Automation of Closed-Form and Spectral Matting Methods for Intelligent Surveillance Applications

AUTOMATION OF CLOSED-FORM AND SPECTRAL MATTING METHODS FOR INTELLIGENT SURVEILLANCE APPLICATIONS

BY

MUHAMMAD ALRABEIAH, B.Sc.

A THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

AND THE SCHOOL OF GRADUATE STUDIES

OF MCMASTER UNIVERSITY

IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

Master of Applied Science (2015) McMaster University

(Electrical & Computer Engineering) Hamilton, Ontario, Canada

TITLE:	Automation of Closed-Form and Spectral Matting Methods for Intelligent Surveillance Applications
AUTHOR:	Muhammad Alrabeiah B.Sc., (Electrical Engineering) King Saud University, Riyadh, Kingdom of Saudi Arabia
SUPERVISOR:	Dr. Jun Chen
NUMBER OF PAGES:	xiii, 76

To my mother Norah and my fiancee Saule

Abstract

Machine-driven analysis of visual data is the hard core of intelligent surveillance systems. Its main goal is to recognize different objects in the video sequence and their behaviour. Such operation is very challenging due to the dynamic nature of the scene and the lack of semantic-comprehension for visual data in machines. The general flow of the recognition process starts with the object extraction task. For so long, this task has been performed using image segmentation. However, recent years have seen the emergence of another contender, image matting. As a well-known process, matting has a very rich literature, most of which is designated to interactive approaches for applications like movie editing. Thus, it was conventionally not considered for visual data analysis operations.

Following the new shift toward matting as a means to object extraction, two methods have stood out for their foreground-extraction accuracy and, more importantly, their automation potential. These methods are Closed-Form Matting (CFM) and Spectral Matting (SM). They pose the matting process as either a constrained optimization problem or a segmentation-like component selection process. This difference of formulation stems from an interesting difference of perspective on the matting process, opening the door for more automation possibilities. Consequently, both of these methods have been the subject of some automation attempts that produced some

intriguing results.

For their importance and potential, this thesis will provide detailed discussion and analysis on two of the most successful techniques proposed to automate the CFM and SM methods. In the beginning, focus will be on introducing the theoretical grounds of both matting methods as well as the automatic techniques. Then, it will be shifted toward a full analysis and assessment of the performance and implementation of these automation attempts. To conclude the thesis, a brief discussion on possible improvements will be presented, within which a hybrid technique is proposed to combine the best features of the reviewed two techniques.

Acknowledgements

This thesis crowns my two-year pursuit of a Master's of Applied Science (M.A.Sc.) degree in the Electrical and Computer Engineering (ECE) department at McMaster university. At the end of this journey, I would like to take the chance to express my sincere gratitude to those whom have supported and helped me all along. Staring with the Saudi Arabia Cultural Bureau (SACB) in Canada, I say thank you very much for funding my studies and for all the personal support I received. To my supervisor, professor Jun Chen, I'd like to express my appreciation for the kind guidance you provided me with throughout these two years, and to my colleague and friend, Mostafa Medra, I want to say thank you for being there for me. Finally, I dedicate my deepest gratitude to the three people whom enclosed me with their endless kindness, support, and love, my mother Norah, my fiancee Saule, and my best friend Hisham.

Notation and abbreviations

OD	Object Detection
OE	Object Extraction
SVM	Support Vector Machine
CFM	Closed-Form Matting
SM	Spectral Matting
RGB	Red, Green, Blue
RLLS	Regularized Linear Least Squares
OLS	Ordinary Least Squares
PSD	Positive Semi-Definite
OSVS	Object-Shape Vector Space
PCA	Principle Component Analysis
MS	Mean Shift
HSV	Hue, Saturation, Value

Contents

List of Figures

Chapter 1

Introduction

Visual information representation is one of the most informative and widely favorable forms of information representation throughout humanity's history. In the modern age, specifically the digital era, representing visual information has taken a major leap forward with respect to two fundamental aspects: availability and content. A quick look through the last century would suffice to show the fast growing development of equipment and devices used for "capturing" and "displaying" the two main forms of representation, images and videos. This, consequently, has made visual information available almost everywhere to everyone. As for the content of the visual information, henceforth referred to as *the visual content*, it has had its share of development as well, arguably a larger share of development compared to availability. To see such improvement in content, one could think of how imagery evolved from producing grayscale images to color ones or from producing high resolution images to three-dimensional (3D) ones. This continuing evolvement has made each of the aforementioned aspects an area of research in itself.

The main theme of this thesis is content analysis of visual information. More

specifically, the focus is on studying the topic of object detection and extraction for visual content analysis. Owing to the large number of applications in which this topic is discussed, studying object detection and extraction is somehow infeasible and may tend to be very complicated if an application is not specified. Thus, the discussion of the topic here will be directed toward intelligent surveillance systems as one of the important and rapidly developing applications. Generally, the visual content of surveillance video sequences convey a lot of information about the surrounding, yet the process of automatically extracting this information from the visual data[1] without human intervention is still posing a major challenge. To a large extent, the reason behind that could be attributed to the lack of semantic-comprehension of visual data in machines, where only primary information, like light and color intensities, could be directly extracted from visual data. Hence, much of the research in the area of visual content analysis has been dedicated to developing tools and techniques with which more meaningful, albeit obscure, information is obtained and analyzed.

Prior to introducing the main statement of this work, brief overviews on some essential topics and concepts need to be presented. To do that, this chapter is divided into three sections. In the first section, a quick look at the literature of *object detection* and *object extraction* from images and video sequences will take place. This literature review helps shape the understanding of both topics and their relation to the problem statement. Within this section, light will be shed on the definitions of some basic concepts and popular approaches having been developed to address major issues related to these two topics. This kind of discussion justifies the need for the second section in which the concept of *matting* and its literature will be briefly introduced. The relation between matting and object detection and extraction will

[1] Visual data refers to the digital "representation" of the visual information captured from a scene.

also be clarified, paving the way for the main problem statement in the third and last section of this chapter.

1.1 Object detection and extraction

An obvious goal in visual content analysis is to give machines the ability to recognize different objects in an image or video sequence; however, recognition is a task that requires high-level understanding of the visual content, which, in turn, could not be grasped from primitive (low-level) visual data. This difficulty gives rise to the need for some sort of intermediate processing stage capable of unearthing more perceptually relevant data from the primitive one. Tasks like localizing the objects of interest in the scene and isolating them from their irrelevant surroundings are very appealing candidates to do the job. These two tasks are widely-known as Object Detection (OD) and Object Extraction (OE), respectively.

For over three decades, OD and OE have been extensively studied yielding several approaches of implementation. On a very basic level, these approaches differ according to the type of visual content they process, whether it is an image or a video sequence. Nevertheless, the difference is subtle and manifested in the extra temporal dimension of videos, i.e., a video is just a time sequences of still images. Based on such distinction, many early attempts, as well as the new ones, have been focusing on addressing the issues of detecting and extracting objects of interest in still images, and once they turn out good results, they get extended to the video case. Hence, it stands the reason if a brief literature review of OD and OE is first introduced for images and then for videos.

1.1.1 Object detection and extraction in images

As stated in [21], the general classical approach for object recognition in computer vision was based on three main processing stages: 1) low-level feature extraction from the image (OD stage), 2) *segmentation* of the image into semantically-related regions (OE stage), 3) recognition of each of the segmented regions. At the first glance, the process looks very elegant and consistent. Despite of that, the processing flow turned out to be a disappointment, for it failed to attain sustainable results making it practically incompetent. The main reason for such failure was mostly attributed to the segmentation stage. In order to see why, a quick look at the concept of segmentation is required. The segmentation process aims to break an image into several regions based on some primitive features of the visual data, like intensity discontinuity and pixel similarity. The pixels forming each region, therefore, either share some common feature or are contained within the same boundary line– [12] makes a good reference for more details on the subject. This reliance on low-level features is believed to be the reason why segmentation is unable of capturing visually meaningful structures, leading to the failure of the classical approach [21].

The major disappointment accompanying the classical approach pushed toward a segmentation-free object detection framework. Getting rid of the segmentation stage meant that object extraction would no longer be required. This movement, revolutionary at its time, resulted in an explosion of techniques with impressive performances, like [8] and [24] among others. Loosely, these techniques are grouped as a single approach called *the sliding window*. This name comes as an obvious result of the way they operate; each of them defines a fixed-size window and runs it across the whole image looking for some a posteriori features [17]. Moreover, some methods

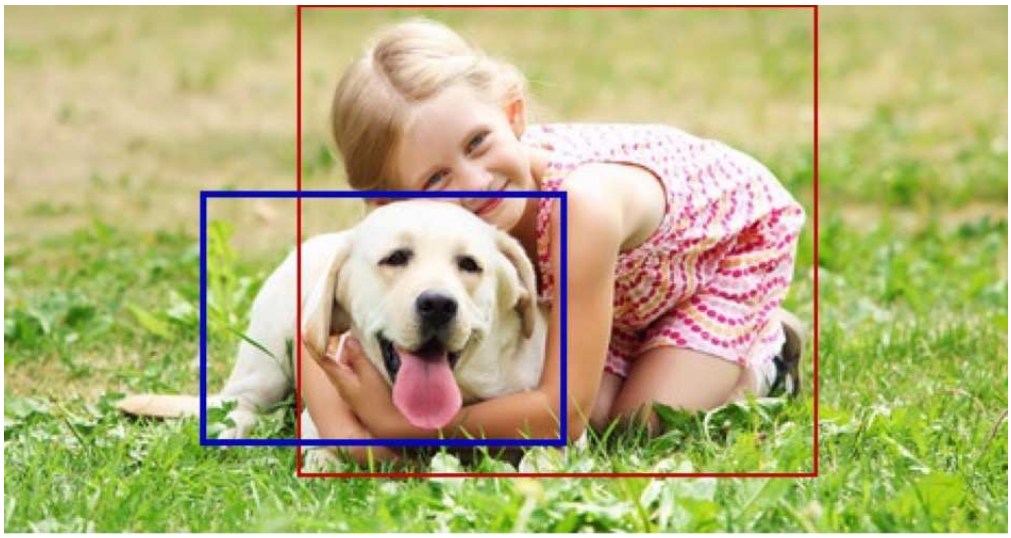

Figure 1.1: The red bounding box cannot only fit the little girl which may cause a detection error. On the other hand, the blue box shows another situation when it is possible to contain close-to-rectangular shapes like the dog's face.

generate a pyramid using different sizes of the same image and, then, run the window across all of them [8]. Many of the sliding window techniques, if not all, rely on a well-trained classifier within each window to test the existence of the object of interest. Commonly this classifier is based on the Support Vector Machine (SVM) learning algorithm. Despite their recorded success in detection, they have a common major drawback. Ironically, it is their source of strength, the sliding window concept. As it is argued in [21], sliding window techniques fail in recognizing objects that do not have close-to-rectangular shapes. Such failure implicitly points toward the localization method; it produces a bounding box around the object of interest, which may not completely contain the object or may include other unrelated objects, see Figure 1.1.

Recently, there have been some voices that call for the return of segmentation in

the object recognition process as a way to tackle the shortcomings of sliding window approaches. Malisiewicz and Alexei's voices are among the clearest ones. In their paper [21], they lay the groundwork for why incorporating segmentation within OD techniques would provide a great boost to their performances. This new direction brought the discussion on OE back to life. Many attempts, [11], [20], and [28] to name a few, have been made to incorporate segmentation in the process of OD and OE. Generally speaking, the majority of the proposed techniques have succeeded in utilizing segmentation to improve their object localization abilities. The key advantage that segmentation has provided is the ability to specify a shape-adaptable detection region instead of the bounding box provided by the sliding window methods. However, one should not think that this puts an end to the whole OD and OE problem; while segmentation brings the so-called *spatial support* to detection techniques, it also introduces an extra layer of computational load.

1.1.2 Object detection and extraction in videos

A natural extension of the discussion on OD and OE is to the case of video sequences. As previously stated, many techniques that turned out to be successful in the case of images are further developed to process videos. However, a main trend with videos is the focus on human recognition scenarios like in surveillance systems or TV sport broadcast. Several approaches have been developed for that end [26], but many of them could be used, with some modifications, to recognize other objects of interest. Both processes, OD and OE, in videos exploit the additional temporal dimension. The variation between consecutive frames resulting from the motion of the object constitutes an important factor for localization and segmentation. As it was in the

case of images, object recognition in video sequences could be performed without using the segmentation stage, i.e., without OE; however, some may argue that OE would provide the spatial support required to improve the performance of current techniques. Thus segmentation has been recently incorporated in many methods, like [3] and [35], and the results are promising.

Detection of salient objects, like detecting humans, in videos could be done in two main stages. First, the variation resulting from motion in the consecutive frames must be detected. Then, regions where those variations are detected are analyzed to localize and identify the object [26]. At first, the need for these two stages may seem confusing, for why would it happen in two stages if the object in motion is the one of interest? This is true in highly controlled environments, which, unfortunately, is not the case for real-world scenes; many sequences contain several moving object that are not of any significance, e.g., the wavy motion of tree leaves or the continuous sprinkling from a water fountain. For that, the two stages are necessary to obtain reasonable detection outcomes. There are roughly three main approaches to object motion detection, some of which incorporate OE by applying segmentation and others do not. These approaches are: background subtraction, optical flow, and spatio-temporal filtering. For the object identification stage, three approaches could be defined as well: shape-based, motion-based, and texture-based. In many of these approaches, techniques that have been developed for the image case are extended to work on videos, like [9] and [33].

Due to the extra boost in performance segmentation gives, many video OE techniques incorporate a segmentation stage. However, the way with which segmentation is implemented differs according to the processing method and the features used to

aid the process. Setting aside the fact that their work took place more than thirteen years ago, Megret and DeMenthon presented an elegant classification of video segmentation techniques [22], one that still holds to this day. They defined three major video segmentation approaches: segmentation with spatial priority, segmentation by trajectory grouping, and joint spatial and temporal segmentation. For the first class of approaches, the segmentation process is performed on a frame-by-frame basis. The extracted regions in the sequence of frames are linked to each other using a certain tracking method, such as motion similarity or motion model fitting. On the other hand, the second class of approaches relies on the temporal grouping where the trajectory of some defined features is tracked throughout the sequence and, then, grouping is performed based on some rules. The third and last class is merely a combination of the previous two. It favors no dimension over the other and operates on the whole video sequence at once. An important observation should be made; out of those three approaches, only the first one can be implemented in real-time processing [10]. That is the case because the other two approaches require the whole video sequence to be available for the segmentation process to take place. However, this advantage does not come without a price; temporal consistency for this approach is an issue. An accurate tracking method must be implemented, or, otherwise, jitters in the motion of the extracted object will be apparent.

