FUZZY LOGIC TECHNIQUES FOR MATTING USING INFRARED & COLOR IMAGE

By

Guodong Gao

A thesis submitted to the
Department of Computer Science
in conformity with the requirements for
the degree of Master of Science

Bishop's Unversity
Sherbrooke, Quebec, Canada
April 2011

Accepted by the Department of the Computer Science, Bishop's University, in partial fulfillment of the requirements for the degree of M.Sc. in Computer Science.

Thesis Examination Committee:

_____**Supervisor**
Dr. Layachi Bentabet

_____**Examiner**
Dr. Madjid Allili

_____**Examiner**
Prof. Mohamed Airouche

_____ **Chairperson**
Dr. Nelly Khouzam

Date:

Abstract

Matting refers to the problem of accurate foreground estimation in images and video. From a computer vision perspective, this task is extremely challenging because it is massively ill-posed—at each pixel we must estimate the foreground and the background colors, as well as the foreground opacity(alpha matte) from a single color measurement. It is one of the key techniques in many image editing, film and video production applications. With the recent advances of digital cameras, using matting techniques to create novel composites or facilitate other editing tasks has gained increasing interests from both professionals as well as consumers. Consequently, various matting techniques and systems have been proposed to try to efficiently extract high quality mattes from both still images and video sequences.

In this thesis, we focus on scenes with human actors. The foreground is the actor that should be put in another context by switching the background. For this purpose, we make use of a second imaging device that is an Infrared (IR) camera. Fuzzy Logic concepts are used to combine the information from the color and infrared cameras in order to produce reliable and accurate matting.

Acknowledgements

I would like to thank my supervisor, Dr. Layachi Bentabet, for his patient guidance and support in my work. Without his able step-by-step guidance I could not have worked out this thesis. He has offered me valuable ideas, suggestions and criticisms with his profound knowledge in mathematics and computer vision. I wish to extend my thanks to my classmate Hui Zhang. I would like to give my special thanks to my parents whose support and encouragement enabled me to complete this work.

Table of Contents

LIST OF FIGURES

Chapter 1: Introduction

1.1 Background

Matting refers to the accurate estimation of foreground objects in still images or video sequences. It was mathematically established by Porter and Duff in 1984 [1, 2], and since then it has been extensively studied and many approaches have been developed and optimized. Some of them have been successfully commercialized, and play an important role in many image and video editing applications, such as film-making, image based modeling and rendering, object-oriented media compression and so on [3].

Accurately separating a foreground object from the background involves determining both full and partial pixel coverage, also known as pulling a matte, or digital matting. Porter and Duff [2] introduced the alpha channel to control the linear interpolation of foreground and background colors for anti-aliasing purposes when rendering a foreground over a new background. For this purpose, the observed image I is modeled as a convex combination of a foreground image F and a background image B. This combination is controlled by a weight parameter α that is referred to as the alpha matte. For a pixel $i = (x,y)$, the matting equation can be expressed as follows:

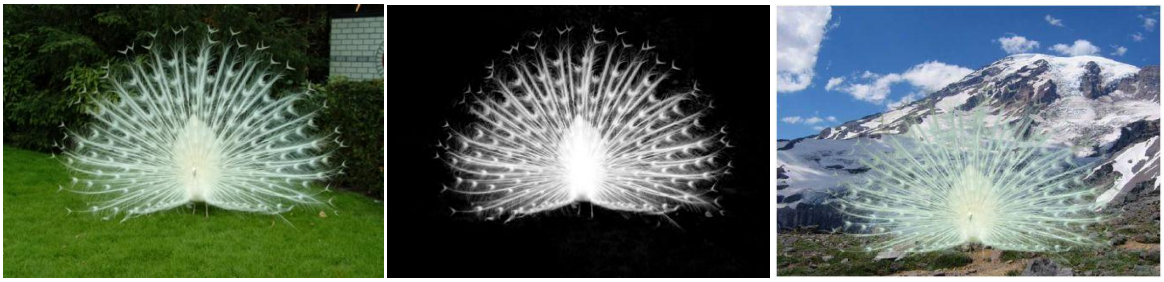
$$I_i = \alpha_i F_i + (1-\alpha_i) B_i, \qquad (1.1)$$

where $0 \leq \alpha_i \leq 1$ reflects the foreground's opacity. If $\alpha_i = 1$, pixel *i* is a definite foreground, and if $\alpha_i = 0$ pixel *i* is a definite background. Otherwise (i.e. $0 < \alpha_i < 1$), it is a mixed pixel. In most natural images, the majority of pixels are either definite foreground or definite background, so the task of accurately estimating alpha values focuses on the mixed pixels. For a given input image, the known information we have for each pixel is the three dimensional color vector $I_i = (I_i^R, I_i^G, I_i^B)$ and the unknown variables are the three dimensional color vectors of F_i^C and B_i^C, and the scalar alpha value α_i, as shown in equation 1.2:

$$\begin{cases} I_i^R = \alpha_i F_i^R + (1-\alpha_i) B_i^R \\ I_i^G = \alpha_i F_i^G + (1-\alpha_i) B_i^G \\ I_i^B = \alpha_i F_i^B + (1-\alpha_i) B_i^B \end{cases} \tag{1.2}$$

Solving the system given by (1.2) involves the determination of 7 unknown variables with only three equations. This is obviously a severely under constrained problem. Most matting approaches proposed in literature require additional information based on the user guidance and/or prior assumptions on image statistics to constrain the problem.

Once a matte has been extracted, the foreground *F* can be easily put in front of a new background, by replacing the original background *B* with a new background image *B*' in equation 1.1, as shown in Figure 1.



(a) Input image (b) Alpha matte (c) composite images

Figure 1: matting compsite example

In this thesis, we provide an approach towards accurate and efficient matting by creating and using extra information. An infrared camera is used to constrain the matting problem. This thesis is composed of four chapters. In the first chapter, we define the matting problem and we show that it is an ill-posed problem that requires extra constraints in order to be solved. In the second chapter, we give a review of some classical methods for image matting using a single view. We concentrated our review on the Bayesian matting approaches [4], and the closed-form solution [5, 6]. These techniques are explained, implemented and tested on real datasets. In the third chapter, we provide an overview of some matting techniques that employ additional sources of information to solve the matting problem. We will present the Defocus technique [7] and the Camera Arrays solution [8]. In the fourth chapter, we will introduce and discuss our fuzzy logic-based techniques for matting. We will show how to build decision list and fuzzy inference systems to combine infrared and color cues. Our techniques will be tested and validated on images pairs that we have acquired. Technical issues such as registration and automatic Trimap construction will also be discussed. A section presenting our conclusion and possible future work is provided at the end of this thesis.

Chapter 2: Matting with a Single View

2. Matting with a single view

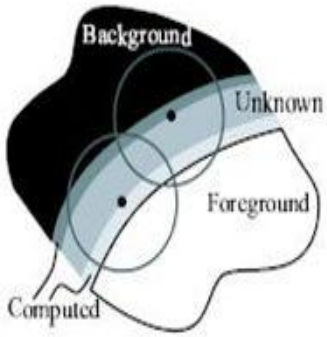
In this chapter, we provide a review of some classical image matting algorithms and systems that use a single view. These techniques are tested and their matting results are given at the end of each section.

2.1 Bayesian Matting

Bayesian matting [4], extracts a foreground element from a background image by estimating color and opacity at each pixel. This approach uses a Bayesian framework for modeling both the foreground and background color distributions with spatially varying sets of Gaussians, and assumes a fractional blending of the foreground and background colors to produce the final output. It then uses a maximum-likelihood criterion to estimate the optimal opacity, foreground and background simultaneously. The model construction and alpha estimation procedure is summarized as follows:

- Segment the input image into three regions: “background”, “foreground” and “unknown”, as shown in Figure 2 (b). This segmentation is done by hand using graphic image editing tool. The resulting image is called a ***trimap***.

