several methods have been developed. In [34] Kundu and Mitra propose edge image thresholding, a very simple approach for reducing the detection of outlier edges caused by noise. It is based on edge magnitudes. Edge pixels with a magnitude smaller than a certain threshold are removed, because they are likely to be the result of noise present in the image. Consequently only strong edges, which most probably correspond to object borders, are used when constructing edge chains. As with threshold based segmentation methods the difficulty is the selection of a suitable threshold value. Again, some a priori knowledge, like the expected edge length, can simplify the choice. Other algorithms use hysteresis to filter the output of an edge detector. Two thresholds T_1 and T_2, $T_1 > T_2$, are defined. Edge pixels with a magnitude greater than T_1 are assumed to be valid, i.e. not induced by noise, and edge pixels with a magnitude smaller than T_2 are considered to be the result of image noise and thus are removed. Pixels that have an edge magnitude in the range $[T_2, T_1]$ are examined more precisely. If they border another pixel that is already markers as edge, then they are marked as edge too. Otherwise they are removed. But filtering edges on the basis of their magnitudes is not the only possibility. One can use different filtering criteria, like the edge length, or the average strength of an edge.

2.3.2 Region Construction from Borders

If we have achieved a complete segmentation in the previous step, which means that the detected borders already partition the image into regions, we are finished. However, in most cases the result after border detection is only a partial segmentation consisting of border parts rather than closed borders. Extracting regions from a partial segmentation is a difficult task that requires some higher level knowledge. An example for such a region construction method is the superslice method [38]. It tries to find relevant regions by comparing detected borders to regions that have been derived by gray value thresholding of the original image. Several different thresholds are used to find suitable regions. Finally those regions that best match the detected region borders are accepted.

2.3.3 Edge Detectors

Since edge detectors play a crucial role in edge based image segmentation, we will give a brief overview of them in the following sections. Edge detectors are used to locate intensity changes within images. As is well known from mathematics, such changes can be described with the help of derivatives. Because an image function depends on two variables, the image coordinates, edge detectors use partial derivatives. One form of describing edges is the image gradient, that is a vector consisting of the partial derivatives of a function. At every function value it points in the direction of the largest change of the function, and its length is proportional to this largest variation. In case of an image function $f(x, y)$ the gradient is defined as follows:

$$\text{grad } f(x,y) = \nabla f(x,y) = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right) \tag{2.4}$$

The gradient magnitude and direction are calculated as:

$$|\nabla f(x,y)| = \sqrt{\left(\frac{\partial f}{\partial x} \right)^2 + \left(\frac{\partial f}{\partial y} \right)^2} \qquad \text{and} \qquad \phi = \arg \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right) \tag{2.5}$$

Edges are characterized by the same properties, magnitude and direction, which can be derived directly from the gradient. The edge magnitude is identical to the gradient magnitude, and the direction of the edge equals the gradient direction minus 90°. Since digital images are discrete, the derivatives used in Equation (2.5) must be approximated by differences of neighboring pixels.

For the detection of edges in an image several edge detectors have been developed. Sonka et al. divide them into three classes. The first group are operators that detect edges by searching for maxima of the first derivative of the image function. An example is the well known Sobel operator [47], which is capable of determining the edge magnitude as well as the edge direction. The second class encompasses operators that search for zero-crossings of the second derivative of the image function in order to find edges. Due to the underlying principle these detectors create closed loops of edges, i.e. there are no gaps that have to be filled in a post-processing step. This property was seen as an advantage in original papers, but there are also applications where this behavior is a drawback. Two examples for such second derivative based operators are the Marr-Hildreth [37] and Laplacian of Gaussian (LoG) [30] detector. The problem with algorithms based on second derivatives is their increased sensitivity to noise compared to methods using first derivatives. The Laplacian of Gaussian operator reduces this vulnerability to noise by smoothing the image with a Gaussian filter prior to computation of second derivatives. Furthermore it can be approximated very efficiently by the difference of two Gaussian filters with different standard deviations, the so called Difference of Gaussians (DoG) operator (see [36] for details). The last category of detectors are parametric edge models. They are based on the assumption that the image intensity function is a sampled and noisy version of an underlying continuous or piecewise continuous function. This continuous image function can be estimated from the discrete image, and certain image properties, like edges, can then be derived from this estimate. To represent the piecewise continuous image function so called facets are used. This leads to the term facet model [22–24] for such an image representation.

2.3.3.1 Scale Space

As stated in the previous section, many edge detectors compute differences between pixels in a local neighborhood. Of course the size of this neighborhood affects the detection result. But what is the right size? The answer to this question depends on the objects that are investigated, and choosing the right scale in advance can be very difficult or even impossible. The solution to this problem is the analysis of an image at different scales. These different scales are obtained by smoothing the original image

stepwise, usually with a Gaussian filter:

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{\left(x^2+y^2\right)}{2\sigma^2}} \tag{2.6}$$

Each of the smoothed images represent one scale in the so called scale space, ranging from the original image, which contains all details, to a very blurred image, where only coarse image structures remain. If, for example, the task is to segment richly textured objects, spurious edges arising from the object textures can be eliminated easily by using a more smoothed image in the scale space instead of the original one. Among the detectors mentioned in the previous section the Laplacian of Gaussian operator is the only one that already takes scale into account, so no scale space is needed when using this detector. The principle is the same. The image is filtered with a Gaussian kernel prior to edge detection. By changing the filter's standard deviation the Laplacian of Gaussian operator is tunable for edges at different scales.

2.3.3.2 Canny Edge Detector

In [6] Canny proposes an edge detector that also makes use of scale space theory. It is intended to be optimal according to the following three criteria:

Good detection: Important edges should not be missed, whereas the detector should not respond to distracting edges

Good localization: The distance between the position of the detected and the real edge should be minimal.

Single response: A single image edge should only result in a single detected edge. Multiple responses to single edges should be minimized.

The first step towards edge detection is Gaussian filtering of the image in order to eliminate unimportant edges, just like described in the scale space theory. After smoothing the image the normal to the edge $\boldsymbol{n}$ is estimated for every pixel as:

$$\boldsymbol{n} = \frac{\nabla(G * f)}{|\nabla(G * f)|} \tag{2.7}$$

Now edges are located by searching for local maxima of the image f convolved with the operator G_n in the direction $\boldsymbol{n}$. G_n is the first derivative of a Gaussian G, again,

in the direction $\boldsymbol{n}$:

$$\frac{\partial G_n}{\partial \boldsymbol{n}} * f = 0, \qquad G_n = \frac{\partial G}{\partial \boldsymbol{n}} \tag{2.8}$$

This leads to:

$$\frac{\partial^2 G}{\partial \boldsymbol{n}^2} * f = 0 \tag{2.9}$$

The search for local maxima in the direction perpendicular to the direction of the edge is called non-maximal suppression. The next step is calculating the edge strength throughout the image. The strength s, that is the magnitude of the gradient of the image function f, is evaluated according to the following formula:

$$s = |\nabla(G * f)| \tag{2.10}$$

Finally the resulting edge image is filtered to get rid of spurious edges. This is done by thresholding with hysteresis, as described in Section 2.3.1.

2.3.4 Snakes

So far we have considered the problem of edge based image segmentation as a two stage task, consisting of an object border detector that is followed by some higher level process for region construction. Active contour models, or snakes, form a different approach. They were invented in 1987 by Kass et al. [33] and can be used to solve a variety of computer vision problems, for instance object segmentation, stereo image matching and object tracking. A snake is an energy minimizing spline $C(s)$ that is guided through the image by internal forces, which depend on the shape of the snake, and external forces, which depend on the image. Thus the snake's energy depends on its shape and location. Kass et al. defined the energy functional to be minimized as follows:

$$E_{Snake} = \int_0^1 E_{internal}(C(s)) + E_{image}(C(s)) + E_{constraints}(C(s))\, ds \tag{2.11}$$

$E_{internal}$: This is the internal energy of the spline due to bending. It can act as a membrane or a thin plate. The behavior is controlled by the parameters α (elasticity) and β (stiffness). If β is set to zero at a certain point of the snake, the snake becomes second order discontinuous at that point and develops a corner

there.

$$E_{internal} = \alpha(s) \left| \frac{\partial C}{\partial s} \right|^2 + \beta(s) \left| \frac{\partial^2 C}{\partial s^2} \right|^2 \tag{2.12}$$

E_{image}: This energy term is derived from the image data that is currently covered by the snake and attracts the snake to salient image features. The energy term is modeled as a weighted combination of three different functionals:

$$E_{image} = w_{line} E_{line} + w_{edge} E_{edge} + w_{term} E_{term} \tag{2.13}$$

E_{line} is a very simple functional, pulling the snake towards lines. Depending on the sign of w_{line}, the snake follows either light lines or dark lines. The second functional E_{edge} attracts the snake to contours with high image gradient. The last functional E_{term} is a termination term that finds terminations of line segments and corners.

$E_{constraints}$: This energy term allows the user to define additional constraint forces designed to help solving a certain problem.

But there are also disadvantages of the snake model. In [7] Caselles et al. reveal two problems. First of all in its original form the model is not able to change the topology of the evolving contour. For example, if the initial snake surrounds several image objects that should be segmented, it is not possible to capture them correctly. Instead the result will most likely be a curve similar to the convex hull of the objects. The second limitation is that the snake model is not geometric, i.e. the energy defined in Equation (2.11) depends on the parameterization of the curve and is not directly related to the geometry of the objects.

2.3.5 Geodesic Active Contours

Geodesic active contours (GAC), proposed by Caselles et al. in [7], are a further development of the snake model of Kass et al. They are defined as the following energy optimization problem:

$$\min_C \Big\{ E_{GAC}(C) \Big\} = \min_C \Big\{ \int_0^{L(C)} g(|\nabla I(C(s))|)\, ds \Big\} \tag{2.14}$$

$L(C)$ is the Euclidean length of the curve C, and g is an edge function that aims at stopping the evolving curve when it arrives at object borders. The obtained edge magnitude values have to lie in the interval $(0, 1]$. Caselles et al. define the edge function g as follows:

$$g(|\nabla I|) = \frac{1}{1 + |\nabla(G_\sigma * I)|^p}, \qquad \text{with } p = 1 \text{ or } 2 \tag{2.15}$$

Another possibility is to use a function that is optimized for natural images, such as the function proposed by Huang et al. in [27]:

$$g(|\nabla I|) = e^{-\eta|\nabla I|^\kappa}, \qquad \text{e.g. with } \eta = 0.1 \text{ and } \kappa = 1.0 \tag{2.16}$$

Note that the user has to define additional constraints, since $C = 0$ always minimizes the GAC energy. The main advantage of geodesic active contours over snakes is that they are geometric and independent of the topology of a problem. This means that there is no need to estimate crucial parameters, like α and β in Equation (2.12), and that the evolving contour can adapt to the topology present in the image. For instance, if a geodesic active contour is applied to segmentation problem containing several objects, the evolving contour automatically splits and merges, according to the objects. Another advantage of geodesic active contours is their well founded mathematical framework, which makes them applicable to many different applications. However, the energy defined in Equation (2.14) has one drawback, it is non-convex. Consequently minimizing it will not lead to a globally optimal solution, but instead a local minimum will be found. This means that the result depends on the initialization of the geodesic active contour.

2.4 Region Based Segmentation

In the previous sections we have described how an image can be segmented by finding borders between regions. The methods presented in this section detect regions directly. Generally region based segmentation methods are better suited in case of noisy or richly textured images, where correct border detection is extremely difficult. To be able to decide if a certain pixel belongs to a certain region a homogeneity criterion is needed. It can be based on gray values, color, texture, shape, etc. Having defined a homogeneity

criterion the resulting regions of the segmented image must satisfy the following two conditions, where $H(R_i)$ is a binary homogeneity evaluation of the region R_i:

$$H(R_i) = \text{TRUE}, \qquad \forall\, i = 1, 2, ..., N \tag{2.17}$$

$$H(R_i \cup R_j) = \text{FALSE}, \qquad \forall\, i \neq j, \quad R_i \text{ adjacent to } R_j \tag{2.18}$$

2.4.1 Region Merging

Region merging begins with an image full of small start regions that fulfill Equation (2.17), but usually do not fulfill Equation (2.18). The easiest way of initializing region merging is to define a start region in every pixel. After initialization adjacent regions that can be combined without violating the chosen homogeneity criterion are being merged. This goes on until no more neighboring regions can be connected maintaining condition Equation (2.17). There exist different versions of this method differing in the way how the algorithm is initialized and the merging criterion.

2.4.2 Region Splitting

Region splitting is the opposite of region merging. It starts with the whole image being one region. This single region usually does not meet criteria Equation (2.17) and therefore has to be split into smaller regions. The splitting continues until all regions fulfill Equations (2.17) and (2.18). Although this approach seems to be dual to region merging, generally the results returned by the two methods are different, even if the same homogeneity criterion is used.

2.4.3 Region Splitting and Merging

As stated in [25], a combination of region splitting and merging can definitely improve the segmentation result. Such methods use a pyramid image structure and work as follows. If a region on a certain pyramid level is not homogeneous, it is split into four regions on the level below. If, on the other hand, there are four similar regions on a pyramid level that have the same parent in the upper level, these regions are merged. When no more splitting or merging is possible any two adjoining regions that can be combined into a homogeneous region are merged, even if they belong to different

pyramid levels or have different parents. Finally an optional last step can remove small regions by uniting them with the most similar adjacent region.

2.4.4 ROI-SEG

In [15] ROI-SEG, an unsupervised color segmentation method based on the combination of subsegmentation results, is presented. Starting from detected regions of interest (ROIs) the proposed algorithm builds several differently focused subsegmentations of the same image. First, the color distribution within each region of interest is modeled by a Gaussian mixture model. Then Bhattacharyya distance values measuring the similarity between pixel colors and Gaussian mixture models are calculated for all pixels in order to sort them. These values are then forwarded to a modified version of the maximally stable extremal region (MSER) detector, which returns a set of connected regions, all having a similar color appearance as the corresponding region of interest. These connected regions are then combined by analyzing a local quality measure. The authors use the mean Bhattacharyya distance value of every connected region for this task. Pixels are assigned the label of the region with the locally lowest mean Bhattacharyya distance. The whole process is illustrated in Figure 2.3.

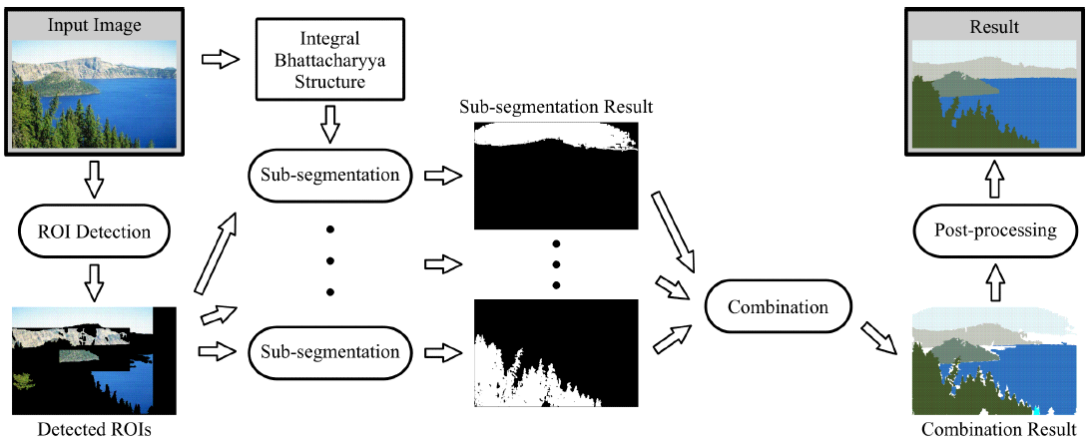


Figure 2.3: ROI-SEG program flow. The figure is taken from [15].

2.5 Total Variation Models

In the previous section we have discussed several classical segmentation methods, ranging from threshold based approaches over edge based algorithms to region based algorithms. For our task of segmenting color face images a combination of edge and region based techniques seems to be the right choice. On the one hand we want to use color and texture information, and on the other hand the segmented regions should coincide with detected region borders. We have decided to use a geodesic active contour model and add a supplemental region term to incorporate color, texture and shape information. This results in the following energy optimization problem:

$$\min_C \Big\{ E_{GAC\,Region} \Big\} = \min_C \Big\{ \underbrace{\int_0^{L(C)} g(C)}_{\text{GAC}} + \lambda \underbrace{\int p(C)}_{\text{Region}} \Big\} \tag{2.19}$$

But, as already stated in Section 2.3.5, geodesic active contours have one major drawback. The defined energy is non-convex, so that the minimization algorithm is likely to get stuck in a local minimum. To still find a satisfying solution a good initialization is necessary. In order to overcome this disadvantage we have developed a model that is based on a weighted total variation (TV) norm, which was introduced by Bresson et al. in [3]. As will be described in Section 2.5.3, this weighted TV-norm minimizes the same energy as a geodesic active contour. However, it is convex and thus allows for calculating a globally optimal solution.

In recent years variational methods have been applied very successfully to a number of inverse problems in computer vision. Such inverse problems are tasks where model parameters have to be estimated from given data, as opposed to direct problems, where data is obtained from a given model. An example of an inverse problem is the reconstruction of an image x from a blurred version Ax of it to which some random noise n has been added:

$$y = Ax + n \tag{2.20}$$

Clearly, reconstructing the original image x from the observed image y is very difficult, sometimes even impossible. To be able to obtain a reasonable solution at least some information about the operator A and the noise n must be available. Inverse problems are typically ill-posed. The opposite are well-posed problems. According to

Hadamard [21] a problem is well-posed if the following three conditions are satisfied:

1. A solution is existent.
2. The solution is unique.
3. The solution depends continuously on the data, i.e. it is stable.

2.5.1 ROF Model

Rudin, Osher and Fatemi were the first who applied variational methods to a computer vision problem, in particular edge preserving image denoising [45]. The goal is to recover the original image $u(x, y)$ from the observed, noisy image $u_0(x, y)$. The relation is stated in Equation (2.21), where n is some additive noise:

$$u_0(x, y) = u(x, y) + n(x, y) \tag{2.21}$$

For a continuous representation denoising can be formulated as a least squares approximation of u:

$$\min_u \int_\Omega |u_0(x, y) - Au(x, y)|^2 \, dx, \qquad \Omega \text{ ... image domain} \tag{2.22}$$

First the image u is transformed via a linear operator A, a blur for example, and afterwards some random noise n is added. This is an inverse problem, just as described before. Image denoising is then connected to the following constrained minimization problem (ROF model), with $\int_\Omega |\nabla u| \, d\Omega$ being the total variation of u:

$$\begin{aligned} &\min_u \int_\Omega |\nabla u| \, d\Omega, \\ \text{constrained by} \quad &\int_\Omega u \, d\Omega = \int_\Omega u_0 \, d\Omega \qquad \text{and} \qquad \int_\Omega (u - u_0)^2 \, d\Omega = \sigma^2 \end{aligned} \tag{2.23}$$

The first constraint ensures that the additive noise has zero mean, whereas the second constraint determines the standard deviation of the noise. Unfortunately, the ROF model in its original form is non-convex.