1.2 Image and video matting

A well-known problem in the literature of image and video processing is the separation of the foreground and background. Salient objects in the scene make the main source of visual information, and, thus, they are collectively considered the foreground of the

image. The background, obviously, is made up of the other less informative objects. Such definitions may sound a bit amorphous, but they have the versatility needed to suite the various applications applying the separation concept. Commonly this process is referred to as *matting*, and it was mathematically formulated in 1984 in the work of Porter and Duff [27]. The matting process generally assumes that the image is a composite of two layers, a background and a foreground, with a compositing factor α called the alpha matte or the opacity factor [31]. This factor is defined for all the pixels in the image, and it takes values in the interval [0,1]. Mathematically, for each pixel of an image I, the compositing equation is:

$$I_i = \alpha_i F_i + (1 - \alpha_i) B_i, \tag{1.1}$$

where I_i is the intensity of *ith* pixel of the image, F_i is the intensity of the *ith* pixel of the foreground layer, B_i is the intensity of the *ith* pixel of the background layer, and α_i is the opacity factor associated with the *ith* pixel. As Equation (1.1) suggests, a pixel of an image I is made of the sum of two pixels from two different layers with a "mixing" factor α that determines the percentage of contribution of each layer. For pixels that belong solely to one of the two layers, the alpha matte takes the value of either 0 or 1. The case of $\alpha = 0$ is for a definite background pixel where $\alpha = 1$ is the case for a definite foreground pixel.

Equation (1.1) is inherently under-constrained since there are more unknown variables than known values, which, in itself, poses the major challenge in the matting process. For grayscale images, each pixel has one known value, the pixel intensity I_i, and three unknown variables, F_i, B_i, and α. The situation gets worsened when dealing with color images; each color channel has its compositing equation, and all of these

equations share the same opacity factor. In this case, and depending on the color space used, there are at least seven unknowns and three known values. This striking observation depicts how challenging, if not impossible, it is to tackle the matting problem knowing only the input image. Therefore, many methods and algorithms have been proposed in the last four decades most of which focus on humanly-provided input. This kind of input is provided in the form of three region map, named *a trimap*, in which a region defines the absolute background, another defines the absolute foreground, and finally the third one defines the unknown region that needs to be estimated. The latter region is usually covering the boundary between the foreground and the background.

Matting methods could be generally classified into four broad categories [31]: color-sampling-based methods, affinity-based methods, methods using optimization by combining sampling and affinities, and lastly matting with extra information. The first category of methods utilizes the fact that pixels in a region may exhibit some level of color correlation, which could be very high in some parts of the image [4]. They imply a local smoothness assumption in a neighborhood. Then, they use samples from nearby foreground and background to estimate the unknown F_i and B_i pixels. Although such methods sound simple, their implementation is not. In addition, the results they provide are often not very accurate. An attempt to combat the problems of color-sampling-based matting is established in the second category, affinity-based methods. These methods avoid enforcing certain color distributions or sampling different neighborhoods. Instead, they define various affinities within a very small neighborhood, as small as a 3x3 neighborhood. Within this window some sort of assumption may apply to derive the relation between pixels, like in [18]. Despite their

improved accuracy, they have some issues like error propagating and accumulating. More robust matting methods could be obtained by leveraging the strengths of the former two categories, which is exactly what the methods of the third category do. They formulate an energy function in which sampling and affinities are used. Then, they plug it in an optimization problem where matting parameters are estimated, e.g., the easy matting method [13]. Finally, with a slight deviation in concept, the fourth category of methods handles the matting problem. These methods require extra information to be provided about the scene. Such additional information usually comes from the device capturing the scene or the settings of the scene itself. Green screen and flash matting, respectively, make famous examples. More details about matting approaches could be found in the survey presented in [31].

As in OD and OE, Section 1.1, video matting could be viewed as an extension of the image matting case. Many of the methods developed for image matting could be altered to handle the video matting problem. However, some different challenges present themselves in this case. One of the major ones is the ability of the method to efficiently process the large amount of data in a video sequence. Average quality video-recording systems usually work with a rate of 30 frames per second, called the frame rate. Hence, the matting process must be conducted within roughly 33 milliseconds for real-time matting to be implemented. If not, there is still the challenge of pulling the alpha matte of each frame in the large stack of images, called the video volume. Moreover, presuming the speed issue is resolved, video matting faces the obstacle of producing a consistent sequence of alpha mattes. In frame-by-frame processing, jittering is a likely problem. Loosely, it could be attributed to the inconsistence of errors resulting from estimating the alpha matte in each frame. Several solutions have

been proposed, most of which heavily rely on human interaction [31].

Unlike its popularity in film and TV production industries, matting is yet to play a key role in areas related to visual content analysis. A main reason behind that could be the lack of automatic efficient image and video matting methods. For matting to be done, most of the existing methods, in all four categories, require human interaction (or human guidance). This is usually done by asking the user to provide a trimap for key frames to alleviate the difficulty and get the matting process started. For a long time, such a dependence made matting an inconvenient choice. However, this has recently changed; with the introduction of efficient interactive image matting methods, like [1], [18], and [19], and the advancement in machine learning, matting is becoming a key player in the field of visual content analysis.

1.3 Thesis statement

Leveraging the strengths of some matting methods to solve the OE problem has been proposed in some recently published research [15], [19], [34]. Offering a slightly different view on the problem, matting presents a new approach for extracting salient objects in surveillance applications. The fundamental advantage distinguishing matting from segmentation is that it pays attention to the concepts of foreground and background. While segmentation breaks down the image or video into several pieces representing different coherent regions of the image, matting incorporates the primary sense of object classification in the process; regions either make up the foreground or background objects in the image or video. Hence, matting stands out as a promising candidate that could overcome some of the shortcomings of segmentation and provide the sought-after spatial support for OD and OE tasks.

This thesis is aimed to provide a detailed review and analysis on the automation of two of the most advanced matting methods, closed-form and spectral matting. These two methods represent a milestone in interactive image matting, and several attempts have been made to automate them. With implementation and performance analysis, the main advantages and shortcomings of those automation attempts will be highlighted, and based on that, the possibility of a hybrid technique combining their major advantages will be briefly discussed. All that is organized as follows: Chapter 2 is dedicated to presenting the two techniques and their theoretical foundation. Chapter 3 will present two of the most advanced attempts having been made to automate these techniques, namely the shape-prior automation and the adaptive component detection and tracking techniques. Performance evaluation and analysis of two automatic techniques will be the core of discussion in Chapter 4. Hybridization and its key advantage and major shortcomings will be discusses in Chapter 5. Finally, Chapter 6 will present some conclusions and possible future work.

Chapter 2

Closed-form and spectral matting methods

Recently, matting has been rediscovered as a promising tool for OE in images and videos. The way in which the matting problem is formulated encodes a primitive sense of classification, i.e., matting breaks down the image into two layers, namely foreground and background, and provides a compositing factor. As explained in Section 1.2, the mathematical formulation of the matting problem results in an under-constrained problem. There have been many successful attempts to tackle the problem, producing a rich literature of techniques. Despite the variety, what most of these techniques have in common is their reliance on a set of user-provided constraints, usually given in the form of a trimap, see Figure 2.1. This used to be the main obstacle for utilizing matting in OE tasks, in which the process must be completely automatic. However, several new proposals came out recently aiming to automate the matting process. In many of them, if not all, a machine learning technique is implemented in an attempt to compensate for the lack of user input. In general, the performance

Figure 2.1: Starting from the left, the first image represents the input image. The second one is an example of a trimap inputted by the user, in which the red region is definite foreground with $\alpha = 1$, blue is definite background with $\alpha = 0$, and white is the unknown region where α is being estimated. Finally, the third image is an example of the possible output matte.

of these newly developed automatic matting techniques depends on how well they manage the trade off between matte accuracy and computational efficiency.

Among all the proposed matting methods, Closed-Form Matting (CFM) and Spectral Matting (SM) [18], [19] stand out as milestones in the matting research. The CFM method formulates the matting problem as a quadratic optimization problem with some linear constraints. Such a formulation is very desirable since it is relatively easy to solve and, when given enough constraints, it often produces a high-quality alpha matte. As for the second method, SM approaches the matting problem from a different angle. Instead of formulating an optimization problem, it follows a segmentation-like process where a set of matting components accounting for different possible regions in the image are produced. Both methods show interesting perspectives on automating the matting process. For that, several automation techniques have been proposed for each of them [15],[30],[34]. In preparation to discuss and analyze the performance of the most successful ones, Chapters 3 and 4, this chapter will

present each of the two interactive methods as well as their theoretical primes. Section 1 will introduce the basic ground upon which the two methods are built. Then, Section 2 and 3 will provide a detailed discussion on CFM and SM, respectively.

2.1 Spatial constancy and the color line model

Both matting methods, CFM and SM, share the same premise that is constructing a pixel affinity matrix called the *matting Laplacian*, but they part ways on how to utilize this matrix in the matting process. Basically, this matrix describes the relation between pixels within a very small spatial window, which is placed in all possible locations across the image. Inside the window, a smoothness model is assumed to be approximately satisfied in the foreground and background layers, so the relation between pixels in the image could be derived. Two different smoothness models are briefly introduced here. They simply account for the difference in the type of image being processed, grayscale or color image. These models are the *spatial constancy model* and the *color line model*.

Understanding the two models is essential to understand how the two methods work. For grayscale images, the assumption is fairly simple; the intensities within the spatial window are assumed to be constant for both layers. This model will henceforth be referred to as the spatial constancy model. Since this model reduces the number of variables within the window, it helps combat the inherent problem of the matting process, being severely under-constrained. However, the assumption is not necessarily true for all locations, and this will be shown later to be the main source of errors in both methods. As for the case of color images, a slightly more sophisticated

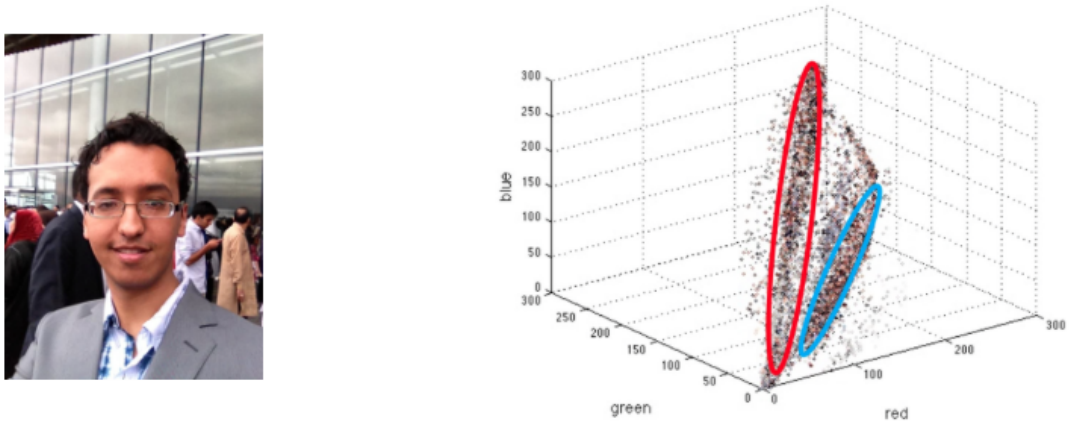

Figure 2.2: An example of an input image and the tendency of its color vectors to form elongated clusters in the RGB space.

smoothness model is used, called the color line model. Before describing it, an interesting observation, pointed out by Omer and Werman [25], needs to be discussed, for it forms the ground for the model. In their work, Omer and Werman noticed that for many images, most of the color vectors tend to align in elongated clusters in the RGB space, see Figure 2.2. Although not all these clusters are elongated ellipsoids, their skeleton could be fitted with a straight line. This observation is almost always true if a small spatial window is considered like in Figure 2.3. Several samples from the same image show an almost-linear distribution of the color vectors. Inspired by this observation, Levin *et al* proposed the color line model [18]. It simply assumes that within a small spatial window, as small as a 3x3 window, the colors of the pixels in both the foreground and background are aligned along a line in the RGB space. In other words, the colors of all pixels in the window could be viewed as a linear mix of two colors. The validity of this model is not upheld in every window of a natural image, yet for reasons that will become clear in Chapter 4, the model is very powerful in describing the relation among the pixels

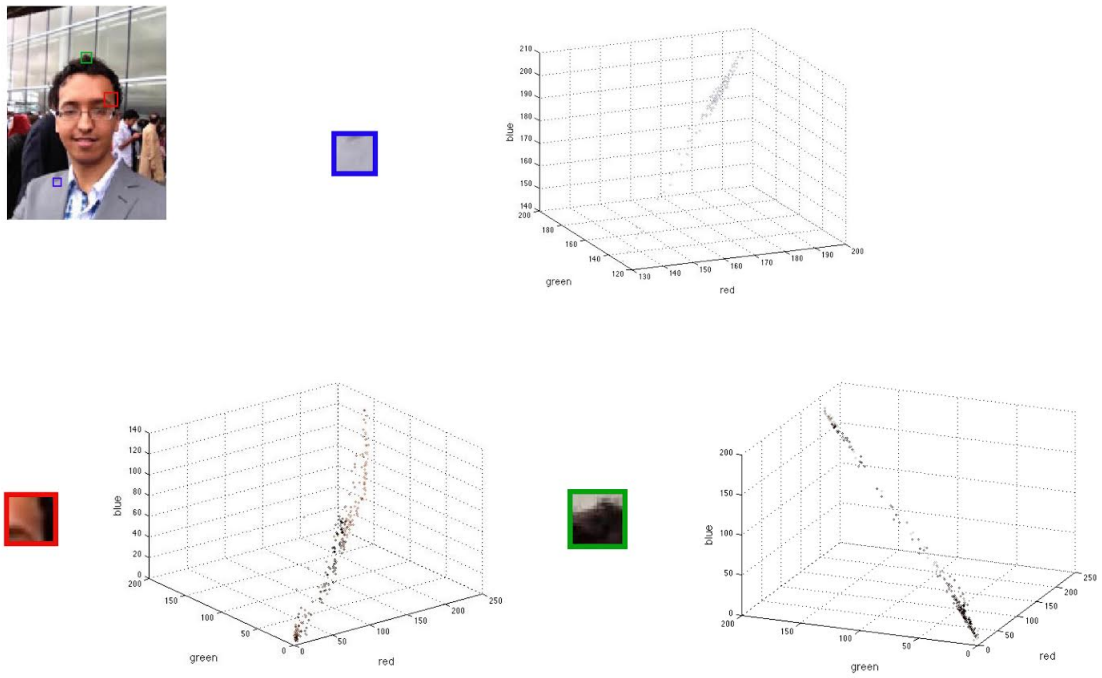

Figure 2.3: Three different 16x16 windows are sampled from the input image. For each window, the distribution of the color vectors is shown. For all three samples, color vectors are aligned in elongated clusters with a linear skeleton.