(a) Input image **(b) User specified trimap** **(c) Bayesian alpha matte**



(d) Neighborhood definition

Figure 2: Bayesian Matting Principle

- Use a continuously sliding window for neighborhood definition. This window marches inward from the foreground to background region as shown in Figure 2 (d). For each window's location, the foreground and background samples are used to build color Gaussian distributions for F and B in area.
- To find a best α for distribution begin by using the maximum a posteriori (MAP) technique, at each unknown pixel I_i to estimate F_i, B_i and α_i [9]:

$$
\begin{aligned}
&\arg\max_{F_i,B_i,\alpha_i} \; P(F_i,B_i,\alpha_i \mid I_i) \\
&= \arg\max_{F_i,B_i,\alpha_i} \; \left[P(I_i \mid F_i,B_i,\alpha_i)P(F_i)P(B_i)P(\alpha_i)\right] \\
&= \arg\max_{F_i,B_i,\alpha_i} \; \left[L(I_i \mid F_i,B_i,\alpha_i)+L(F_i)+L(B_i)+L(\alpha_i)\right]
\end{aligned}
\tag{2.1}
$$

In equation (2.1), $L(\cdot)$ is the log likelihood $L(\cdot)=\log P(\cdot)$. The problem now is reduced to defining the log likelihoods *L(C| F, B, α)*, *L(F)*, *L(B)* and *L(α)*. The first term is modeled as the difference between the observed image and the image that would be predicted by the estimated F_i, B_i and α_i:

$$
L(I_i \mid F_i,B_i,\alpha_i) = \frac{-\left\| I_i - \alpha_i F_i - (1-\alpha_i)B_i \right\|^2}{\sigma_{I_i}^2} \tag{2.2}
$$

where the color variance $\sigma_{I_i}^2$ is measured locally. It models the error with a Gaussian probability distribution centered at $\bar{I} = \alpha F + (1-\alpha)B$ with standard deviation σ_{I_i}.

To estimate $L(F_i)$, firstly the foreground colors set, in the nearby region defined by the sliding window, is partitioned into several clusters using the method of Orchard and Bouman [10]. Secondly, for each cluster, the weighted mean $\bar{F}$ and covariance matrix Σ_F are calculated to estimate an oriented Gaussian. The likelihood $L(F_i)$ is then defined as $-(F_i-\bar{F})^T \sum_F^{-1}(F_i-\bar{F})/2$. The background's likelihood $L(B_i)$ is calculated in a similar manner using the background samples. $L(\alpha_i)$ is treated as a constant.

Equation (2.1) is solved in two steps. In the first step, α_i is considered constant and F_i and B_i are estimated according to:

$$\begin{bmatrix} \sum_F^{-1} + I_3 \cdot \alpha_i^2 / \sigma_i^2 & I_3 \cdot \alpha_i \cdot (1-\alpha_i) / \sigma_i^2 \\ I_3 \cdot \alpha_i \cdot (1-\alpha_i) / \sigma_i^2 & \sum_B^{-1} + I_3 \cdot (1-\alpha_i)^2 / \sigma_i^2 \end{bmatrix} \begin{bmatrix} F \\ B \end{bmatrix} = \begin{bmatrix} \sum_F^{-1} \overline{F} + I_i \cdot \alpha_i^2 / \sigma_i^2 \\ \sum_B^{-1} \overline{B} + I_i \cdot (1-\alpha_i)^2 / \sigma_i^2 \end{bmatrix} \quad (2.3)$$

Where I_3 is a 3×3 identical matrix. In the second step, F_i and B_i are considered constants and α_i is estimated as:

$$\alpha_i = \frac{(I_i - B_i)(F_i - B_i)}{\left\| F - B \right\|^2} \quad (2.4)$$

The steps described in equations (2.3) and (2.4) are repeated until a stable solution is reached. When the input image is composed of smooth regions and given a well-specified trimap, the Bayesian matting approach can generate accurate mattes. However, when these assumptions are violated, such as in highly textured images, the Gaussian model will be unable to model the high order statistics of color distributions. Also, if the correlation between the pixels in the unknown region and those in the background and the foreground regions is weak, as in the regions containing fine and long details, the Bayesian approach will result in coarse and inaccurate matting. Bayesian Matting tend to use all samples without considering their legitimacy, thus it may introduce significant errors when the sample set is not properly constructed. The later Robust Matting system [11] presents a sample selection procedure to re-examine the collected samples and only pick out a small number of "good" ones to use, thus can generate more accurate and robust results.

Finally, Bayesian matting is computationally expensive since it involves parameters estimations and several iterations to converge to a stable solution for each pixel. We have implemented the Bayesian approach and tested it on some images. The matting results in

figure 3 show that Bayesian matting can fail if the image contains some texture or fine details such as fur or hair.

Figure 3: Some Bayesian matting results

2.2 Closed Form Solution

The closed form matting [4, 5] is an affinity based approach, which derives the alpha matte value of a pixel in the unknown region using affinity measures with samples from the background and the foreground regions. Unlike sampling-based approaches, such as the Bayesian matting, which require the definition of a trimap, closed form solution uses fewer samples given under the form of "s*cribbles*" (see figure 4). Besides, the foreground *F* and the background *B* are assumed to be locally constant within a sliding window (typically a 3 by 3 or a 5 by 5 window). This assumption allows constructing a quadratic cost function in α from which *F* and *B* are analytically eliminated. The minimization of the cost function leads to the estimation of the alpha matte.

The underlying assumption made in closed form solution is the color line model, which states that if the foreground (or background) color in a local window ***w*** is constant, α would be a linear transformation of *I* that is given by:

$$\alpha_i = \sum_c a^c I_i^c + b, \forall i \in w, \qquad (2.5)$$

where *c* refers to color channels, *i* is a pixel index, a^c and b are constants in ***w*** and depend on *F* and *B* [12]. The overall matting cost function is then defined as

$$J(\alpha, a, b) = \sum_{j \in I} \left(\sum_{i \in w_j} (\alpha_i - \sum_c a_j^c I_i^c - b_j) + \varepsilon \sum_c {a_j^c}^2 \right) \qquad (2.6)$$

Furthermore, a^c and b can be eliminated by algebraic manipulations, yielding a quadratic cost in α alone:

$$J(\alpha) = \alpha^T L \alpha \qquad (2.7)$$

Here α is an $N\times 1$ vector, where N is the number of pixels in the unknown region. L is called *the matting Laplacian* and it is the most important analytical result of the closed form solution. It is an $N\times N$ symmetric matrix whose $(i, j)^{th}$ element is:

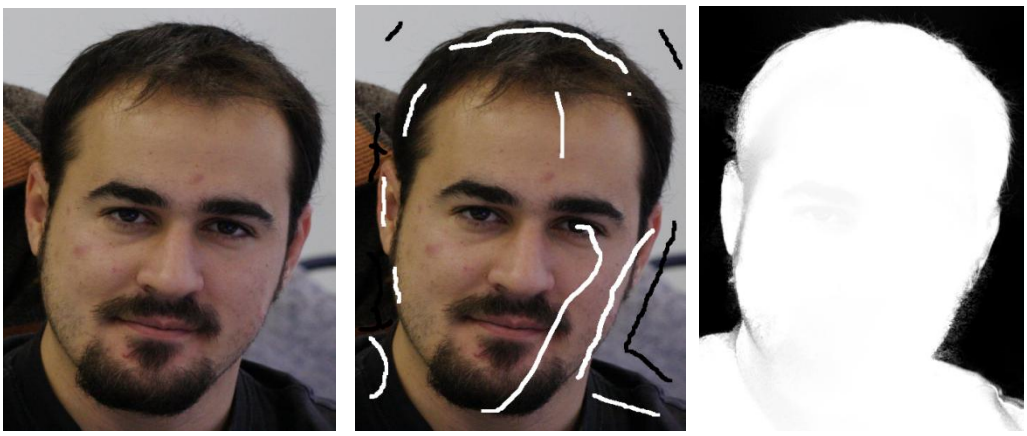
$$L(i,j)=\sum_{k|(i,j)\in w_k}(\delta_{ij}-\frac{1}{|w_k|}(1+(I_i-\mu_k)(\sum_k+\frac{\varepsilon}{|w_k|}I_3)^{-1}(I_j-\mu_k))) \quad (2.8)$$

where $\sum_k$ is a 3×3 covariance matrix, μ_k is a 3×1 mean vector of the colors in a window w_k, and I_3 is the 3×3 identity matrix. The problem of matting is now reduced to find the α that minimizes equation (2.7).

$$\alpha = argmin(\alpha^T L\alpha) \qquad ((2.9)$$

The linear system can be solved by adding the user-supplied constraints on the matte provided via a scribble-based GUI or a trimap.