In [10] Chambolle et al. restated the image denoising problem. By changing the original constraint $\int_\Omega (u - u_0)^2 \, d\Omega = \sigma^2$ to $\int_\Omega (u - u_0)^2 \, d\Omega \leqslant \sigma^2$, Chambolle et al. obtained the following unconstrained, non-convex minimization problem, where $\lambda > 0$

is a Lagrange multiplier:

$$\min_u \Big\{ E_{ROF} \Big\} = \min_u \Big\{ \int_\Omega |\nabla u| \; d\Omega + \frac{1}{2\lambda} \int_\Omega \big(u - u_0\big)^2 \, d\Omega \Big\}, \tag{2.24}$$

The first term in Equation (2.24) is named regularization term and minimizes the variance of u, but preserves discontinuities, like edges, at the same time. The second term is called data fidelity term and minimizes the difference between u and u_0. It uses a L^2-norm, and thus the model is often referred to as TV-L^2-model. However, using the L^2-norm in the data fidelity term results in a loss of contrast in the denoised image, a severe drawback of this version of the ROF model.

2.5.2 TV-L^1 Model

In order to overcome the drawbacks of the ROF model researchers have developed different modifications of the original ROF model (see [1] for an overview). For instance, in [11] Chan et al. propose using the L^1-norm instead of the L^2-norm for the data fidelity term in Equation (2.24):

$$\min_u \Big\{ E_{TV\text{-}L^1} \Big\} = \min_u \Big\{ \int_\Omega \Big| \nabla u \Big| \, d\Omega + \lambda \int_\Omega \Big| u - u_0 \Big| \, d\Omega \Big\} \tag{2.25}$$

The derived model is called TV-L^1 model and has some attractive features. It outperforms the ROF model in removing impulse noise, like salt-and-pepper noise for example, and it preserves the contrast that is present in the image. This makes the TV-L^1 model very useful for shape denoising, as presented by Nikolova et al. in [39], and selection of features of a certain scale, as proposed by Chen et al. in [12]. Consequently the order in which structures in the image disappear depends completely on their geometry (e.g. area, length) and not on their contrast. Unfortunately these favorable properties come with a main disadvantage. The TV-L^1 model is not strictly convex, which means that the global minimum is not unique. The existence of more than one globally optimal solution makes the optimization task more difficult.

2.5.3 Weighted Total Variation

The weighted total variation approach was introduced by Bresson et al. in [3]. The novelty of their method compared to the TV-L^1 model is the weighting g of the total

variation term:

$$\min_u \Big\{ E_{TV_g} \Big\} = \min_u \Big\{ \int_\Omega g \, |\nabla u| \; d\Omega + \lambda \int_\Omega \Big| u - u_0 \Big| \, d\Omega \Big\} \tag{2.26}$$

$$\int_\Omega g \, |\nabla u| \; d\Omega = TV_g(u) \qquad \text{... weighted TV} \tag{2.27}$$

In [4] Bresson et al. proved that if u is a characteristic function 1_{Ω_C} of a set Ω_C whose boundary is denoted C, and u is allowed to vary continuously between $[0, 1]$, and g is an edge detector, Equation (2.27) describes a geodesic active contour as defined in Equation (2.14):

$$TV_g(u = 1_{\Omega_C}) = \int_\Omega g \, |\nabla 1_{\Omega_C}| \; d\Omega = \int_C g \, ds = E_{GAC}(C) \tag{2.28}$$

Furthermore these constraints turn the weighted total variation into a convex functional, so that a globally optimal solution can be derived. This is a major advantage over geodesic active contours described in Section 2.3.5, which are non-convex.

2.6 Segmentation of Face Images

For many years now face detection and localization systems have been the subject of research activities, and a lot of different approaches have been developed. A very brief overview can be found in [50]. The invented systems include methods that use skin color in order to locate a face in an image [8, 26, 51], statistical methods like Active Shape Models (ASM) and Active Appearance Models (AAM) [16, 17, 35], neural networks [43, 44], boosting [52, 55] and support vector machines [40]. Since the main purpose of these algorithms is face detection and localization, most of them return only the locations and dimensions of rectangles containing detected faces. However, some of the mentioned methods provide more detailed information, like Active Shape and Active Appearance Models, which return location and contour for every face.

In contrast to the variety of different face detection and localization approaches, the problem of segmenting face images has not gained much attention in the research community up to now. As stated by Subasic et al. in [50], no method for segmentation of passport photographs into face, hair, shoulder, background and padding frame has been developed preliminary to their expert system approach.

2.6.1 Expert System

In [50] Subasic et al. present an rule based expert system that uses domain knowledge for segmentation of canonical face images. Their method consists of two main steps, a low-level segmentation algorithm and the expert system itself. Figure 2.4 illustrates the proposed method.

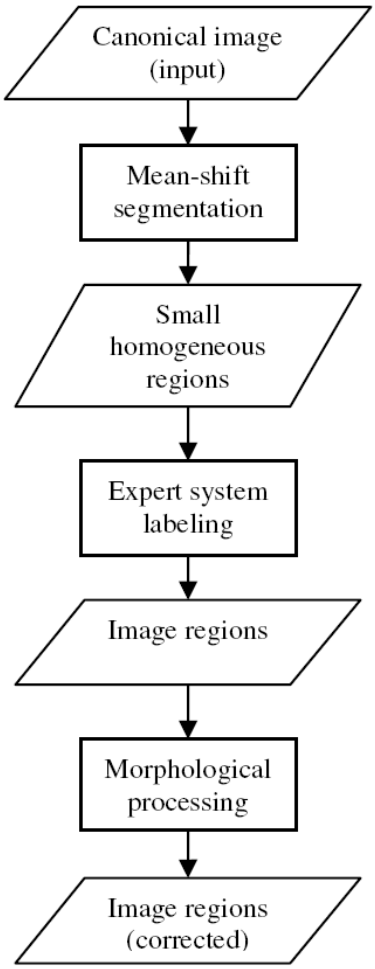


Figure 2.4: Block diagram of the proposed expert system taken from [50]

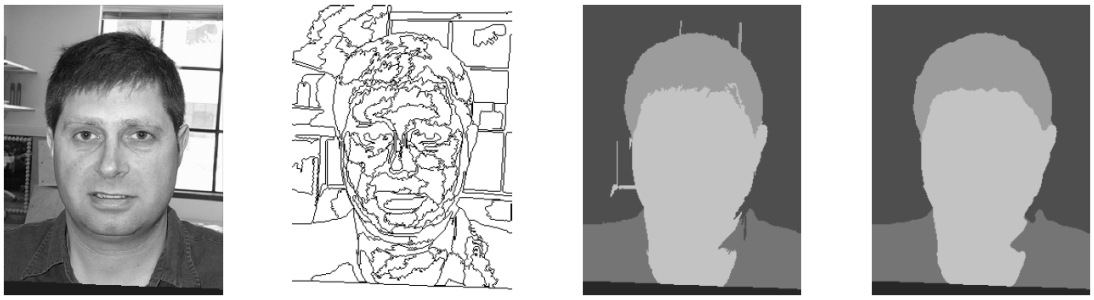
In the first stage the image is divided into a set of homogeneous regions by mean-shift filtering [13, 14]. The user has to determine the trade-off between the number of obtained regions and the likeliness that this first segmentation discards important details. Less initial regions reduce the processing time for the second stage, but increase the probability of losing essential details at the same time.

After partitioning the image into a set of homogeneous regions, the expert system follows. It consists of a set of rules, which are designed with knowledge about the scene in typical passport photographs, and facts, which are determined by a number of attributes, like region properties (e.g. color and texture) and region neighborhood information. A rule consists of two parts, the "if" part and the "then" part. The "if" part specifies the facts that have to be fulfilled in order to make the rule applicable, and the "then" portion defines the actions that are performed when executing the rule. Typical actions are of course the assignment of labels (face, hair, shoulder, background and padding frame) to certain regions of the initial segmentation and the adjustment of parameters of other rules. The inference engine of the expert system automatically examines the current facts, and then chooses one of the applicable rules for execution. If several rules can be chosen, the rule with the highest priority is taken. Usually rules are ordered in a way so that individual image regions are segmented consecutively, namely in the following sequence: padding frame, face, hair, shoulder and background. Of course applying a rule can change the facts. In this case the inference engine has to check the applicability of all rules again. However, it is not possible for a rule to change the facts in a manner that would activate rules of an already segmented image region. For example, a rule for the hair region can not cause a rule for the face region to become applicable again. Also rules usually do not change the affiliation of already labeled regions, except for two special cases. A face region in ear's position can be changed into a hair region, and a hair region in typical shoulder position can be changed into a shoulder region.

The padding frame, if existing, is very easy to segment, because it has a predefined color. Therefore it is processed first. Afterwards the segmentation process continues with the face region, since this is the only region that is certainly present in the image. The algorithm picks a seed region between the person's eyes and starts color based region growing. A region grows by adding neighboring regions with similar properties. Rules for such a region growing process are defined for every region, but simple region growing is not sufficient in most cases. Thus the majority of rules of the expert system are designed to handle special situations. Examples for this kind of rules are the relaxation of parameters in case of poor illumination, or the detection of beards based on high entropy and expected position within the image. After the face region has been segmented, the hair region follows. Again, a seed region, this time located above the already segmented face, initiates a region growing process, and specific situations

are treated with appropriate rules. The next region to be segmented is the shoulder region. The algorithm chooses a seed region that lies below the face region and starts the same process as for the previous two regions. The last region is the background region. With the region growing approach a correct segmentation of the background region is very difficult because of the large variations in the appearance of this region. Hence it is determined by elimination. After having segmented all other regions, the remaining, still unlabeled image areas are considered background.

Finally the segmentation result is refined by morphological post-processing, which consists of the following steps: closing of holes in regions, gray value morphological opening, dilation and erosion, as well as filtering based on the region size. A segmentation example of the expert system is shown in Figure 2.5.



(a) Input image **(b)** Mean-shift segmentation **(c)** Result of expert system **(d)** Result after morphological post-processing

Figure 2.5: Segmentation example of the expert system taken from [50]

2.6.2 AdaBoost Classification

In [18] a method for segmentation of passport photographs based on AdaBoost (short for adaptive boosting) classification is presented. Again, the goal is to divide a canonical face image into face, hair, shoulder, background and padding frame region. Like the expert system described in the previous section this method uses a two step approach. The first stage is the same as for the expert system, a mean-shift algorithm that partitions the image into a set of homogeneous regions.

The second stage consists of several classifiers for the individual image regions. Like in case of the expert system, the padding frame can be segmented easily, whereas the segmentation of the background region is very difficult. Consequently the background

region is determined by elimination. This results in three regions that have to be classified, in particular the face, hair and shoulder region. For each of these regions a separate classifier is trained using the AdaBoost method [52]. The classifiers rely on several different features, like color values, position coordinates, region probability maps (see Section 3.2.1 for details) and various texture features (e.g. statistical values calculated from intensity values or wavelet filters combined with oriented edge detectors).

To train the classifiers a set of hand labeled face images is necessary. First every training image is segmented into homogeneous regions by the mean-shift algorithm, and the features used for classification are calculated for each of these regions. Then the three classifiers for face, hair and shoulder region are trained using the AdaBoost method. AdaBoost is an iterative machine learning algorithm that constructs a strong classifier as a linear combination of simple (weak) classifiers. In every iteration the weak classifier that produces the smallest error on the current training set is chosen and added to the strong classifier in order to improve the overall performance. After each iteration the weights of training samples that have been classified correctly in the previous iteration are decreased, whereas the weights of wrongly classified samples are increased. In this way the algorithm always focuses on hard training samples when adding a new weak classifier. When the overall error is small enough or the maximum number of iterations has been reached the training is stopped.

To segment an image all three trained classifiers are applied to the homogeneous regions of the image obtained by mean-shift segmentation. A homogeneous region is then labeled according to the classifier with the highest response, provided that this maximum response is greater than a certain threshold. Homogeneous regions that received no strong response are labeled as background. Finally the segmentation result is refined by morphological post-processing.

2.7 Discussion

In this chapter we have presented various segmentation methods, ranging from simple thresholding over edge based approaches to region based methods. As already mentioned in Section 1.3, these simple approaches are likely to fail in our case. Therefore we need more sophisticated methods, like total variation models. Such variational models

have been applied very successfully to a number of different computer vision problems, among others image segmentation. Thus we have derived a total variation model that allows us to combine an edge based approach, in particular a geodesic active contour, with a region-term encompassing color, texture and shape information. A disadvantage of variational models is the computational cost involved in finding an adequate solution. How we solve the derived model, and how knowledge is incorporated into the segmentation process is described in the next chapter.

We have concluded this chapter with a description of two methods that aim at solving the same problem as we intend to. One is an expert system that introduces knowledge into the segmentation process by well defined rules. The other uses classifiers trained with AdaBoost. Since both methods are especially designed for segmentation of face images in the context of machine readable travel documents specified by ICAO, we will compare our algorithm to them in Chapter 4. We expect our method to achieve a higher performance than these algorithms, because our approach uses prior knowledge right from the beginning of the segmentation process, i.e. the result does not depend on an initial low-level segmentation. While we incorporate knowledge at all stages of the algorithm, which are definition of start regions, region growing and post-processing, the expert system and the AdaBoost classifier only use knowledge in the labeling stage, but not in the segmentation stage itself. In the next chapter a detailed description of our method follows.

Chapter 3

Segmentation Algorithm

Contents

3.1 Overview

In this chapter we describe our segmentation algorithm in detail. We start with a short overview of the individual stages of the proposed method in this section. Next in Section 3.2 some preliminary steps that have to be carried out prior to segmentation are outlined. Then we present the region growing mechanism, which is based on the energy optimization problem stated in Equation (2.19). We show how this optimization problem can be solved. This is done in Section 3.3. After that the whole program flow is described in Section 3.4. We illustrate how we achieve a segmentation by defining start regions and letting them grow. The last segmentation stage is post-processing, which is explained in Section 3.5. Finally we conclude Chapter 3 with the description of our background classifier in Section 3.6.

In order to solve the task of segmenting color passport photographs we use a knowledge-based approach. The basic stages of the proposed method are as follows:

Initialization: Before the actual segmentation process can start some preliminary steps are necessary. These are calculation of shape information based on ground truth data, image pre-processing and calculation of gradient and texture information.

Definition of start regions: This stage and the next stage, namely region growing, are performed iteratively. First the region growing process is initialized by defining a start region. Then this start region grows and, after the growing process has stopped, the resulting region is used to guide the definition of a new start region for another region. For example, after the face region has been found, a small region above the face region is used to initialize the hair region growing process. In this way knowledge about the scene is incorporated into the segmentation process.

Region growing: The region growing process uses gradient, color, texture and shape information and is based on a total variation framework. Again, knowledge about the individual regions in the image is used to guide the growing process.

Post-processing: The post-processing stage corrects the segmentation result based on general as well as region-specific rules designed with knowledge about the scene. General rules, for example, include removal of small outlier regions and morphological processing. An example of a region-specific rule is the removal of hair regions that have no contact to the face region.

Background classification: The background classifier is an addition to the segmentation tool. It allows the passport photograph inspection framework to reject images with a non-uniform background at an early stage. To determine whether the background region is uniform or not, our classifier examines the standard deviation and the gradient magnitude within the background region.

As one can see, the proposed segmentation algorithm incorporates prior knowledge whenever possible, in particular in the definition of start regions, the region growing process and the post-processing stage. In the following sections all five steps are described in detail. Figure 3.1 illustrates the whole program flow of our algorithm. Most of the face images used throughout this chapter are taken from [5].

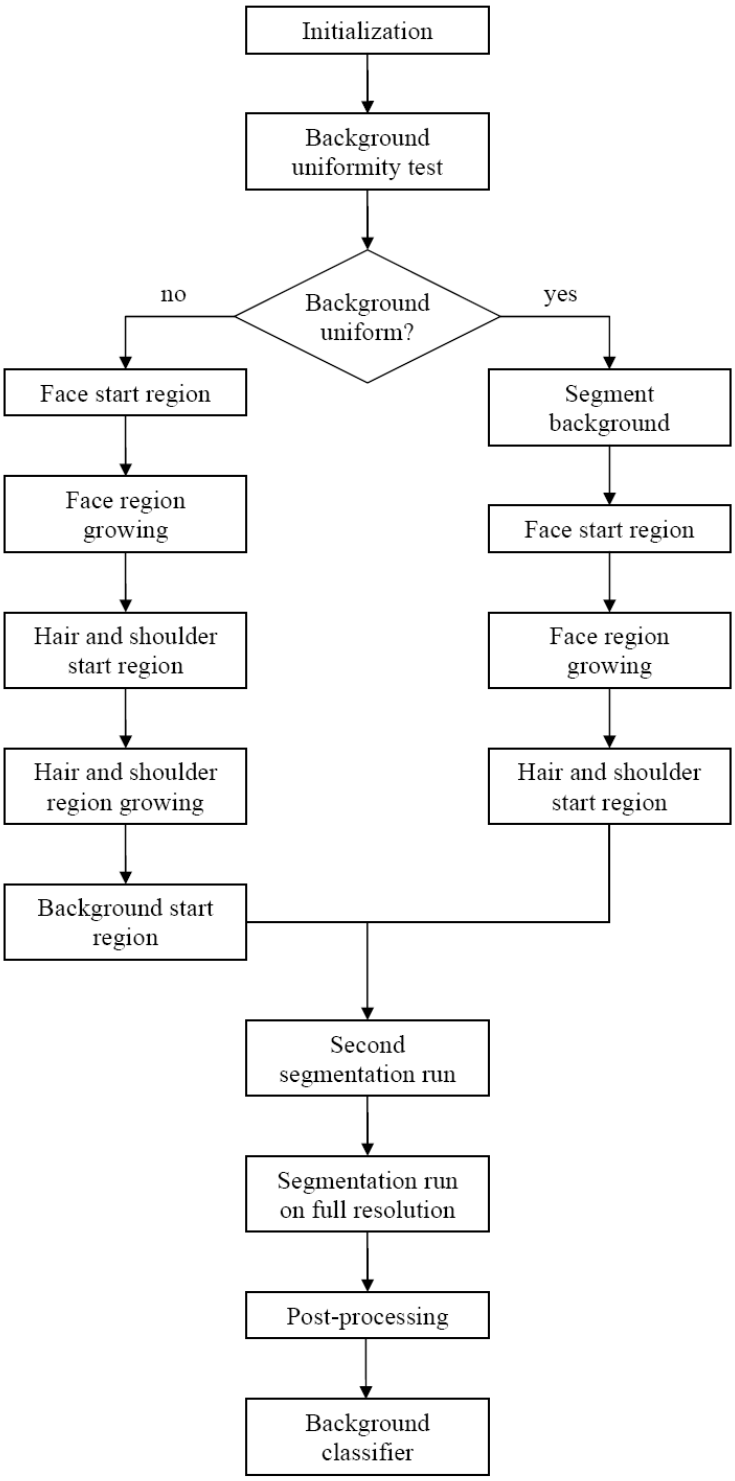


Figure 3.1: Program flow of our algorithm

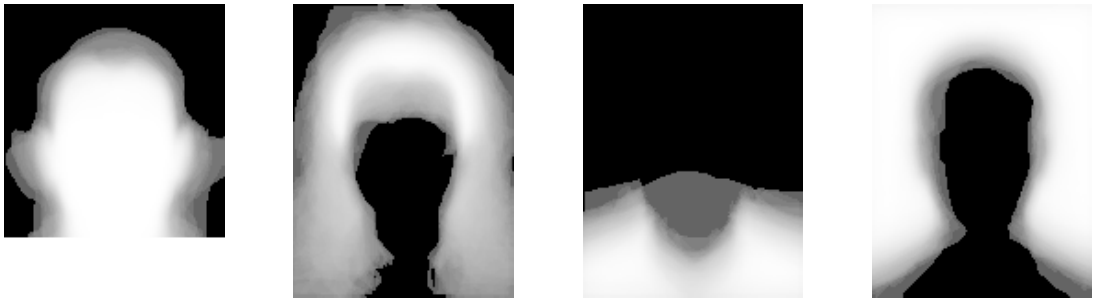
3.2 Initialization

3.2.1 Calculation of Shape Information

To increase the performance of our segmentation tool, we use knowledge about the region distribution in typical canonical face images: the face is located in the center, the hair adjoin to the upper head, the shoulders are underneath the face, and the background is usually located in the upper image corners and, depending on the hairstyle, can pass down the image on both sides of the face. This knowledge must be incorporated in the segmentation process in order to avoid regions to be found in completely uncharacteristic image areas. For example, it would make no sense to detect shoulder regions located above the head, or to find face regions near the image corners only because of color similarity. Since we have to deal with arbitrary backgrounds, such situations are not uncommon. To solve this problem, our method uses a shape probability map for each region. The probability map is two-dimensional and has the same size as the image. For each location in the image the map contains a value representing the probability that the corresponding region appears at this location. In this way the probability map supports regions in proper places, whereas regions in uncommon places are attenuated.