2.2 Closed-form matting method

As stated in the previous section, constructing the matting Laplacian matrix is the core of both matting methods. This matrix is derived from the basic compositing equation, Equation (1.1), and the smoothness model assumed. Owing to the fact that color images, especially the ones represented with the RGB color system, are prevalent, the derivation of the CFM method and its implementation are going to be discussed for color images in this section. However, for other types of color systems or grayscale images, the discussion follows along the same lines of this one with some subtle differences.

The CFM method formulates the interactive matting process as a constrained quadratic optimization problem [18], in which the cost function is derived from the compositing equation and the color line model. Given a small spatial window (w_k), as small as 3x3 window, the colors of the foreground and background pixels are assumed to obey the following two linear equations:

$$F_i^c = \beta_i^c F_1^c + (1 - \beta_i^c) F_2^c, \tag{2.1}$$

$$B_i^c = \gamma_i^c B_1^c + (1 - \gamma_i^c) B_2^c, \tag{2.2}$$

where:

- γ_i^c and β_i^c are the mixing factors of the *ith* pixel in the *cth* color channel within the spatial window.
- F_1^c is the first foreground color intensity in the *cth* color channel.
- B_1^c is the first background color intensity in the *cth* color channel.

- F_2^c is the second foreground color intensity in the *cth* color channel.
- B_2^c is the second background color intensity in the *cth* color channel.
- F_i^c is the foreground color intensity of the *ith* pixel in the *cth* color channel within the spatial window.
- B_i^c is the background color intensity of the *ith* pixel in the *cth* color channel within the spatial window.

As Equations (2.1) and (2.2) show, two intensities from each color channel are mixed with different factors to make the color of the*ith* pixel of the foreground and background.

Now, within the same window w_k, each of these two equations is substituted in Equation (1.1) for each color channel to form the compositing equations of the *ith* pixel:

$$\begin{aligned} I_i^r &= \alpha_i F_i^r + (1-\alpha_i) B_i^r, \\ I_i^g &= \alpha_i F_i^g + (1-\alpha_i) B_i^g, \\ I_i^b &= \alpha_i F_i^b + (1-\alpha_i) B_i^b. \end{aligned} \tag{2.3}$$

Using Equations (2.1) and (2.2) and after some rearranging, for any pixel in w_k, the following system of linear equations is obtained:

$$y = Hx$$

$$\begin{bmatrix} I_i^r - B_2^r \\ I_i^g - B_2^g \\ I_i^b - B_2^b \end{bmatrix} = \begin{bmatrix} F_2^r - B_2^r & F_1^r - F_2^r & B_1^r - B_2^r \\ F_2^g - B_2^g & F_1^g - F_2^g & B_1^g - B_2^g \\ F_2^b - B_2^b & F_1^b - F_2^b & B_1^b - B_2^b \end{bmatrix} \begin{bmatrix} \alpha_i \\ \alpha_i \beta_i \\ (1-\alpha_i)\gamma_i \end{bmatrix}. \tag{2.4}$$

This system of linear equations is assumed to always have a unique solution despite

where in the image the spatial window is placed. At the moment, such assumption may seem unreasonable, but it will be justified when the contribution of the foreground and background colors get eliminated later on. Since the target in the process is finding α, it suffices to invert matrix H and take the result of multiplying its first row with the vector y. This is given by the following very important equation:

$$\begin{gathered}\alpha_i = a_1 I_i^r + a_2 I_i^g + a_3 I_i^b - a_1 B_2^r - a_2 B_2^g - a_3 B_2^b \\ \alpha_i = \textstyle\sum_{c=1}^{3} a_c I_i^c + b,\end{gathered} \tag{2.5}$$

where $I_i^r, I_i^g, I_i^b \equiv I_i^1, I_i^2, I_i^3$, $b = -(a_1 B_2^r + a_2 B_2^g + a_3 B_2^b)$, and finally, a_1, a_2, a_3 are the first row elements of H^{-1}.

Equation (2.5) states that the value of the opacity factor α could be locally represented as a scaled sum of the color intensities and a free variable b. This representation is very powerful because it decouples the alpha value from the texture complexity in the window. In many spatial windows across the image, pixels fully belong to either the foreground or the background. Hence, the opacity factor usually has a constant value (0 or 1) that could be simply obtained by setting the three scaling factors in Equation 2.5 to 0 and setting b to 1 or 0. Even with the locations where the window contains mixed pixels, the linear representation performs well, see Figure 2.4 for an example.

For all pixels in window w_k, the alpha values are found by minimizing a local cost function given by:

$$J_k(\alpha, a, b) = \sum_{i \in w_k} (\alpha_i - (\sum_{c=1}^{3} a_k^c I_i^c + b_k))^2 + \epsilon \sum_{c=1}^{3} (a_k^c)^2, \tag{2.6}$$

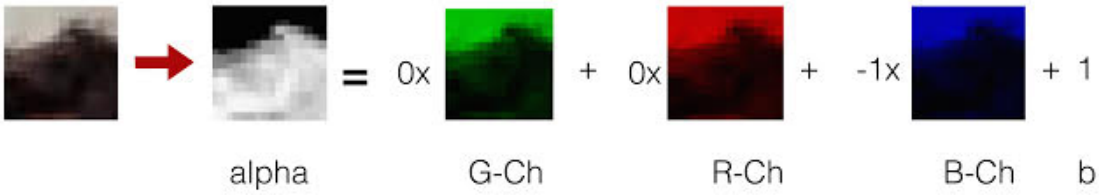

Figure 2.4: The window contains pixels from the foreground and background. By scaling the blue channel and adding 1, the required alpha matte is found. It is not very accurate, yet it captures the difference between both layers.

where k here refers to the number of the window and ϵ is a regularization factor.

This is just another formulation of a Regularized Linear Least Squares (RLLS) problem. The objective is to minimize the squared error between the alpha values and the estimated values given by the linear representation in Equation (2.5). The need for a regularization term in there could be justified with numerical stability; in some regions in the image, the very small spatial window may include pixels with the same color, making the minimization problem hard to solve without a bias. This could be better viewed when Equation (2.6) is rewritten in its fundamental form:

$$J_k(\alpha, a, b) = \|G_k \bar{x}_k - \bar{\alpha}\|_2^2, \tag{2.7}$$

where:

$$G_k = \begin{bmatrix} I_1^1 & I_1^2 & I_1^3 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ I_9^1 & I_9^2 & I_9^3 & 1 \\ \sqrt{\epsilon} & 0 & 0 & 0 \\ 0 & \sqrt{\epsilon} & 0 & 0 \\ 0 & 0 & \sqrt{\epsilon} & 0 \end{bmatrix}, \quad \bar{x_k} = \begin{bmatrix} a_k^1 \\ a_k^2 \\ a_k^3 \\ b_k \end{bmatrix}, \quad \bar{\alpha_k} = \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_9 \\ 0 \\ 0 \\ 0 \end{bmatrix},$$

and the alpha values in $\bar{\alpha}$ and the color intensities in G_k are index according to

their order in the spatial window. In the case of uniform color, matrix G_k would be noninvertible, which makes the closed-form solution of the Ordinary Least Squares (OLS) problem unattainable. This regularized problem is minimized with respect to x using:

$$\bar{x_k} = (G_k^T G_k)^{-1} G^T \bar{\alpha_k}. \tag{2.8}$$

Substituting back in Equation (2.7) and following some algebraic manipulations, a quadratic form in one variable, α, is obtained:

$$J_k(\alpha) = \alpha^T L_k \alpha, \tag{2.9}$$

where: $L_k = [G_k(G_k^T G_k)^{-1} G_k]^T [G_k(G_k^T G_k)^{-1} G_k]$.

Equation (2.9) is, indeed, the living testimonial for the ingeniousness of the CFM method. This, in general, could be attributed to two main reasons. First, the equation provides a closed-form formulation to the local cost function in which several unknowns, namely a_k^1, a_k^2, a_k^3, and b_k, are analytically eliminated making alpha the only variable in the equation. In addition, the function is quadratic with a Positive Semi-Definite (PSD) matrix L_k, making the process of optimizing with respect to alpha computationally very efficient.

With the derivation above in mind, a global optimization problem could be obtained, the solution of which is the sought-after image alpha matte. Exploiting the convexity of the cost functions within all windows in an image, the sum of these local functions produces a good global cost function retaining the two fundamental advantages mentioned before. After some tedious algebraic simplifications, including an element-by-element expansion of the sum of all the local cost functions and the

rearrangement of variables, the global function is found to be:

$$J(\alpha) = \alpha^T L \alpha, \tag{2.10}$$

where matrix L is the matting Laplacian matrix capturing pixel affinities within all possible 3x3 spatial windows. Besides its positive semi-definiteness, L is a sparse symmetric matrix with rows summing to 0. Each element in L represents the relation between two pixels in the image. This relation is given by the following equation:

$$(i,j)th\ element = \sum_{(i,j)\in w_k \& k\in image} \{\delta(i,j) - \frac{1}{n}[1 + (\bar{I}_i - \bar{\mu_k})^T(\Sigma_k + \frac{\epsilon}{n}\Sigma_{idn})^{-1}(\bar{I}_j - \bar{\mu_k})\}, \tag{2.11}$$

in which $\delta(i,j)$ is the Direct delta function, n is the number of pixels in the kth spatial window w_k, I_i and I_j are the color vectors of the ith and jth pixels, Σ_{idn} is a 3x3 identity matrix, μ_k is the 3x1 color mean vector in window w_k, and Σ_k is a 3x3 color covariance matrix of the same window. The last two are given as follows:

$$\bar{\mu_k} = \begin{bmatrix} \mu_k^r \\ \mu_k^g \\ \mu_k^b \end{bmatrix} = \begin{bmatrix} \frac{I_1^r + \ldots + I_9^r}{n} \\ \frac{I_1^g + \ldots + I_9^g}{n} \\ \frac{I_1^b + \ldots + I_9^b}{n} \end{bmatrix},$$

$$\Sigma_k = \frac{1}{n} X X^T \tag{2.12}$$

$$X = \begin{bmatrix} I_1^r - \mu_k^r & I_2^r - \mu_k^r & \ldots & I_9^r - \mu_k^r \\ I_1^g - \mu_k^g & I_2^g - \mu_k^g & \ldots & I_9^g - \mu_k^g \\ I_1^b - \mu_k^b & I_2^b - \mu_k^b & \ldots & I_9^b - \mu_k^b \end{bmatrix}.$$

One final remark on Equation (2.11): the sum is over all possible windows encompassing pixels i and j.

In order to find the alpha matte of an image using CFM, the global cost function, Equation 2.10, must be minimized subject to some constraints. The need for constraints stems from the fact that the null space of L is not empty– an important characteristic that has led to the development of the SM method. This means there are many potential minimizers of the cost function. However, almost all these solutions make no sense visually. For instance, the trivial solution of all one vector is a minimizer and belongs to the null space of L, yet it indicates that all the image pixels belong to the foreground. This solution is not only visually unacceptable, but it violates the very basic objective of the matting process, foreground and background separation. Therefore, some constraints reflecting a visual sense for the two layers are required to bias the solution toward the actual matte. This is usually accomplished by user-based input, in which the user provides information on some definite foreground and background pixels. Incorporating the user input, the final optimization problem is given by:

$$\min_{\alpha} \quad \alpha^T L \alpha \tag{2.13a}$$

$$\text{subject to} \quad (\alpha - b_s)^T D_s (\alpha - b_s) = 0, \tag{2.13b}$$

where:

- b_s is a vector containing the alpha values provided by the user.
- D_s is a diagonal matrix with the diagonal elements taking the value 1 for constrained pixels and 0 otherwise.

This quadratic function is minimized by differentiating it and setting the derivative

to zero. The result is a sparse linear system:

$$(L + \lambda D_s)\alpha = \lambda B_s. \tag{2.14}$$

When enough constraints are provided by the user, a very good alpha matte could be pulled. Usually these constraints are given as two sets of pixel-specified alpha values. For example, one set may define the alpha values of some pixels belonging completely to the foreground, and the other may define alpha for pixels belonging to the background. The location and number of these user-constrained pixels highly influence the quality of the extracted matte. Unfortunately, there is no specific rule for that. However, [18] provides some tips on where to place the constraints and how they could be inputted. One final note is that this form of user-provided input must not be confused with the classical trimap. They may serve a similar purpose, but they are different; the input of the CFM is very sparse and concerns the alpha values of some pixels while a trimap breaks down the whole image into three layers, a background, foreground, and unknown region

2.3 Spectral matting method

The ability of the matting Laplacian matrix to model the relation between pixels is its source of strength. This is clearly demonstrated with the quality of the matte the CFM produces. Minimal user-provided input, compared to other matting methods, is often enough to get that result. A quick look at the cost function in optimization Problem 2.13 reveals that the matte is the minimizer subject to a set of constraints. This obvious observation may give rise to the interesting question of whether the

matte will still be the minimizer without the constraints or not. In other words, is α in the null space of the matting Laplacian? Such question has been addressed by Levin et al in [19], leading to the development of the novel matting approach referred to as *Spectral Matting (SM)*. This method does not only provide a powerful method for matte extraction, but it also opens a new door for the automation of the matting process.

SM, in its core, expresses a strong similarity to a segmentation approach known as spectral segmentation. In fact, the development of this method was highly influenced by the likes of [23] and [32]. The basic idea of spectral segmentation is to construct a graph-Laplacian matrix that captures the distinct clustering of pixels in the image. A very similar idea is adopted by the SM method. However, the two approaches have their differences, the main of which is rooted in the matting Laplacian ability to capture fuzzy clustering of pixels instead of hard clustering. In order to understand how SM works and what gives it the edge over spectral segmentation, the concepts of *spectral analysis* and *matting components* must be briefly reviewed.

