UNIVERSITY OF CALIFORNIA, SAN DIEGO

Learning-Based Trimap Generation for Video Matting

A thesis submitted in partial satisfaction of the requirements for the degree
Master of Science

in

Computer Science

by

Kyoung-Rok Lee

Committee in charge:

Professor Truong Q. Nguyen, Chair
Professor David J. Kriegman
Professor Serge J. Belongie

2010

The thesis of Kyoung-Rok Lee is approved and it is acceptable in quality and form for publication on microfilm and electronically:

_____

_____

_____
Chair

University of California, San Diego

2010

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGEMENTS

I would like to thank my family for all the support through my Masters program. I would like to acknowledge Professor Truong Nguyen for his support as the chair of my committee. I would also like to acknowledge Stanley Chan and Natan Jacobson of the Video Processing Lab for the valuable discussion regarding Spectral Matting.

ABSTRACT OF THE THESIS

Learning-Based Trimap Generation for Video Matting

by

Kyoung-Rok Lee
Master of Science in Computer Science
University of California, San Diego, 2010
Professor Truong Q. Nguyen, Chair

Object extraction is a critical operation for many content-based video applications. For these applications, a robust and precise extraction technique is required. This thesis proposes an efficient and accurate method for generating a trimap for video matting. We first segment the foreground using motion information and neighboring pixel coherence via graph cuts. Also, we estimate the parameters of a Gaussian Mixture Model for the foreground and background with segmented foreground and estimated static background. Next, we classify the pixels of each frame into models by performing maximum likelihood classification and generate a trimap which is an image consisting of three regions: foreground, background and unknown. Finally, we use the trimap as a guide in spectral matting for video matting. Our experimental results show that the proposed method yields accurate and natural object boundaries.

Chapter 1

Introduction

Extraction of moving objects from a video is an important issue in computer vision and video processing. With object extraction, we are able to composite a scene with background and foreground objects. The background can be a natural scene or computer generated scene. The extracted object can be used for scene analysis such as object recognition and classification. However, accurate foreground estimation is an inherently ill-posed problem because there are a large number of unknowns in the computation. Noise and imperfect motion estimation can affect segmentation results. Although difficulty is found in the general case, the task can be simplified if the scene has a static background with moving objects. Such situations can be found in many applications, including video conferencing, security videos, video-based tracking, compositing video and motion capture. This is the focus of this thesis.

The simplest way to extract a moving object in a static background is using a frame difference. Pixel colors in the current frame are compared to the colors in the previous frame. If the colors are similar, they are classified as ”background” pixels and if not, they are classified as ”foreground” pixels. However, the accuracy is affected by noise, as noisy pixel can be incorrectly classified as an object pixel. Also, it fails to detect an object when the object stops moving since the pixel difference is zero. For example, the method only detects the moving head while ignoring the stationary body.

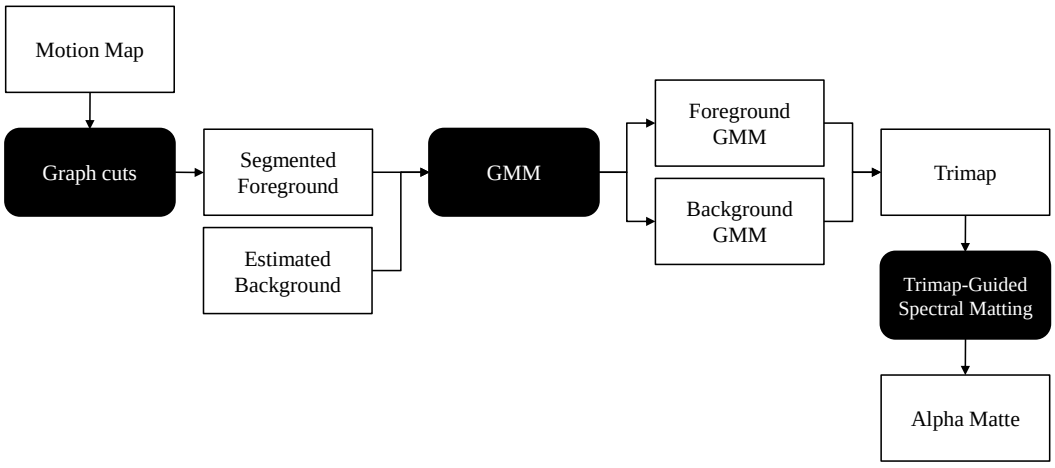


Figure 1.1. Block diagram of the proposed algorithm.

Finally, if the surface of the object is flat, color differences exist only near the object's silhouette and the method yields a hole in the detected object.

Many computer vision and video processing research groups have produced useful algorithms for robust and accurate object segmentation in video. One of the most recent and relevant work is Simple Interactive Object Extraction (SIOX) proposed by (Friedland et al., 2006). They utilize color characteristics for foreground segmentation assuming that the foreground objects are perceptually different from the background. The method creates a set of color representative for foreground and background by clustering the color space driven by user input. Then it assigns all the image pixels to foreground or background by a weighted nearest neighbor search. The algorithm yields comparatively accurate object boundaries and copes well with noise. However, a drawback of the method is that it only yields a binary mask for segmented objects. Every pixel is determined as either foreground or background. Simple binary classification is not a good approach to segmentation since sometimes boundary pixels cannot be classified as either foreground or background. For example, complex structures such as hair or fur are hard to be determined whether they belong to foreground or background. To

obtain accurate and natural result, we should consider partial coverage of a background pixel around the foreground object's boundary, which might include transparency and motion blurring of the foreground element. For accurate separation, image and video matting techniques have been extensively studied (Chuang et al., 2002; Levin et al., 2008b; Smith and Blinn, 1996). However, they require either user interaction such as a trimap which is an image consisting of three regions of foreground, background and unknown, or special setting, e.g. uniform background color and color restriction.