To calculate the probability maps, ground truth data is needed. We use 439 hand labeled face images in order to obtain the maps. A counting algorithm iterates through all ground truth images and constructs the probability maps for all regions simultaneously. First each probability map is initialized, so that all values are zero. Then for every image those values in a probability map covered by the corresponding region are incremented. When all images have been processed, all values are divided by the number of images. After that values near one indicate places where a certain region has appeared many times, whereas values near zero represent untypical image areas for this region. Figure 3.2 shows the probability maps of the face, hair, shoulder and background region. As one can see, some minor corrections of the probability maps are necessary. For the face region the probability is increased in the lower image area. This removes the shape penalty for wider necks. Another correction has to be applied to the background probability map, which shows a lower probability close to the image borders because of the padding frame, which is often present in the images from our dataset. To correct this, the background region probability is increased near the image

border.



(a) Face **(b)** Hair **(c)** Shoulder **(d)** Background

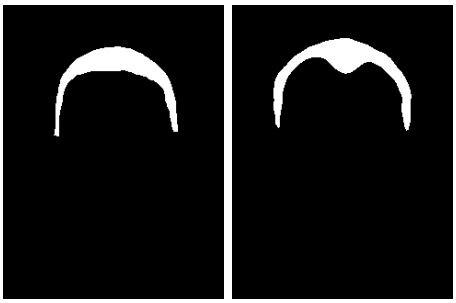
Figure 3.2: General shape probability maps calculated from 439 hand labeled face images. The higher the probability of a pixel is the whiter it appears in the image.

A special case is the hair region. While the shape probability concept works well for the face and shoulder region, the hair region cannot be represented appropriately by a single shape probability distribution, because the hairstyle can differ significantly from person to person. Figure 3.2b shows the problem: while the probability map calculated from all 439 images is suitable for personal photographs of people with rather short hair, it causes large segmentation errors on images showing people that wear long hair. As one can see in Figure 3.2b, the image area with the highest probability, that is the area covered by most hairstyles, represents short hair very well. In contrast to this the probability for longer hair is rather small in this shape probability map, causing the segmentation algorithm sometimes to fail on images of persons with long hairstyle. Especially if the image background is cluttered, the lower hair region is likely to be labeled as background region.

To overcome this problem, we use three different shape probability maps for the hair region: one for short, one for medium and one for long hair. Since the shape of the hair region directly affects the shape of the background region (the background region is large for small hair regions and vice versa), also three different shape probability maps are calculated for the background region. The calculation for these probability masks is carried out in the same manner as described before. The sole difference is that for each probability mask only the corresponding images (short, medium or long hair) take part in the counting process. To avoid having to classify every image by hand, several hair masks are used for each of the three different hairstyles. The class of the hair

mask that best fits the hand labeled hair region determines to which probability mask an image contributes. The best fit is defined in the sense of minimum overlap error. A logical XOR-operation is performed on both regions (hair mask and hand labeled hair region), and the number of set pixels in the operation's result describe the overlap error. Figure 3.3 shows some of the different hair masks, and Figure 3.4 depicts the different shape probability maps for the hair and the background region respectively.

The same hairstyle classification procedure is carried out during the segmentation process. There it decides which shape probability maps should be activated based on a first guess of the hair region present in the image. This is described in more detail in section Section 3.4.2.3.



(a) Short hairstyle

(b) Medium hairstyle

(c) Long hairstyle

Figure 3.3: Some of the hair masks used for hairstyle classification

3.2.2 Image Pre-Processing

At this stage the input image undergoes some basic pre-processing steps: color space transformation, contrast enhancement and median filtering.

(a) Hair and background for short hairstyle

(b) Hair and background for medium hairstyle

(c) Hair and background for long hairstyle

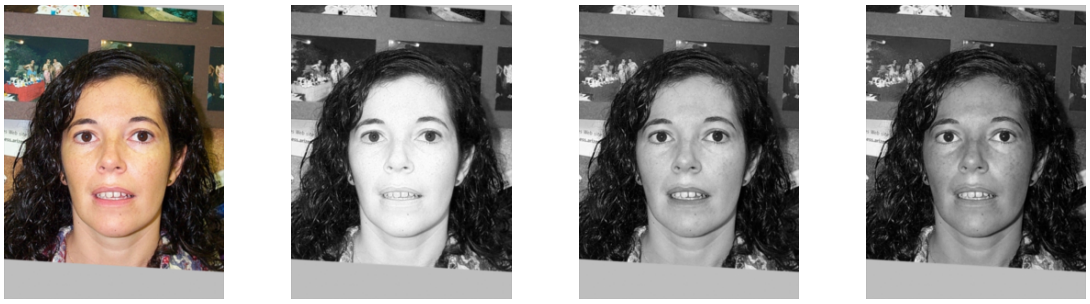
Figure 3.4: Special shape probability maps calculated from 439 hand labeled face images

3.2.2.1 Color Space Transformation

First, the image is transformed from the RGB to the Lab color space (Figure 3.5). While images consist of a red (R), green (G) and blue (B) channel in the RGB space, a Lab image is represented as luminance (L) and two color channels (a, b). The luminance L ranges from 0 (black) to 100 (white), the color channel a from -127 (green) to 128 (red), and the color channel b from -127 (blue) to 128 (yellow). The reason for this color space transformation are the drawbacks of the RGB space for color analysis. As outlined by Vezhnevets et al. in [51], the RGB space suffers from highly correlated color channels, significant perceptual non-uniformity and mixing of luminance and chrominance data. Perceptual uniformity is a color space property favored in skin color detection systems. Since skin color is rather a perceptual phenomenon than a physical property of an object, a color representation similar to the color sensitivity of the human visual system seems to be well suited for such systems. In perceptually uniform color spaces a small change of a component's value is perceived approximately equally across the range of

that value. Lab is a color space that has this property.

Furthermore luminance and chrominance data are separated in this color space. Many of the works in skin color modeling and detection try to reduce the dependency of skin color in images on illumination conditions by dropping the luminance component and, by doing so, even gain speed. However, Vezhnevets et al. point out that the omission of the luminance component does not improve the discrimination of skin and non-skin color. Kakumanu et al. even conclude in [32] that dropping the luminance component actually reduces the skin detection performance. In our case dropping the luminance component has not been a choice anyway, because a complete segmentation of the image is required, instead of just detecting skin pixels. To distinguish, for instance, several different gray levels (e.g. gray background, white shirt and black hair), the luminance component is necessary as it holds significant information.



(a) Image **(b)** R channel **(c)** G channel **(d)** B channel

(e) L channel **(f)** a channel **(g)** b channel

Figure 3.5: Image with RGB and Lab color channels

3.2.2.2 Contrast Enhancement

The second pre-processing step is contrast enhancement. The face images from our test dataset showed an unbalance in the Lab channels. It seems that the luminance

channel L of typical face images has a rather good contrast (except for outlier images taken under very bad illumination conditions), whereas the two color channels a and b are very dull. Since a region's color distribution is described by a Gaussian model, the segmentation process would mainly be driven by the dominant luminance channel. Contrast enhancement performed on each channel separately eliminates this problem [20], as depicted in Figure 3.6.

(a) L channel **(b)** a channel **(c)** b channel

Figure 3.6: Color channels of contrast enhanced Lab image

3.2.2.3 Median Filtering

Finally a median filter with kernel size $[3, 3]$ is applied to the image. It reduces both image noise and diversity within regions, but preserves borders at the same time. The noise reduction is especially important for the calculation of the gradient image described in the next section. Figure 3.7 shows the pre-processed image.

Figure 3.7: Pre-processed image

3.2.3 Calculation of Gradient and Texture Information

The region growing process uses gradient and texture information among others in order to segment a region. Hence a gradient magnitude image is calculated from the input image. To reduce the influence of unimportant edges, the input image is filtered with a Gaussian kernel prior to gradient calculation. Using the following equation proposed by Huang et al. in [27] for natural images we obtain the gradient image (Figure 3.8a):

$$g(|\nabla I|) = e^{-\eta|\nabla I|^{\kappa}}, \qquad \text{e.g. with } \eta = 0.1 \text{ and } \kappa = 1.0 \tag{3.1}$$

Then we derive the texture information from the gradient image. The texture information will only be used as an additional feature to color in the hair segmentation process. Here it provides a good differentiation criterion, since hair is richly textured in most cases. Hence it is sufficient to distinguish between image areas with rather dense and image areas with quite sparse texture. A typical example where texture plays an important role is the segmentation of gray hair on a uniform, gray background. But it also helps to improve the differentiation between hair and face region, because skin usually does not show much texture. To obtain the texture information, we calculate the local standard deviation of the gradient image. Each pixel in the texture image contains the standard deviation of a local neighborhood around the corresponding pixel in the gradient image. Finally the texture values are scaled in order to range from 0 to 255. Figure 3.8b illustrates the texture image.

(a) Gradient **(b)** Texture

Figure 3.8: Gradient image and texture image derived from it

3.3 Region Growing

For better understanding the region growing process is described at this point before the other sections. As already mentioned, the basic idea is to define a start region and then extend it to image areas with similar color and texture properties respectively. In addition the image gradient and knowledge about the shape of the expected regions are incorporated into the growing process. All in all the growing process is based on the following four features:

- Color
- Texture
- Gradient
- Shape

For our growing algorithm we combine a geodesic active contour model with a region model that incorporates color, texture and shape information into the growing process, as stated in Equation (2.19). Due to its non-convexity we replace the GAC-term in this equation with the weighted total variation $\mathrm{TV_g}$ according to Equation (2.28) and obtain the following energy optimization problem:

$$\min_u \Big\{ E_{TV_g\, Region} \Big\} = \min_u \Big\{ \underbrace{\int_\Omega g\, |\nabla u|\; d\Omega}_{\mathrm{TV_g}} + \lambda \underbrace{\int_\Omega u f\, d\Omega}_{\mathrm{Region}} \Big\} \tag{3.2}$$

For the weighting g we use the edge measure $g(|\nabla I|) = e^{-\eta |\nabla I|^\kappa}$, which is optimized for natural images. With the parameter λ the influence of the $\mathrm{TV_g}$-term and region-term on the segmentation result can be controlled. A small λ means that the result is primarily determined by the $\mathrm{TV_g}$-term, whereas a high λ makes the region-term to the main contributor. We define the function f as a probability ratio that forces the segmentation to partition the image into homogeneous regions. The involved probabilities encompass color, texture and shape information for the corresponding region:

$$\min_u \Big\{ E_{TV_g\, Region} \Big\} = \min_u \Big\{ \underbrace{\int_\Omega g\, |\nabla u|\; d\Omega}_{\mathrm{TV_g}} + \lambda \underbrace{\int_\Omega u \log \frac{p_2}{p_1}\, d\Omega}_{\mathrm{Region}} \Big\} \tag{3.3}$$

The two probabilities p_1 and p_2 in Equation (3.3) represent the foreground, that is the currently growing region and the background or remaining image respectively. Thus we have a formulation in which one region competes against another region, often referred to as background. Note that in this context the term background does not refer to the background region that we want to segment in the passport photograph, but to the antagonist of the growing region under investigation.

3.3.1 Solving the TV_g-Region Model

Solving total variation models is a demanding task. The reason for this is the non-differentiability of the L^1-norm in the TV-term at zero. Many different approaches exist, ranging from explicit time marching algorithms to graph cut methods. A short overview with corresponding references can be found in [41]. We decided to use an iterative method proposed by Chambolle et al. in [9]. It is based on introducing a second variable v and leads to the following convex energy optimization problem:

$$\min_{u,v}\Big\{E_{TV_g\,Region}\Big\} = \min_{u,v}\Big\{\int_\Omega g\Big|\nabla u\Big|\,d\Omega + \frac{1}{2\theta}\int_\Omega (u-v)^2\,d\Omega + \lambda\int_\Omega v\log\frac{p_2}{p_1}\,d\Omega\Big\} \quad (3.4)$$

Introducing a second variable v leads to a third term in the TV_g-region model, the connection term $\frac{1}{2\theta}\int_\Omega (u-v)^2\,d\Omega$. The parameter θ controls the influence of the TV_g-term and the region-term respectively. The minimization task is now split into two steps. First the energy is minimized in terms of u with v being fixed, and then the energy is minimized in terms of v with u being fixed. These two steps are iterated until convergence:

$$1. \qquad \min_u\Big\{\int_\Omega g\,|\nabla u|\,d\Omega + \frac{1}{2\theta}\int_\Omega (u-v)^2\,d\Omega\Big\} \quad (3.5)$$

$$2. \qquad \min_v\Big\{\frac{1}{2\theta}\int_\Omega (u-v)^2\,d\Omega + \lambda\int_\Omega v\log\frac{p_2}{p_1}\,d\Omega\Big\} \quad (3.6)$$

According to Chambolle et al. Equation (3.5) can be solved by using a dual variable $\boldsymbol{p} = \frac{\nabla u}{|\nabla u|}$:

$$v = \text{constant} \qquad \begin{aligned} u^{n+1} &= v + \theta\,\mathrm{div}\,\boldsymbol{p} \\ \boldsymbol{p}^{n+1} &= \frac{\boldsymbol{p}^n + \frac{\tau}{\theta}\nabla u}{1 + \frac{\frac{\tau}{\theta}|\nabla u|}{g}} \end{aligned} \quad (3.7)$$

In practice the timestep τ has to be less or equal than $\frac{1}{4}$ in order to achieve convergence. For Equation (3.6) we obtain:

$$u = \text{constant} \qquad \begin{aligned} &\frac{1}{\theta}(v-u) + \lambda \log \frac{p_2}{p_1} \stackrel{!}{=} 0 \\ &\Rightarrow v = u - \lambda\theta \log \frac{p_2}{p_1}, \qquad v = \max(0, \min(1, v)) \end{aligned} \tag{3.8}$$

After every iteration of the region growing process (Equations 3.7 and 3.8) the probabilities p_1 and p_2 can be updated according to the actual segmentation. However, to decrease the runtime of our algorithm new probability values are only calculated after significant changes in the region topology. Furthermore we also keep track of the variation of u and v for all pixels in the image. Pixels where the variation of these variables is very low are considered to be steady, and thus they are excluded from further iterations. In this way we gain a notable speedup, especially for larger images.

3.3.2 Probability Calculation

Equation (3.3) just allows two regions to compete against each other, but we have to segment up to four different regions. The easiest way of solving this problem is to let the individual regions grow successively. But this leads to a dependency of the result on the growing order. Earlier growing regions have an advantage over later ones. The region that grows first has the highest chance of adapting itself to yet unlabeled areas, while regions growing afterwards would not come across so many still unlabeled pixels. Hence the first region would grow most easily, whereas the last region would be most hindered, because many of the unlabeled image areas would already be occupied by the other regions. It is of course possible for a later growing region to displace another region that has grown earlier. However, this is rather unlikely, because the earlier grown region has already adapted itself to the newly added image areas, which means that its probability p_1 in these areas has risen.

To avoid that one of the regions is preferred, the regions grow alternately. We start with the first region and perform one iteration of the growing process, then we move on to the next region and again perform one iteration. This process continues until all regions have completed the first iteration. After that the growing algorithm starts the second iteration on all regions, and so on. In this way we minimize the influence of the growing order on the result.

3.3.2.1 Region Probability p_1

To obtain the probability p_1 in Equation (3.3), we build a Gaussian model in the feature space (see [32] for a survey of different distribution modeling techniques). As stated by Vezhnevets et al. in [51], parametric modeling methods such as Gaussian models are well suited for problems with limited training data due to their interpolation and generalization ability. Hence for our task of extending rather small start regions (the training data) a parametric method is well suited. The feature space is either three dimensional (three color channels) or four dimensional (three color channels and one texture channel), depending on the growing region. The model parameters, which are mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$, are derived from all pixels that are currently inside the region using the following equations:

$$\boldsymbol{\mu} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{c}_i, \qquad \boldsymbol{\Sigma} = \frac{1}{n-1}\sum_{i=1}^{n} (\boldsymbol{c}_i - \boldsymbol{\mu})(\boldsymbol{c}_i - \boldsymbol{\mu})^T \tag{3.9}$$

These parameters are updated every few iterations during the growing phase. Having calculated the Gaussian model the joint probability density function (pdf), defined as

$$p(\boldsymbol{c}) = \frac{1}{(2\pi)^{\frac{N}{2}} |\boldsymbol{\Sigma}|^{\frac{1}{2}}} e^{-\frac{1}{2}(\boldsymbol{c}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{c}-\boldsymbol{\mu})}, \tag{3.10}$$

gives us the desired probability p_1. Here $\boldsymbol{c}$ is a three or four dimensional feature vector respectively, and $p(\boldsymbol{c})$ represents the probability that a pixel with a certain color (and texture) belongs to the region from which the Gaussian model was derived.

3.3.2.2 Background Probability p_2

To derive the probability p_2 in case of several simultaneously growing regions, we use their Gaussian models. The calculation is almost the same as in case of the growing region's probability term p_1. The only difference is that we now have to combine multiple Gaussian models. We us a simple maximum operation, because it is fast and gives good results. For each pixel p_2 is the maximum of the probability values derived from the models of the current background regions, i.e. all regions except the currently growing one. In this way the growing region encounters maximum resistance and will only extend to image areas that have a smaller probability for all other regions. Note that there are no additional computational costs involved in building the Gaussian

models for calculating p_2, because usually, if multiple regions are present in the image, we let all of them grow simultaneously. And this means that we need these models anyway for calculation of the regions' probabilities p_1.

Furthermore we also have to deal with the case of only one growing region. For example, when extending the face start region, we do not have any knowledge about the other regions. Hence we can not define a competing region for the growing face region. Instead we simply define a fixed threshold for p_2. With this value we can adjust how easy or hard it is for the region to grow. The higher the value the harder it is for the region to extend.