2.3.1 Spectral analysis

Spectral analysis represents the foundation of spectral segmentation. The goal is to decompose the image into several disjoint sets of pixels using a graph-Laplacian matrix (L_{graph}), which tries to capture the affinities among pixels. To see that, consider an image represented as a collection of several distinct sets of pixels C_1, C_2, ..., and C_k, see Figure 2.5. Each of these sets is assigned an indicator vector $\bar{m^{c_k}}$, the *ith* element

of which is given by:

$$m_i^{c_k} = \begin{cases} 1, & i \in C_k \\ 0, & otherwise \end{cases} . \tag{2.15}$$

In ideal cases, when the affinity matrix accurately captures the pixel clustering, this set of k indicator vectors $\{\bar{m^{c_1}},\bar{m^{c_2}},\ldots,\bar{m^{c_k}}\}$ makes an orthogonal basis to the null space of the graph-Laplacian matrix L_{graph} [19]. In fact, each vector of these is a zero-eigenvector of L_{graph} (i.e., the eigenvector with eigenvalue = 0). Utilizing such property, this set could be easly found by applying eigendecomposition to L_{graph} and rotating the zero-eigenvectors.

In practice, this decompose-and-rotate process does not produce the sought-after set of vectors. The main reason for that could be traced to the failure of the affinity matrix in recognizing the distinct clustering in an image. Normally, natural images have a complex structure that cannot be captured with affinity matrices like those proposed for spectral segmentation [32]. They require a higher-level of affinity modeling, which is something the matting Laplacian L has, to some extent, managed to do.

2.3.2 Matting components

The basic definition of the matting process states that matting aims to decompose an image into two layers and provide a compositing factor for each pixel in the image, see Section 1.2. This definition will be slightly altered to introduce the notion of matting components; any image I would be decomposed into several layers $\{F^1, F^2, \ldots, F^k\}$, instead of just two, and each pixel in that image would be a linear combination of the corresponding pixels of those layers scaled with several opacity factors $\{ \alpha_i^1, \alpha_i^2,\ldots,\alpha_i^k$

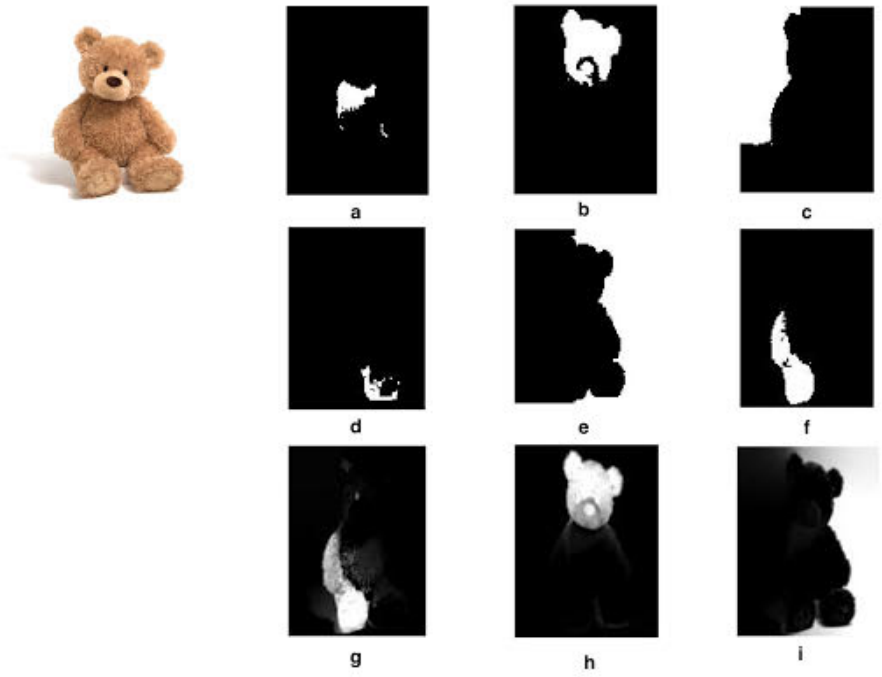

Figure 2.5: This is an example of hard segments and matting components. Images a-f are 6 hard segments obtained from the input image while images g-i are examples of matting components resulting from decomposing the input image with SM method.

}. For each pixel in I the following two equations hold:

$$\begin{aligned} I_i &= \sum_{j=1}^{k} \alpha_i^j F_i^j \ , \\ \sum_{j=1}^{k} \alpha_i^j &= 1 \ . \end{aligned} \tag{2.16}$$

The vector $\bar{\alpha^j}$ whose elements are the opacity factors of the *jth* layer is called a *matting component*. Any image has several matting components associated with its layers, see Figure 2.5.

At first, matting components and indictor vectors may seem to do the same job, which is not completely true. While they both decompose the image into several layers, these layers have a very different nature. The indicator vectors serve the purpose of partitioning an image into several hard segments. That is, each of these vectors is a binary vector. It basically indicates whether a pixel belongs to a region in the image or not. On the other hand, matting components produce a set of fuzzy layers. Any pixel of the image would be the composite of one or several layers. Another difference between the two is the orthogonality of the set. Matting components do not make an orthogonal set of vectors like the indicator vectors do since each element may assume a value in the interval [0,1]. Despite that, the matting components are still obtained from the null space of the matting Laplacian L.

2.3.3 Extraction of matting components

With the concept of matting components in mind, a natural question that comes to mind is how these components are extracted. In case of hard segments, the indicator vectors are in the null space of L_{graph}, and they form an orthogonal set of vectors. Therefore, they could be found with a rotation operation to the zero-eigenvectors of

L_{graph}. However, for matting components, nothing in the previous discussion hints to how they are extracted from L, so their relation to L needs first to be explored. Then, an answer to the question could be presented.

The matting components are found to be in the null space of L if they satisfy the following claim [19]:

- Claim 1: Let $\{\alpha^1,\ldots,\alpha^k\}$ be the k matting components of image I. These components lie in the null space of L if for every small spatial window w_k, one of the following conditions is satisfied:

    1. A single matting component α^k has an element $\alpha_i^k > 0$ within the window (In such case, the component is said to be active in the window w_k).
    2. Two components α^{k1} and α^{k2} are active within w_k, and the colors of the corresponding layers F^{k1} and F^{k2} within the same window form two different lines in the color space.
    3. Three components α^{k1}, α^{k2} and α^{k3} are active within w_k, and the colors of the corresponding layers F^{k1}, F^{k2} and F^{k3} within the same window are uniform and different, making the set of these three color vectors linearly independent.

The proof of this claim is given in [19]. Note that satisfying this claim yields a set of matting components that lies in the null space of L, but it does not necessarily mean that these components are orthogonal.

The process of extracting the matting components relies heavily on the aforementioned claim. If the components are proved to be in the null space of the matting Laplacian, the process of extracting them seems straightforward. One thing that

needs to be accounted for here is the orthogonality issue. The components could be extracted from the zero-eigenvectors of L with a general linear transformation instead of a rotation. In contrast, when the components of an image do not satisfy the claim, which is the situation with most of natural images, the null space of L may not exist or may not have multiple zero-eigenvectors. Despite that, the extraction operation would still be possible but under one condition, the image must have a relatively distinct clustering. That means pixels must form a large set of visually discernible regions. The less the regions are perceptually recognized, the less efficient the SM method performs.

Assuming the discernibility requirement is held for a natural image, the matting components could be recovered from the smallest eigenvectors of L. The SM problem could be posed as an optimization problem:

$$\min_{\alpha} \quad \sum_{i,k} |\alpha_i^k|^\gamma + |1-\alpha_i^k|^\gamma \tag{2.17a}$$

$$\text{subject to} \quad \bar{\alpha}^k = E\bar{y}^k, \tag{2.17b}$$

$$\sum_k \alpha_i^k = 1, \text{for all i} \in I, \tag{2.17c}$$

where:

- The cost function: $\sum_{i,k} |\alpha_i^k|^\gamma + |1-\alpha_i^k|^\gamma$ is a sparsity measure function with $\gamma \in$ [0,1]. This exponent controls the shape of the curve around the two main local minima of each term of the summation, see Figure 2.6.

- E is a NxK matrix containing the K smallest eigenvectors of L, $E = [e^1, \ldots, e^K]$ (N is the number of pixels in the image).

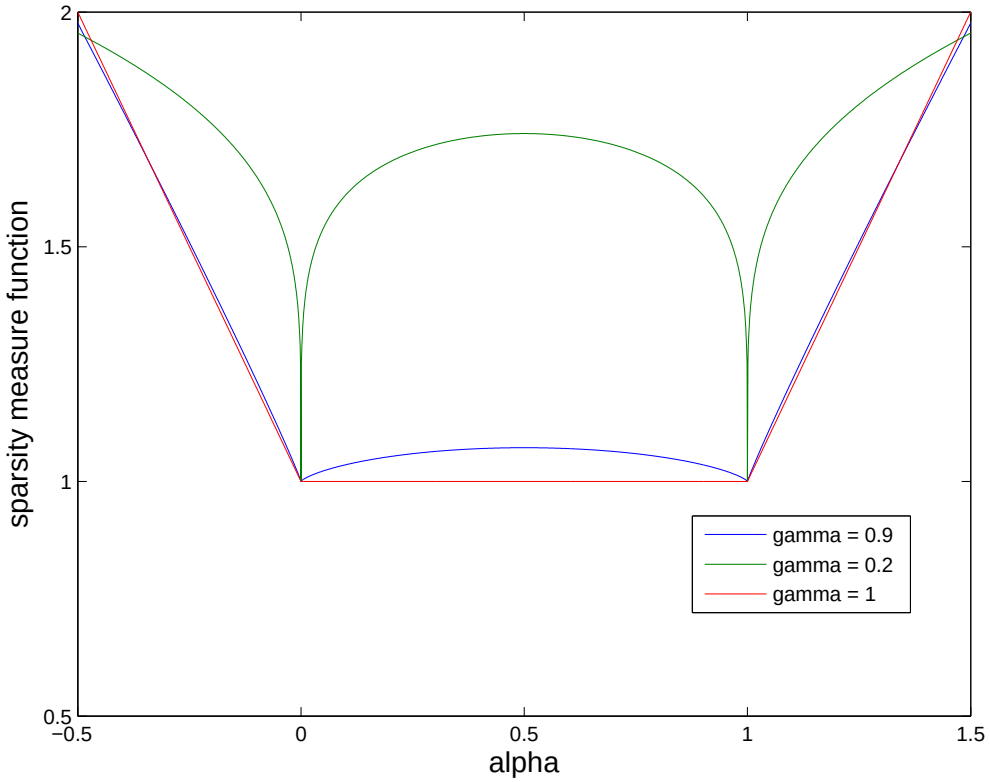

Figure 2.6: These are different plots of the sparsity measure function for several γ values.

Problem (2.17) is obviously non-convex, which makes it hard to solve. However, this difficulty could be alleviated with a model locally approximating the cost function using a quadratic function, see [19]. Given an initial set of values for all α^k, the problem becomes:

$$\min_{\alpha} \quad \sum_{i,k} u_i^k(\alpha_i^k)^2 + v_i^k(1-\alpha_i^k)^2 \tag{2.18a}$$

$$\text{subject to} \quad \bar{\alpha}^k = E\bar{y}^k, \tag{2.18b}$$

$$\sum_{k} \alpha_i^k = 1, \text{for all i} \in I, \tag{2.18c}$$

where: $u_i^k \propto |\alpha_i^k|^{\gamma-2}$ and $v \propto |1-\alpha_i^k|^{\gamma-2}$. This problem is convex with some linear constraints, and solving it iteratively starting from a very good set of initial vectors amounts to a very good final set of matting components.

An important issue that needs to be addressed before implementing the SM method is the initialization of the matting components. Apparently, the result of the optimization problem in (2.18) is highly influenced by the choice of the initial set of vectors. An elegant way of tackling this problem is also proposed in [19], which relies on the *k-means* clustering algorithm. Using the smallest eigenvectors of L a set of indicator vectors is obtained, in a very similar way to the one introduced in [32]. Then, this set is projected onto the range space of matrix E exploiting the fact that E is a tall matrix with orthonormal columns. Mathematically, this is given as:

$$\bar{\alpha}_{int}^k = EE^T m^{C_k}. \tag{2.19}$$

This projection results in a set of vectors each of which is a linear combination of the

columns of E. Note that, the initial matting components still sum to one along each row:

$$\sum_k \bar{\alpha}^k = EE^T(\sum_k m^{C_k}) = EE^T\bar{1} = \bar{1}. \tag{2.20}$$

One final remark on the extraction process: when the pixels of an image are distinctly clustered, the optimization problem in (2.17) could be dropped, and the SM process would be reduced to a two-step process: k-means hard clustering and projection onto the range space of E.

Chapter 3

Automation of closed-form and spectral matting methods

Chapter 2 introduced the theoretical grounds and mathematical models of two state-of-the-art matting methods, CFM and SM. Both of them are implemented interactively, i.e., the process of extracting the alpha matte is guided with user-provided constraints. In case of CFM, the user must specify the alpha values in some small regions within the image. As for SM, constraints are provided in several ways, the simplest of which would be the manual selection of the foreground layers by the user. With the interactive nature of the two methods, they fail to meet the basic requirement of visual content analysis, automatic OE. However, owing to the interesting perspectives each of them provides, several automation attempts have recently been made [15][16][30][34]. In all of them, the accuracy of the pulled matte is generally acceptable, but it is yet to be equivalent to the one pulled with the interactive methods. The automation problem gets even harder when the content being processed is a

video sequence; extraction of an accurate matte sequence is no longer the only problem; temporal consistency of the matte poses another major challenge. Overall, it is safe to say that automatic methods face the same obstacle, managing the trade-off between quality and computational complexity.

Recently, two different, but equally-intriguing techniques have been proposed to address the automation problem of CFM and SM for video sequences, [15] and [34]. What makes them stand out is their practicality compared to other techniques. Approaching the problem from two different directions, they manage to produce some relatively good results. One of them compensate for the absence of user input with some form of prior knowledge about the shape of the foreground object. The other utilizes the texture and color information of each frame to produce the matting sequence. They both have their advantages and shortcomings that could be traced back to the fundamental issue of complexity-accuracy trade-off.