In this thesis, we propose an efficient and accurate object extraction method without any user interaction or extra information. Figure 1.1 shows a diagram of our proposed algorithm. Our method consists of three modules. The graph cuts module segments objects using motion information and neighboring pixel coherence via graph cuts. The GMM Module estimates the parameters of a Gaussian Mixture Model from a segmented object and estimated background, and classifies pixels into each model by performing maximum likelihood classification, generating a trimap. Finally, the trimap-guided spectral matting module creates an alpha matte for video using the trimap as a guide.

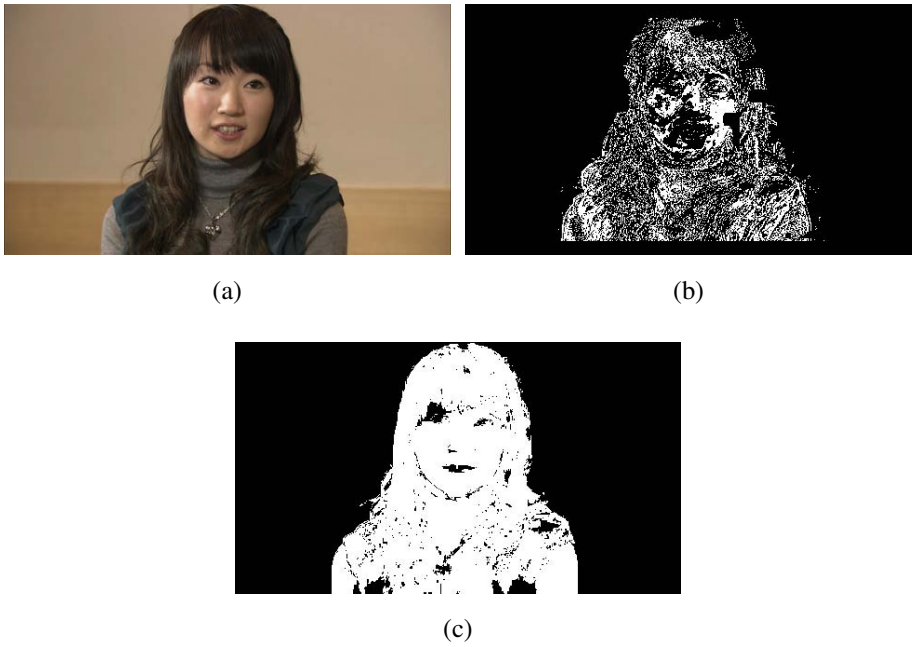
The remainder of the paper is organized as follows: Chapter 2 describes the graph cuts algorithm for foreground segmentation, Chapter 3 describes learning-based trimap generation, and Chapter 4 presents the trimap-guided spectral matting method. Experimental results are provided in Chapter 5 and the thesis is concluded in Chapter 6.

Chapter 2

Foreground Object Segmentation via Graph Cuts

2.1 Motion map

We use optical flow for estimating the inter-frame motion at each pixel in a video sequence (Lucas and Kanade, 1981). Optical flow provides a vector field that shows the direction and magnitude of intensity changes from one image to the other. Given two neighboring frames, previous frame I^{t-1} and current frame I^t, we can estimate motion using the optical flow method. We create a motion map that presents the regions where reliable motion exists. To get a more reliable result, an additional validity bit is added to each pixel in order to indicate which motion vector of the pixel is relevant (Chuang et al., 2002). If the color difference between the pixel in the frame warped by the estimated motion vector and the pixel in the current frame is greater than a certain threshold, this validity bit is set to 0, which implies unreliable. Otherwise, the validity bit is set to 1, which implies that the pixel's motion vector is trustworthy. In the motion map, a pixel is labeled as 1 if the magnitude of motion vector is larger than a threshold and its validity bit is 1. We ignore the direction of motion vector since we are interested only in the



(a) (b)

(c)

Figure 2.1. Foreground segmentation via graph cuts (Nana sequence). (a) original image; (b) motion map; and, (c) binary map of segmented foreground.

existence of motion.

$$M_p = B_p \wedge (\|O_p\| > \tau_m) \tag{2.1}$$

where B_p is the validity bit, O_p is the motion vector, and τ_m is the threshold. Figure 2.1(b) shows the motion map of a frame using the proposed approach.

2.2 Graph cuts

Motion information alone is insufficient to segment moving objects. As shown in Figure 2.1(b), the motion map is too sparse and inaccurate to identify the moving objects. A denser motion map is needed for accurate identification of moving objects. For the foreground segmentation, we use the graph cuts algorithm to overcome inaccurate and noisy motion measurement.

Every pixel $p \in P$ in a frame is assigned a label f_p in some label set A, where

f_p = 1 for the moving object and f_p = 0 for the background. We hypothesize that the pixels that have motion are likely to be object pixels, and the pixels are likely to have the same label if neighboring pixels have similar color. Our goal is to find a labeling f that assigns each pixel $p \in P$ a label $f_p \in A$ satisfying that f is both consistent with the motion information and piecewise smooth. The problem can be formulated in terms of energy minimization.

$$E(f) = \gamma E_s(f) + E_d(f) \tag{2.2}$$

where

$$E_s(f) = \sum_{\{p,q\} \in N} V_{\{p,q\}}(f_p, f_q) \quad (image\ smoothness\ term) \tag{2.3}$$

$$E_d(f) = \sum_{p \in P} D_p(f_p) \quad (motion\ data\ term) \tag{2.4}$$

Parameter γ is the relative weight of the motion data and image smoothness terms and N is a set of all pairs of 4-connected neighboring pixels. The value of γ specifies how strongly neighboring pixels are correlated. A large value of γ causes pixels to group strongly with nearby pixels with similar color, and the result will contain larger homogeneous clusters. On the contrary, if γ is small, then nearby pixels bond weakly and the output will look much more like that obtained by simply thresholding the motion map. Thus, if an input motion map contains a lot of noise, a large value of γ is needed, in order to smooth over the larger clusters of noisy pixels.