3.3.2.3 Multimodal Probability Calculation

For the face, hair and shoulder region a single Gaussian model in the feature space is usually sufficient to represent the region properties well. However, the situation is different for the background region. While a uniform background region can of course also be represented by a single Gaussian model, such a model is likely to fail in case of a non-uniform background region. For example, if there is a great color diversity present in the background, the Gaussian model might adapt itself to a subset of these colors only. As a result image areas containing one of the remaining colors might be covered by another region with a more appropriate Gaussian model. On the other hand it is also possible that the model adapts to the whole set of different colors and becomes very general. Hence the background region can easily displace other regions due to its unspecific color model.

To avoid these problems, we use multiple Gaussian models for the background region. Using the mean-shift segmentation algorithm we identify all different subregions within the background region. Then for each of these subregions a Gaussian model is derived. The final probability p_1 for every pixel is obtained in the same way as the probability values p_2 when multiple regions are growing simultaneously (Section 3.3.2.2). It is defined as the maximum among all probability values derived from the subregion models. Figure 3.9 shows how the background start region is partitioned into subregions after applying the mean shift segmentation algorithm. After the background region has grown, a new subdivision is performed in the next probability update step. However, the background region can easily displace other regions due to its multimodal probability calculation. Imagine, for instance, that a few pixels corresponding to a

person's hair are covered by the background region. If these few pixels are turned into one of the background region's subregions by the mean-shift algorithm, the hair region would most likely be replaced by the background region. That is why we define a minimum subregion area. Only subregions found by the mean-shift algorithm that are big enough are considered in the probability calculation.

(a) Background start region

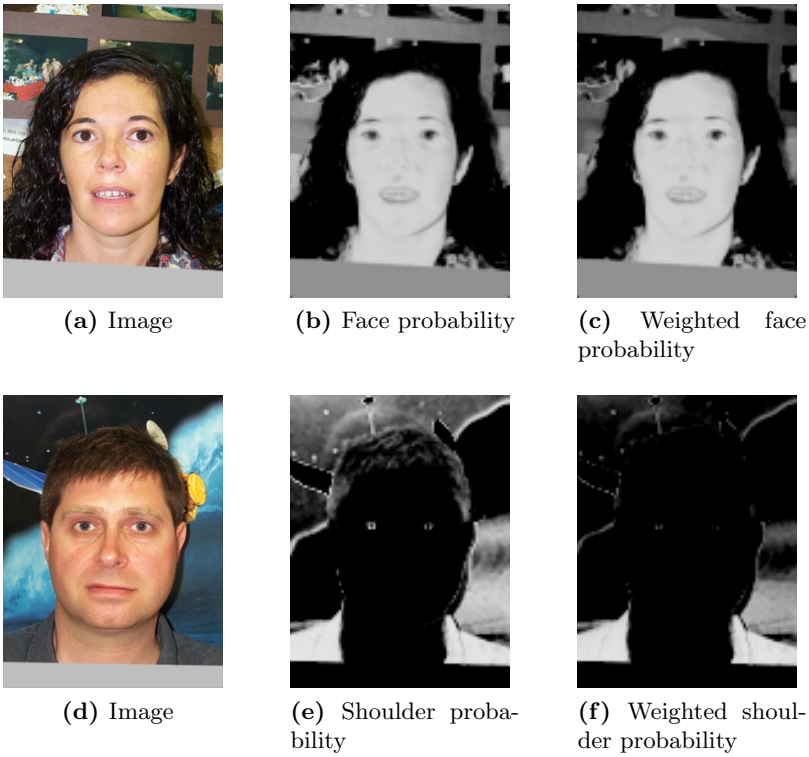
(b) Subregions within background start region

(c) Grown background region

Figure 3.9: The left image shows the image overlaid with the background start region (green). In the middle image one can see various subregions detected by the mean shift algorithm. On the right the grown background region and its subregions are presented. Note that the different subregion colors serve only as a visualization tool and have no further meaning. For the regions that we want to segment (as well as their start regions) the following colors are used throughout this document: face is red, hair is yellow, shoulders are blue and background is green. The padding frame is depicted in purple.

3.3.2.4 Incorporation of Shape Information

The last step of the probability calculation is the incorporation of shape information. This is achieved by weighting the calculated probabilities p_1 with the corresponding shape probability maps described in Section 3.2.1. Since the probability values for p_2 are derived from the values of p_1 (except for the case when a fixed threshold is used), the shape information is also included there. In Figure 3.10 the effect of weighting the probabilities is presented. One can see how a region's probability is diminished in improper image areas. For example, the face probability is reduced in outer image areas, whereas it remains nearly unchanged in the center.



(a) Image **(b)** Face probability **(c)** Weighted face probability

(d) Image **(e)** Shoulder probability **(f)** Weighted shoulder probability

Figure 3.10: The upper row shows the effect of incorporating shape information into the probability calculation for the face region. Note how the probability is reduced in improper image areas. The same effect is presented for the shoulder region in the lower row.

3.4 Program Flow and Definition of Start Regions

3.4.1 Overview

Before segmenting a certain image region, an initial region, the start region, must be defined. This initial region serves as starting point for the region growing process. The only constraint on the input face images is that they must be canonical so that a person's eyes lie on predefined positions within the image. These positions are usually chosen in a way to ensure that the face appears centered in the image. Furthermore, the face region is the only region that is certainly present in the image. As already stated in Section 1.1, all other regions may or may not appear in the image. So the only assumption that we can make at the beginning of the segmentation process is that there is a face located in the middle of the image.

According to this the algorithm defines a start region in the image center. This start region is now used to initiate the region growing process in order to find the face region. After the growing process has converged, two new start regions are defined. One start region is located directly above the actual face region and serves as starting point for the hair region. The other starting region is located directly underneath the actual face region and corresponds to the shoulder region. Again, a region growing process is initiated that allows both regions to grow simultaneously. Finally after convergence of the process, the start region for the background region is defined in the upper image corners, and a further growing run is started. Now the first segmentation run is complete. Every region has grown once. During this first segmentation run we use a rather strict parameter setting for the growing process. We do so because we want to avoid that regions extend too far and grow into other regions. Having a first estimation for the regions the result is refined by a second segmentation run. The parameters for the growing process can be relaxed during this second run, because all regions in the image now grow simultaneously, competing against each other.

At this point it makes sense to recap the parameters that influence the region growing process. The previous description of the growing algorithm pointed all involved parameters out. On the one hand there are fixed parameters that are never changed during the segmentation process (e.g. τ and θ), on the other hand some of them are used to steer the region segmentation in a certain way. In this section we focus on the program flow and therefore are interested in the parameters that guide the region growing. In the following we also outline the differences in setting these parameters for the individual regions.

Definition of Start Region Of course the highest influence comes from the definition of the start region, since the region growing algorithm tries to extend this initial region to image areas with similar color and texture respectively. Hence defining suitable start regions is crucial, because even small deviations from the desired start point for a certain image region can cause the region growing process to fail in finding this region. Fortunately we have some prior knowledge about the scene which makes the definition of appropriate start regions much easier. The definition of a start region is also the point where a region is assigned a certain label. Again, this is possible because of knowledge about the scene shown in the image.

In contrast to this it is also possible to use low-level segmentation techniques, which

involve no knowledge about the scene, at first. Afterwards one can apply the region growing process to the regions found in the previous step in order to refine the segmentation. An early version of our segmentation tool was based on such an approach. For obtaining the initial segmentation we used the ROI-SEG algorithm described in Section 2.4.4. Beside the fact that defining the start regions using prior knowledge is much more robust, a further disadvantage of this approach is the additional labeling effort. Here, contrary to the knowledge based segmentation where the labeling is inherent in the segmentation process, labels must be assigned to the segmented regions in an extra step. Therefore information about the regions' topology and with it knowledge about the scene is needed. So, while the first approach uses knowledge right from the beginning, the latter involves knowledge only in the labeling step, performing the segmentation step completely free of knowledge about the scene. Examples for the latter method are the expert system described in Section 2.6.1 and the AdaBoost classifier outlined in Section 2.6.2. As will be shown in the Chapter 4, our method is superior to methods involving knowledge only in the labeling stage.

Feature Space The features used for segmentation have a high effect on the result too. We use color as well as texture information, but not for all regions. While the three color channels L, a and b are used for every region, the texture information is only provided when segmenting the hair region. Experiments showed that for the face, shoulder and background region color information alone is sufficient, and moreover texture information is rather obstructive for these regions. Especially the face region suffered from adding texture as additional feature, either leaving large areas around eyes, nose and mouth uncovered due to increased texture in theses regions, or adapting itself to these regions and hence growing into the also richly textured hair region.

A performance gain by using texture as an additional feature was only achieved for the hair region. The hair of most people shows a rather dense texture (exceptions, for instance, are people with hair so dark that individual hair streaks are no longer visible in the passport photograph). This information can be used in cases where a person's hair color is very similar to the color of the background, e.g. an elder person with gray hair in front of an also gray background region (gray is an often chosen color for uniform backgrounds). Another observation regarding hair is its frequently shaded appearance. In many cases the hair region contains several shades from the same color depending on hairstyle and reflectance properties. Imagine, for example, an image showing a woman

with dyed hair. Often the hair roots have a different (the original) color than the rest of the dyed hair. But also the color of natural hair can vary significantly within an image. To reduce the influence of these color shades, the four features used for the segmentation of the hair region are scaled discriminatively.

Feature scaling is a technique commonly used in data clustering algorithms. The idea is that different features are differently important for clustering. Because of that a weight factor is assigned to each feature according to its importance. Features with higher weights have greater importance than features with lower weights. A detailed description of feature scaling can be found in [19] and the references mentioned therein. The authors describe a technique called Adaptive Feature Scaling, which improves the performance of clustering gene microarray data. Their algorithm also takes into account that a single feature can have a different importance for individual clusters, and therefore every feature has several weight factors, one for each cluster.

In our case it is sufficient to use feature scaling only for the hair region. For segmenting the other regions only color information is used, and, as already mentioned in Section 3.2.2.1, we treat the luminance component L as equally important as the color components a and b. For the hair region the texture feature is assigned a weight factor of one, i.e. its value ranges from 0 to 255 just as the values for the three color features normally do. The color channels a and b are multiplied with a weight factor of 0.5, hence their range is halved. Since the luminance component L is mainly responsible for the shaded appearance of hair in images, its importance is even further reduced by assigning a weight factor of 0.25.

Shape Probability Maps The shape probability maps also play an important role, as they can severely influence the segmentation result. Although they are pre-calculated we can manipulate the region growing process with their help. This is achieved by choosing the most suitable probability maps for the hair and background region based on an estimation of the hairstyle after the first segmentation run. Additionally the maps can be multiplied with a weight factor in order to increase or decrease their influence. Experimentally we found the following suitable weight factors for the individual regions: 2 for face, 1 for hair, and 4 for shoulder as well as background region. While the shape information has medium influence on the face region, the effect on the hair region is rather low, and the effect on the shoulder and background region is rather high. This comes from the fact that hair can have various different shapes due to the variety of

possible hairstyles. So the contribution of shape information to the first segmentation run is cut back for the hair region. In contrast to this the shoulder and background region must be limited more strongly to avoid the shoulder region to extend to upper image regions and the background region to grow into the center.

Minimum Background Probability Parameter The region growing process uses a probability ratio that indicates likeliness to which of two different regions a pixel belongs to. The two probability values are derived from two Gaussian models in the feature space, one for each of the competing regions. The first region, which is the region that is currently growing, is considered to be the foreground region, the second region is the background region. Note that in this context the term background region does not refer to the actual background region in the image, but instead it stands for a growing region's opponent. One can specify a value for the minimum background region probability in order to make it more or less difficult for a region to grow. A high value for the minimum background probability makes it harder for a region to expand, and a low value simplifies the growing process.

Usually a minimum background probability is only specified if there is only one region currently growing. In this case it can be seen as a threshold that controls how strong the region grows. For the face, shoulder and background region the same value is used. However, for the hair region a smaller value has to be used, because the hair region uses a fourth dimension in the feature space (three color channels and texture).

In the case of letting all regions in the image grow simultaneously the minimum background probability can be set to minus infinity, which means that the background region probability is purely derived from the feature space models, as described in section Section 3.3.2.2. This allows each region to grow freely from any threshold, only restricted by the other competing regions.

Thus with the help of the minimum background probability parameter we can control the growing behavior of various regions. As previously mentioned, we want the regions to grow very conservatively in the first segmentation run, and allow them to compete with each other in the second run. This can be achieved by setting the minimum background probability parameter appropriately.

Probability Calculation Mode Finally one can choose between single and multimodal probability calculation for the growing process of a certain region. If single modal probability calculation is enabled, only one Gaussian model is calculated in the feature space. Multimodal probability calculation uses several Gaussian models, and the final probability is the maximum of all the probabilities derived from these Gaussian models. The only region in the image that usually grows multimodally is the background region. For all other regions one can assume at least a certain degree of homogeneity, so one Gaussian model in the feature space is sufficient. But for the background region this assumption is unrealistic, since the image background is arbitrary. This ranges from completely uniform to very cluttered backgrounds.

3.4.2 First Segmentation Run

In order to decrease the run time of our method the segmentation is carried out using a multiresolution approach. First the geometrical resolution of the input image is reduced and the segmentation process is applied to this smaller image. The result of this first step is then scaled up to the original resolution again and serves as starting point for a second segmentation process. Since the image is already roughly segmented, this second run is quite fast, although the image's full resolution is used. After this overview that should help understanding the main concept of our segmentation tool the following sections will describe the whole program flow in detail.

3.4.2.1 Background Uniformity Test

Before actually starting with the face region, which is the only region surely present in the image, we try to estimate whether the background is uniform or not. We define a test region located in the upper image where usually the background is located. Then the standard deviation of the pixels' color values within the test region is checked. If it is small enough, the background region is considered to be uniform and we start with the segmentation of the background region instead of the face region. We do so because if the background is uniform, it is very easy to segment, even without any knowledge about the other regions. And later, when segmenting the other regions, already knowing the background region is a big benefit, especially when defining the hair start region. The test region serves as the starting region of the background growing process. The only difference to a normal background region growing process

is the fact that in this case multimodal probability calculation is turned off, because we assume a uniform region. After this first step we have an estimation of whether the background is uniform or not, and, if the background seems to be uniform, the corresponding region in the image. The background uniformity test is visualized in Figure 3.11.

(a) Test region indicates non-uniform background.

(b) Test region indicates uniform background.

(c) Grown uniform background region

Figure 3.11: Background uniformity test. In the left image the test region indicates a non-uniform background. In contrast to this the background seems to be uniform in the middle image. We let grow the uniform test region and obtain an estimation of the true background region (right image).

3.4.2.2 Face Region

Now the algorithm tries to segment the face region. Figure 3.12 shows the corresponding start region located in the image center. To increase the performance of our algorithm, we use a skin detector based on two statistical Gaussian mixture models in color space [31]. One model represents the skin class, and the other the non-skin class. Using Equation (3.11) with probabilities derived from these two models one can classify a pixel as skin or non-skin. The classification result depends on the specified threshold T. The higher this threshold is the less skin pixels will be detected:

$$\frac{P(\boldsymbol{c}|skin)}{P(\boldsymbol{c}|nonSkin)} \geqslant T \tag{3.11}$$

Our algorithm now applies the skin detector to the face start region. Pixels that are labeled as non-skin are removed from the start region. In doing so we gain robustness in

cases where parts of the face are covered, e.g. by glasses or a beard. If such outlier pixels were not removed, we would obtain a very bad initial face color model for segmentation, so that the face region would easily grow into unsuitable image regions. Especially the hair region would often be displaced by the face region, since the color of hair and beard are usually the same.

But one problem remains. If a person's skin appears very unnatural in an image, for example due to bad illumination conditions, the fact that we only have predefined color models can lead to high false positive or false negative rates. Also people can have different ethnicity, and the predefined color models might not be well suited for all possible ethnicities. Luckily the classification result can be tuned by the threshold value. Furthermore we already know where we can expect skin pixels in an image, although we do not know their exact positions. The idea is now to start with a very high threshold value, thus only very few skin pixels will be detected. Then the threshold is lowered until a sufficiently large portion of the start region is classified as skin. In the worst case the skin appearance in the image is so bad that the threshold has to be lowered to zero. The result is that all pixels in the start region are classified as skin, thus the whole start region is used. Having now defined the face start region properly we can execute the growing algorithm in order to obtain a first version of the face region.

(a) Entire face start region

(b) Skin inside face start region

(c) Skin inside face start region

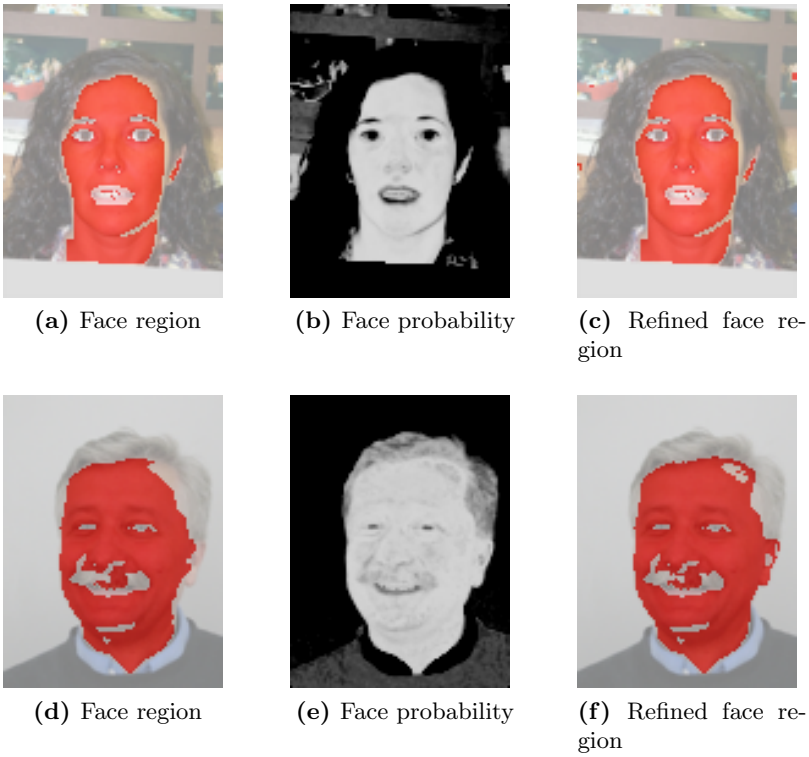
Figure 3.12: The face start region is located in the image center. Applying the skin classifier gives us the final start region. While the changes caused by the classifier are minor in the middle image, the right image shows a case where the start region is reduced significantly by the the classifier.

3.4.2.3 Hair and Shoulder Region

The next step is the definition of start regions for the hair and shoulder region. We know that hair is usually located above the face. Special cases are bald and half-bald people, which have no hair or only little hair near the ears respectively. The position of the shoulder regions is underneath the head, in the left and right image corner. To find suitable start regions, we use the already segmented face region. But we must take into account that this first version of the face region is a rather conservative estimation of the true face region. Recall that we want to avoid the case where a region grows into another region in the first segmentation run. Hence the parameters during the first segmentation run are quite strict, preventing the face region to grow into image regions with moderate color dissimilarities. As a result the face region usually does not cover the entire true face region after the first segmentation run, because the border zones have slightly different colors in most cases. Especially the border zone between face and hair, where the hairline starts, is often missed by the first face segmentation run.