We have implemented and tested this approach on some images as shown in Figure 4. The ***scribbles*** in white and black (Figure 4 b) are used to provide samples from pure background and foreground. The matting result is shown in Figure 4 (c). The closed form approach gives poor results for those regions of background trapped inside foreground. In this case, if no scribble is put in the trapped region, the result will not be accurate.



(a)Input image **(b) Scribbles** **(c) Alpha matte**

Figure 4: Closed-form Matting Example

As pointed out by Wang and Cohen [1], there are two possible drawbacks of affinity-based approaches. First, unlike sampling-based approaches, most approaches focus on first estimating alpha values, and only then estimate true foreground colors for unknown pixels based on pre-computed alphas, rather than estimating them jointly for an optimal solution. Secondly, the alpha matte is estimated in a propagation fashion, from known pixels to unknown ones, thus small errors could be propagated and accumulated to produce bigger errors. As a result, mattes generated by affinity-based approach are sometimes less accurate than those generated by sampling-based approach.

Chapter 3: Matting with Multiple Views

3.1 Introduction

As addressed in the previous chapters, matting with a single view is a severely under-constrained problem, since 7 unknown variables need to be estimated from 3 equations for every unknown pixel. Both sampling-based and affinity-based approaches are making a close guess of the unknown variables ahead of time to significantly reduce the size of the solution space. Alternative solutions propose to add more constraints by utilizing additional cameras or cameras settings.

Unfortunately, the matting systems with extra information need extra effort and cost. Some of them may require specially designed capturing devices, like in early matting system. The input image is often captured against a single or multiple constant-colored background(s), which is known as blue screen matting [13]. In this approach, knowing the background greatly reduces the difficulty for extracting an accurate matte. Other methods require capturing an object using different camera settings over multiple shots. Typical matting methods that employ different capturing settings include flash and non-flash image pairs [14], matting with camera arrays [8], and defocus matting [7].

In this chapter, we will provide a review of some classical existing image matting algorithms and systems with multiple views, and give a summary of their results.

3.2 Defocus Matting

The Defocus matting uses a multi-sensor camera to capture multiple synchronized video streams, as shown in Figure 5. Video streams from the same scene and different depth of fields can be captured by using three cameras with beam splitters.

The defocus matting employs a pinhole camera whose purpose is the creation of a large depth of field. The foreground and background sensors have large apertures, thus creating narrower depths of field. The foreground sensor produces sharp images for objects within about 0.5m of depth of the foreground object and defocuses objects farther away. The background sensor produces sharp images for objects from about 5m to infinity and defocuses the foreground object. The Defocus processing procedures as the follows:

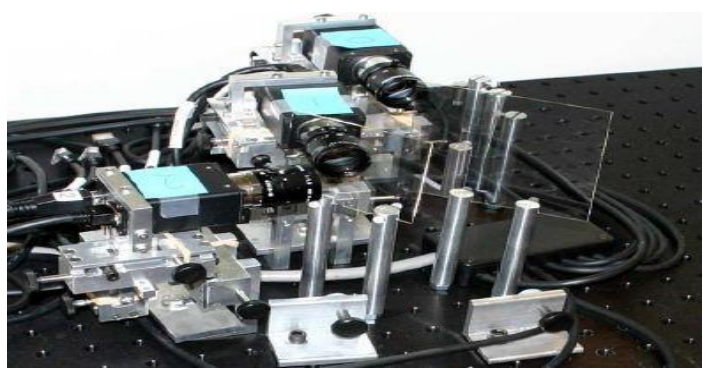
1) Capture three input video streams, I_F, I_P, I_B ;

2) For each frame:-

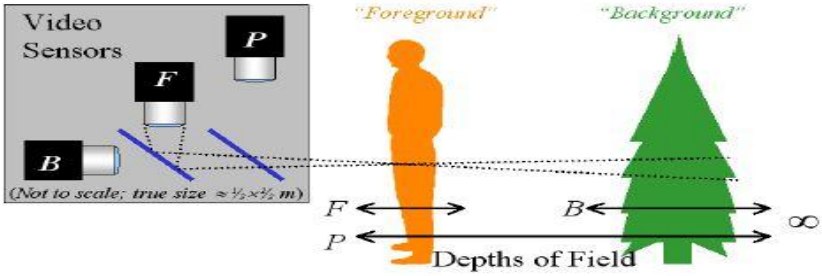
- Automatically extract trimap
- Estimate scene from trimap
- Estimate occluded B from adjacent frames

3) Repeat until error is minimized to search for α,F,B that minimize $\|C_z(\alpha,F,B,g_z,h_z) - I_z\|$, where $C_z(\alpha,F,B,g_z,h_z)$ is defocus lens composite, g_z and h_z are aperture masks of varying radius

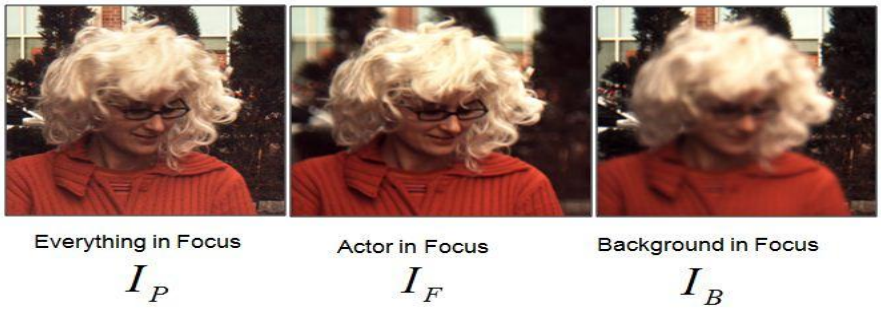
4) Post-process α



(a) The multi-parameter camera captures video streams



(b) Multi-parameter video camera



(c)Frames come from three pixel-aligned video streams with varying focus

Figure 5: Defocus Matting Principle

Because the background camera's depth of field is very large and there is no parallax between these cameras, a background with widely varying depths can still be well approximated as a plane for the purpose of matting.

Given all three streams, matting problem becomes an optimization problem. For each frame, the optical formation of the three input images I_F, I_P, I_B is expressed as a function of unknown images α , F and B using a model of the defocus blur. For each pixel there are seven unknown "scene" values α , $F_{\{r, g, b\}}$, and $B_{\{r, g, b\}}$ and nine constraint values $I_{P\{r, g, b\}}$, $I_{F\{r, g, b\}}$, $I_{B\{r, g, b\}}$ from the sensors, so the problem is over-constrained. An optimizer is used for seek to minimize an error function of the sum-squared differences between the simulated images and

the observed ones. Trimap is automatically created by using depth-from-defocus to speed up the optimizer's convergence, and choose initial values for the unknowns that are likely near the true solution. The initial foreground values F_0 are created by automatically painting known foreground colors into the unknown regions. The initial background values B_0 are created by reconstructing occluded areas from neighboring frames and then painting into never-observed areas. The initial coverage values α_0 are computed by solving the pinhole compositing equation using F_0 and B_0. Reconstructed from adjacent frames and in painted; Figure (d) matte via pinhole composition. The matting is then posed as an error minimization problme for a sigle frame of the video.

As addressed by the authors in [7], there are only 25% of the incident light that can be received by the sensors behind two beam splitters. Thus the system requires stronger illumination than does a normal camera. Also, a significant depth discontinuity between foreground and background is required that is limited to scenes where the foreground and background are visually distinguishable.