The image smoothness term E_s measures spatial coherence between nearby pixels by penalizing discontinuities.

$$V_{\{p,q\}}(f_p, f_q) = e^{-\|I_p - I_q\|^2/\sigma^2} \tag{2.5}$$

This term $V_{\{p,q\}}$ indicates a penalty for discontinuity between pixel p and pixel q. The function $V_{\{p,q\}}$ is large when nearby pixels are similar, and it approaches zero when the color difference is much greater than σ.

The motion data term E_d indicates how appropriate a label f_p is for the pixel p given the observed data. We specifically use the motion map from the previous section

as the observed data by using

$$D_p(f_p) = \begin{cases} (K-m)f_p + m & if\, M_p = 1 \\ m & if\, M_p = 0 \end{cases} \tag{2.6}$$

where

$$K = \max \sum_{\{p,q\} \in N} V_{\{p,q\}}(f_p, f_q) \tag{2.7}$$

where M is the motion map. m is used for minimizing the risk of false motion estimation. Even if a pixel has no motion, the pixel still has a possibility of being an object pixel, and E_s will capture spatial consistency and will connect constant intensity regions with moving pixels. The term m makes it happen by assigning the small amount of energy to the pixel which has no motion. When the motion estimation is accurate, m can be smaller. On the other hand, m has to be larger when motion estimation is inaccurate. If the pixel has motion, the pixel is likely to have value 1 as its label f_p.

The moving object segmentation can be achieved by finding the labeling that minimizes the energy E and this can be solved by the graph cuts algorithm (Boykov et al., 2001; Boykov and Kolmogorov, 2004; Howe and Deschamps, 2004; Ahn and Byun, 2006).

Figure 2.2 illustrates the graph formed for a 3×3 image and the basic segmentation operations performed in graph cuts. The graph cuts algorithm begins by building an undirected graph. The graph is defined as a set of nodes and a set of edges connecting neighboring nodes. The nodes of the graph represent image pixels. There are also two terminal nodes: a source node and a sink node, and they represent ”object” and ”background” labels, respectively. A node in the graph is connected to exactly six other nodes: the source, the sink, and 4-connected neighbors. Each edge determines how strongly the nodes are connected to one another.

The weights of the neighbor edges between pixel nodes are determined by Equation (2.3) and the edges between the pixel nodes and the source and sink can be obtained from Equation (2.4). Once the graph is constructed, the push-relabel algorithm is used

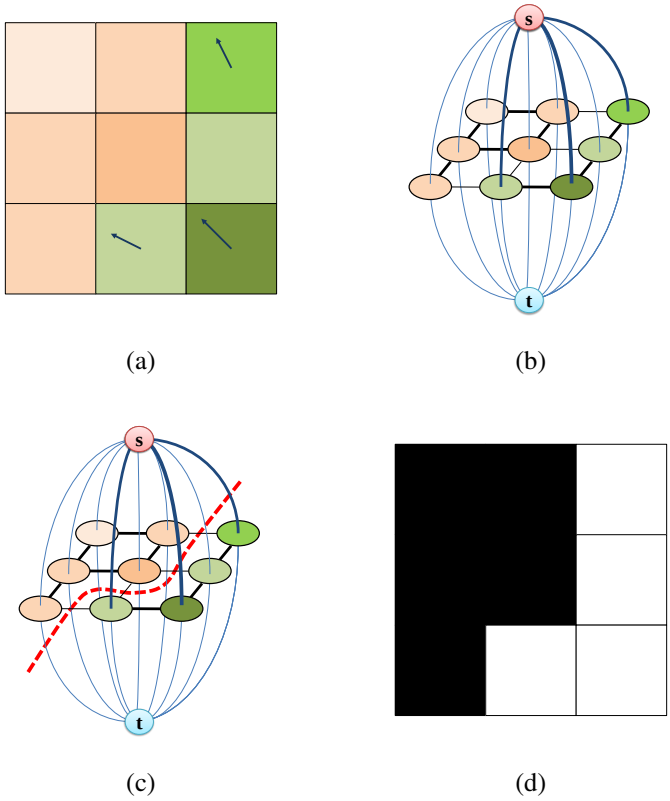


Figure 2.2. A simple foreground segmentation example for a 3x3 image. (a) an image with motion; (b) a graph where the weight of each each is reflected by the edge's thickness; (c) graph cut achieved by finding the labeling minimizing Equation (2.2); and, (d) binary map for foreground.

for separating the source from the sink (Goldberg and Tarjan, 1986). The results of foreground object segmentation are shown in Figure 2.1(c).

As shown in Figure 2.1(b), when only the optical flow method is used to estimate foreground, the boundaries are not specific and estimation performance is poor. As a result, the motion map is very ambiguous to determine the foreground. However, when the graph cuts method is used, the quality of foreground segmentation is improved. This is demonstrated by Figure 2.1(c).