For their importance, this chapter is devoted to introducing the two automation techniques and their implementation. The first technique automates the CFM with a shape prior, and it will be presented in Section 3.1. The other technique is designed to automate the SM method with adaptive component detection and matching, and it will be discussed in Section 3.2.

3.1 Automation of closed-form matting with shape prior

Since the CFM method completely relies on the user-provided information to create a bias toward the actual matte vector, the shape-prior automation technique [34]

compensate for such input with a set of learned basis vectors. This set approximates the so-called *Object-Shape Vector Space (OSVS)*. The basic idea here is to define a vector space that encloses all possible shapes taken by the foreground object– shape here refers to the silhouette of the object. A set of basis vectors are found for this space and supplied to the optimization problem of CFM, problem (2.13), instead of the user-inputted constraints. This technique was actually proposed for surveillance applications, in which the foreground object is a walking person. However, it could be extended to other applications and objects.

For surveillance applications, this technique requires a human detector, like the detector in [8], and a relatively large set of training data, called shape database (see Figure 3.1), and it is implemented in a two-stage fashion (see Figure 3.2). The process starts with the human detector providing a bounding box in which a pedestrian is located. This window is fed to the automated CFM method. The window matting process, the second stage of the whole automatic matting process, is performed using the shape basis matrix, which is obtained form the shape database. The output of this last stage is an alpha matte of the frame. One question could arise here: what happens if the frame contains more than one pedestrian? Although not directly addressed in [34], the use of a person detector in the first stage indicates that the second stage can only handle one person per window. This observation means that for situations with multiple pedestrians, breaking down the frame into several indexed windows is necessary. Processing these windows could be done either sequentially or in parallel depending on the hardware capabilities.

In mathematical terms, the automation process is accomplished with an optimization problem similar to (2.13) and a Principle Component Analysis (PCA) learning

Figure 3.1: A sample of the training database. Several mattes of different people are used to find a basis matrix for the shape space.

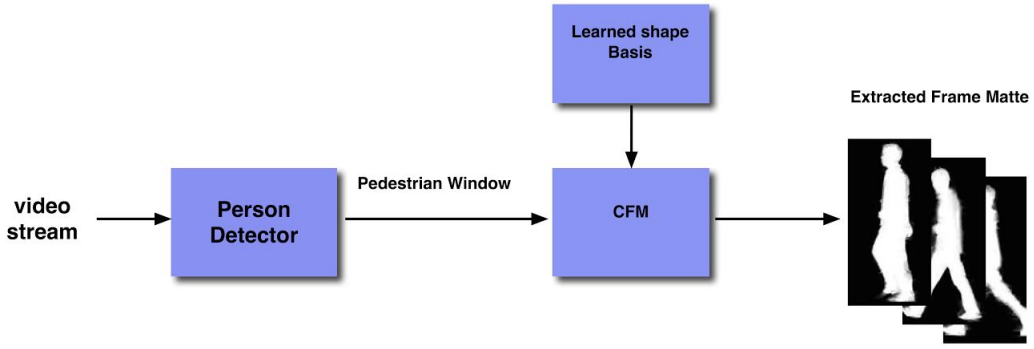

Figure 3.2: A block diagram describing the automation of the CFM process using shape-prior.

technique. Before starting the matting process, PCA is used to extract the shape basis matrix V from the database, in which every matte is converted into a large N-dimensional vector. This is done in a way similar to the one proposed in [29]. Using V, any shape could be modeled as:

$$\bar{S}(u) = V\bar{u} + \bar{\Delta}, \tag{3.1}$$

where $\bar{S}$ is the shape vector V is an NxM basis matrix (M represents the number of basis vectors), $\bar{\Delta}$ is the mean shape obtained from averaging all shapes in the database, and $\bar{u}$ is a vector of the basis coefficients.

With the shape model, the optimization problem in (2.13) becomes:

$$\min_{\alpha} \quad \alpha^T L \alpha \tag{3.2a}$$

$$\text{subject to} \quad (\alpha - (V\bar{u} + \Delta))^T(\alpha - (V\bar{u} + \Delta)) = 0. \tag{3.2b}$$

This problem is convex in both u and α, and strong duality could be also proven [14]. Hence, the solution is obtained by finding the derivative of the Lagrangian and setting it to zero. That way, the solution of (3.2) is found by solving the following sparse linear system:

$$\begin{bmatrix} (L + \lambda\Sigma_{idn}) & -\lambda V \\ -\lambda V^T & \lambda V^T V \end{bmatrix} \begin{bmatrix} \bar{\alpha} \\ \bar{u} \end{bmatrix} = \lambda \begin{bmatrix} \Delta \\ -V^T\Delta \end{bmatrix}. \tag{3.3}$$

The constant λ in (3.3) is the Lagrange multiplier, and it is usually set to around 0.01– more details on the choice of this constant will be given in Chapter 4 when the performance of this technique is analyzed.

3.2 Automatic spectral matting with adaptive component detection and matching

In spite of the ingeniousness of the original SM method, its performance without user intervention is very limited. This could be attributed to two fundamental facts: lack of information on the nature of the foreground object and multiplicity of non-trivial minimizers in the null space. The former is very obvious and could be understood from the mathematical derivation of the method, not having prior information on the foreground. For the latter reason, Levin *et al* [19] have shown that exhaustive search might be used to automatically find the right combination of components that minimizes the cost function in (2.13). However, this techniques is not reliable at all; it, for one, ignores the fact that the number of matting components in natural images is relatively high, which makes the exhaustive search unfeasible for implementation. In addition, the null space of L forms a hyperplane in R^N with infinite number of solutions. The most trivial of them is the constant vector, all ones or zeros.

Bearing in mind the aforementioned obstacles, Hu *et al* [15] have proposed a technique that mainly uses hue information to automate the video matting process. The system comprises multiple stages and requires no prior knowledge, see Figure 3.3. The process starts in a different way compared to the shape prior technique. There is no object detector here– a step that would be shown to be problematic in Chapter.4. Unlike the shape prior automation of the CFM method, SM needs automation on different levels. The very basic one of them is determining the number of matting components, which is exactly what the first stage, adaptive component detection, does using Mean Shift (MS) algorithm [5]. Since it works on color and location information

of pixels, it could find the clustering of these pixels in the so-called *feature space*. Usually, such clustering reflects the number of the perceptually significant regions in the image. Determining the number of components[1], K, paves the way for the extraction of matting components, and this is one of the two main tasks the second stage does. The other task is to pull a full alpha matte by combining the foreground matting components. This is done with the aid of the average hue of each component and a basic assumption on the background components; they must occupy most of the boundary of the image. This assumption basically means the pixels on the boundary of the image most probably belong to the background. Therefore, the component with the most boundary pixels is assumed to belong to the background. Classifying the components into three groups, foreground, background, and unknown region, prepares for the process of final matte extraction. It is accomplished with an exhaustive search for the minimizer of a cost function incorporating the matting Laplacian. The next stage in the process is component matching based on hue information. It matches the foreground components of the previous frame to the ones just found for the new frame. This step speeds up the classification process and imposes some form of consistency on the matte sequence.

The second stage, SM with hue information and exhaustive search, is the core process of the whole technique since it determines the alpha matte of the frame. It is conducted with the average hue information of each component– a component here refers to an indicator vector not a matting component. Such average is calculated after transforming the frame to the HSV color space. Using the hue angle between all components, two possible classification scenarios could take place:

[1]These components found by the mean shift algorithm are hard segments given in the form of indicator vectors. The should not be confused with the matting components extracted from the smallest eigenvectors of L.

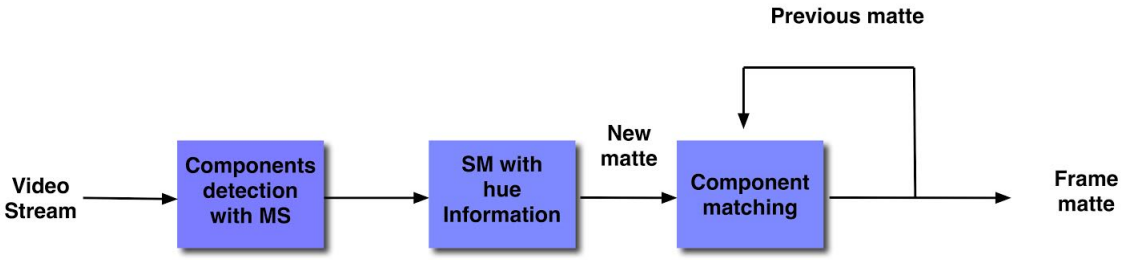

Figure 3.3: A block diagram illustrating the automation of the SM method using adaptive component detection and matching.

First scenario

If all angles are smaller than a threshold, $\pi/5$, then, a single foreground and a single background components are found with the following equations:

$$\begin{aligned} C_B &= \arg\max_{i \in k}(C(i) \cap I_{boundary}), \\ C_F &= \arg\max_{i \in k}(C_B^H - C^H(i)), \end{aligned} \tag{3.4}$$

where C_B is the background component, $C(i)$ is the *ith* component, $I_{boundary}$ is the one-pixel frame defining the boundary of the image, C_B^H is the hue angle of the background component C_B, $C^H(i)$ is the hue angle of the *ith* component, and finally, C_F is the foreground component. The threshold here is given based on empirical testing [15]. Now, matting components are extracted from the smallest eigenvectors of L, and with the aid of the one foreground and one background components, a foreground matting component is found and grouped with the unknown matting components. This leaves out some background matting components, which reduces the total number of matting components to $\tilde{K} \leq K$. With these $\tilde{K}$ matting components, exhaustive search could

be applied to find the minimizer of the cost function:

$$J(b) = y^T(\tilde{E}^T L \tilde{E})y, \tag{3.5}$$

where $\tilde{E}$ is the matrix containing the $\tilde{K}$ matting components and y is the optimization variable. It is a binary variable selecting the matting components that make up the final alpha matte.

Second scenario

The second scenario, obviously, occurs when the angles exceeds the threshold. In this case, multiple components belonging to either the foreground, the background, or the unknown region are expected. Thus, a check-then-merge process would be performed on adjacent components using the technique proposed in [30]. This process checks every two adjacent components for the possibility of merging. This is done based on the algorithm given below:

Algorithm 1
if $(C^s(i) \geq T_{C(i)})$ & $(C^s(j) \geq T_{C(j)})$
$C(i) \cup C(j)$
else if $\lvert C^v(i) - C^v(j)\rvert \leq Th_v$
$C(i) \cup C(j)$
else do nothing

where $C^S(i)$ and $C^V(i)$ are, respectively, the saturation and intensity of the *ith* component. The thresholds $T_{C(i)}$ and Th_v are determined empirically. Th_v is set to 0.1,

and $T_{C(i)}$ is calculated by [30]:

$$T_{C(i)} = \frac{1}{1 + \beta C^v(i)}, \beta \in [1, 4]. \tag{3.6}$$

Following the check-then-merge process, a foreground and a background components are identified using Equation (3.4). These foreground and background components are checked with the non-adjacent unknown components for any possibility of merging. The one foreground component that results from the previous process is grouped with the unknown components– in a similar way to the one used in the first scenario. To generate the final set of $\tilde{K}$ matting components, the smallest eigenvectors of L are used to extract the K matting components, and with the help of the grouped components, one foreground matting component is found and grouped with the unknown ones. Using this reduced set of matting components, the cost function, given in Equation (3.5), is minimized with exhaustive search to obtain the final alpha matte.

Chapter 4

Implementation and performance analysis

With the conceptual and theoretical discussions on CFM, SM, and their automation using shape-prior and component detection and matching techniques, one could have a good grasp on how the two matting methods operate and how they have been automated. However, a full understanding of the potential and capabilities of the two automation techniques developed for each matting method requires a deeper discussion of their implementation and a thorough analysis to their performances. Chapters 2 and 3 have shown some glimpses of their capabilities and provided a very general discussion on how they work. Therefore, this chapter is devoted to addressing the topics of implementation and performance analysis and providing a full assessment of the automation techniques.

The shape-prior automation and the component detection and matching techniques represent an important step toward developing a practical OE method for the purposes of visual content analysis. Among other automatic techniques, the outcome

these two provide is very good, which may hint to their adequacy for applications like intelligent surveillance. However, it is yet to say that they have managed to fulfill the requirements of automatic OE. Despite their good results, there are many implementation and performance issues that need to be addressed and overcome before adapting them for any real-world application. These issues fall in one of the two main categories of challenges that any OE technique or method faces: computational complexity and extraction accuracy.

This chapter discusses the computational complexity (implementation) and matting accuracy (performance) of the two automation techniques in three sections. Section 4.1 is dedicated to the first automation technique, shape-prior automation of CFM. It starts by describing its implementation and presenting some issues related to that. Then, it moves to analyzing the matting results of the technique and comparing it to the results of the interactive method. Section 4.2 follows a very similar order of discussion for the second automation technique, automation of SM with component detection and matching. Finally, for the sake of thoroughness, Section 4.3 provides a comparative analysis of the two techniques and a third technique that attempts to combine CFM and SM and automate them with shape-prior knowledge.

4.1 Implementation and performance analysis of shape-prior automation

In Section 3.1, shape-prior knowledge was introduced as a means of automation for the CFM method. This knowledge is derived from a large set of binary images of the object silhouette using PCA, and it is used in the optimization problem given

Figure 4.1: An sample of the database used in [34]. Each of the training images is a spatially-registered binary image.

in (2.13) instead of the user-provided constraints. Here, a detailed discussion on the implementation of this technique and its issues along with a performance analysis are presented.

Implementation

Before starting the automatic matting process, the proposed technique requires a set of shape basis vectors V. This is usually obtained from applying PCA on a shape database, which is done only once. PCA is used in a way similar to the one proposed in [29]. The training data are given in the form of a collection of binary images describing several poses and view-angles of random people, see Figure 4.1. In [34], the training images were spatially registered to centralize the different shapes. Despite how good this step may sound, it might become a source of troubles for real-time implementations. As it is going to be discussed soon, the pedestrian in the input window may assume different scales, poses, and locations, which may hint to the need to a more diverse shape database (more random scales, poses and locations, see Figure 3.1).