To overcome this problem, we refine the segmented face region. To do so, we calculate the probabilities for all image pixels according to the current face color model, that is a Gaussian model derived from the pixels of the current face region. Next we estimate the distribution of the probability values within the current face region by calculating their mean and standard deviation. Finally we enlarge the segmented face region by adding pixels that have a probability value greater or equal than the mean probability minus the standard deviation. Figure 3.13 shows the effect of this extension process. The enlarged face region is much closer to the border of the true face region than the originally segmented region. However, in some cases the segmented region is extended too far, for example crossing the border between face and hair and stopping amid the hair region. Fortunately this situation is not that harmful, because the enlarged region is only used for defining start regions for segmenting the hair and shoulders. In most cases even a too far extended face region leads to suitable start regions. But to be on the safe side we check the size of the enlarged region. If it is too large compared to the original face region, we treat it as outlier and use the originally segmented face region instead.

Now that we have prepared the face region we calculate its convex hull. Note that there can be some outlier face pixels located around the face region. To avoid distortions



(a) Face region **(b)** Face probability **(c)** Refined face region

(d) Face region **(e)** Face probability **(f)** Refined face region

Figure 3.13: The left column shows the grown face regions. The middle column shows the probabilities derived from the current face color model. The higher the probability of a pixel is the whiter it appears in the image. The right column shows the refined face regions. While the changes between the grown face region and its refined version are very small in the first example (upper row), the second row shows a case where refining the face region has significant influence.

caused by these pixels, we use only the largest part of the face region for calculating the convex hull. The portion of the convex hull that lies above the eyes and is not already covered by another region (e.g. uniform background region) is then used as start region for the hair region. To define the shoulder start region, we use three points. One is the lower left image corner, one is the lower right image corner, and the last one is a point located approximately in the middle of the person's neck. One has to bear in mind that the two corner points must not necessarily be the corner pixels, since the image corners might be covered by a padding frame. The third point is calculated using the lower boundary of the extended face region. It lies centered between the left and right image border, slightly above the enlarged face region's lower boundary. Defining the third point in this way gives us robustness in cases where the face region reaches all the

way down to the lower image border. Finally that part of the spanned region that is not already covered by the extended face region or any other region forms the shoulder start region. Both start regions are shown in Figure 3.14.

(a) Non-uniform background

(b) Uniform background

Figure 3.14: Hair and shoulder start regions

The next step depends on the background uniformity estimated at the beginning of the segmentation procedure. If the background seems to be uniform, we directly jump to the second segmentation run, which means that all regions currently present in the image grow simultaneously with relaxed parameters. This is described in Section 3.4.3.

If the background seems to be non-uniform, the hair and shoulder start regions are extended by the growing algorithm. The two regions grow simultaneously to avoid that one region grows into the other due to color similarity. Imagine, for instance, a person with black hair wearing a dark pullover. If the hair region extends at first, it could easily grow into the shoulder region and even displace it completely. Vice versa the shoulder region could grow up into the hair region. Although it is very unlikely that the entire hair region would be displaced by the shoulder region in the latter case, both scenarios probably lead to a wrong estimation of the person's hairstyle.

To estimate the hairstyle, we use the pre-calculated hair masks that have already been described in Section 3.2.1. The classification process is exactly the same. After the growing process for hair and shoulder regions has finished, we compare the grown hair region to every hair mask by a logical XOR-operation. The hair mask that results in the smallest overlap error determines the hairstyle class (short, medium or long hair), and with it the shape probability maps for the hair and background region that are used in all further growing processes.

3.4.2.4 Background Region

Now we can define the starting region for the background region. As already mentioned, this starting region is located near the upper image corners. In contrast to the starting region definition used in the very first step of the segmentation procedure, the estimation of background uniformity, this time the starting region reaches down to the lower image corners, but is only allowed to cover yet unlabeled pixels. To improve the robustness, the background starting region is cut back slightly at borders with already segmented regions. This is carried out using morphological processes. Figure 3.15b shows the background start region.

However, there are cases where no background is visible in the image due to a very voluminous hairstyle. But in the most of such cases the hair region does not completely cover the corresponding image region after the first segmentation run because of the stricter parameter setting. As a result some of the yet unlabeled hair pixels are defined as background start region, and the background and hair region start competing against each other for the same region, namely the hair region. To gain robustness for this kind of images, we define a minimum area for the background start region. If the start region is smaller than this threshold, it is very likely that it only consists of outlier pixels, and therefore the start region is removed. This means that no background region will be detected.

(a) Face, hair and shoulder region

(b) Face, hair and shoulder region with background start region

Figure 3.15: The left image shows the face, hair and shoulder region after letting hair and shoulder region grow. Note that the regions do not entirely cover the true image regions due to the rather strict parameter setting during the first segmentation run. The right image shows the background start region.

At this point we have finished the first estimation of the regions present in the image. We have segmented the face region and, dependent on the background uniformity, start regions or even estimations for the other regions. If we assume a uniform background, the algorithm has segmented the background and face region, and has also defined start regions for hair and shoulder. If the background seems to be non-uniform, the algorithm has segmented the face, hair and shoulder region, and has defined a background start region (except for cases where the background is entirely covered by hair). In both cases we are now ready for the second segmentation run, in which all regions currently present in the image grow simultaneously.

3.4.3 Second Segmentation Run

In the second run the algorithm tries to refine the rather coarse first image segmentation by letting the different regions compete against each other. Therefore we relax the parameter setting by lowering the minimum background probability parameter as described in Section 3.4.1. As already mentioned, we could set this parameter to minus infinity so that there is no additional value constraining the competing regions. However, doing so can easily result in regions growing into improper image areas, because a minimum background probability value of minus infinity enforces a complete segmentation of the image. Although this is the desired behavior of a segmentation algorithm, we have to be careful at this stage of our method. Imagine, for example, a small object with reddish color similar to the face region located somewhere in the background. Furthermore let's assume that the remaining background has a completely different color, as for instance gray. On the one hand the probability that this small object belongs to the background would be very low due to color dissimilarity. On the other hand the probability for this region being a face region would also be low, since it is located in an inappropriate image area. Setting the minimum background probability parameter to minus infinity would force the algorithm to assign a certain label to the small object in the background. The algorithm would choose the label with the highest probability among all labels, regardless of how small this maximum probability actually is. As a result regions that are very tough to segment correctly are assigned a false label, instead of just marking them as unknown. So in our example it could happen that the small object is labeled as face, although it would be much more reasonable to label it as unknown for now. Thus we use a very low, but finite value

for the minimum background probability parameter during the second segmentation run. We ignore tough regions that have only a low probability for any of the labels at the moment and deal with them later in the post-processing stage. There we can use knowledge about the regions' constellation to assign labels to all still unknown image regions.

Note that during this second segmentation run some of the regions may disappear. This can happen in cases where a region is not actually present in the image, a common example are shoulders that are completely covered by hair. After the first segmentation run the upper part of the true hair region might be correctly marked as hair, whereas the lower part might be wrongly labeled as shoulders. This is a direct result of the way we define the hair and shoulder start region. Now if the hair region is strong enough during the second segmentation run, it can entirely displace the false shoulder region. Unfortunately in the majority of cases such false shoulder regions can resist and do not vanish. For this reason we have to check the shoulder regions in the post-processing stage and correct the false ones.

The last step before the post-processing stage is up-scaling. At this point we have a segmented image on reduced resolution. Hence we scale the current result up to the original image resolution, and let then grow all regions once more, just like we did in the previous step. Finally, after the growing process has converged, we have a refined version of our previous result (Figure 3.16). This segmentation result is now forwarded to the post-processing stage.

(a) Image overlaid with regions

(b) Pure regions

Figure 3.16: Segmentation result after second segmentation run

3.5 Post-Processing

The post-processing stage enhances the segmentation result by removing or relabeling of regions and examining yet unlabeled image areas. To do so, we again use prior knowledge about the scene in the image. One can think of the post-processing stage as being like an expert system, consisting of several rules. Some of them are general and are valid for the whole image, whereas others are specific to certain regions. All the rules are checked consecutively and, if suggested by the current rule, corrections are made. In the following rules description when we speak of regions we mean single regions, that is although regions belong to the same label they are treated separately (e.g. the left and the right shoulder region, if not connected, are considered two separate regions).

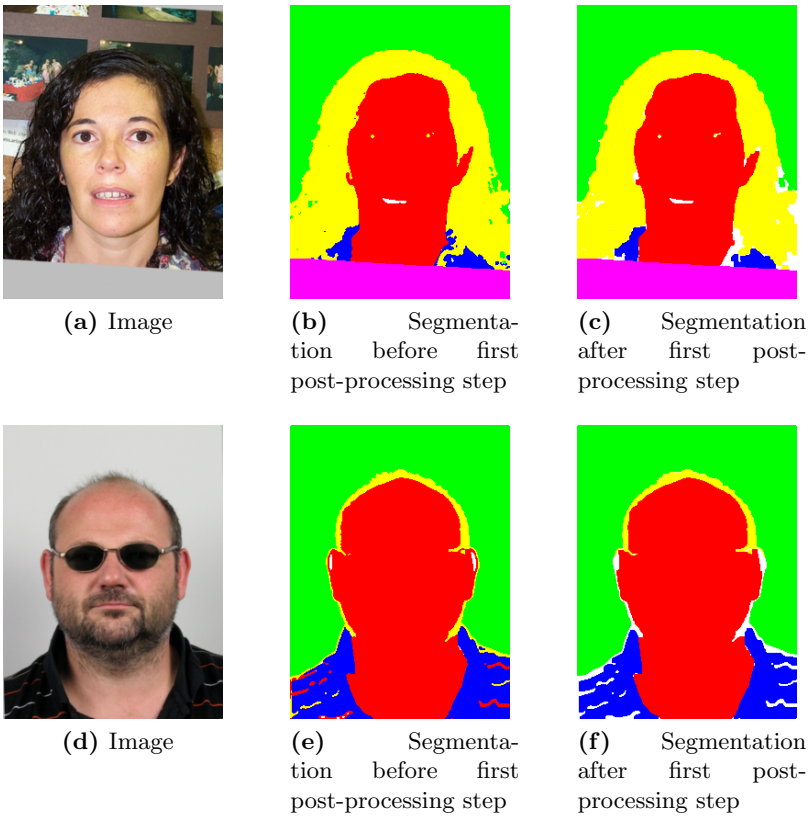
3.5.1 Morphological Processing and Small Region Removal

The first post-processing step consists of morphological processing and small region removal. This rule is general and therefore applied to the whole image. First every single region is morphologically opened using a small disk as structure element. The main purpose of this operation is to eliminate misclassifications along region borders. Due to color similarity it is quite common that the borderline between face and background or face and shoulders is labeled as hair. In other cases the face region extends along the borderline between shoulder and background region. Morphological opening is perfectly suited to remove such unwanted disturbances. However, one must pay attention when dealing with images showing a person with short hair. Although correctly segmented, morphological opening could easily remove the entire hair region.

But how to distinguish between a properly detected short hair region and a disturbing hair region along the borderline between face and background? The decision is based on the experience that in most cases the hair region of people with short hair reaches roughly down to the ears' onset, which lies approximately on one line with the eyes in a canonical image. And because the eyes' positions are known from the canonization process, the decision is based on their y-coordinate. The part of the hair region that lies above the eyes is stored temporarily as a backup to be able to restore it after morphological opening. So even if the hair region has vanished due to the opening process, the valid part of it can be restores with the backup mask. The minor drawback

that arises of this morphological step is the loss of finer structures, especially in the hair region (e.g. hair ends sticking out into the face, shoulder or background region).

The second part of the first pre-processing step is the removal of small regions. The regions we want to detect in our face images are rather big, hence very small regions are likely to be either the result of noise present in the image or simply misclassified regions (e.g. eyes labeled as hair because of dominant color similarity). In both cases we can reject such regions and label them as unknown for now. The effects of the first post-processing step are visualized by Figure 3.17.



(a) Image **(b)** Segmentation before first post-processing step **(c)** Segmentation after first post-processing step

(d) Image **(e)** Segmentation before first post-processing step **(f)** Segmentation after first post-processing step

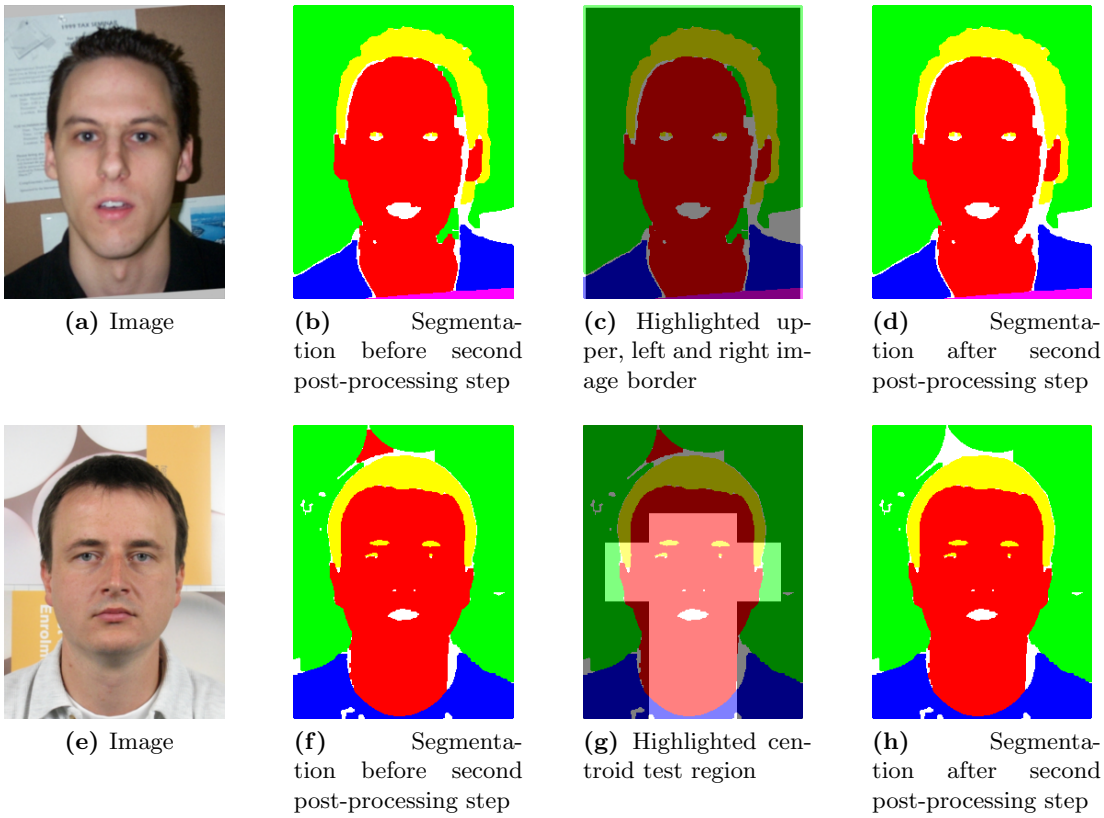
Figure 3.17: Two examples showing the effect of morphological processing and small region removal

3.5.2 Removal of Improbable Background and Face Regions

The second post-processing step is the removal of improbable background and face regions. A valid background region must have contact with the upper, left or right

image border, that means the actual image border or the corresponding padding frame. There are only a few, extremely rare cases where this definition fails. For example, images of people with a very unusual hairstyle can contain a valid background region that is completely surrounded by the hair region and has no contact to the image border. Nevertheless our test database does not contain such a special case, and in consideration of the rareness of this kind of cases we ignore them.

Proper face regions must have a centroid lying within a predefined region in the image center. This centroid test region is designed in a way so that the main face region as well as the ears and the neck can pass the test, but outlier face regions detected in the background are rejected. Figure 3.18 illustrates the second post-processing step.



(a) Image

(b) Segmentation before second post-processing step

(c) Highlighted upper, left and right image border

(d) Segmentation after second post-processing step

(e) Image

(f) Segmentation before second post-processing step

(g) Highlighted centroid test region

(h) Segmentation after second post-processing step

Figure 3.18: The upper row shows how improbable background regions, located between face and hair/shoulder region, are removed. The lower row illustrates the removal of an outlier face region located above the head.

3.5.3 Correction of Nested Regions and Removal of Improbable Hair and Shoulder Regions

The next post-processing step is the correction of nested regions and the removal of improbable hair and shoulder regions. A nested region is a region that is completely surrounded by another region. As one can imagine, such regions do not occur in usual face images, except for ears that are enclosed by hair, the whole face surrounded by hair (due to long hairstyle) or skin enclosed by the shoulder region (due to a special blouse). As already mentioned in the description of the second post-processing step, the rare case of a background region surrounded by hair is ignored. A similar case, shoulder region surrounded by hair, is also neglected because of its rarity. All in all the following cases of nested regions are treated as segmentation errors and hence are relabeled according to the surrounding region's label:

- Face region entirely enclosed by the background region
- Hair region entirely enclosed by the face, shoulder or background region
- Shoulder region entirely enclosed by the face, hair or background region
- Background region entirely enclosed by the face, hair or shoulder region
- Unknown region entirely enclosed by any other region

In detail the nested region correction works as follows. The algorithm checks every single region and examines its neighborhood. It starts with the regions labeled as background and then processes the other labels in the following order: shoulder, hair, face and unknown. It is important that the unknown regions are processed last, because under some circumstances a nested region is labeled as unknown. Such unknown regions have weaker constraints that allow them to become relabeled more easily. That means that a nested region labeled as unknown at first can be assigned a real labeled in the final run, where all unknown regions are examined.