(a)Trimap from defocus **(b) Foreground "outpainted" to fill space**

(c) Background **(d) α-matte**

Figure 6: Defocus Matting Example

3.3 Matting using Camera Arrays

A matting approach using a camera array has been proposed in [8]. It can extract high quality foreground mattes from video sequences in natural video matting. The actual system is shown in Figure 7:

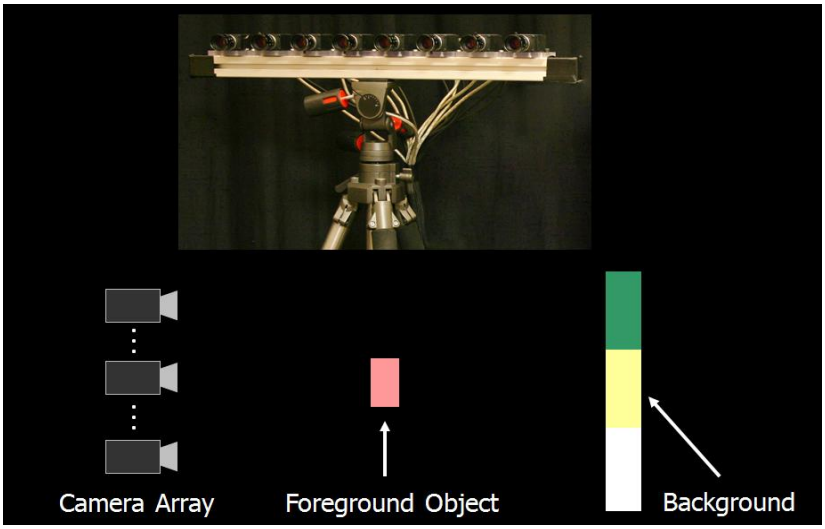


Figure 7: The Camera Array Matting System

The key idea of the system is that relative parallax in the array images can be used to separate between the foreground and background. It adds constrains to the matting problem by capturing the foreground object in front of several background colors, given a sufficiently textured background. The algorithm proceeds as follow:

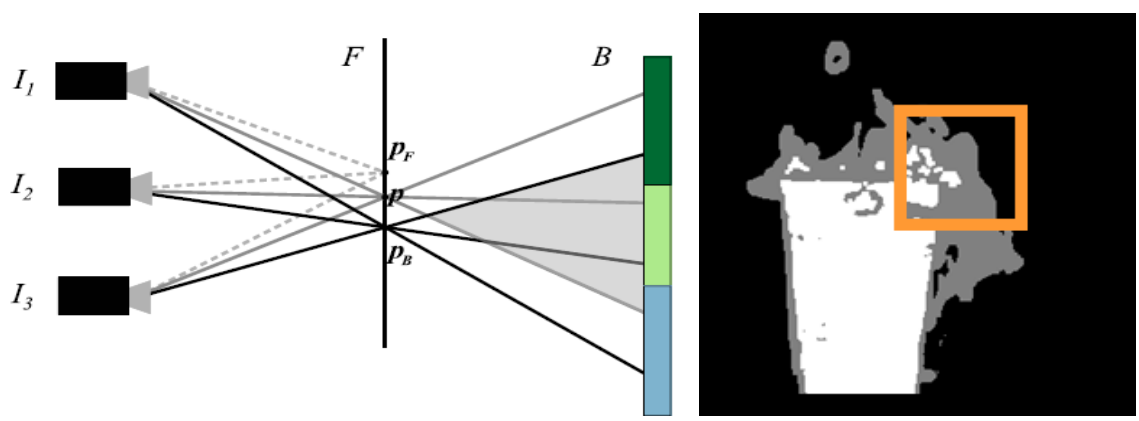
1) Automatically find the dominant foreground depth plane and then perform a local depth search to account for non-planarity
2) Compute *var(I)* and *mean(I)* of color values from each camera projected to their corresponding depths
3) Threshold the variance *var(I)* to get trimap: aotomatically compute a trimap based on variance

The trimap is constructed by filtering *var(I)* with a 9×9 pixel median filter to smooth out small fluctuations due to noise. Then using a relatively standard approach of erosion and dilation combined with double-thresholding to create a trimap with conservatively wide

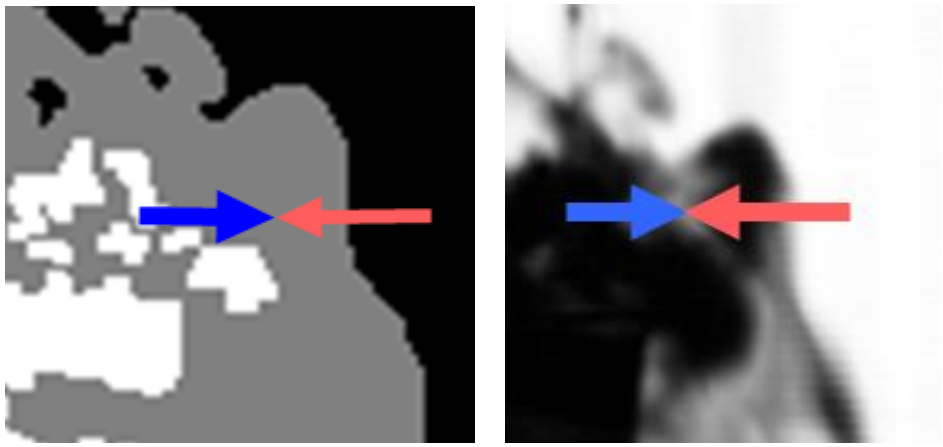
unknown regions. Low variance is considered definitely foreground (*Foreground = Low var(I)*), the white region shown Figure 8, while high variance is labeled definitely background (*Background = High var(I)*), the black region, and the unknown region contains the remaining pixels.

4) Estimate *var(F)*, *var(B)*: propagate the variance from the background and foreground regions to the unknown region; Propagate the *mean(B)* from the background to the unknown region
5) Compute α and F

Given n images of a scene, the following matting equation of a given scene point p: $I_i(p) = \alpha(p)F_i(p) + (1-\alpha(p))B_i(p)$, where $I_i(p)$ corresponds to the intensity of point p recorded in image i. $F_i(p)$ and $B_i(p)$ are the foreground and background values that, as a function of the transparency $\alpha(p)$, are mixed to give $I_i(p)$. The transparency of the point is view-independent and hence α is fixed across all images.



(a) Propagating image statistics **(b) Propagating Variance and Mean**



(c) Propagating Variance and Mean

Figure 8: Camera Arrays Matting Computation

For each pixel in the unknown region, the arrows refer to the nearest pixels of background and foreground.

As shown in Figure 8 (a), the three cameras on the left I_1, I_2 and I_3 observe a foreground object F and a background object B. The point p has a trimap label of unknown, and p_F and p_B are the points nearest to p labeled, respectively, foreground and background. To solve for α and αF, the measurements var$(I(p))$,var$(F(p))$, var$(B(p))$, mean$(I(p))$, and mean$(B(p))$; However, just only var$(I(p))$ and mean$(I(p))$ can be measured. The var$(F(p))$, var$(B(p))$, and mean$(B(p))$ need be estimated from the measurements at p_F and p_B. Note that rays going through the point's p and p_B hit overlapping regions (indicated by the shaded triangle as shown in Figure 8-(a) on the background and, as a result, the mean and variance of the two points should indeed be similar. For each pixel in the unknown region, the author estimate the variance of F from the nearest pixel in the labeled foreground region and estimate the variance and mean of B using the mean and variance from the nearest pixel in the background region, as shown in Figure 8-(c):

$$\begin{cases} \mathrm{var}(F(P)) \approx \mathrm{var}(I(p_F)) \\ \mathrm{var}(B(P)) \approx \mathrm{var}(I(p_B)) \\ mean(B(P)) \approx mean(I(p_B)) \end{cases} \qquad (3.1)$$

where $I(P_F)$ is the nearest pixel in the foreground; $I(P_B)$ is the nearest pixel in the background.