Chapter 3

Trimap Generation Using a Color-Learning Method

3.1 Background estimation

Background estimation is not an easy task since many factors can affect the results such as gradual illumination changes and camera noise. For example, an object's appearance might change slightly resulting in the case where some object pixels produce less pixel difference than the differences produced by artifacts. To distinguish real movement and unexpected artifacts is an important task for background detection. In our approach, we assume that the object pixels produce a large difference at least once over a certain time period while the static background does not. First, we partition a frame into 25 equal sections and accumulate the color changes in each section for a certain time interval. We assume that at least one of the sections has no foreground objects, which means it is purely static background. We choose the section that has minimal change and save its deviation per pixel as e, assuming that the minimal change is caused only by illumination changes or camera noise. We use e as a threshold to determine which pixels belong to the static background. The background estimation is based on the pixel's recent history. Pixels in each frame at current time t are compared with the

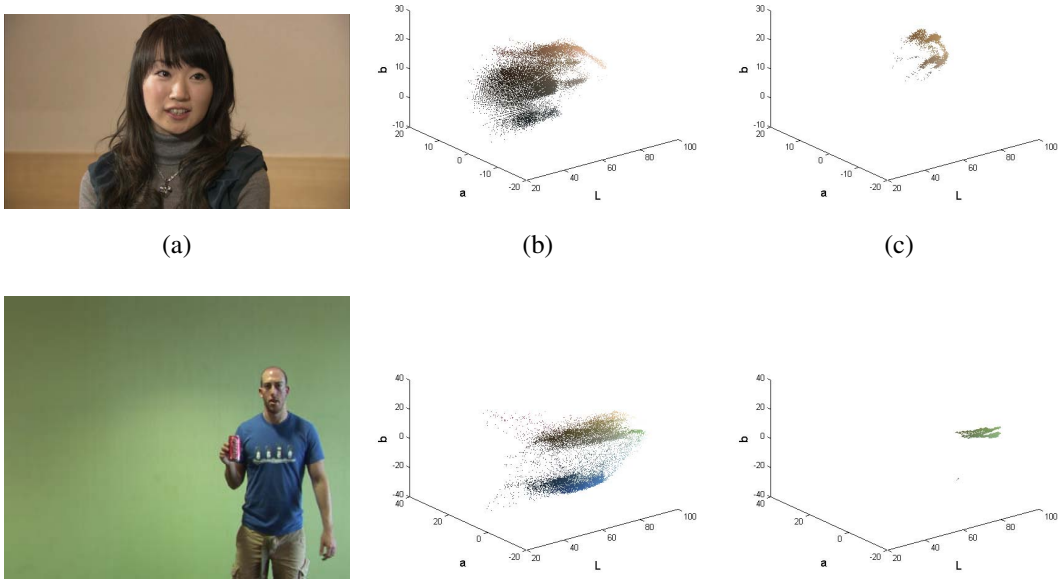


Figure 3.1. Foreground and background color distributions (Nana and Natan sequence). (a) Nana frame; (b) Nana foreground color distribution; (c) Nana background color distribution; (d) Natan frame; (e) Natan foreground color distribution; and, (f) Natan background color distribution.

previous N frames (N=15 in this thesis). If the difference is larger than e in any of the frames, the pixel is regarded as an object pixel and it is pruned from the background map. e is updated when the portion of background pixels in a frame becomes less than 35% to adapt to new illumination changes. Figure 3.3(a) shows background samples after background estimation.

3.2 Trimap generation

We assume that the foreground and background colors are mostly distinctive. In this assumption, each color distribution can be characterized by a Gaussian Mixture Model (GMM). We estimate two GMM classes, G_o and G_b, using the segmented object

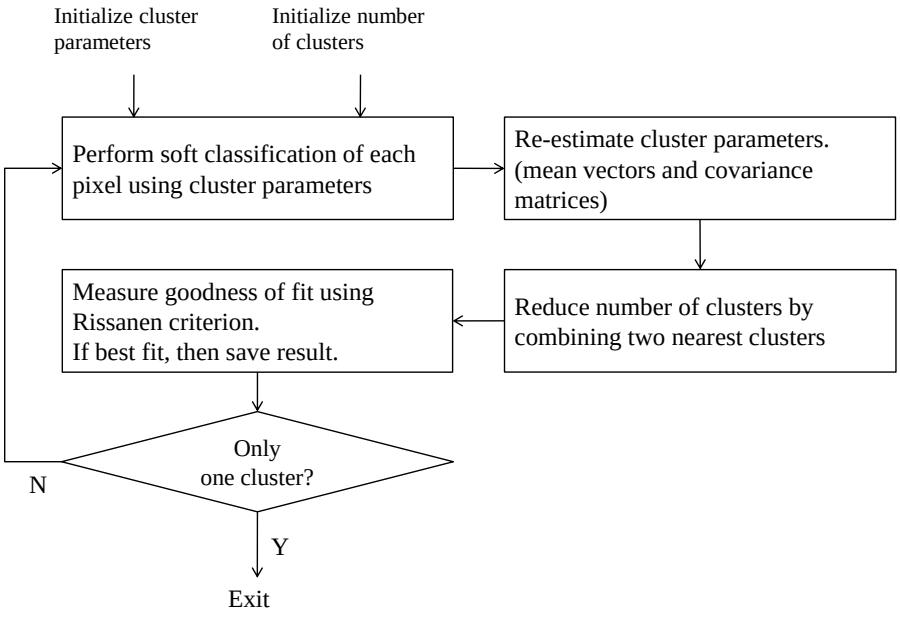


Figure 3.2. GMM cluster algorithm.

and estimated background. Figures 3.1(b) and (e) show the foreground color distribution of Nana (Figure 3.1(a)) and Natan (Figure 3.1(d)) sequences, whereas Figures 3.1(c) and (f) show the corresponding background color distribution. Note that the foreground colors and background colors are separately distributed. For example, in the Nana sequence, skin color and background color seem similar, but in Lab space these two colors can be differentiated from each other.