First the current region's neighboring regions must be determined. In order to get the neighbors the region is subtracted from its morphologically dilated version. This gives us the region's contour which helps us to get the labels of all neighboring pixels by a simple logical AND-operation of the contour and the labeled image. Pixels that are yet unlabeled (unknown) and pixels belonging to the padding frame are removed from

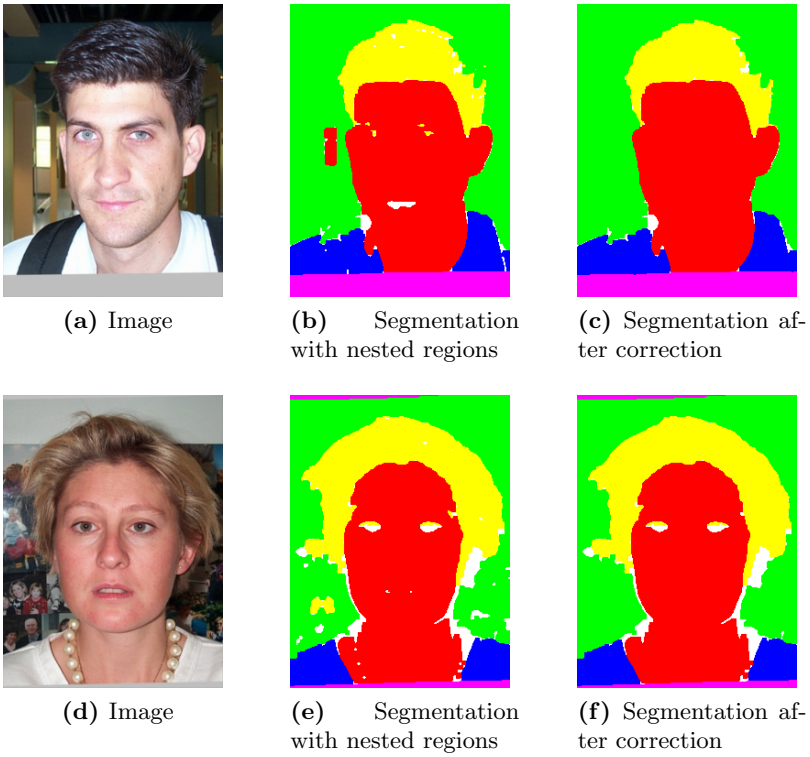
the set of neighboring pixels, because they don not hold any significant information for the process of relabeling nested regions. Unfortunately nested regions are often surrounded by a lot of unlabeled pixels, and only a few of their neighboring pixels have a real label. For the purpose of making the algorithm more robust the contour is allowed to grow from a width of one pixel up to a certain thickness. After every contour growing step the ratio between reliable neighbors, that is neighboring pixels that have a real label, and neighboring pixels labeled as unknown or padding frame is checked. If the neighboring pixel set contains enough neighbors that have a real label, it is marked as reliable and the growing process stops. An additional set containing only the neighbors that are directly adjacent to the region currently examined (determined during the first run of the contour growing process) is also stored, because these direct neighbors are needed later. As one can imagine, if the contour width is greater than one pixel, it is possible that the neighboring pixels set contains pixels with the same label as the region (if, for example, two nested regions of the same label are only one pixel apart). Such pixels having the same label as the currently examined region must also be removed from the set to ensure the correct functionality of our algorithm.

Now that we have a region's neighborhood information we further examine the region in order to check whether the region is an invalid nested region that needs relabeling or not. First we verify that the set containing the neighboring pixels is reliable. Only if we can trust the neighborhood information, the relabeling process continues with the next step, the investigation of the currently examined region's size.

Sometimes we have to deal with rather large regions in the image that have only one other neighboring region. For example, one can think of an image showing a person with long hair, where the background region is completely separated from the other regions by the hair region. Although they have only one neighbor such valid regions must of course not be relabeled. Luckily these valid regions are usually adjacent to the image border. If a region adjoins to the image border, it is not relabeled, no matter how many neighboring regions it has.

This additional verification step before relabeling nested regions increases the robustness of the relabeling algorithm. In some cases the segmentation algorithm fails to correctly classify the border zones between regions and labels them as unknown. This can lead to the situation that one region, let's say region A, directly adjoins to a second region B, but between A and a third region C lies a border zone of unknown pixels.

So we only find pixels of region B, but no pixels of region C in the neighborhood of region A. If there are enough pixels of region B adjacent to region A, the neighborhood information is marked as trustable and region A would be relabeled as region B, although region A should ideally have two neighbors, namely region B and region C. The problem of missing a region's neighbor particularly affected the shoulder regions, for example, when there was a zone of unknown pixels between the shoulder and face region, but not between the shoulder and background region. Without the verification step such a shoulder region would most likely be relabeled as background region. Note that the relabeling criterion based on a region's vicinity to the image border does not apply to unknown regions. Since we do not have any information about the true label of an unknown region, such regions are assigned the most probable label that we can find, which in case of nested regions is the label of the surrounding region. Figure 3.19 visualizes the correction of invalid nested regions.



(a) Image **(b)** Segmentation with nested regions **(c)** Segmentation after correction

(d) Image **(e)** Segmentation with nested regions **(f)** Segmentation after correction

Figure 3.19: Two examples in which invalid nested regions are corrected

For the removal of improbable hair and shoulder regions a region's neighborhood must be examined too. Because of this the removal step is carried out together with

the previously described correction of nested regions. Hair regions with no connection to the face make no sense and are obviously outliers. If possible, such hair regions are relabeled according to the rules for correcting nested regions, else they are labeled as unknown.

Improbable shoulder regions are regions that adjoin the hair region and have similar color. Under certain circumstances hair can be mislabeled as shoulder region in images where long hair almost completely covers the shoulders of a person. If this is the case, we usually have to deal with a correctly segmented and labeled upper part of the hair region and a misclassified lower part labeled as shoulder region. To decide whether a shoulder region has to be relabeled as hair region or not, we must examine the neighborhood of the shoulder region as well as the similarity between the shoulder and the hair region. Only shoulder regions that are directly adjacent to a hair region can be meaningfully relabeled and take part in the similarity test. If a shoulder region also passes this test, i.e. its color and texture is similar to the color and texture of the hair region, the shoulder region is changed into a hair region. In most cases when it is necessary to relabel such improbable shoulder regions the true shoulder region is also visible in the image, even though only small parts of it are present at the lower image border. To find these true shoulder regions, the region growing process is once more initiated with the corresponding start region. But this time the region is only allowed to grow in yet unlabeled pixels near the lower image border. We choose the y-coordinate of the centroid of the former shoulder region as limit. Only unlabeled pixels below this limit are considered in the growing process. The removal of improbable hair and shoulder regions is depicted in Figure 3.20.

3.5.4 Labeling Unknown Regions

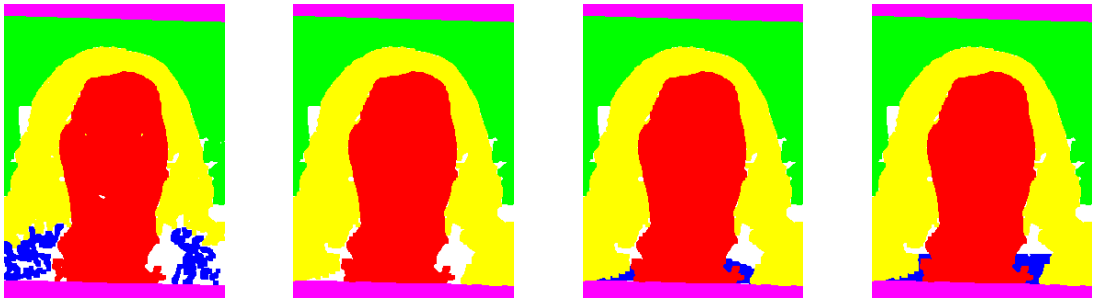
The fourth and last post-processing step deals with regions that are still unlabeled. Since nested regions have already been corrected, these unknown regions are border regions between other, labeled regions. This means that they have more than one neighboring region, and we have to decide for every pixel which label among the labels of all neighbor regions is most suitable. From our experiments we know that such border regions are rather elongated. An example is the common case of an unlabeled region extending along the border line between the face and the hair region, a result of the morphological process described in section Section 3.5.1. To classify the unknown

(a) Image

(b) Segmentation with improbable hair regions

(c) Segmentation after relabeling

(d) Image



(e) Segmentation with improbable shoulder regions

(f) Segmentation after relabeling

(g) Segmentation with new shoulder start region

(h) Segmentation with newly grown shoulder region

Figure 3.20: In Figures 3.20a-3.20c an example of removing improbable hair regions is presented. Figures 3.20d-3.20h show a case where the shoulder regions are very similar to the adjacent hair region, and therefore are relabeled as hair. Afterwards a new shoulder region segmentation process is initiated.

regions, the neighboring pixels of every unlabeled pixel are examined. Of course just neighboring pixels that already carry a valid label are considered. If there are only pixels of one certain label among them, the decision is easy and the unknown pixel is assigned this sole label. But if there are more valid labels available, we must choose the most probable one. In order to do so a probability map is calculated for all four labels as specified in section Section 3.3.2.1. Each of the four maps contains the probability values for one label based on the image content. The unknown pixel is now assigned the label that, among all neighboring labels, has the highest value at the pixel's position in its probability map.

One last thing to mention is that several iterations through the image pixels are necessary for a complete elimination of unlabeled pixels. The reason for this is the

sequential processing of the unlabeled pixels. We start in the top left image corner, go down the first column, then the second column and so on until we reach the bottom right image corner. If the segmented and labeled image would be updated immediately after an unknown pixel has been changed, outflow effects could occur. To avoid such effects, an update is performed only after all unlabeled pixels have been processed. The drawback of this approach is that several runs are necessary to assign all unknown pixels a valid label.

Finally the previous post-processing step, the correction of nested regions, has to be carried out once more. This is necessary, because it is not guaranteed that all nested regions were revised during the first correction run due to the possibility of unreliable neighborhood information caused by unlabeled pixels (see Section 3.5.3 for details). Now that all pixels have been assigned a certain label any invalid nested regions still present can be eliminated. Usually there are only a few, if any, nested regions left, so this time the correction step is quite fast. Note that reversing the order of the last two post-processing steps, that means first assigning a certain label to all unknown pixels and then correcting nested regions, would of course prevent us from having to call the correction step twice. However, the image segmentation should be as good as possible when assigning labels to unknown pixels solely based on information about neighboring regions. Hence we first refine the segmentation by correcting nested regions, then all unknown pixels are assigned a certain label, and lastly still existing invalid nested regions are revised.

3.6 Background Classification

The final background classification is based on the background region's standard deviation as well as on the gradient magnitude within it. As already mentioned, we want to determine whether the background is uniform or not. The principle is much the same as for the first background uniformity estimation. The only differences are that we now have the segmented background region instead of just a test region, and that we also take gradient information into account. The aim of the first uniformity information was to aid the segmentation process. So the most important criterion was color uniformity. Slight edges appearing in the background were only of minor importance, so that we could neglect them. But now we have to decide whether the background in an image is uniform, i.e. it has a uniform color and contains no edges, or not.

Before examining standard deviation and gradient magnitude the background region is slightly cut back, because border zones between the background region and other regions can easily lead to misclassification. Especially near the hair region there are often distortions, like hair ends sticking out into the background region. The segmentation algorithm fails to correctly label such fine structures as hair and marks them as background. To avoid that the background is classified as non-uniform due to these distortions, the background region is pruned more strongly near the hair region.

$$std(BG) > T_1 \quad \Rightarrow \quad \text{background is non-uniform} \tag{3.12}$$

$$\begin{aligned} &P = \{\text{pixel } p(x,y) \mid (p(x,y) \in BG) \ \&\ (|\nabla p(x,y)| > T_2)\} \\ &|P| > T_3 \quad \Rightarrow \quad \text{background is non-uniform} \end{aligned} \tag{3.13}$$

After preparing the background region we examine the standard deviation within the reduced region. If it is greater than a certain threshold, the background is classified as non-uniform (Equation (3.12)), else we further check the gradient magnitude. In a uniform background region the number of pixels with a gradient magnitude greater than a threshold must not exceed a certain limit (Equation (3.13)). This definition allows a few outlier pixels with high gradient magnitude in a uniform background region, so that we gain some robustness against noise and distortions near region borders.

3.7 Discussion

In this section we have described our knowledge based segmentation algorithm. Segmentation is achieved by letting well defined start regions grow using gradient, color, texture and shape information. The proposed method integrates prior knowledge whenever possible, in particular in the definition of start regions, the region growing process and the post-processing stage. The main advantage of our algorithm over methods that also use domain knowledge, like the expert system described in Section 2.6.1, is that we incorporate knowledge right from the beginning of the segmentation process. The expert system approach performs low-level segmentation first, and then uses a set of rules in order to assign labels to the segmented regions. In contrast to this our method already assigns labels to regions when defining their start regions, so no additional labeling effort arises.

In the next chapter we will present the results obtained with our algorithm. Furthermore we will compare them to results achieved by the expert system (Section 2.6.1) and the AdaBoost classifier (Section 2.6.2), and it will be shown that our method outperforms both.

Chapter 4

Results

Contents

4.1 Overview

In this chapter we present the results of our algorithm. First in Section 4.2 we outline the dataset used for evaluating the performance of our algorithm as well as the performance of the expert system and the AdaBoost classifier described in Section 2.6.1 and Section 2.6.2 respectively. Afterwards we define several error metrics (Section 4.3). In Section 4.4 the achieved results are shown, and in Section 4.5 we present a detailed comparison of the three methods. Finally some qualitative results of our algorithm are depicted in Section 4.6.

4.2 Dataset

To obtain the results we used a data set that consists of 320 face images. One part of these images was taken from the Caltech face database [5], the other portion are proprietary images provided by Siemens PSE Graz, Biometrics Center. 197 of the 320 face images show a uniform background, while the background in the remaining 123 images is non-uniform. Since homogeneous backgrounds can be segmented more easily than non-homogeneous ones, the segmentation error is usually lower for images with a uniform background. All images have been canonized, as this is the only constraint on the input passport photographs. Furthermore we received hand labeled ground truth data for all images from Siemens PSE. These annotated images allow us to calculate a variety of error metrics on our segmentation results, and to compare our algorithm to other methods. The annotated images also enable us to easily extract potential padding frames and take them into account when segmenting an image.

4.3 Error Metrics

4.3.1 Per Region Error Metrics

The error metrics described in this section are defined in [2]. For each segmented region we use two error rates, the false positive and the false negative rate. They are defined as follows:

- **False positive rate (FP):** The false positive rate describes the error of segmenting a certain region in places where this region actually does not appear in the ground truth data.
- **False negative rate (FN):** The false negative rate describes the error of not segmenting a certain region in places where this region actually does appear in the ground truth data.

However, one has to bear in mind that precise pixel wise segmentation is a very hard task, even for a human. One just has to think of border zones between regions. Determining uniquely to which one of two neighboring regions a border pixel belongs can be very challenging. Because of this the error rate calculation tolerates a small

uncertainty at region borders. This is achieved by enlarging the segmented and the hand labeled region respectively, like shown in Figure 4.1. For the false negative rate the segmented region has to be enlarged prior to error calculation, for the false positive rate the size of the hand labeled regions has to be increased. As suggested in [2], we choose an uncertainty in the region boundaries of $0.005 \times imageWidth$.

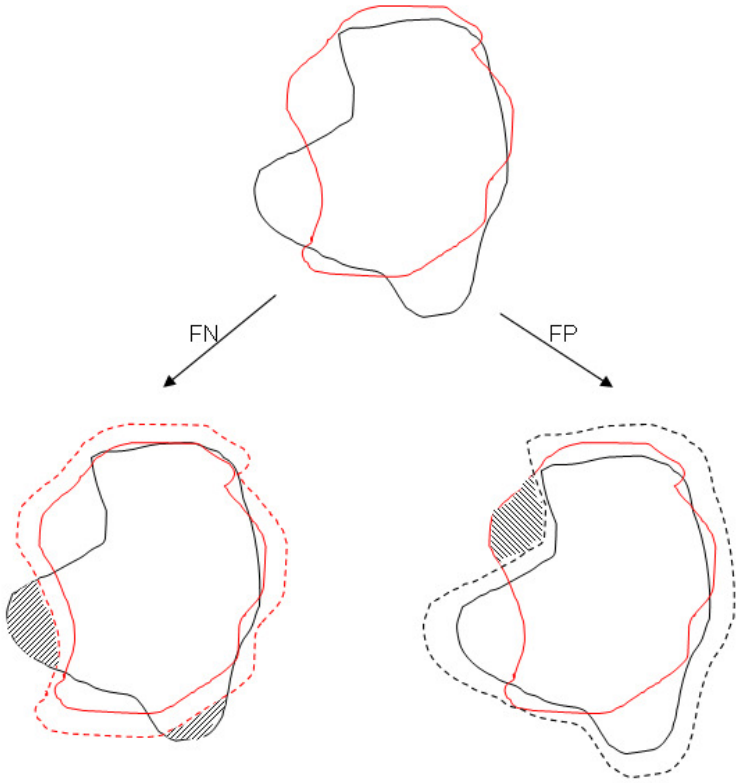


Figure 4.1: Border uncertainty. The segmentation result is drawn in red, the ground truth is plotted in black. Errors are represented by shaded areas. The figure is taken from [2].

After adjusting the corresponding region size the two error rates can be calculated with the following formulas. They represent the relative error per region:

$$FP = \frac{numRegionFalsePositivePixels}{\max(numRegionPixelsGroundTruth, 0.01 \times imageArea)} \times 100 \quad [\%] \quad (4.1)$$

$$FN = \frac{numRegionFalseNegativePixels}{\max(numRegionPixelsGroundTruth, 0.01 \times imageArea)} \times 100 \quad [\%] \quad (4.2)$$

The denominator in Equations 4.1 and 4.2 has a lower bound of $0.01 \times imageArea$. This saturation term prevents the error rates from shooting up for very small regions, i.e. regions that consist of only a few pixels. Otherwise every single wrongly labeled pixel would dramatically increase the error rate for such tiny regions.

4.3.2 Overall Image Error Metrics

The error rates defined in Equations 4.1 and 4.2 are calculated per region. Additionally we use the following three error metrics for determining the segmentation error on the whole image:

$$misclassified = \frac{numMisclassifiedPixels}{imageArea} \times 100 \quad [\%] \tag{4.3}$$

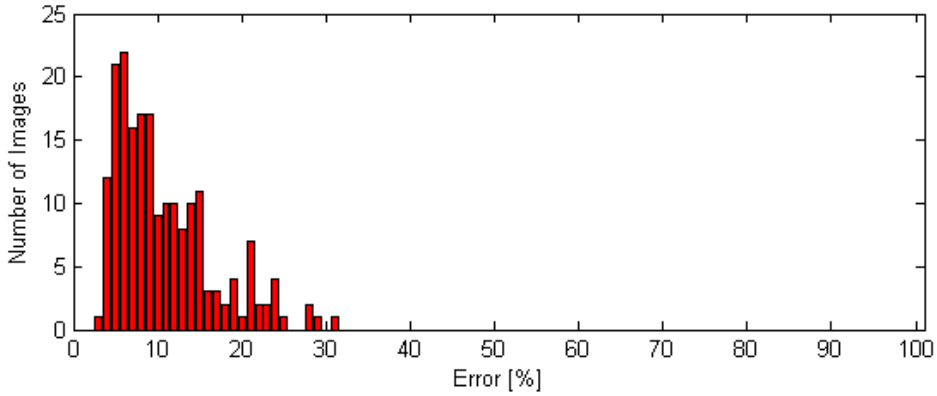
$$unclassified = \frac{numUnclassifiedPixels}{imageArea} \times 100 \quad [\%] \tag{4.4}$$

$$totalError = misclassified + unclassified \quad [\%] \tag{4.5}$$

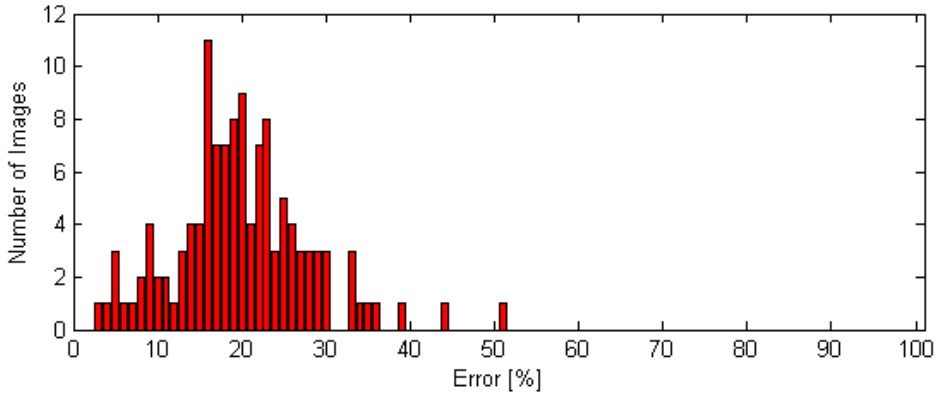
4.4 Quantitative Results

In this section we compare our method to the expert system and the AdaBoost classification algorithm outlined in Section 2.6.1 and Section 2.6.2 respectively. We evaluated all three approaches on the dataset described in Section 4.2 using the error metrics defined in the previous section.