For one frame image of ith ($i \in [1,n]$) camera captures the sample video, I, F and B are all treated as sampling the random variables $\{I_i(p)\}_{i=1}^{n}$, $\{F_i(p)\}_{i=1}^{n}$ and $\{B_i(p)\}_{i=1}^{n}$. Taking the variances to the matting equation $I_i = \alpha F_i + (1-\alpha)B_i$ gives

$$\mathrm{var}(I) = \mathrm{var}[\alpha F + (1-\alpha)] = \alpha^2\,\mathrm{var}(F) + (1-\alpha)^2\,\mathrm{var}(B) \qquad (3.2)$$

Get a quadratic equation in α via algebraic manipulation, shown as equation 3.3:

$$\alpha^2[\mathrm{var}(F) + \mathrm{var}(B)] - 2\,\mathrm{var}(B)\alpha + [\mathrm{var}(B) - \mathrm{var}(I)] = 0 \qquad (3.3)$$

by assuming that F and B are statictically independent. The solutions to the quadradic equation are:

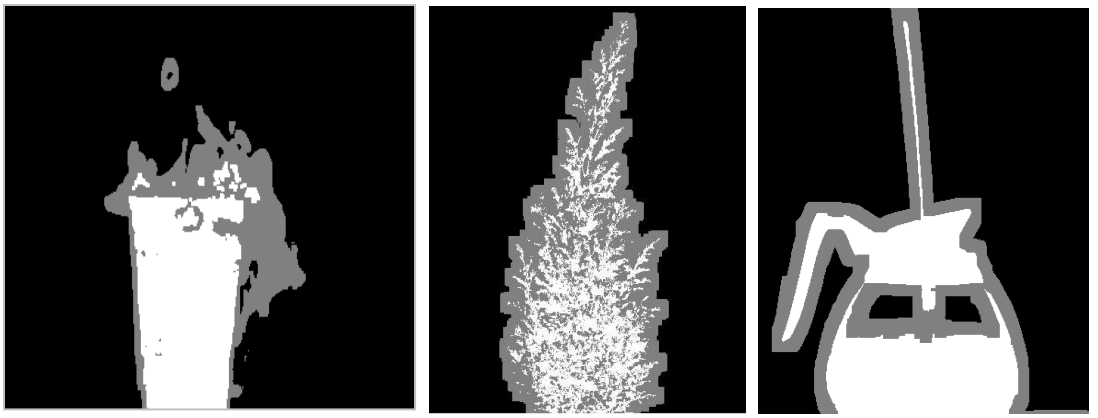
$$\alpha = \frac{\mathrm{var}(B) \pm \sqrt{\Delta}}{\mathrm{var}(F) + \mathrm{var}(B)} \qquad (3.4) \qquad \text{where}$$

$$\Delta = \mathrm{var}(I)[\mathrm{var}(F) + \mathrm{var}(B)] - \mathrm{var}(F)\,\mathrm{var}(B) \qquad (3.5)$$

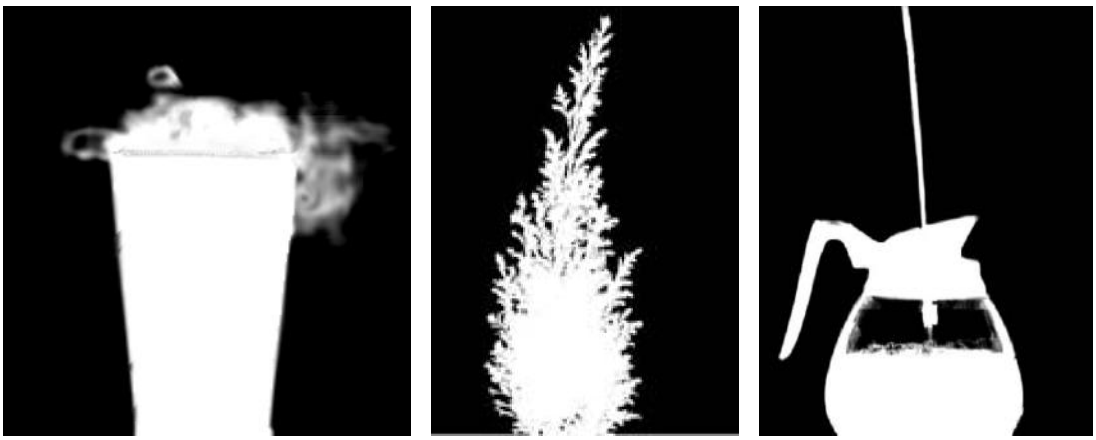
Thus, provides two algebraic sulitons; however, when $\mathrm{var}(B) >> \mathrm{var}(F)$, as is the case in practice, one of the solutions is often greater than 1 making it inconsistent with the constraint that $\alpha \in [0,1]$, thus needs to be discarded. In case that both are valid, the average is taken as the solution. Algebraic analysis reveals all possible cases and the final solution can be summarized as follows:

$$\alpha = \begin{cases} 0 & \mathrm{var}(I) > \max(\mathrm{var}(B), \mathrm{var}(F)) \\ \dfrac{\mathrm{var}(B) + \sqrt{\Delta}}{\mathrm{var}(B) + \mathrm{var}(F)} & \mathrm{var}(B) < \mathrm{var}(I) \le \mathrm{var}(F) \\ \dfrac{\mathrm{var}(B) - \sqrt{\Delta}}{\mathrm{var}(B) + \mathrm{var}(F)} & \mathrm{var}(F) < \mathrm{var}(I) \le \mathrm{var}(B) \\ \dfrac{\mathrm{var}(B)}{\mathrm{var}(B) + \mathrm{var}(F)} & \dfrac{\mathrm{var}(B)\,\mathrm{var}(F)}{\mathrm{var}(B) + \mathrm{var}(F)} \le \mathrm{var}(I) \le \min(\mathrm{var}(F), \mathrm{var}(B)) \\ 1 & \mathrm{var}(I) < \dfrac{\mathrm{var}(B)\,\mathrm{var}(F)}{\mathrm{var}(B) + \mathrm{var}(F)} \end{cases} \tag{3.6}$$

The first and last lines of equation 3.6 represent the cases in which no valid solutions exist; Lines 2 and 3 represent the case in which one of the solutions is outside the range [0,1], and can be safely chosen the other solution. Line 4 is when both solutions are valid, need set α to be their average. Note that two discontinuities occur. One occurs when the value of var(I) switches the solution from lines 2 or 3 to line 4, and the second is from line 4 to 5. These are due to switching from an exact solution to an approximate solution. For both solutions the discontinuity is no more than $\frac{\mathrm{var}(F)}{\mathrm{var}(B) + \mathrm{var}(F)} = \frac{1}{1+c}$, where c is defined as the ratio between background and foreground variances, i.e., *var(B) = c*var(F)*. To summarize, for estimate mean and variance statics uses equation 3.1 and compute α using equation 3.6. Discontinuities will occur when switching from one solution to another, but will be small enough if *var(B) >> var(F)*.



(a)



(b)

Figure 9: Trimaps

Figure 9 shows some trimaps and alpha mattes extracted by the cameras arrays system. Obviously, it is so efficient that can handle not only hair and trees, but also transparent objects such asfluide and smoke, which is nearly impossible to achieve using the defocus matting system to extract mattes in nearly realtime. However, as addressed in the matting survey[1], this approach suffers from some limitations. The alpha matte is assumed to be fixed and not view-dependent. However, this is not true for some objects/meterials which present self-occlusion. Also, the variance of the background is assumed to be several orders of magnitude

larger than that of the foreground, which may not be the case. Finally, the system is specifically designed for indoor scenes with controllable backgrounds, thus it is hard to extend to general outdoor scenes.

Chapter 4: Image Matting Using IR & Color Cues

4.1 Overview

In our matting approach, we focus on scenes with human actors. The foreground is the actor that should be put in another context by switching the background. For that, we make use of a second imaging device that is an Infrared (IR) camera. The information from the infrared camera is combined with the one from the color camera in order to facilitate the matting and increase its accuracy. The infrared image allows a rapid extraction of the human actor from the background; whereas, the color image provides the additional information that will serve to sharply define the contours of the actor.

In this chapter, we will propose three solutions to calculate the alpha-matte. The IR and color image histograms are modeled as *p.d.fs* (Probability Density Functions) in the first solution. In the second and the third ones, we derive fuzzy membership functions that are combined to calculate the alpha-matte.