For the GMM process, we use Lab color space (Labs, 1996), which is a conceptually uniform color space. The number of clusters in each model is estimated by using the expectation-maximization (EM) algorithm (Dempster et al., 1977) together with an agglomerative clustering strategy (Bouman, 1997). The estimation is based on the Rissanen order identification criterion known as minimum description length (MDL) (Rissanen, 1983). Figure 3.2 shows the overview of clusting operation. For building a model, the algorithm first initializes the number of clusters and cluster parameters. It

calculates the cluster means by choosing a proper number of samples from training data, and the covariances are assigned as the covariance of the whole data set. After this initialization, the algorithm enters a loop and combines the two nearest clusters until only one cluster remains. The algorithm saves the set of clusters when a minimum of the Rissanen criterion occurs. The estimated cluster parameters are used in the classification process.

After the GMM process, we classify current frame pixels into the classes, G_o and G_b.

$$T_p = \begin{cases} 1 & if\ P(I_p|G_o) > P(I_p|G_b)\ and\ |P(I_p|G_o) - P(I_p|G_b)| > \delta_G \\ 0 & if\ P(I_p|G_o) < P(I_p|G_b)\ and\ |P(I_p|G_o) - P(I_p|G_b)| > \delta_G \\ 0.5 & otherwise \end{cases} \tag{3.1}$$

The term $P(\cdot)$ in Equation (3.1) is the log likelihood and it describes on how the pixel I_p fits into the given two GMM classes.

$$P(I_p|G) = \log(\sum_{k=1}^{K} \pi_k S(f_p|G)) \tag{3.2}$$

where K is the number of subclasses in class G and the probability density function for the pixel $S(I_p|G)$ is given by

$$S(I_p|G) = \frac{1}{(2\pi)^{c/2}} |R_k|^{-1/2} \exp(-\frac{1}{2}(I_p - \mu_k)^\top R_k^{-1}(I_p - \mu_k)) \tag{3.3}$$

where the parameters are defined as

c : color channels

π_k : the mixing coefficient represents the probability that a pixel has subclass k.

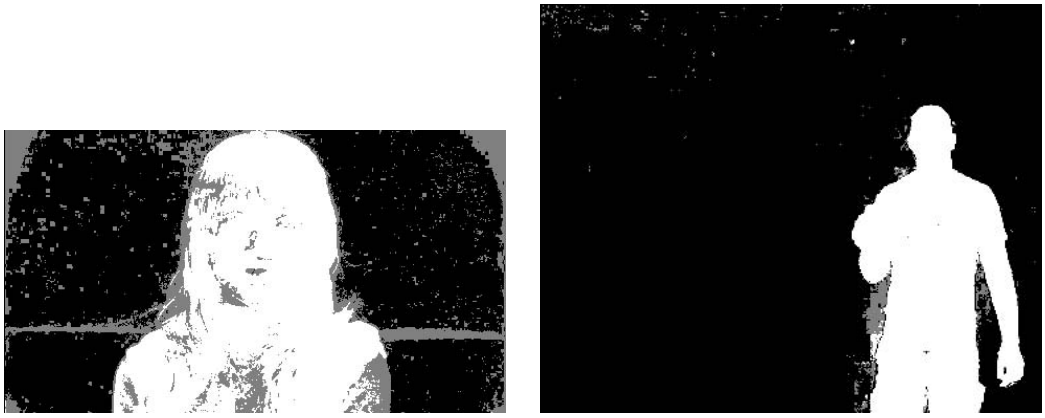
μ_k : the c dimensional spectral mean vector for subclass k.

R_k : the $c \times c$ spectral covariance matrix for subclass k.

δ_G in Equation (3.1) is a small constant which is required for determining the uncertain region. In Figures 3.1(e) and (f), brown and blue colors are dominant in the foreground color distribution, while the background color is generally green. However, since segmented object includes some of the background pixels near its boundary, the foreground color distribution contains green color. Thus, the boundary becomes an uncertain region. If the absolute difference between $P(I_p|G_o)$ and $P(I_p|G_b)$ is less than or equal to δ_G, the pixel belongs to the uncertain region. Figure 3.3 shows the results of the trimap generation process for the two sequences. Note that generated trimaps are reasonably accurate. The complex area such as hair in Figure 3.3(c) is determined as an uncertain region since the thin hair may be transparent and it includes both foreground and background colors. Also, object boundary pixels and pixels that have similar color become uncertain regions.

(a)

(b)



(c)

Figure 3.3. Trimap generation (Nana and Natan sequence). (a) estimated background; (b) segmented foreground from Chapter 2; and, (c) generated trimap.

Chapter 4

Trimap-Guided Spectral Matting

4.1 Closed-form natural matting

Accurate object extraction is an important task in computer vision and video processing because it applies to image editing and video production. Many matting methods have been proposed to extract a high quality matte from video sequences. Alpha matting or digital matting was first introduced by (Porter and Duff, 1984). Figure 4.1 shows a result of alpha matting. The method uses an opacity value alpha matte to control the proportion of foreground pixel values F_i and background pixel values B_i that contribute to a single image pixel value $I_i(i = (x, y))$. The operation is summarized by the compositing equation:

$$I_i = \alpha_i F_i + (1 - \alpha_i) B_i \tag{4.1}$$

where α_i is the foreground opacity. The alpha matting task is challenging since it is an ill-posed problem; we must estimate not only the foreground color and the background color, but also the alpha matte from a single color measurement.