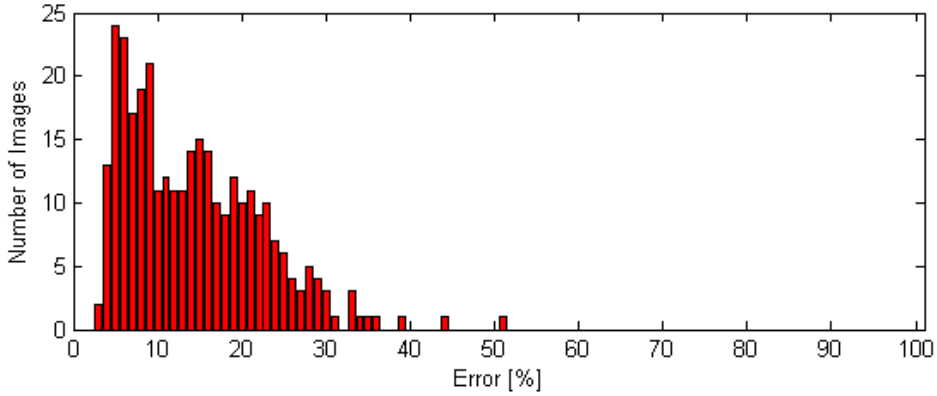
4.4.1 Expert System



(a) Images with uniform background



(b) Images with non-uniform background



(c) All images

Figure 4.2: Expert system error histograms

		Mean [%]	Std [%]	Min [%]	Max [%]
Face	FP	4.51	5.67	0.00	31.29
	FN	4.44	3.10	0.21	17.45
Hair	FP	12.16	11.11	0.00	69.43
	FN	33.81	29.87	0.09	100.00
Shoulder	FP	8.64	13.93	0.00	104.31
	FN	15.38	20.11	0.00	100.00
Background	FP	12.73	18.80	0.02	140.02
	FN	1.58	2.40	0.00	14.74
Overall image	Misclassified	9.11	5.36	2.28	29.03
	Unclassified	1.19	1.59	0.00	8.98
	Total error	**10.30**	5.85	2.94	30.30

Table 4.1: Expert system results on images with uniform background

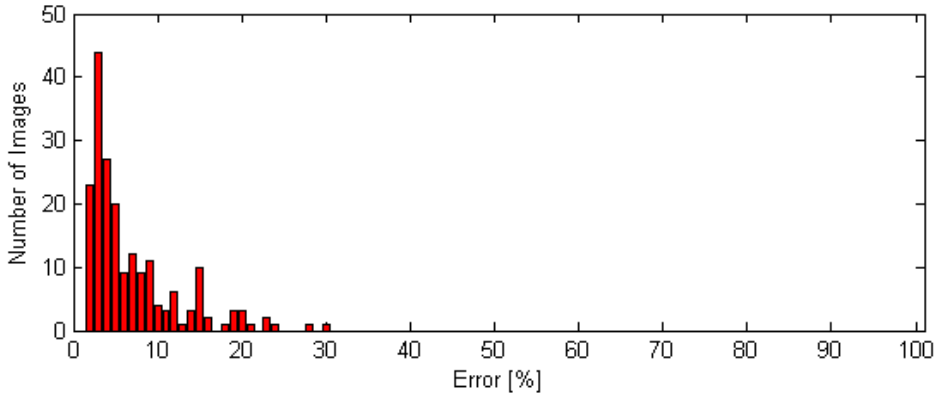
		Mean [%]	Std [%]	Min [%]	Max [%]
Face	FP	2.56	4.11	0.00	21.70
	FN	7.60	4.75	0.01	25.06
Hair	FP	40.75	46.11	0.00	338.35
	FN	17.38	26.43	0.01	100.00
Shoulder	FP	121.56	156.56	0.11	697.75
	FN	20.11	22.54	0.00	88.75
Background	FP	10.31	10.70	0.01	62.76
	FN	14.66	14.14	0.00	71.87
Overall image	Misclassified	16.67	6.84	2.82	37.40
	Unclassified	2.77	3.50	0.00	26.93
	Total error	**19.44**	7.92	2.82	50.40

Table 4.2: Expert system results on images with non-uniform background

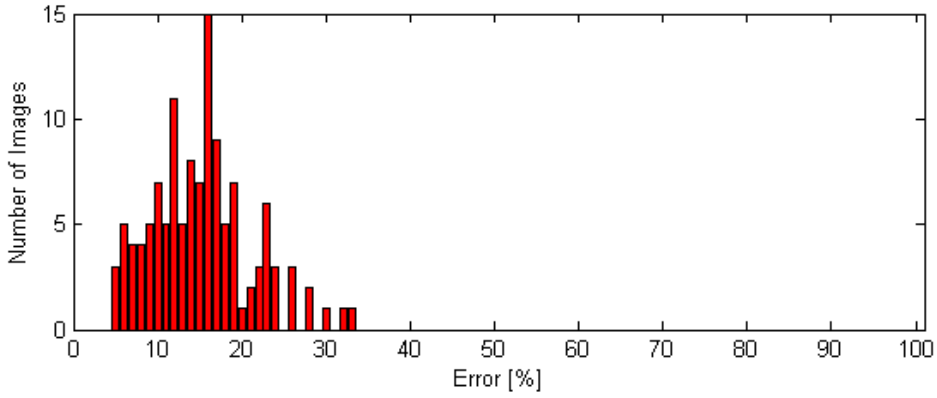
		Mean [%]	Std [%]	Min [%]	Max [%]
Face	FP	3.76	5.21	0.00	31.29
	FN	5.66	4.11	0.01	25.06
Hair	FP	23.15	32.91	0.00	338.35
	FN	27.50	29.65	0.01	100.00
Shoulder	FP	52.04	111.90	0.00	697.75
	FN	17.20	21.17	0.00	100.00
Background	FP	11.80	16.19	0.01	140.02
	FN	6.61	10.98	0.00	71.87
Overall image	Misclassified	12.01	7.01	2.28	37.40
	Unclassified	1.80	2.61	0.00	26.93
	Total error	**13.81**	8.05	2.82	50.40

Table 4.3: Expert system results on all images

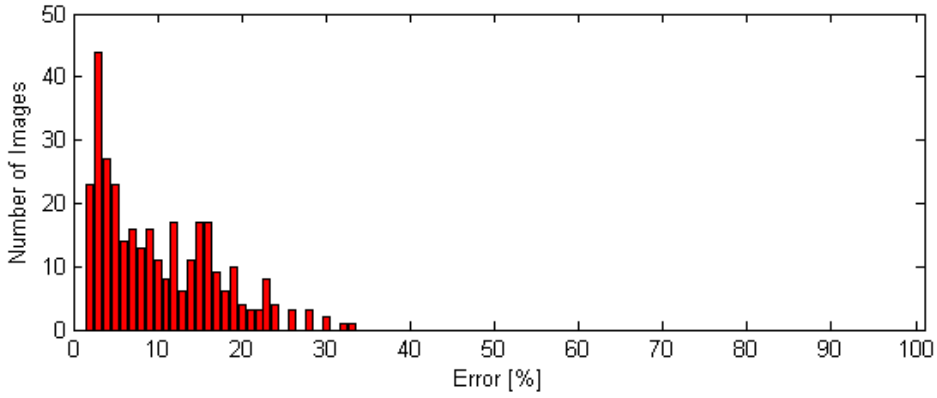
4.4.2 AdaBoost



(a) Images with uniform background



(b) Images with non-uniform background



(c) All images

Figure 4.3: AdaBoost error histograms

		Mean [%]	Std [%]	Min [%]	Max [%]
Face	FP	2.98	4.16	0.01	32.24
	FN	1.41	1.85	0.00	10.66
Hair	FP	4.80	12.07	0.00	141.68
	FN	16.87	16.30	0.42	100.00
Shoulder	FP	7.91	25.74	0.00	316.52
	FN	4.64	11.49	0.00	96.77
Background	FP	9.74	17.59	0.13	137.05
	FN	0.81	2.22	0.00	26.87
Overall image	Misclassified	6.39	5.52	1.23	29.22
	Unclassified	0.00	0.00	0.00	0.00
	Total error	**6.39**	5.52	1.23	29.22

Table 4.4: AdaBoost results on images with uniform background

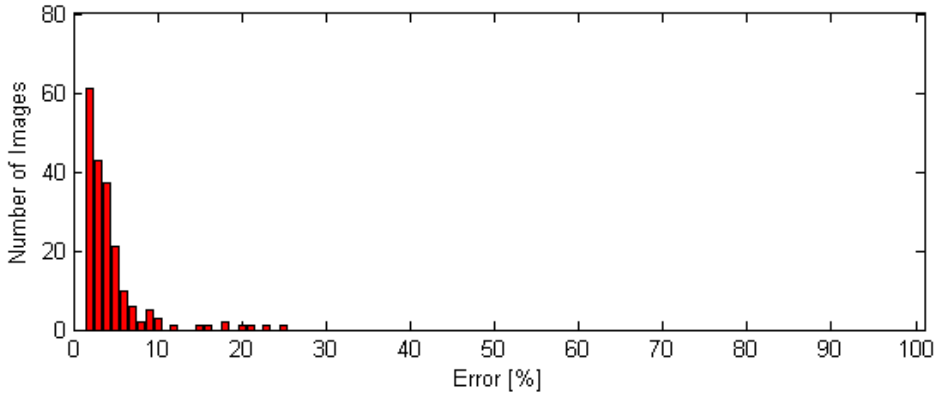
		Mean [%]	Std [%]	Min [%]	Max [%]
Face	FP	3.52	4.58	0.16	39.07
	FN	3.11	3.62	0.03	24.24
Hair	FP	26.69	37.09	0.46	302.52
	FN	9.87	8.60	0.65	45.34
Shoulder	FP	130.24	175.93	0.04	968.30
	FN	7.92	12.16	0.00	82.86
Background	FP	9.80	11.32	0.00	64.66
	FN	8.55	5.26	0.15	22.03
Overall image	Misclassified	14.75	5.97	4.27	32.68
	Unclassified	0.00	0.00	0.00	0.00
	Total error	**14.75**	5.97	4.27	32.68

Table 4.5: AdaBoost results on images with non-uniform background

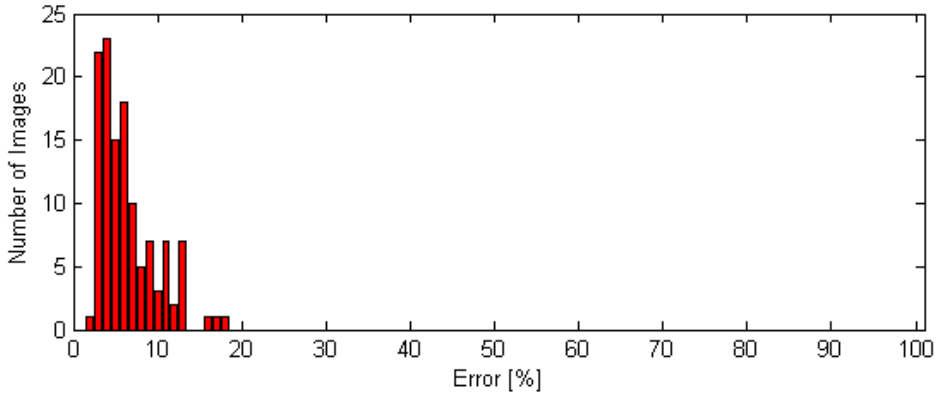
		Mean [%]	Std [%]	Min [%]	Max [%]
Face	FP	3.19	4.33	0.01	39.07
	FN	2.06	2.80	0.00	24.24
Hair	FP	13.21	27.00	0.00	302.52
	FN	14.18	14.25	0.42	100.00
Shoulder	FP	54.93	125.68	0.00	968.30
	FN	5.90	11.84	0.00	96.77
Background	FP	9.76	15.47	0.00	137.05
	FN	3.79	5.27	0.00	26.87
Overall image	Misclassified	9.60	7.00	1.23	32.68
	Unclassified	0.00	0.00	0.00	0.00
	Total error	**9.60**	7.00	1.23	32.68

Table 4.6: AdaBoost results on all images

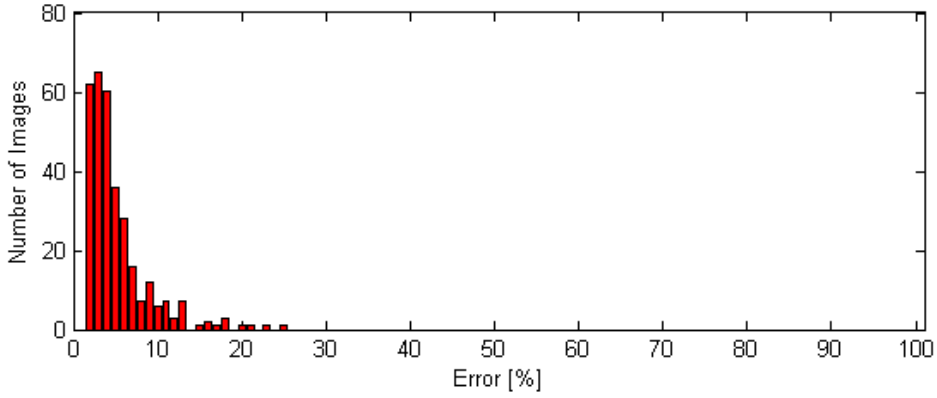
4.4.3 Our Algorithm



(a) Images with uniform background



(b) Images with non-uniform background



(c) All images

Figure 4.4: Error histograms of our algorithm

		Mean [%]	Std [%]	Min [%]	Max [%]
Face	FP	3.19	3.02	0.01	16.49
	FN	0.92	2.73	0.00	32.58
Hair	FP	16.91	43.98	0.00	421.02
	FN	8.93	9.33	0.00	79.78
Shoulder	FP	3.21	10.58	0.00	115.25
	FN	5.92	17.21	0.00	100.00
Background	FP	0.68	2.60	0.00	22.43
	FN	1.93	3.21	0.00	33.05
Overall image	Misclassified	3.92	3.67	1.39	24.43
	Unclassified	0.00	0.00	0.00	0.00
	Total error	**3.92**	3.67	1.39	24.43

Table 4.7: Results of our algorithm on images with uniform background

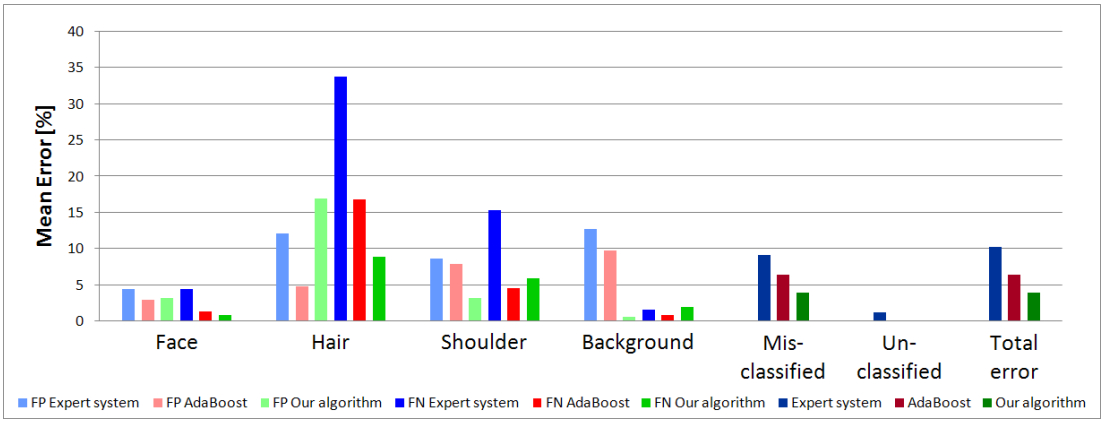
		Mean [%]	Std [%]	Min [%]	Max [%]
Face	FP	3.30	5.00	0.10	29.67
	FN	2.70	4.57	0.00	26.63
Hair	FP	21.90	49.06	0.00	317.83
	FN	12.06	14.09	0.00	91.37
Shoulder	FP	11.00	31.38	0.00	230.84
	FN	11.46	14.58	0.00	95.80
Background	FP	4.52	5.34	0.04	34.94
	FN	3.22	4.82	0.00	24.33
Overall image	Misclassified	5.93	3.42	2.00	17.94
	Unclassified	0.00	0.00	0.00	0.00
	Total error	**5.93**	3.42	2.00	17.94

Table 4.8: Results of our algorithm on images with non-uniform background

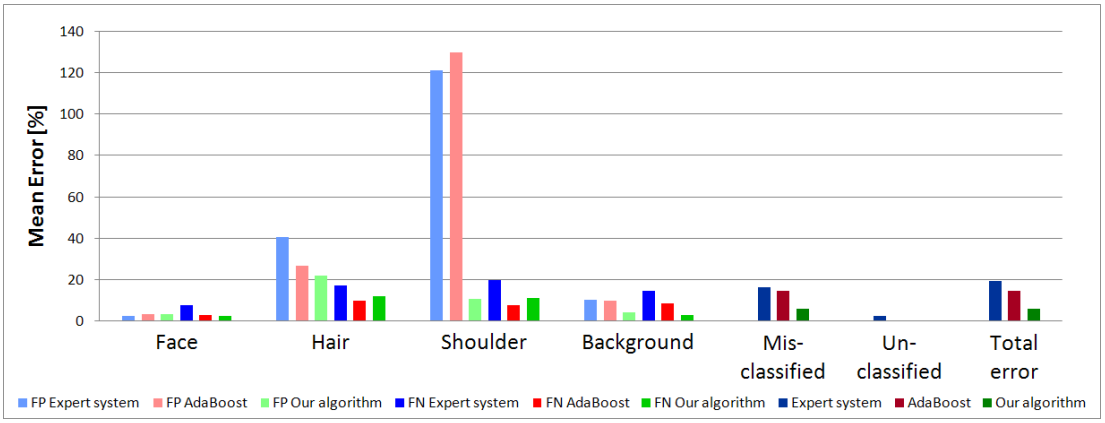
		Mean [%]	Std [%]	Min [%]	Max [%]
Face	FP	3.23	3.89	0.01	29.67
	FN	1.60	3.65	0.00	32.58
Hair	FP	18.83	45.99	0.00	421.02
	FN	10.13	11.48	0.00	91.37
Shoulder	FP	6.21	21.44	0.00	230.84
	FN	8.05	16.45	0.00	100.00
Background	FP	2.16	4.30	0.00	34.94
	FN	2.43	3.95	0.00	33.05
Overall image	Misclassified	4.69	3.71	1.39	24.43
	Unclassified	0.00	0.00	0.00	0.00
	Total error	**4.69**	3.71	1.39	24.43

Table 4.9: Results of our algorithm on all images

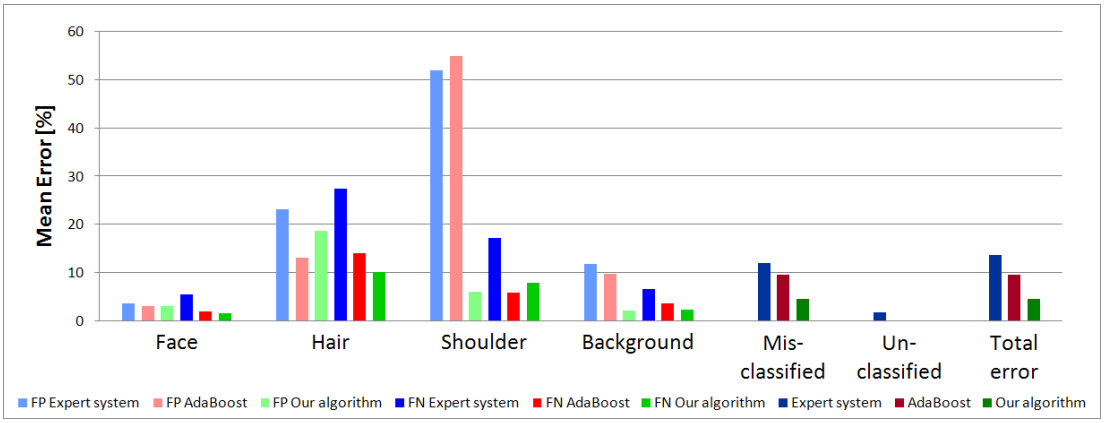
4.4.4 Comparison



(a) Images with uniform background



(b) Images with non-uniform background



(c) All images

Figure 4.5: Comparison of our algorithm to the expert system and AdaBoost. The charts present mean error rates and mean overall errors.