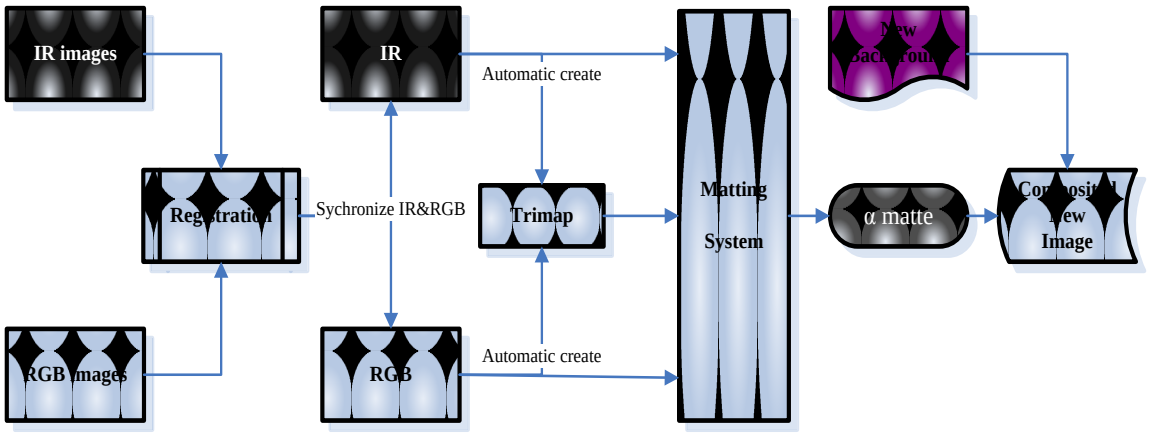


Figure 10: Our matting method process flow diagram

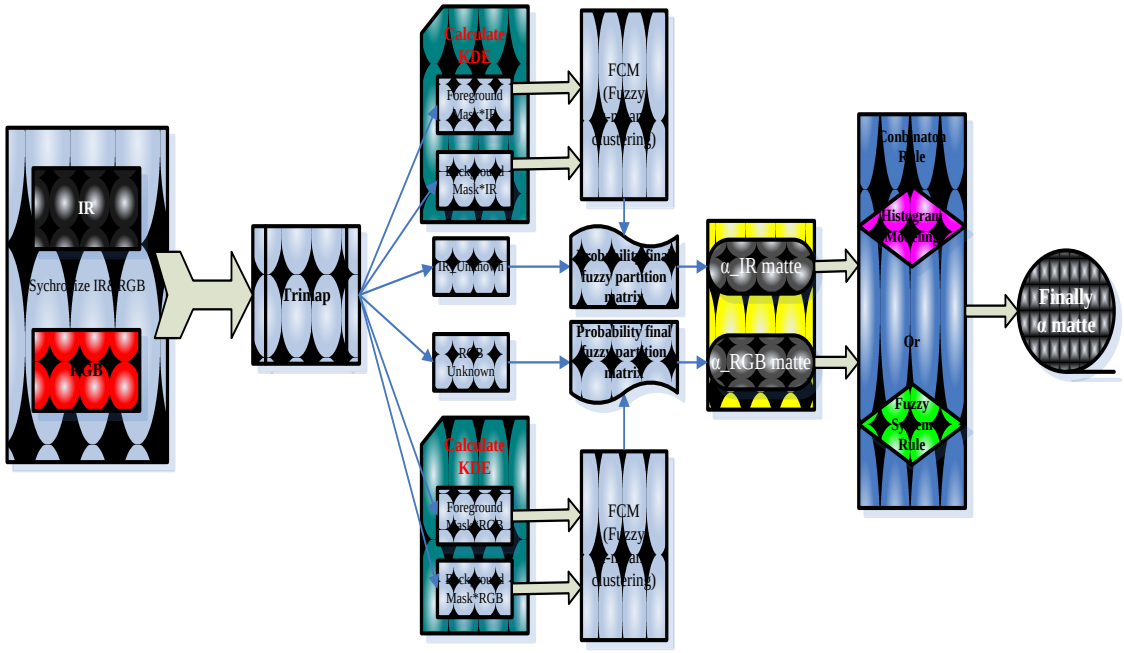


Figure 11: Matting system module process flow diagram

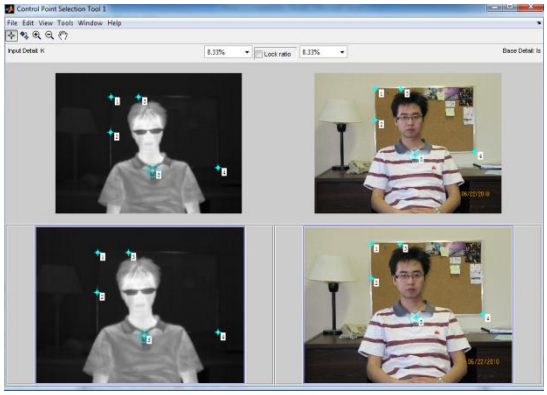
Figure 10 is an overview of our matting method and figure 11 shows the processing details of our matting system.

4.1.1 Registration

The cameras used in our system might have different fields of view angles and lenses. The resulting images will differ in size and scale (see figure 12-(*a*) and 12-(*b*). Therefore, an image registration [17, 18, 19] process is a necessary step.

In our experiments, the registration step is carried out using Matlab. Indeed, Matlab provides a graphical tool to set control points in the IR and color images as shown in figure 12-(c). Then, the control points are paired and used to infer a 2D spatial transformation [20]. This transformation is applied on the IR image to match it with the color image. Figure 12-(d) shows the registration result.

(a) Color image 4000 by 3000 **(b) Infrared image 320 by 240**



(c) Control point selection **(d) Registration result 4000**

Figure 12: A Registration Example Using Matlab

4.1.2 Trimap Construction

As we mentioned previously, one of the important factors that impacts the performance of a matting algorithm is the accuracy of the trimap. Ideally, the unknown region in the trimap should only cover truly mixed pixels. In other words, the unknown region around the foreground boundary should be as thin as possible to achieve the best possible matting results.

The methods proposed recently in the literature require a significant amount of user effort and are often undesirable in practice, especially for objects with large semitransparent regions

or holes. In our work, we used the algorithm proposed by Hui [22] and which constructs the unknown region iteratively as follows:

Let Ω_F^n and Ω_U^n be the foreground and unknown regions at iteration ***n***. We define $\Omega_{F-U}^n = \Omega_F^n \cup \Omega_U^n$. At iteration ***n+1***:

$$\Omega_U^{n+1} = \Omega_U^n \cup \partial\Omega_{F-U}^n \qquad (4.1)$$

where $\partial\Omega_{F-U}^n$ is the set of pixels that are added to the unknown region. If $x \in \partial\Omega_{F-U}^n$ then:

1) $x \notin \Omega_{F-U}^n$ and

2) $IR(x) > T_1$ and

3) $\exists z \in \left(\Omega_{F-U}^n \cap N_8(x)\right) | \left|IR(z) - IR(x)\right| < T_2$

Where $N_8(\cdot)$ is the set of 8-Neighbors $\boldsymbol{T_1}$ and $\boldsymbol{T_2}$ are thresholds set by the user. The algorithm described above is applied on some sample images and the results are given in figure 13.

4.1.3 Foreground and Background Histogram Modeling

As a result of trimap construction, the image domain is split into three separate regions: the foreground $\boldsymbol{\Omega_f}$, the background $\boldsymbol{\Omega_b}$ and the unknown region $\boldsymbol{\Omega_u}$ (See Figure 13).

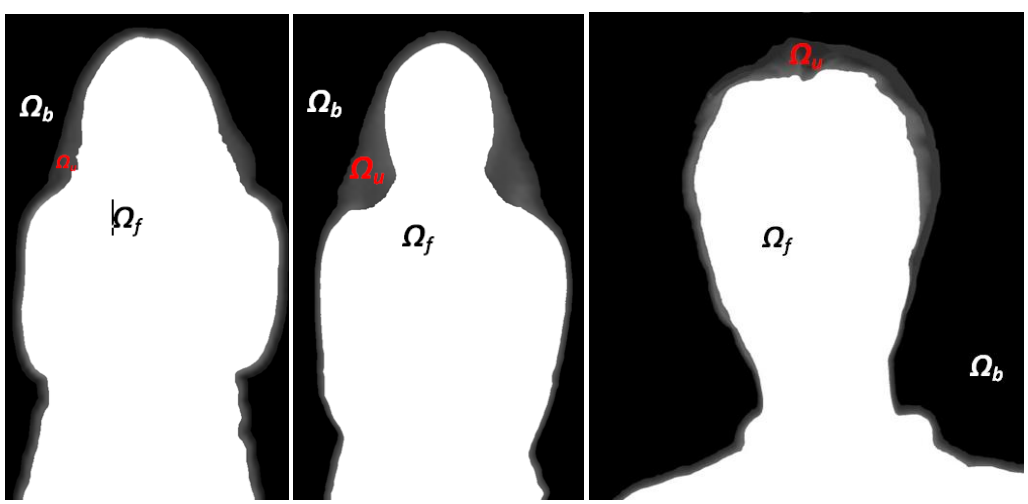
The color and infrared distributions in $\boldsymbol{\Omega_f}$ and $\boldsymbol{\Omega_b}$ define the patterns of the foreground and the background objects. Distances to these distributions could be used to estimate α in $\boldsymbol{\Omega_u}$. The *p.d.fs* of the background and foreground regions in the color and infrared images are derived using Kernel Density Estimation (KDE) [23]. For a given set of data points (x_1, x_2, …, x_n), the KDE defines a *p.d.f* as:

$$f_h(x) = \frac{1}{nh}\sum_{i=1}^{n} K(\frac{x - x_i}{h}) \qquad (4.2)$$

where h is a scale parameter and $K(\cdot)$ is a Kernel function. In our case, we used a Gaussian kernel:

$$K(x) = e^{-\frac{x^2}{2}}, \forall x \in IR \qquad (4.3)$$

(a) The initial images



(b) Automatic construct trimap results from IR images

Figure 13: Trimaps Of The Test Images

The use of the KDE allows to smooth the noisy histograms and to model efficiently multimodal distributions as it is the case in color and infrared histograms. The resulting *p.d.fs* are denoted $f_F^{IR}(\cdot)$, $f_F^{C}(\cdot)$, $f_B^{IR}(\cdot)$ and $f_B^{C}(\cdot)$.

Figure 14 shows an example of foreground and background histograms and the resulting *p.d.fs* obtained with Kernel Density Estimation. Except for the background in the infrared image which is uniform, all the other histograms are multimodal. Indeed, the actor's face and clothes contain several colors. The body's temperature also differs between the skin, the hair, the clothes and the glasses.

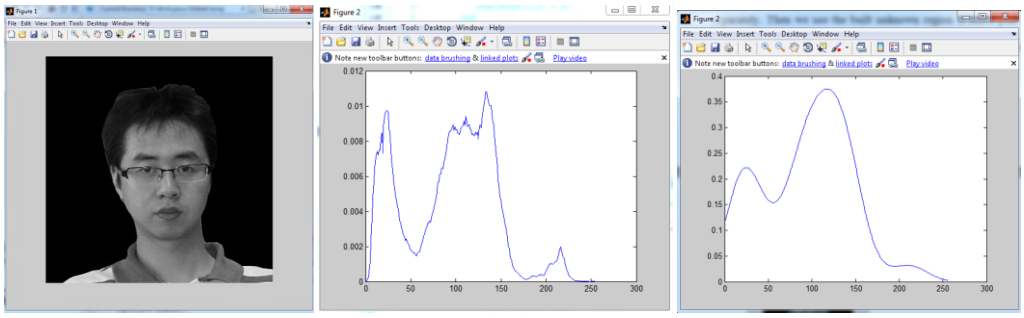
At this stage, it is possible to extract an initial estimate of α. Indeed, for a given pixel in $\mathbf{\Omega_u}$, it is possible to evaluate its probability to belong to the foreground or to the background, of the color image or the infrared image. Let $x \in \Omega_u$ be a pixel whose α value is unkown. We define the value of α according to the color image as:

$$\alpha_{color} = \frac{f_F^C(x)}{f_F^C(x) + f_B^C(x)} \tag{4.4}$$

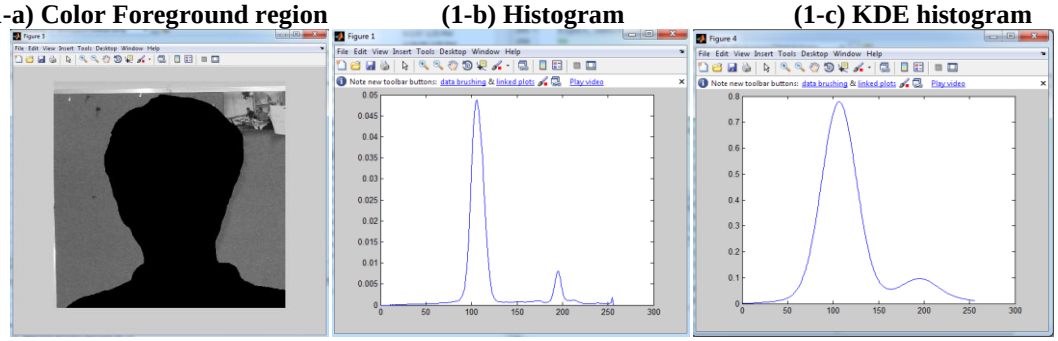
Similarly, α according to the infrared image is :

$$\alpha_{IR} = \frac{f_F^{IR}(x)}{f_F^{IR}(x) + f_B^{IR}(x)} \tag{4.5}$$

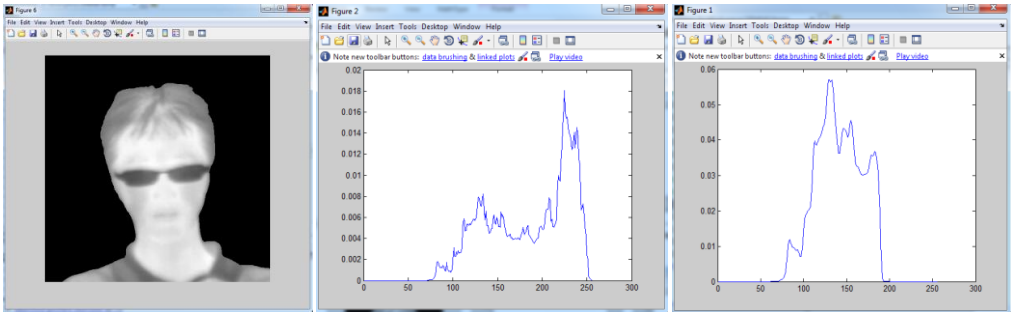
In equation (4.4) and (4.5), if the *p.d.f* value of the foreground is very large in comparison with the *p.d.f* value of the background then α tends to 1. In the inverse case, α tends to zero. If the pixel has the same probability of belonging to the foreground and to the background then $\alpha = 0.5$.



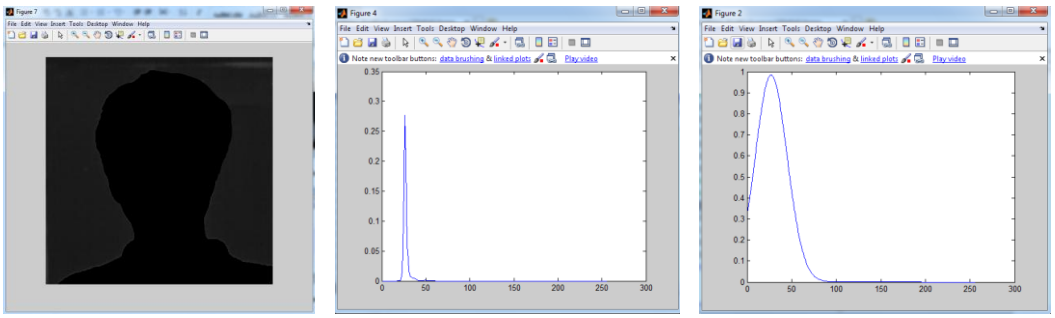
(1-a) Color Foreground region (1-b) Histogram (1-c) KDE histogram