One state of the art alpha matting methods is the Closed-form natural matting approach proposed by (Levin et al., 2008a). This method derives a cost function from local smoothness assumptions on foreground and background colors F and B, and shows that the terms F and B can be eliminated in the compositing equation, yielding a quadratic

(a) (b) (c)

Figure 4.1. Alpha matting method. (a) the original image; (b) the alpha matte; and, (c) the composite image and zoomed-in area of the image.

cost function in α, which can be solved as a sparse linear system. It is assumed that each F and B is a linear combination of two colors over a small window (3×3) around each pixel, which is referred to as the color line model (Omer and Werman, 2004). Under this assumption, α values in a small window w can be expressed as

$$\alpha_i = \sum_c a^c I_i^c + b, \quad \forall i \in w \tag{4.2}$$

where c indexes the three color channels and a^c and b are constants in the window. Thus, the matting cost function (Levin et al., 2008a) is defined as

$$J(\alpha, a, b) = \sum_{j\in I} \left(\sum_{i\in w_j} \left(\alpha_i - \sum_c a_j^c I_i^c - b_j \right)^2 + \varepsilon \sum a_j^{c2} \right) \tag{4.3}$$

where $\varepsilon \sum a_j^{c2}$ is included for numerical stability. It is necessary that the image pixels I_i are all constant values in the window w in order to determine a^c and b uniquely.

To simplify this equation, let us define matrices $G_j \in \mathbb{R}^{(|w_j|+|c|)\times 4}$ and $\overline{\alpha}_j \in \mathbb{R}^{(|w_j|+|c|)}$. The first $|w_j|$ rows of G_j are given by $\left[I_i^{c\top}\ 1\right],\ i \in w_j$ and the last three rows are given by $[\sqrt{\varepsilon} Id_3 \quad 0]$, where Id_n is an identity matrix of size $n\times n$. The first $|w_i|$ entries of $\overline{\alpha}_j$ are given by $\alpha_i,\ i \in w_j$ and the last three entries are all zeros. With this notation, $J(\alpha, a, b)$ can be rewritten as

$$J(\alpha, a, b) = \sum_{j\in I} \|G_j v_j - \overline{\alpha}_j\|^2 \tag{4.4}$$

where $v_j = \begin{bmatrix} a_j^c \\ b_j \end{bmatrix}$. We can estimate v_i for each window w_j, as

$$v_j = \arg\min_v \|G_j v_j - \overline{\alpha}_j\|^2 = (G_j^\top G_j)^{-1} G_j^\top \overline{\alpha}_j \tag{4.5}$$

Now, a^c and b are eliminated from the cost function, yielding a quadratic cost in parameter α alone:

$$\begin{aligned} J(\alpha) &= \sum_{j \in I} \left[\overline{\alpha}_j^\top (Id_{|w_j|+3} - G_j (G_j^\top G_j)^{-1} G_j^\top) \overline{\alpha}_j \right] \\ &= \alpha^\top L \alpha \end{aligned} \tag{4.6}$$

where L is an $N \times N$ matrix, whose $(i,j)^{th}$ element is

$$L(i,j) = \sum_{q|(i,j) \in w_q} (\delta_{ij} - \frac{1}{|w_q|}(1 + (I_i - \mu_q)^{\mathrm{T}} (\Sigma_q + \frac{\varepsilon}{|w_q|} Id_3)^{-1} (I_j - \mu_q))) \tag{4.7}$$

δ_{ij} is the Kronecker delta, Σ_q is a 3×3 covariance matrix, and μ_k is a 3×1 mean vector of the colors in a window w_k. The matrix L is called as the Matting Laplacian and the constructed cost function is quadratic in the alpha mattes. Consequently, this cost function can be minimized by solving a linear system.

4.2 Trimap-guided spectral matting

A new approach called Spectral Matting is proposed by (Levin et al., 2008b) as a further analysis of natural matting. The approach generalizes the compositing equation by assuming that the input image is modeled as a convex combination of K image layers

$$I_i = \sum_{k=1}^{K} \alpha_i^k F_i^k \tag{4.8}$$

The α_i^k parameters are the matting components of the image. The paper proposed that the smallest eigenvectors of the matting Laplacian L span the individual matting components of the image, therefore recovering the matting components of the image can be done by finding a linear transformation of the eigenvectors. They first initialize α^k

by applying a k-means algorithm on the smallest eigenvectors, and projecting the set of binary vectors that indicate each resulting cluster onto the span of the eigenvectors. The matting components are found by minimizing an energy function

$$\sum_{i,k} |\alpha_i^k|^\omega + |1 - \alpha_i^k|^\omega, \; where \; \alpha^k = Ey^k \tag{4.9}$$

However, one challenge in the method is determining the proper number of matting components for a given image. It is difficult to choose the number of matting components automatically for a given video sequence. Also, performing a k-means algorithm to determine the best combination of eigenvectors is time consuming. To solve these problems, (Chan et al., 2010) uses guided information to obtain the alpha matte. Given a trimap T, the task reduces to determining a linear combination vector y that minimizes the cost.

$$\min_y \|T - Ey\|_2^2 \tag{4.10}$$

where E is a matrix of eigenvectors of L and y is a linear combination vector. A regularization term is added to T to penalize the cost when the value α is not close to either 0 or 1. The cost function becomes

$$\min_y \|T - Ey\|_2^2 \; + \lambda \sum_{i \in I} (-\alpha_i^2 + \alpha_i) \tag{4.11}$$

where $\alpha = Ey$. Placing this constraint into the objective function, we have

$$\min_y \|T - Ey\|_2^2 \; + \lambda(- \|Ey\|_2^2 + \mathbf{1}^\top Ey) \tag{4.12}$$

where $\mathbf{1}$ denotes a vector of ones. Equation (4.12) is quadratic and can be solved by a standard sparse linear system solver such as the MATLAB backslash operator. Figures 5.2 and 5.3 show the results for trimap-guided spectral matting.