4.4.5 Background Classification

Our algorithm classifies 99.06% of all image backgrounds correctly, using the following tresholds in Equations 3.12 and 3.13: $T_1 = 25$, $T_2 = 0.5$ and $T_3 = 0.005 \times imageArea$. Only three out of 320 images, one with a uniform and two with a non-uniform background, are misclassifed. This result is slightly better than the outcome of a (yet unpublished) background classifier developed at the University of Zagreb, which uses two neural networks, one for homogeneous and one for non-homogeneous backgrounds. It makes a correct decision in approximately 98% of the cases.

4.5 Discussion

The results obtained in the previous section demonstrate that our algorithm outperforms both, the expert system and the AdaBoost classifier. The expert system has the worst overall performance of all three methods. Even on the set of images with uniform backgrounds it has a mean total error of 10.30%. Interestingly the mean false negative rate for the hair region is also very hight (33.81%), although the background region is uniform and therefore contrasts strongly with the hair region in most cases. As expected, the performance decreases on the set of images with non-uniform backgrounds. Here the mean false positive rate for the shoulder region is extremely high. 121.56% is far from any acceptable value. The mean total error on this set is 19.44%. Not surprisingly the error values on the set of all images are somewhere in the middle between the values for the other two sets (the mean total error is now 13.81%).

The AdaBoost classifier generally shows a higher performance than the expert system. The mean total error is 6.39% on uniform background images, 14.75% on non-uniform background images, and 9.60% on all images. However, on the set of images with non-uniform backgrounds the AdaBoost approach suffers from a very high false positive rate for the shoulder region too, inacceptable 130.24%.

Our algorithm shows a very good performance on all three datasets. Most error values are smaller as for the other two methods, or at least approximately equally good. We obtain a mean total error of 3.92% for uniform background images, 5.93% for non-uniform background images, and 4.69% for all images. As one can see in Figure 4.5, our method is quite robust and does not suffer from any extremely high error values, like the expert system and the AdaBoost classifier. This is a result of the

good generalization ability of our algorithm. By using only general knowledge about typical passport photographs we avoid that our method specializes in a certain set of images. In contrast to this the expert system is rather vulnerable to failing on new image sets, because it uses very specific rules.

Note that, when looking at the charts in Figure 4.5, one has to bear the duality of the false positive and false negative rate in mind. One can always minimize one of the rates by letting the other one grow. Hence a good performance is only given if both error rates are adequately low.

4.6 Qualitative Results

In this section we conclude Chapter 4 with the presentation of a few of our segmentation results. Figure 4.5 shows some examples of well segmented images, and in Figure 4.6 cases in which our algorithm has difficulties are illustrated. While the well segmented examples do not need further discussion, the incorrectly segmented ones deserve closer attention.

In the first example of Figure 4.6 the tie, which should be labeled as shoulder region, is misclassified as face region due to color similarity. The second example shows a case where the hair region grows into the background region. The reason for this are color and texture similarity (both the hair region and the part of the background that contains the stairway are fairly textured). The third row in Figure 4.6 presents an example in which the face regions displaces the hair region at the location of the ears, again due to color similarity. What is more problematic in this example is the fact that quite a few hair ends are covered by the background region, a common situation in images where thin hair ends stick out into the background region. These hair ends can then lead to wrong decision of the background classifier described in Section 3.6. The last examples shows an image in which our classifier can not distinguish between hair and shoulder region, and therefore the shoulder region is mislabeled as hair (see also Section 3.5.3).

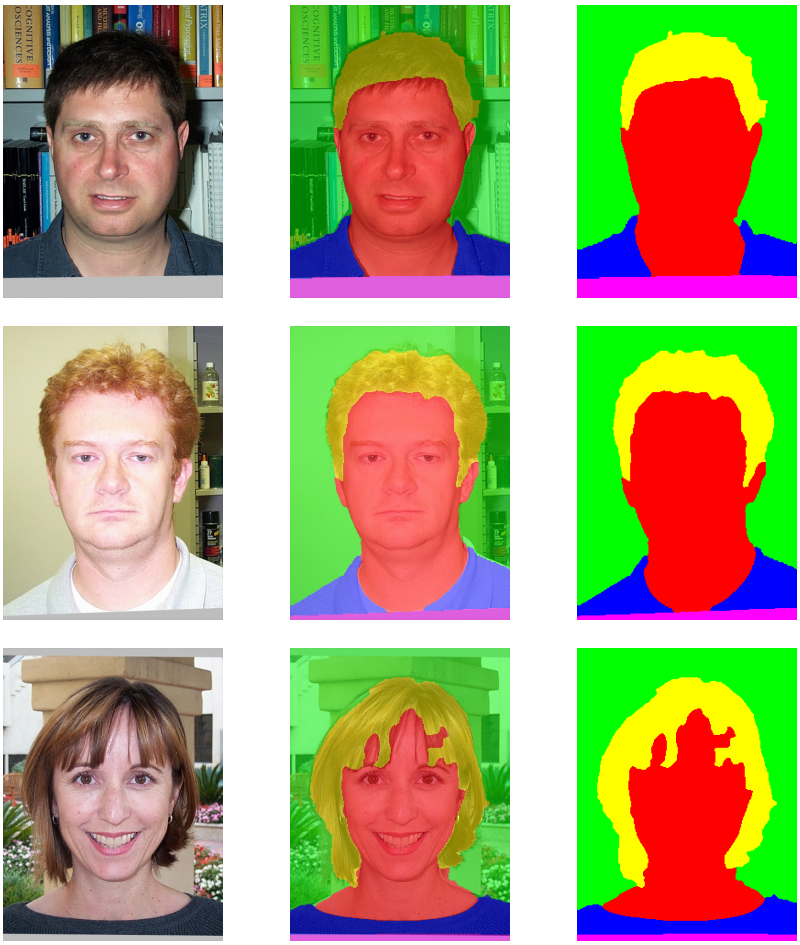


Figure 4.6: Well segmented images

Figure 4.5: Well segmented images (continued). The left column shows the input images, the middle column depicts the images overlaid with regions, and the right column illustrates the pure segmentation results.

Figure 4.6: Imperfectly segmented images. The left column shows the input images, the middle column depicts the images overlaid with regions, and the right column illustrates the pure segmentation results.

Chapter 5

Conclusion and Outlook

Contents

5.1 Conclusion

In this master's thesis we presented an unsupervised, knowledge based segmentation algorithm that partitions canonical face images into face, hair, shoulder and background region. The developed tool is intended to be part of an automatic passport photograph inspection framework, which checks passport photographs in terms of minimal quality requirements defined by ICAO. In addition we included a background classifier that distinguishes between uniform and non-uniform image backgrounds in our tool.

In Chapter 2 we first described different classical segmentation methods. Then we introduced our approach, which is a combination of a geodesic active contour and a functional that encompasses region information. This model allows us to incorporate gradient, color, texture and shape information into the segmentation process. Since the geodesic active contour is implemented with a weighted TV-norm, we also took a closer look at convex variational models.

The main part of this work is devoted to the description of our method. In Chapter 3 we showed how the proposed algorithm segments face images by letting well defined start regions grow, and how our approach integrates prior knowledge about typical

passport photographs whenever possible. In particular domain knowledge is involved in the definition of start regions, the region growing process and the post-processing stage.

In Chapter 4 we compared our algorithm to two other methods that also aim at solving the problem of segmenting face images. One is a rule based expert system, and the other is an AdaBoost classifier. We showed that our approach is superior to them. One reason for this is surely its good generalization ability. The knowledge that we incorporate into the segmentation process is very general and can be applied to any canonical face image. Thus our algorithm is not optimized for certain image sets, like, for example, the expert system, which is based on very specific rules. Finally we concluded the chapter with some qualitative results of our segmentation method.

5.2 Outlook

The results depicted in Figure 4.6 reveal that there is still room for future research. As one can see in Table 4.9, the hair region has the highest error rates of all regions. Furthermore improving the detection of hair will not only enhance the segmentation result, but also the background classifier. We have already encountered a significant performance gain by introducing individual shape probability maps for different hairstyles. However, the activation of a certain probability map depends on an estimation of the hairstyle after the first segmentation run. Consequently hair segmentation errors during this first segmentation run can have a considerably negative impact on the overall result. In order to improve the segmentation of hair one might use more advanced texture descriptors, like higher order moments or Gabor filters.

Another problem are images of people with long hair where the hair and shoulder color are very similar. In these cases our algorithm sometimes rejects the shoulder region and labels it as hair, a consequence of our post-processing policy of relabeling improbable shoulder regions.

Also glasses can cause problems. They severely affect the definition of the hair start region, and in the final segmentation they are often labeled as hair due to color similarity. To overcome this problem a glasses detector might be integrated into the segmentation algorithm.

Bibliography

[1] J. F. Aujol, G. Gilboa, T. Chan, and S. J. Osher. Structure-texture image decomposition: Modeling, algorithms, and parameter selection. *International Journal of Computer Vision*, 67(1):111–136, 2006.

[2] J. A. Birchbauer, A. Falkner, B. Frühstück, V. Krivec, S. Loncaric, and P. Scharfetter. ICAO knowledge based facial image segmentation - user requirements specification v1.0. Technical report, 2006.

[3] X. Bresson, S. Esedoglu, P. Vandergheynst, J. P. Thiran, and S. J. Osher. Global minimizers of the active contour/snake model. *International Conference on Free Boundary Problems: Theory and Applications*, 2005.

[4] X. Bresson, S. Esedoglu, P. Vandergheynst, J. P. Thiran, and S. J. Osher. Fast global minimization of the active contour/snake model. *Journal of Mathematical Imaging and Vision* , 28(2):151–167, 2007.

[5] Caltech. Caltech face database. http://www.vision.caltech.edu/archive.html.

[6] J. F. Canny. Finding edges and lines in images. Master's thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, 1983. Supervisor: J. Michael Brady.

[7] V. Caselles, R. Kimmel, and G. Sapiro. Geodesic active contours. *International Journal of Computer Vision*, 22(1):61–79, 1997.

[8] D. Chai and K. N. Ngan. Face segmentation using skin-color map in videophone applications. *IEEE Transactions on Circuits and Systems for Video Technology*, 9(4):551–564, 1999.

[9] A. Chambolle. An algorithm for total variation minimization and applications. *Journal of Mathematical Imaging and Vision*, 20(1-2):89–97, 2004.

[10] A. Chambolle and P.-L. Lions. Image recovery via total variation minimization and related problems. *Numerische Mathematik*, 76(2):167–188, 1997.

[11] T. F. Chan and S. Esedoglu. Aspects of total variation regularized L^1 function approximation. *SIAM Journal of Applied Mathematics*, 65(5):1817–1837, 2005.

[12] T. Chen, W. T. Yin, X. S. Zhou, D. Comaniciu, and T. S. Huang. Total variation models for variable lighting face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* , 28(90):1519–1524, 2006.

[13] D. Comaniciu and P. Meer. Mean shift analysis and applications. In *Proceedings of IEEE International Conference on Computer Vision*, pages 1197–1203, 1999.

[14] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence* , 24(5):603–619, 2002.

[15] M. Donoser and H. Bischof. ROI-SEG: Unsupervised color segmentation by combining differently focused sub results. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.

[16] G. J. Edwards, A. Lanitis, C. J. Taylor, and T. F. Cootes. Statistical models of face images: Improving specificity. *Image and Vision Computing* , 16(3):203–211, 1998.

[17] G. J. Edwards, C. J. Taylor, and T. F. Cootes. Interpreting face images using active appearance models. In *Proceedings of IEEE International Conference on Automatic Face and Gesture Recognition*, pages 300–305, 1998.

[18] FER. Segmentation of id photographs using adaboost algorithm. Technical report, Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia, 2008.

[19] H. Geng, X. Deng, and H. Ali. Mining gene microarray data with adaptive feature scaling. In *Proceedings of IEEE International Conference on Electro Information Technology*, pages 6pp.–, 2005.

[20] R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Prentice Hall, 3rd edition, 2008.

[21] J. Hadamard. Sur les problèmes aux dérivées partielles et leur signification physique. *Princeton University Bulletin*, 13:49–52, 1902.

[22] R. M. Haralick. Digital step edges from zero-crossings of second directional derivatives. *IEEE Transactions on Pattern Analysis and Machine Intelligence* , 6(1):58–68, 1984.

[23] R. M. Haralick and L. G. Shapiro. *Computer and Robot Vision, Volume 1*. Addison-Wesley, 1992.

[24] R. M. Haralick and L. T. Watson. A facet model for image data. *Computer Graphics and Image Processing* , 15(2):113–129, 1981.

[25] S. L. Horowitz and T. Pavlidis. Picture segmentation by a direct split and merge procedure. In *Proceedings of IEEE International Conference on Pattern Recognition*, pages 424–433, 1974.

[26] R.-L. Hsu, M. Abdel-Mottaleb, and A. K. Jain. Face detection in color images. *IEEE Transactions on Pattern Analysis and Machine Intelligence* , 24(5):696–706, 2002.

[27] J. Huang and D. Mumford. Statistics of natural images and models. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 541–547, 1999.

[28] ICAO. International civil aviation organization. http://www.icao.int.

[29] ISO/IEC. Editor's working draft text for revision of biometric data interchange formats - part 5: Face image data. Technical report, 2007. JTC 1/SC 37 N 2113.

[30] B. Jähne. *Digital Image Processing - Concepts, Algorithms and Scientific Applications*. Springer-Verlag, 2nd edition, 1993.

[31] M. J. Jones and J. M. Rehg. Statistical color models with application to skin detection. *International Journal of Computer Vision* , 46(1):81–96, 2002.

[32] P. Kakumanu, S. Makrogiannis, and N. Bourbakis. A survey of skin-color modeling and detection methods. *Pattern Recognition*, 40(3):1106–1122, 2007.

[33] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. In *Proceedings of IEEE International Conference on Computer Vision*, pages 259–268, 1987.

[34] A. Kundu and S. K. Mitra. A new algorithm for image edge extraction using a statistical classifier approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence* , 9(4):569–577, 1987.

[35] A. Lanitis, C. J. Taylor, and T. F. Cootes. Automatic interpretation and coding of face images using flexible models. *IEEE Transactions on Pattern Analysis and Machine Intelligence* , 19(7):743–756, 1997.

[36] D. Marr. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. W.H. Freeman and Company, 1982.

[37] D. Marr and E. C. Hildreth. Theory of edge detection. *Proceedings of the Royal Society of London*, B-207(1167):187–217, 1980.

[38] D. L. Milgram. Region extraction using convergent evidence. *Computer Graphics and Image Processing* , 11(1):1–12, 1979.

[39] M. Nikolova, S. Esedoglu, and T. F. Chan. Algorithms for finding global minimizers of image segmentation and denoising models. *SIAM Journal of Applied Mathematics*, 66(5):1632–1648, 2006.

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

[41] T. Pock, M. Unger, D. Cremers, and H. Bischof. Fast and exact solution of total variation models on the gpu. In *CVPR Workshop on Visual Computer Vision on GPUs*, pages 1–8, 2008.

[42] A. Rosenfeld and P. de la Torre. Histogram concavity analysis as an aid in threshold selection. *IEEE Transaction on Systems, Man and Cybernetics*, 13(3):231–235, 1983.

[43] H. A. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* , 20(1):23–38, 1998.

[44] H. A. Rowley, S. Baluja, and T. Kanade. Rotation invariant neural network-based face detection. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 38–44, 1998.

[45] L. I. Rudin, S. J. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, 60(1–4):259–268, 1992.

[46] P. K. Sahoo, S. Soltani, A. K. C. Wong, and Y. C. Chen. A survey of thresholding techniques. *Computer Vision , Graphics and Image Processing* , 41(2):233–260, 1988.

[47] I. Sobel and G. Feldman. A 3x3 isotropic gradient operator for image processing. Presented at a talk at the Stanford Artificial Project in 1968, unpublished but often cited (e.g. in Pattern Classification and Scene Analysis, Duda, R. and Hart, P., John Wiley and Sons, 1973, pages 271-272), 1968.

[48] M. Sonka, V. Hlavac, and R. Boyle. *Image Processing, Analysis, and Machine Vision*. PWS, 2nd edition, 1999.

[49] M. Storer, M. Urschler, H. Bischof, and J. A. Birchbauer. Face image normalization and expression/pose validation for the analysis of machine readable travel documents. In *Proceedings of OAGM/AAPR Conference*, pages 29–39, 2008.

[50] M. Subasic, S. Loncaric, and J. A. Birchbauer. Expert system segmentation of face images. Expert Systems with Applications, In Press, Available online 11 May 2008, 2008.

[51] V. Vezhnevets, V. Sazonov, and A. Andreeva. A survey on pixel-based skin color detection techniques. In *Proceedings of GraphiCon*, pages 85–92, 2003.

[52] P. A. Viola and M. J. Jones. Robust real-time face detection. *International Journal of Computer Vision* , 57(2):137–154, 2004.

[53] J. S. Weszka, C. R. Dyer, and A. Rosenfeld. A comparative study of texture measures for terrain classification. *IEEE Transaction on Systems, Man and Cybernetics*, 6(4):269–286, 1976.

[54] J. S. Weszka and A. Rosenfeld. Histogram modification for threshold selection. *IEEE Transaction on Systems, Man and Cybernetics*, 9(1):38–52, 1979.

[55] D. Zhang, S. Z. Li, and D. Gatica Perez. Real-time face detection using boosting in hierarchical feature spaces. In *Proceedings of IEEE International Conference on Pattern Recognition*, pages 411–414, 2004.