The automation technique [34] starts with a first stage that implements a person

detector to provide the matting stage with an input window in which a pedestrian is located. This is usually accomplished with a detection algorithm like HOG [8]. The found window is fed to the matting stage in which a learned shape basis matrix is used to find the alpha matte. The dependence of the second stage on person detectors makes it susceptible to any problem the detector may incur [34]. Some common issues with using person detectors are the location of the pedestrian in the window and its scale; the detector usually can not provide this kind of information. Consequently, performing the matting process using the basis matrix, learned from a spatially registered shape database, requires the spatial alignment of the shape basis and the person within the window.

This person detector problem is tackled in [34] with a spatial-transformation optimization substage. This substage is combined with the original optimization substage given in (3.2) to make up the matting stage in Figure (3.2). The process starts with the assumption that the person is in the center of the window, so the original optimization substage is performed resulting in an initial matte α_{int} and a basis coefficient vector u_{int}. This matte is fed to the spatial optimization substage, given as:

$$\arg\min_{\delta t} \quad \|\alpha_{int} - (V(w(x;t+\delta t))u_{int} + \Delta((w(x;t+\delta t)))\|_2^2. \tag{4.1}$$

Here, $w(.;.)$ is the mapping of a pixel in location x in the window into the location given by $w(x;t+\delta t)$ in the shape basis in which t is the transformation parameter and δt is its increment found by minimizing (4.1). Once the new increment δt is found, the shape basis is spatially transformed accordingly, and, then, the first optimization substage is repeated to generate new updates of the matte α and the basis coefficient vector u. The process continues alternating between the two substages until the

	Figure 4.2	Figure 4.3
Number of training samples	215 spatially aligned binary shape images	146 diverse binary shape images
Number of principle components used	182	146
Size of all sample images	80 x 40	160 x 80
λ	0.01	several values

Table 4.1: Parameters

optimal solution is found or the maximum number of iterations is reached.

Regardless of the improved quality of the outcome (see Figure 4.4) the spatial optimization increases the overall computational cost. The cost function in the optimization problem (4.1) may look quadratic at first, but this is not true due to its relation to the optimization parameter δt. This forms the main source of complexity here. To tackle that, [34] proposed minimizing it locally using the first order Taylor expansion around the initial value of t. This step is incorporated with the assumption that spatial transformation is restricted to the simple translation operation to reduce complexity. Such assumption seems reasonable for surveillance applications, as stated in [34], and it alleviates the computational burden; however, it ignores the fact that person detectors cannot guarantee a fixed scale of the person within the window. Therefore, scale must be included in the spatial alignment optimization, which amounts to a slight increase in complexity.

Performance

Figure 4.2 shows the results of the two-substage matting process of a short sequence (these results are given in [34]). The input windows are cropped manually and fed to the matting process (see Table 4.1 for details on implementation parameters). Each of the mattes in Figure 4.2 either contains some pixels from the background with alpha values greater than zero or misses some pixels from the foreground. This kind of error does not propagate to the subsequent windows. Thus, the matte sequence becomes inconsistent. Generally, this problem could be attributed to the simplified spatial optimization and imperfect shape basis matrix. As stated earlier, the transformation process in [34] is restricted to the translation operation only, for people usually appear in upright positions in a surveillance video sequence. Such assumption helps reduce the complexity of the spatial optimization substage, yet it ignores the fact that person detectors cannot control the scale of the person within the window. For that, many matting errors occur. Moreover, the imperfect shape basis matrix used in the first optimization substage takes a toll on the final outcome [34]. Although the idea of deriving a shape basis matrix from a set of binary shapes seems elegant, the diversity of shape poses and scales makes it practically very hard to do so with limited set of samples. Even if these sample shapes are random and not registered, finding the perfect shape basis is still challenging and requires a very large database. An example showing that is presented in Figure 4.3-(b). This is a matte pulled using a diverse database, consisting of 146 training samples, and without spatial optimization. It is obvious how the quality significantly decreases without accurate spatial optimization and with a small database.

Beside the need for a large and diverse database and for spatial optimization,

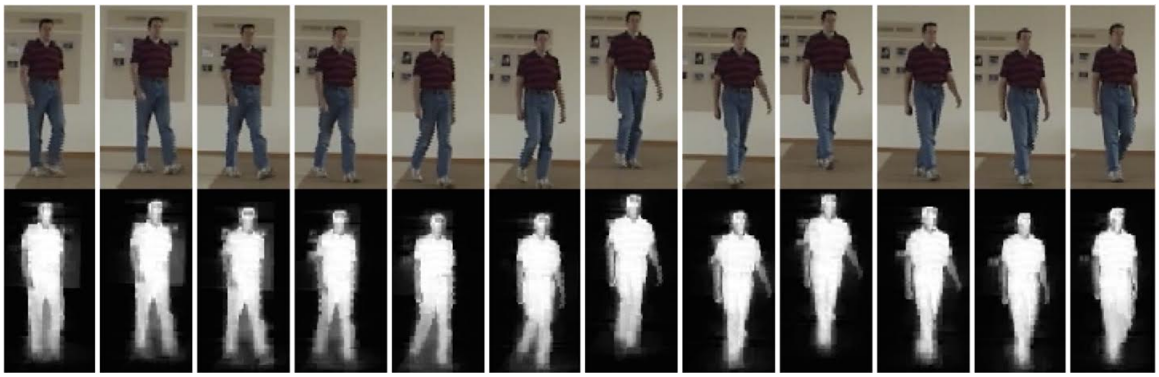

Figure 4.2: This is one of the examples given in [34]. The first row shows an input sequence of windows each of which is a 80x40 pixel of size (very low quality). The second row is the alpha mattes obtained with two substage optimization process.

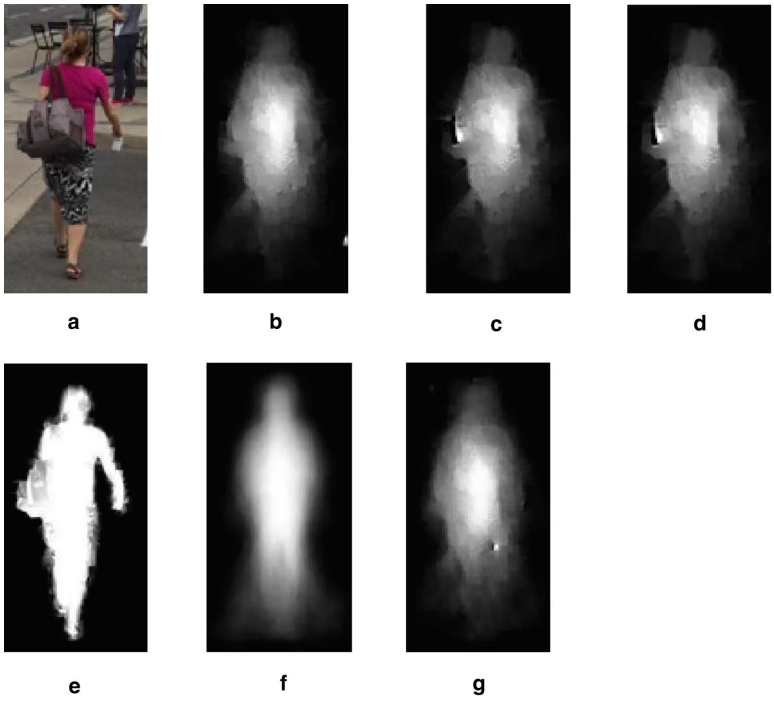

Figure 4.3: (a) is the input window. (b)-(d) are the extracted mattes without spatial optimization and with different values of λ, 0.25 ,0.1, and 0.09, respectively. (e) is the matte extracted interactively with CFM. (f) is the mean-shape of the database. (g) is the matte extracted with $\lambda = 1$.

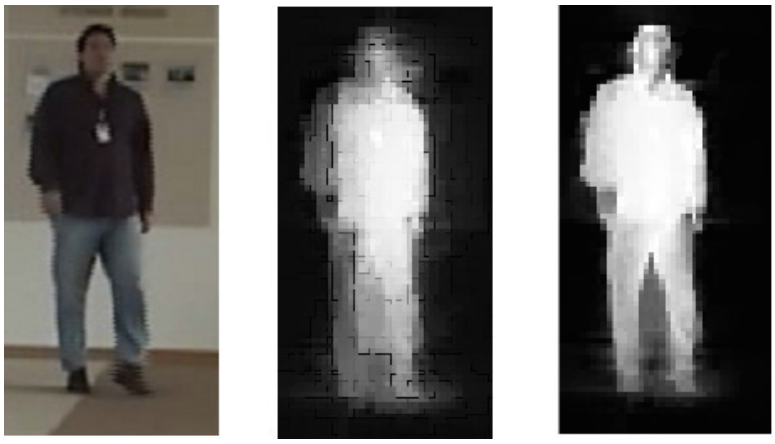

Figure 4.4: This example is given in [34] to illustrate the role of spatial alignment. The left image is the input, the middle one is the extracted matte without spatial alignment, and the last on eis with spatial alignment

this automation technique has a parameter that highly influences the quality of the matte. This parameter is the Lagrange multiplier λ, see Equation (3.3). Figures 4.3-(b)-(d) and (g) show the mattes extracted with different λ values. The more this value increases, the closer the final matte gets to the shape of the mean of the training database, Figure 4.3-(f). When restricted to values between 0.001 to 0.5, the shape gets more recognizable. Owing to the poor quality of the matte, the change with respect to λ is very limited in Figure 4.3. However, although it is not mentioned in [34], investigating the role of λ mathematically would help explain this behaviour. Form Equation (3.3), the following two equations are obtained:

$$(L + \lambda\Sigma_{idn})\alpha - \lambda V u = \lambda\Delta, \tag{4.2}$$

$$-\lambda V^T\alpha + \lambda V^T V u = -\lambda V^T\Delta. \tag{4.3}$$

Theoretically, matrix V contains the largest M (where M is always $\ll$ the number

of pixels N) eigenvectors of the covariance matrix of the database [29]. Hence, it is an orthonormal tall matrix with size NxM, and V^TV and VV^T result in identity and projector matrices, respectively. This makes the second term in Equation(4.3) equal to λu. In addition, the elements of L, typically, fall in the interval [-1,1], so as λ gets larger, $\lambda\Sigma_{idn}$ in the first term in Equation (4.2) dominates, leading to: $(L + \lambda\Sigma_{idn}) \cong \lambda\Sigma_{idn}$ for very large λ. With these two observations and following some algebraic manipulations, α is found to be given with this equation:

$$(\Sigma_{idn} - VV^T)\alpha = (\Sigma_{idn} - VV^T)\Delta, \tag{4.4}$$

where $(\Sigma_{idn} - VV^T)$ is a projector matrix onto a subspace perpendicular to the span of V. Equation (4.4) suggests that large values of λ drive the solution of α to be equal to Δ, i.e., α becomes the average shape of the database. This interpretation makes sense because it is aligned with the fact that the constraint in Problem (3.2) does not hold tightly, as a result of the imperfection of the shape basis matrix V.

4.2 Implementation and performance analysis of component detection and matching

Adaptive component detection and matching [15] attempts automating the SM method by decomposing it into three stages (see Figure 3.3), each of which handles one automation issue. These issues are: determining the number of components, finding the ones belonging to the foreground, and matching the extracted mattes throughout the sequence. The performance of this technique is generally good. However,

there are some shortcomings that need to be addressed and solved. In the following, implementation-related issues and performance analysis are presented:

Implementation

The general flow of this automatic matting process is shown in Figure 4.5. The first stage answers one of the fundamental questions in SM, how many matting components are there? The proposed way to determine the number is the Mean Shift (MS) algorithm [5],[7]. It decomposes the frame into several homogeneous regions based on the dominant color and the location of pixels, see Figure 4.6 for an example of that. Although that sounds exactly like what the SM does, what gives MS the edge is its ability to adaptively determine the number of regions. Therefore, using MS results in a good estimation of the number of matting components. One issue that usually accompanies the use of MS is the need to determine its bandwidth parameter. This is a well-addressed issue in the literature [6], yet determining the bandwidth adaptively from the input data increases the computational load.

The second stage of the process is actually the hard core of the technique. It classifies the detected regions into three groups: foreground, background, and unknown regions. To achieve that, the color information and the frame edge pixels are used. Operating in the HSV color space, the average hue angle of each region is calculated, as Figure 4.5 shows. Together with the assumption that the region with the most pixels from the edge of the frame makes a definite background component, two different components are identified, one for the foreground and the other for the background. Typically, the number of components in the unknown group is small, which is a result of the merging process or the fact that the frame contains a few

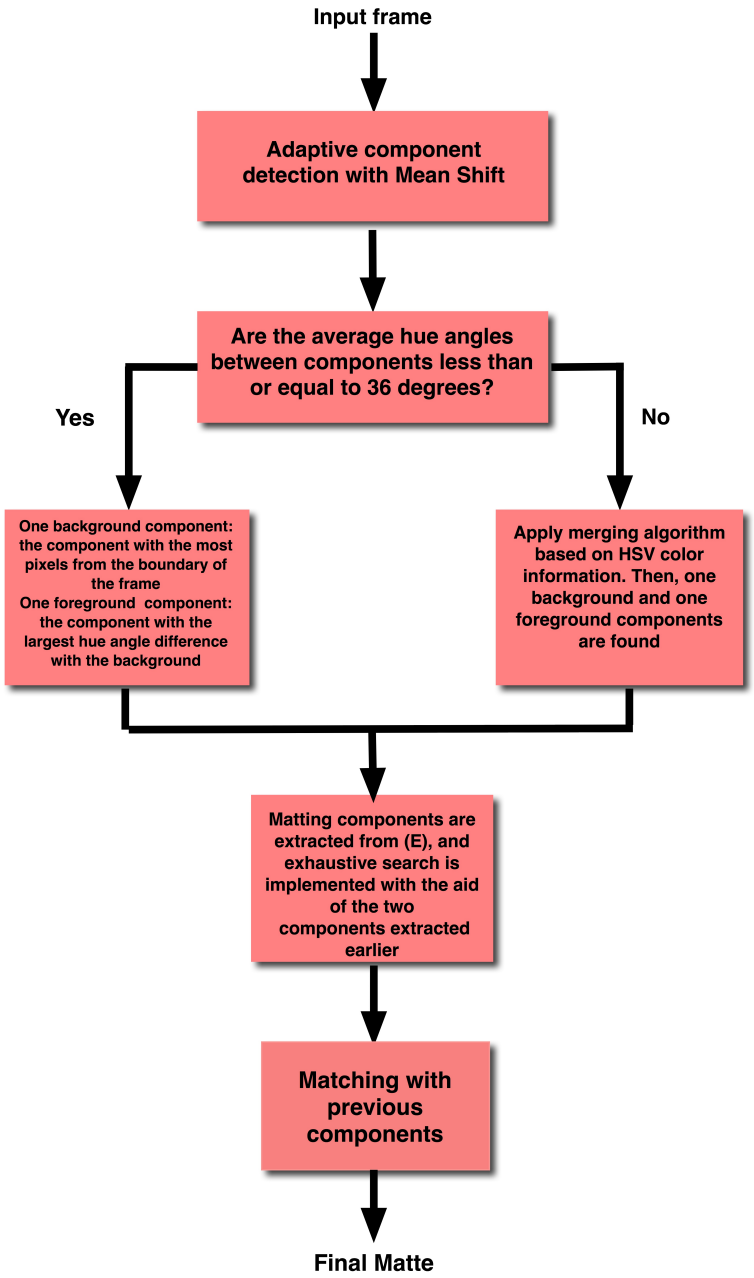

Figure 4.5: A block diagram showing the major steps of the automatic matting process.