Chapter 5

Results

All presented experiments are performed on an Intel Core2Duo 2.53GHz processor with 4GB RAM. We tested two sample sequences, Nana and Natan. The execution time depended on the resolution of the sequence. For a 480 x 270 sequence, the total computation time is 74 seconds; graph cuts, GMM, and the trimap-guided spectral matting algorithms took 2, 12, and 60 seconds, respectively. For comparison, the spectral matting (Levin et al., 2008a) took about 7 minutes per frame. Figures 5.2 and 5.3 show some results for video matting using a learning based trimap; original images, generated trimaps, estimated object mattes, and composites into a new scene using the extracted object based on the object matte. In Figure 5.1 we compared our result with a conventional binary segmentation method, SIOX (Friedland et al., 2006). The proposed method yields more accurate boundary of object since spectral matting is ideal for segmenting complex structures such as hair or fur. Another advantage of our method is that this algorithm doesn't require any user interaction at all while other methods require user-defined trimap selection or special setting. More experiments and results of our method can be found at: http://videoprocessing.ucsd.edu/∼ultralkl/projects/VideoMatting/

(a)

(b)

(c)

Figure 5.1. Comparison between proposed method and SIOX. (a) original image and zoomed-in area of the original image; (b) proposed method and zoomed-in area of the proposed method; and, (c) SIOX and zoomed-in area of SIOX.

(a)

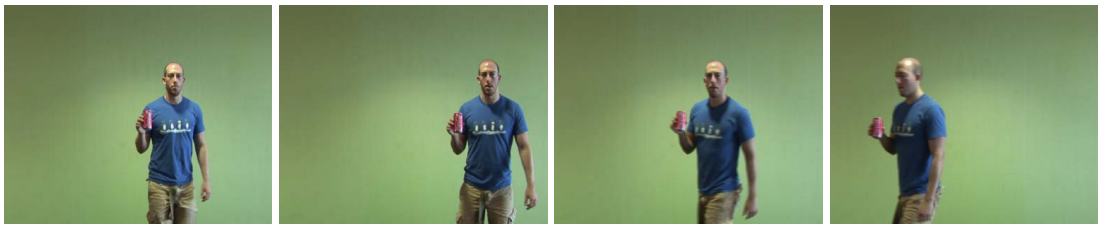
(b)

(c)

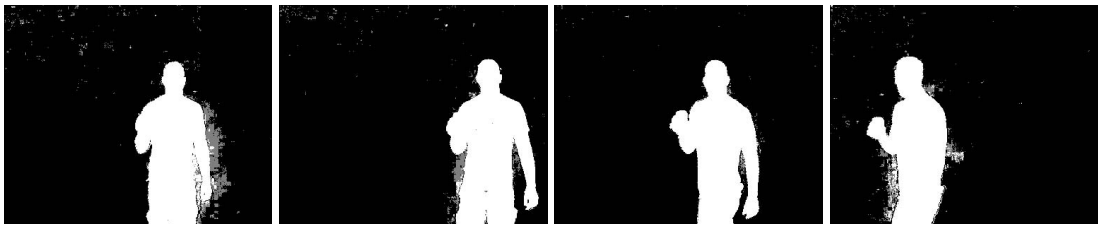
(d)

(e)

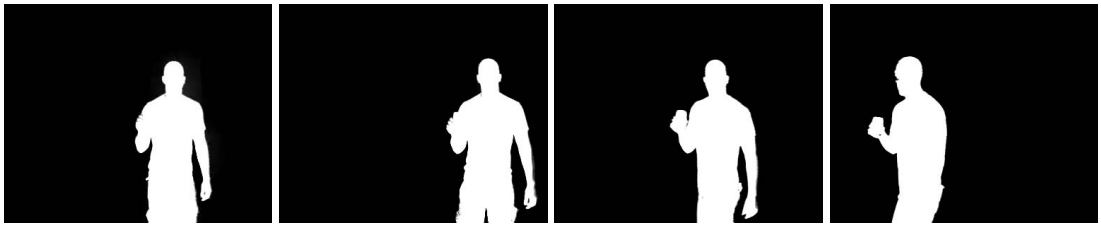
Figure 5.2. Results of Nana sequence: (a) original frames; (b) trimaps; (c) alpha mattes; (d) composites; and, (e) zoomed-in areas of the composites.



(a)

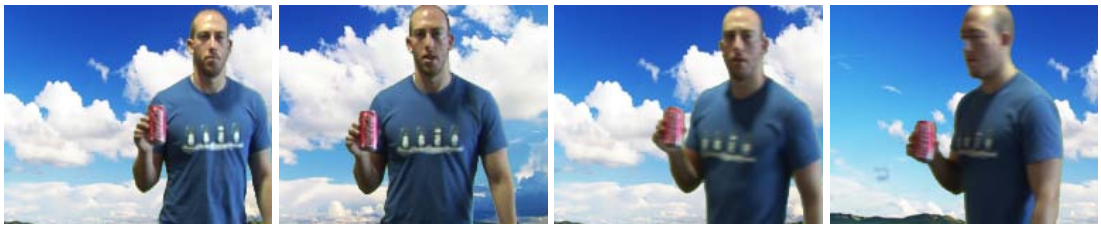


(b)



(c)

(d)



(e)

Figure 5.3. Results of Natan sequence. (a) original frames; (b) trimaps; (c) alpha mattes; (d) composites; and, (e) zoomed-in areas of the composites.

Chapter 6

Conclusion

This thesis proposes a video matting method based on trimap generation in video with stationary background. The method produces more accurate and natural results than the currently prevailing approach based on binary segmentation. The graph cuts method overcomes the effects of noise by aggregating information from a local neighborhood around each pixel, while exploiting motion information. The learning method using GMM generates reliable trimaps and trimap-guided spectral matting yields accurate and high quality mattes which be used for a variety of applications. The proposed method is tested on two sequences, Nana and Natan, and yields accurate object boundaries which are robust to motion estimation error.

However, the computation time of this method is still slow since it includes some sophisticated algorithms that require intensive computation. Due to the size of the Laplacian matrix L is $W \times H$, where W and H are the width and height of the image, the method requires a lot of memory which limits the input video's spatial size. Also, it often fails in some cases; when the motion of the object is too small to obtain a reasonable motion map, where background and foreground are not distinctive. Figure 6.1 shows two examples of false extraction. In the case of Figure 6.1(a), the method failed to generate a correct trimap because colors present in the jeans are similar to those present in the background. Figure 6.1(b) shows an appropriate trimap of a person, but

(a)

(b)

Figure 6.1. Limitations. (a) original frame, trimap, and alpha matte of Natan1 sequence; and, (b) original frame, trimap, and alpha matte of Natan2 sequence.

the boundaries are blurry and inaccurate in the matte. If we take advantage of video information, such as motion information and temporal coherence, and then use that information as additional constraints for spectral matting, we can achieve better results. In the future, we plan to extend the proposed method to video with non-static background using global motion estimation. In addition, we intend to apply the proposed work to complicated video sequences which may fail to satisfy the requirement of distinctive colors in the background and foreground. Finally, we will investigate depth estimation techniques to generate multiple alpha layers from a single video sequence. With accurate segmentation of alpha layers, we will be able to reconstruct the geometry of a scene, ultimately being able to convert from 2D to 3D stereo video.

References

Ahn, J.-H., and Byun, H., 2006: Accurate foreground extraction using graph cut with trimap estimation. In *Lecture Notes in Computer Science*, 1185–1194. Springer Berlin / Heidelberg. ISBN 978-3-540-68297-4. doi:http://doi.acm.org/10.1007/11949534_120.

Bouman, C. A., 1997: Cluster: An unsupervised algorithm for modeling gaussian mixtures.

Boykov, Y., and Kolmogorov, V., 2004: An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Trans. Pattern Anal. Mach. Intell.*, **26**(9), 1124–1137. ISSN 0162-8828. doi:http://dx.doi.org/10.1109/TPAMI.2004.60.

Boykov, Y., Veksler, O., and Zabih, R., 2001: Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell.*, **23**(11), 1222–1239. ISSN 0162-8828. doi:http://dx.doi.org/10.1109/34.969114.

Chan, H., Lee, K.-R., and Nguyen, T., 2010: Fast spectral matting methods. In preparation.

Chuang, Y.-Y., Agarwala, A., Curless, B., Salesin, D. H., and Szeliski, R., 2002: Video matting of complex scenes. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, 243–248. ACM, New York, NY, USA. ISBN 1-58113-521-1. doi:http://doi.acm.org/10.1145/566570.566572.

Dempster, A. P., Laird, N. M., and Rubin, D. B., 1977: Maximum likelihood from incomplete data via the em algorithm. *JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B*, **39**(1), 1–38.

Friedland, G., Jantz, K., Lenz, T., Wiesel, F., and Rojas, R., 2006: A practical approach to boundary accurate multi-object extraction from still images and videos. In *ISM '06: Proceedings of the Eighth IEEE International Symposium on Multimedia*, 307–316. IEEE Computer Society, Washington, DC, USA. ISBN 0-7695-2746-9. doi:http://dx.doi.org/10.1109/ISM.2006.9.

Goldberg, A. V., and Tarjan, R. E., 1986: A new approach to the maximum flow problem. In *STOC '86: Proceedings of the eighteenth annual ACM symposium on Theory of computing*, 136–146. ACM, New York, NY, USA. ISBN 0-89791-193-8. doi: http://doi.acm.org/10.1145/12130.12144.

Howe, N. R., and Deschamps, A., 2004: Better foreground segmentation through graph cuts. Technical report, Smith College. http://arxiv.org/abs/cs.CV/0401017.

Labs, H., 1996: Hunter lab color scale. *Insight on Color 8 9*.

Levin, A., Lischinski, D., and Weiss, Y., 2008a: A closed-form solution to natural image matting. *IEEE Trans. Pattern Anal. Mach. Intell.*, **30**(2), 228–242. ISSN 0162-8828. doi:http://dx.doi.org/10.1109/TPAMI.2007.1177.

Levin, A., Rav-Acha, A., and Lischinski, D., 2008b: Spectral matting. *IEEE Trans. Pattern Anal. Mach. Intell.*, **30**(10), 1699–1712. ISSN 0162-8828. doi:http://dx.doi.org/10.1109/TPAMI.2008.168.

Lucas, B. D., and Kanade, T., 1981: An iterative image registration technique with an application to stereo vision (ijcai). In *Proceedings of the 7th International Joint Conference on Artificial Intelligence (IJCAI '81)*, 674–679.

Omer, I., and Werman, M., 2004: Color lines: Image specific color representation. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, **2**, 946–953. ISSN 1063-6919. doi:http://doi.ieeecomputersociety.org/10.1109/CVPR.2004.62.

Porter, T., and Duff, T., 1984: Compositing digital images. In *SIGGRAPH '84: Proceedings of the 11th annual conference on Computer graphics and interactive techniques*, 253–259. ACM, New York, NY, USA. ISBN 0-89791-138-5. ISSN 0097-8930. doi: 10.1145/800031.808606.

Rissanen, A., 1983: universal prior for integers and estimation by minimum description length. *Annals of Statistics*, **11**(2), 417–431.

Smith, A. R., and Blinn, J. F., 1996: Blue screen matting. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, 259–268. ACM, New York, NY, USA. ISBN 0-89791-746-4. doi: http://doi.acm.org/10.1145/237170.237263.