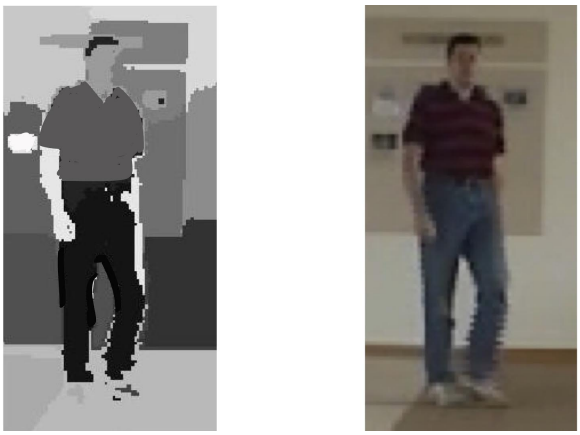

Figure 4.6: One of the windows in Figure 4.2 is decomposed into several regions.

distinct regions. As a result of that, exhaustive search is implemented in the third stage to recover the final matte, the minimizer of the cost function in Equation (3.5). For natural images, especially in surveillance applications, there are always several regions with average hue differences exceeding $\pi/5$, refer to Section 3.2, which makes the merging process the sole reason of reducing the number of distinct regions. A major drawback associated with that is misclassification since merging is performed based on low-level cues like hue, saturation, and intensity. Some regions belonging to the foreground may get merged with others from the background and vice versa, and this forms one of the main sources of errors in this technique.

Following the extraction of the matte, the third stage enforces consistency on the sequence by matching the extracted foreground matting components of the current frame with the previous one using the color information. This step helps produce some form of consistency in the sequence. However, high consistency is barely achieved in this stage. This is because it requires matching on the first stage, number of components, as well as the second stage, region classification, and not only matching the final matting components. That means the first stage of the process must produce the

same regions in every frame and the merged regions must also be the same throughout the sequence. Such a requirement is almost impossible to satisfy due to the variability of brightness in the video sequence and the small changes in the object's poses.

Performance

This automatic matting technique was designed to be a generic video matting technique. Its performance, as shown in [15], is very impressive. However, since the focus here is on surveillance applications, it is implemented on a short sequence of the windows given in Figure 4.2 to test its performance. The results are displayed in Figure 4.7. The second row corresponds to the mattes extracted automatically. A striking observation could be made here, the mattes do not exactly capture the foreground. That is the case for two reasons. The first is related to the aforementioned errors produced by the merging process. Some regions in the window are merged with the background because of average hue proximity, and few others are merged with the foreground for the same reason. The second factor contributing to this mistake is error propagation. Motivated by brightness and pose changes, the decomposition of each frame is different, and sometimes this helps compensate for errors made previously. Nevertheless, with the matching process operating based on low-level cues, some of these corrections may get eliminated, resulting in some sort of error consistency.

At this point, the following question might be deemed important: what happens if these issues are to be avoided? Naturally a substantial improvement in the quality of the matte would be expected. This is not completely true. In [15], the technique was implemented on high-quality videos (high resolution), which is not a common feature for surveillance sequences. To see the effect of resolution reduction on the

performance, one could look back at Figure 4.7 and notice the third row of mattes. These mattes are produced with the interactive SM method. It may seem surprising at first to see how inaccurate these mattes are. Despite the slight improvement, there are still some obvious errors. The reason behind that is traced back to an issue in the SM method itself. In its theoretical grounds, SM requires the conditions of Claim 1 in Section 2.3.3 to be satisfied for perfect decomposition (multiplicity of zero-eigenvalues). If not, the decomposition could still be performed, but its quality is subject to the distinctiveness of the regions composing the image. In a few words, it relies on the quality of the image. The windows in Figure 4.7 are of very low quality, 80 x 40 resolution. Therefore colors and regions are not easily distinguished. Once the quality of the image improves and the two previously discussed issues are overcome, this technique would be able to produce a better matte than the interactive SM method does, and that is exactly what Figure 4.8 shows.

One final remark about this technique concerns its ability to extract foreground objects in crowded scenes. Since the focus here is on surveillance applications, a serious issue arises when there are many pedestrians in the scene; unlike the shape-prior automation technique, the design of this one does not incorporate a person detectors to single out pedestrians. This compromises its ability in real-life applications. Fortunately, there are many ways to overcome this. Several detection and tracking algorithms are developed based on MS, [2] and [36] to name a few. Thus, because the first stage utilizes MS algorithm for component detection, the process could be further improved to do the detection operation.

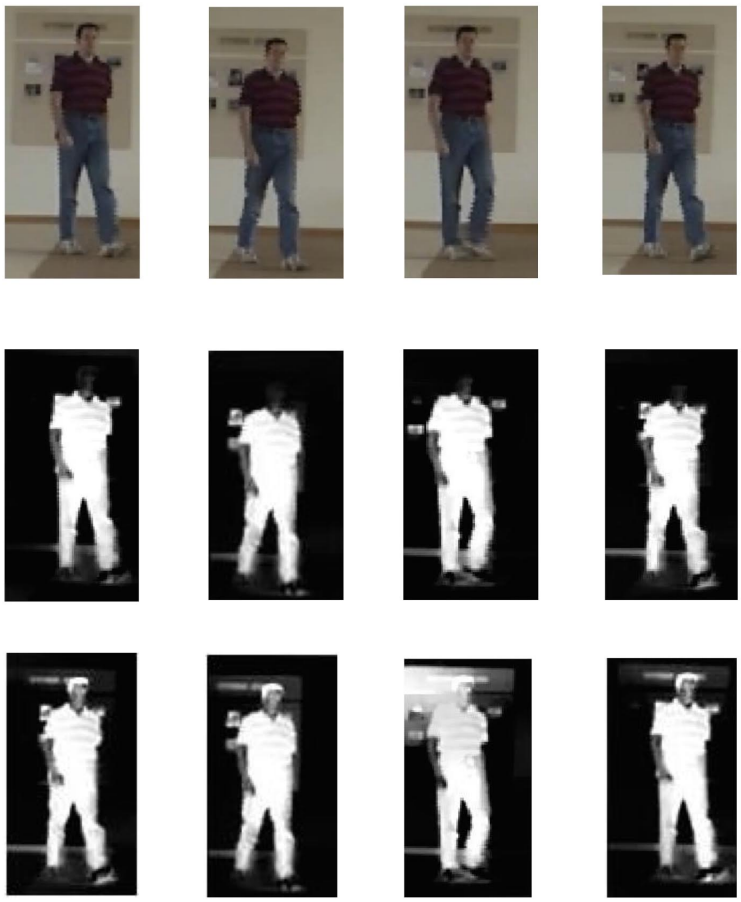

Figure 4.7: a sample of the sequence in Figure 4.2. The first row shows the input windows. The second row shows the extracted matte with the automatic technique. Finally, the third row shows the mattes extracted manually with the SM method.

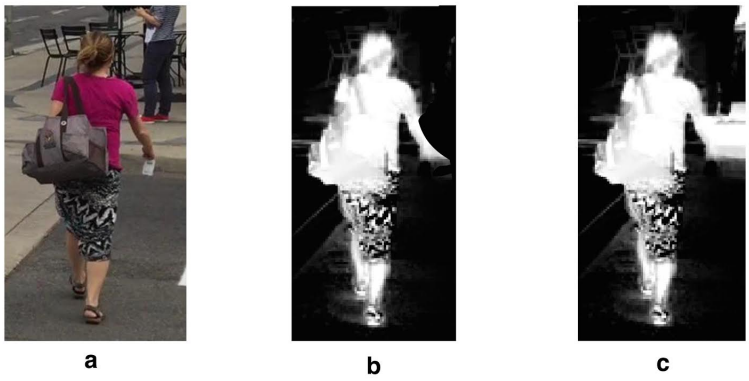

Figure 4.8: This is an example of the performance of the automatic matting technique (the adaptive component detection and matching) when the quality of the image is good. (a) is the input image. (b) is the matte extracted with the automatic technique. (c) is the matte pulled with the interactive SM method.

4.3 Comparative analysis

Both techniques have their strengths and shortcomings, and a comparative analysis of their overall performance in surveillance applications is very helpful to get some perspective on how to improve them. Computational complexity is the first and most important issue owing to the target application. Surveillance sequences require a real-time OE process, which is a challenge for the second technique. Its greatest computational issue is rooted in its core process, SM. It requires eigendecomposition to obtain the smallest eigenvectors, which is a costly process considering the size of the matting Laplacian L. For instance, the input image in Figure 4.8 is of size 160 pixels x 80 pixels making the size of L 12800x12800. Eigendecomposition for such matrix using MATLAB© running on a Macintosh computer with a 2.9GHz CPU and 8GB memory takes around 13 minutes. Unfortunately, reducing the size of the input image would not be of any help because the quality of the matte is directly proportional to it. The more details the image contains, the more discernible its structure is.

The other issue the two techniques need to address is the quality of the pulled matte. Shape-prior automation performs better than component detection and matching with respect to low quality input sequences, the common case for surveillance applications. In addition, the prior knowledge about the shape of the foreground object makes it more reliable compared to the other technique, in which low-level cues, like color information, are used. As the quality of the input frame increases, more details form the scene are being captured. Hence, the shape basis matrix starts to fail in reconstructing these details since the variability in poses, scales, and locations is very large. In this case, component detection and matching might seem more capable of handling these additional details. However, its major problem stems from its dependency on low-level cues and the component matching process. In Figure 4.8, the extracted matte is very good. In fact, it is better than the one extracted with the interactive SM method. Such good result is not guaranteed with all input images, for the merging process in the second stage is highly prone to errors.

Chapter 5

Hybrid Automation Technique

Chapter 4 have revealed some interesting insights on the performance and complexity of the shape-prior automation and the adaptive component detection and matching techniques. It has shown that the former technique is more practical for surveillance applications since it has lower computational cost than the latter technique. However, its reliance on a shape basis matrix, which is derived from some training shape database, makes the quality of the final matte significantly lower than the one obtained with the interactive CFM method [34]. In this small chapter, a simple way to improve the quality of the extracted matte using shape-prior automation is proposed and analyzed. It does not completely resolve the quality issue, but it serves as an illustration of what could be done to improve the automatic matting process.

The proposed technique improves the extracted matte by leveraging the quality advantage of the SM method. The matting components extracted from the matting Laplacian L are incorporated in the optimization process of the shape-prior automation. Thus, this technique has been called the "hybrid automation technique." The constraint in problem (3.2) verifies the validity of the extracted matte α using the

shape model $Vu + \Delta$. In this hybrid technique, the shape model is still used as a validation tool, but the constraint and the cost function of (3.2) are both modified to restrict the choice of α to the matting components $\{\alpha^1, \ldots, \alpha^k\}$. In mathematical terms, this is given by the following optimization problem:

$$\min_{b} \quad y^T(E^T L E)y \tag{5.1a}$$

$$\text{subject to} \quad (Ey - (Vu + \Delta))^T(Ey - (Vu + \Delta)) = 0, \tag{5.1b}$$

$$y_i \in \{0, 1\} \quad \text{for all i} = 1, 2, \ldots, K, \tag{5.1c}$$

where y is a binary vector selecting the matting components that make up the foreground, and E is the matrix of all K matting components, K is determined using the mean shift algorithm. The squared error between the selected components Ey and its reconstruction using shape basis matrix $Vu+\Delta$ is a good measure of how compatible this matte with the shape of the foreground object. This simple hybridization of SM and shape prior knowledge helps improve the final matte quality and rule out the need for spatial alignment. Examples of its performance are given in Figures 5.1 and 5.2. Overall, the resulting mattes are fairly good when compared to the other two techniques.

Aside from the promising outcomes, there are two major implementation obstacles this technique needs to address in order to be of any practical use. First, y is a binary variable that makes the optimization problem non-convex and hard to solve. Such issue requires a form of constraint relaxation or a heuristic algorithm to be overcome. For example, the results in Figures 5.1 and 5.2 are obtained after relaxing the second constraint and assuming that $y \in R^K$. Although the pulled mattes have fair qualities

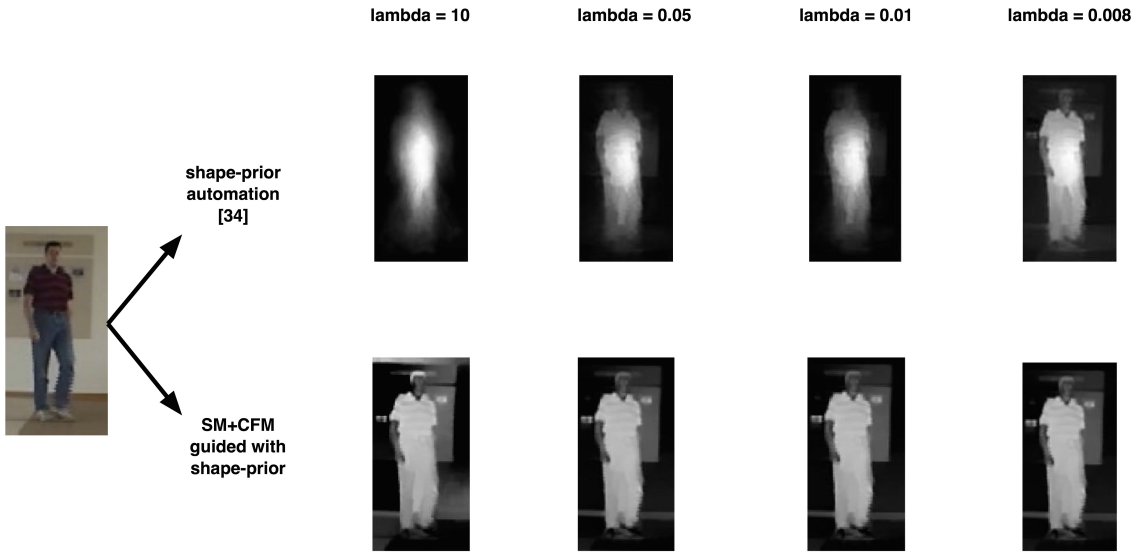

Figure 5.1: A window of the video sequence in Figure 4.2 is fed to the shape-prior automation technique without spatial alignment and the hybrid shape-prior guided CFM-plus-SM technique. The results are given for different values of λ. Note the slight independency of the second technique from the values of λ

compared to the mattes in the first row, they fall to exclude some background matting components. All the mattes in the second row in Figures 5.2 and 5.1 assign non-zero values to the opacity factors of some background pixels. This observation places clear emphasis on the need for a more sophisticated algorithm to solve the non-convexity problem. The second obstacle for this technique is the high computational cost accompanying the eigendecomposition process of L. Being the backbone of the SM method, this process is inevitable, and short of a customized hardware, the process could not be performed in real-time.

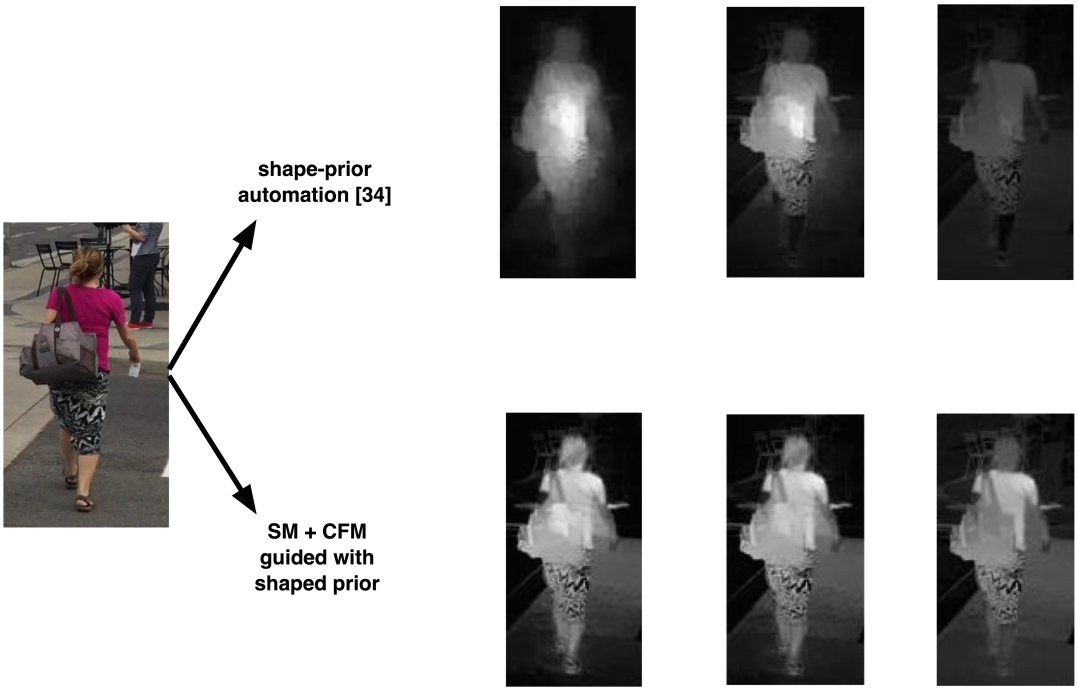

Figure 5.2: A higher resolution frame is fed to shape-prior automation and shape-guided CFM-plus-SM. The results are obtained for different values of λ, starting from the left: 0.1, 0.01, and 0.001, respectively. The quality of the matte is significantly improved with the latter technique

Chapter 6

Conclusion and future work

Conclusion

Closed-Form Matting (CFM) and Spectral Matting (SM) have recently revealed some interesting perspectives on the image foreground and background separation problem. Posing the matting process as either a constrained quadratic optimization problem or a component-wise grouping problem, they identified matting as a real contender for the object extraction task– a task traditionally accomplished with segmentation. Because they were proposed as interactive matting methods, much of the current research is devoted to develop automation techniques. Exploiting their interesting features, shape-prior automation and adaptive component detection and matching have managed to produce some good results compared to the existing automatic matting or segmentation techniques. However, getting to the point of practical implementation still requires some effort. There are some performance and implementation shortcomings that need to be addressed before taking that step.

Shape-prior automation technique uses a form of prior knowledge to tackle the

automation problem of the CFM method. Such knowledge is learned from a predefine database of the silhouette of the foreground objects. This technique has been developed for surveillance applications, and it has achieved a fairly good performance. It utilizes the famous Principle Component Analysis (PCA) method to learn a basis matrix for the shape vector space. Then, it finds a minimizer for the constrained CFM problem that could be constructed with the basis matrix. Despite the promising results, the technique still has some shortcomings, most of which are related to the second stage. Since the technique relies on person detection, the second stage must incorporate a spatial optimization substage. It tries to align the shape bases with the actual location of the person in the window. This increases the computational load, and when a relaxation technique is adopted, errors become inevitable. The quality of the pulled matte is also subject to the imperfection of the basis matrix; learning a set of global bases of the so-called shape space is very hard, if not impossible, due to the extreme variability of object shapes, poses, and scales.

The second automation technique, adaptive component detection and matching, attempts guiding the SM process using some low-level cues, like color information and pixel locations. Depending on the discernibility principle, it implements the non-parametric mean shift algorithm to decompose the input frame into distinct regions. Then, it uses the average color information of each region to find a definite foreground and a definite background regions. These regions help classify the matting components into three groups: foreground, background, and unknown. The final matte is a combination of the foreground components and some of the other unknown ones. Such combination is extracted by minimizing a quadratic cost function using

exhaustive search algorithm. To guarantee matte consistency, a component matching method is implemented at the end. it matches the foreground components of the previous frame with the currently extracted ones. In reality, this automation technique produces some good results. In fact, in some situations it outperforms the interactive SM method. Like the previous technique, it has its own technical issues. Not incorporating a person detection stage causes a dramatic failure when it is used for surveillance applications; this technique assumes the input frame has one specific foreground object, a situation rarely happening for this kind of applications. In addition, its reliance on low-level cues makes it susceptible to extraction errors, which, in turn, makes it unreliable. Color information and pixels locations are not enough to describe objects in natural scenes, let alone using them to perform the classification process.

Incorporating sophisticated machine learning methods is the best approach to take in order to improve these techniques or develop new ones. A closer look to their major drawbacks indicates that they are the result of the absence of prior knowledge or the use of a very primitive learning technique. Not to suggest that advance learning techniques hold the magical solution, but following the analysis of performance and implementation in Chapter 4, advance learning is strongly needed to combat the shortcomings and overcome the challenges. After all, the OE process has to be done by a machine!

Future work

Based on the discussion and analysis presented in Chapters 3 and 4, future work will focus on utilizing advance learning techniques to automate the CFM and SM

methods. A glimpse of that effort is revealed at the end of Chapter 4. In the form of hybrid automatic technique, the CFM optimization problem is constrained with the primitive shape-prior knowledge and the set of matting components in an effort to guide the matting process. The preliminary results are, to some extent, promising, so more development is needed to refine and enhance the performance. Another possible method of automation will take advantage of the extensive literature on image descriptors. They could be utilized to identify regions belonging to the foreground or the background, and accordingly the matting process is performed.

Bibliography

[1] N. Apostoloff and A. Fitzgibbon, "Bayesian video matting using learnt image priors," in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, vol. 1, June 2004, pp. I–407–I–414.

[2] C. Beleznai, B. Fruhstuck, and H. Bischof, "Human detection in groups using a fast mean shift procedure," in *International Conference on Image Processing (ICIP)*, vol. 1, Oct 2004, pp. 349–352.

[3] S. Chen, J. Zhang, Y. Li, and J. Zhang, "A hierarchical model incorporating segmented regions and pixel descriptors for video background subtraction," *IEEE Transactions on Industrial Informatics*, vol. 8, no. 1, pp. 118–127, Feb 2012.

[4] Y.-Y. Chuang, B. Curless, D. H. Salesin, and R. Szeliski, "A bayesian approach to digital matting," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, 2001, pp. II–264–II–271.

[5] D. Comaniciu and P. Meer, "Mean shift: a robust approach toward feature space analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*,

vol. 24, no. 5, pp. 603–619, May 2002.

[6] D. Comaniciu, V. Ramesh, and P. Meer, "The variable bandwidth mean shift and data-driven scale selection," in *Proceedings of the eighth IEEE International Conference on Computer Vision, ICCV*, vol. 1, 2001, pp. 438–445.

[7] D. Comaniciu and P. Meer, "Robust analysis of feature spaces: color image segmentation," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition.*, Jun 1997, pp. 750–755.

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

[9] N. Dalal, B. Triggs, and C. Schmid, "Human detection using oriented histograms of flow and appearance," in *Computer Vision ECCV*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2006, vol. 3952, pp. 428–441.

[10] D. Forsthoefel, D. S. Wills, and L. M. Wills, "Unsupervised video leap segmentation for fast detection of salient segment transformations in mobile sequences," in *15th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, Sept 2012, pp. 728–733.

[11] B. Fulkerson, A. Vedaldi, and S. Soatto, "Class segmentation and object localization with superpixel neighborhoods," in *IEEE 12th International Conference on Computer Vision*, Sept 2009, pp. 670–677.

[12] R. C. Gonzalez and R. E. Woods, *Digital image processing*, 3rd ed. Prentice-Hall, Inc., 2002.

[13] Y. Guan, W. Chen, X. Liang, Z. Ding, and Q. Peng, "Easy matting - a stroke based approach for continuous image matting," *Computer Graphics Forum*, vol. 25, no. 3, pp. 567–576, 2006.

[14] H. Hmam, "Quadratic optimization with one quadratic equality constraint," DSTO Document, Tech. Rep., 2010.

[15] W. C. Hu and J. F. Hsu, "Automatic spectral video matting," *Pattern Recognition*, vol. 46, no. 4, pp. 1183 – 1194, 2013.

[16] W.-C. Hu, J.-J. Jhu, and C.-P. Lin, "Unsupervised and reliable image matting based on modified spectral matting," *Journal of Visual Communication and Image Representation*, vol. 23, no. 4, pp. 665 – 676, 2012.

[17] C. H. Lampert, M. Blaschko, and T. Hofmann, "Beyond sliding windows: Object localization by efficient subwindow search," in *IEEE Conference on Computer Vision and Pattern Recognition, 2008. CVPR 2008*, June 2008, pp. 1–8.

[18] A. Levin, D. Lischinski, and Y. Weiss, "A closed-form solution to natural image matting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 2, pp. 228–242, Feb 2008.

[19] A. Levin, A. Rav Acha, and D. Lischinski, "Spectral matting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 10, pp. 1699–1712, Oct 2008.

[20] Z. Lin and L. Davis, "Shape-based human detection and segmentation via hierarchical part-template matching," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 4, pp. 604–618, April 2010.

[21] T. Malisiewicz and A. Efros, "Improving spatial support for objects via multiple segmentation," in *British Machine Vision Conference (BMVC)*, vol. 37, 2007, pp. 1782–1786.

[22] R. Megret and D. DeMenthon, "A survey of spatio-temporal grouping techniques," DTIC Document, Tech. Rep., 2002.

[23] A. Y. Ng, M. I. Jordan, Y. Weiss *et al.*, "On spectral clustering: Analysis and an algorithm," *Advances in neural information processing systems*, vol. 2, pp. 849–856, 2002.

[24] F. P. O. Tuzel and P. Meer, "Region covariance: A fast descriptor for detection and classification," *Computer Vision-ECCV 2006*, vol. 3952, pp. 589–600, 2006.

[25] I. Omer and M. Werman, "Color lines: image specific color representation," in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, vol. 2, June 2004, pp. II–946–II–953 Vol.2.

[26] M. Paul, S. Haque, and S. Chakraborty, "Human detection in surveillance videos and its applications - a review," *EURASIP Journal on Advances in Signal Processing*, vol. 2013, no. 1, 2013.

[27] T. Porter and T. Duff, "Compositing digital images," *SIGGRAPH Comput. Graph.*, vol. 18, no. 3, pp. 253–259, Jan. 1984.

[28] A. Rosenfeld and D. Weinshall, "Extracting foreground masks towards object recognition," in *IEEE International Conference on Computer Vision (ICCV), 2011*, Nov 2011, pp. 1371–1378.

[29] M. Turk and A. Pentland, "Face recognition using eigenfaces," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 1991.*, Jun 1991, pp. 586–591.

[30] J. Wang and C. Li, "Spectral matting based on color information of matting components," in *Advances in Wireless Networks and Information Systems*, ser. Lecture Notes in Electrical Engineering. Springer Berlin Heidelberg, 2010, vol. 72, pp. 119–130.

[31] J. Wang and M. F. Cohen, *Image and video matting: a survey.* Now Publishers Inc, 2008.

[32] Y. Weiss, "Segmentation using eigenvectors: a unifying view," in *The Proceedings of the Seventh IEEE International Conference on Computer Vision, 1999.*, vol. 2, 1999, pp. 975–982.

[33] B. Wu and R. Nevatia, "Detection and tracking of multiple, partially occluded humans by bayesian combination of edgelet based part detectors," *International Journal of Computer Vision*, vol. 75, no. 2, pp. 247–266, 2007.

[34] T. Yu, X. Liu, S. Lim, N. Krahnstoever, and P. H. Tu, "Automatic surveillance video matting using a shape prior," in *IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, Nov 2011, pp. 1761–1768.

[35] D. Zhang, O. Javed, and M. Shah, "Video object segmentation through spatially accurate and temporally dense extraction of primary object regions," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2013*, June 2013, pp. 628–635.

[36] H. Zhou, Y. Yuan, and C. Shi, "Object tracking using SIFT features and mean shift," *Computer Vision and Image Understanding*, vol. 113, no. 3, pp. 345 – 352, 2009, special Issue on Video Analysis.