(2-a) Color Background region (2-b) Histogram (2-c) KDE histogram

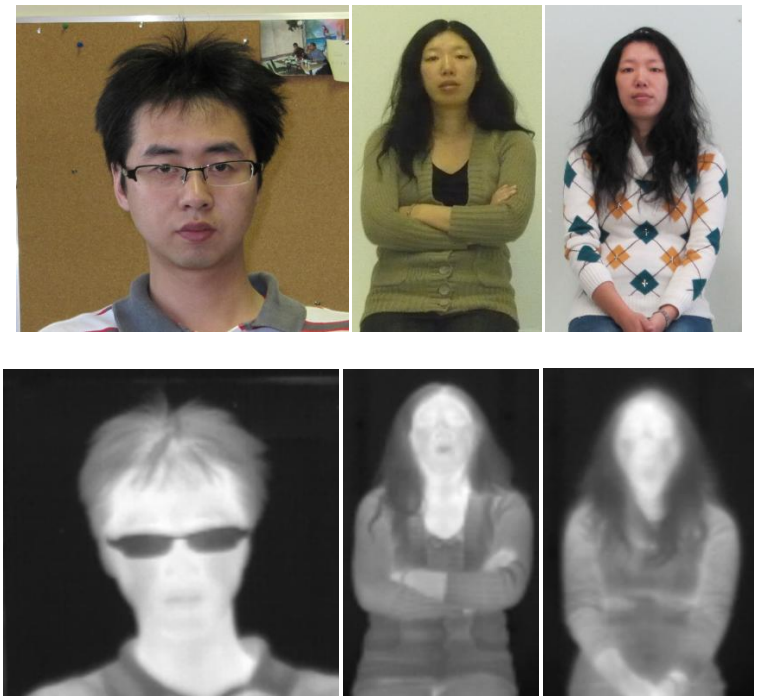


(3-a) IR foreground region (3-b) Histogram (3-c) KDE histogram



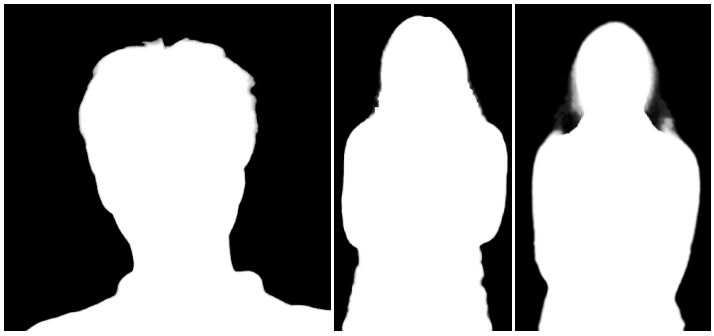
(4-a) IR Background region (4-b) Histogram (4-c) KDE histogram

Figure 14: Histogram Modeling In Color And IR Images



(a) The test original IR and RGB images

(b) α-matte results of RGB with KDE



(c) α-matte results of IR with KDE

Figure 15: Matting Results

This technique has been applied on same image pairs and the results are given in Figure 15. The α-matte extracted from the IR image has a sharp boundary (see Figure 15-(c)) in area areas where the transition between the foreground and the background is supposed to be smooth. The transparency near the hair area is not reflected in the resulting α-matte.The α-matte of the hair seems to capture the whole area; but, it misses the fine details. Also, the value of α in some dense hair areas is under estimated, because of its low heat emission.

The α-matte extracted from the color image has a better structure in the hair area where long fine structures are more visible than in IR(see Figure 14-(b)). Howerer, the α-value decrease, in a abrupt way specialy for the hair area. In some regions, a band with intermediate α-values appear at the borders of the foreground. In this band, $f_F^C(\cdot) \approx f_B^C(\cdot) \approx 0$, which gives a value of $\alpha \approx 0.5$. For these pixels the value of α should be higher; but their $f_F^C(\cdot)$ is small. This is due to the fact that the pixel belongs to a cluster whose prior probability in the overall image is low. This is the case of the hair whose area is small in comparison with the total area of the foreground. As a result, $f_F^C(\cdot)$ is small and since $f_B^C(\cdot)$ is also small then $\alpha \approx 0.5$. To improve the estimation of α using color and infrared, it is clear that we should:

Instead of using directly the *p.d.f s*, where the prior probability is already integrated, we define a membership measure that is not impacted by the size of the foreground and background components;

Use both infrared and color to estimate α and not each image alone.

To address 1), we propose to use the concept of fuzzy membership functions instead of the *p.d.fs*.

4.1.4 Fuzzy C-Mean Clustering

Fuzzy c-means (FCM) [24] is a method of clustering which allows one piece of data to belong to two or more clusters. In fuzzy clustering, each point has a degree of membership to multiple clusters, rather than belonging completely to just one cluster. Let $X=\{x_1,\ldots,x_M\}$ be a finite unlabeled data set and $\boldsymbol{C}$ be the number of clusters; $2\leq C\leq M$ and let the set of all real $C\times M$ matrices. A fuzzy C-Partition of X is represented by a matrix $U=[\mu_{ik}]\in R^{C\times M}$, where $\mu_{ik}=\mu_i(x_k)$ represents the degree of membership of $\boldsymbol{x_k}$ to cluster $\boldsymbol{i}$, and verifies the following constraints:

$$\begin{aligned} &\mu_{ik}\in[0,1] && 1\leq i\leq C;\ 1\leq k\leq M \\ &\sum_{i=1}^{C}\mu_{ik}=1 && 1\leq k\leq M \\ &\sum_{k=1}^{M}\mu_{ik}>0 && 1\leq i\leq C \end{aligned} \tag{4.6}$$

U is used to describe the clusters of X, and a good partition U of X is yielded by the minimization of the *FCM* objective functional [25]:

$$J_q(U,V:X)=\sum_{k=1}^{M}\sum_{i=1}^{C}(\mu_{ik})^q\,\|x_k-v_i\|_A^2 \tag{4.7}$$

where $q\in[1,+\infty[$ is a weighting exponent called the fuzzifier, and $V=(v_1,\ldots,v_C)$ the vector of the cluster centers. $\|x\|_A=\sqrt{x^TAx}$ is any inner product norm where A is any positive definite matrix. Approximate optimization of by the *FCM* algorithm is based on iteration through the following necessary conditions for its local extrema:

FCM Theorem ([25]): Assume $q \geq 1$ and $\|x_k - v_i\|_A^2 > 0$, $1 \leq i \leq C$, $1 \leq k \leq M$. (U, V) may minimize J_q only if:

$$\mu_{ik} = \left[\sum_{j=1}^{C} \left(\frac{\|x_k - v_i\|_A}{\|x_k - v_j\|_A} \right)^{\frac{2}{q-1}} \right]^{-1} \tag{4.8}$$

$$v_i = \frac{\sum_{k=1}^{M} (\mu_{ik})^q \cdot x_k}{\sum_{k=1}^{M} (\mu_{ik})^q} \tag{4.9}$$

The FCM algorithm consists of iterations alternating between Equations (4.6) and (4.7). This algorithm converges to either a local minimum or a saddle point of J_q.

In the context of gray level image segmentation, a fast version of the FCM can be used . The proposed algorithm is based on one-dimensional attribute such as the gray level. Let H be the histogram of image of L-levels, where L is the number of gray levels. Each pixel, l, has a feature that lies in the discrete set:

$$X = \{0, 1, ..., L-1\} \tag{4.10}$$

In this case relation (4.5) can be written as follows:

$$J_q(U, V : L) = \sum_{l=0}^{L-1} \sum_{i=1}^{C} (\mu_{il})^q \cdot H(l) \cdot (l - v_i)^2 \tag{4.11}$$

where $\mu_{il} = \mu_i(l)$ represents the membership degree of gray level ***l*** to cluster ***i***.

Thus, the computation of membership degrees of $H(l)$ pixels is reduced to that of pixel with ***l*** as gray level value. The algorithm is outlined in the following steps:

Step1) Select the number of clusters *C*, *2≤C≤L*, the termination criteria $\varepsilon > 0$, the value of *q>1* and initialize the partition matrix U according to:

$$\sum_{i=1}^{C} \mu_{il} = 1 \qquad l = 0,1,...,L-1$$

Step2) Compute the prototypes of the clusters and update the partition *U* as follows:

$$v_i = \frac{\sum_{l=0}^{L-1} (\mu_{il})^q \cdot H(l) \cdot l}{\sum_{l=0}^{L-1} (\mu_{il})^q} \quad i = 1,...,C \qquad (4.12)$$

$$\tilde{\mu}_{il} = \left[\sum_{j=1}^{C} \left(\frac{(l - v_i)}{l - v_j} \right)^{\frac{2}{q-1}} \right]^{-1} \qquad (4.13)$$

Step3) $E = \sum_{i=1}^{C} \sum_{l=0}^{L-1} \left| \tilde{\mu}_{il} - \mu_{il} \right|$

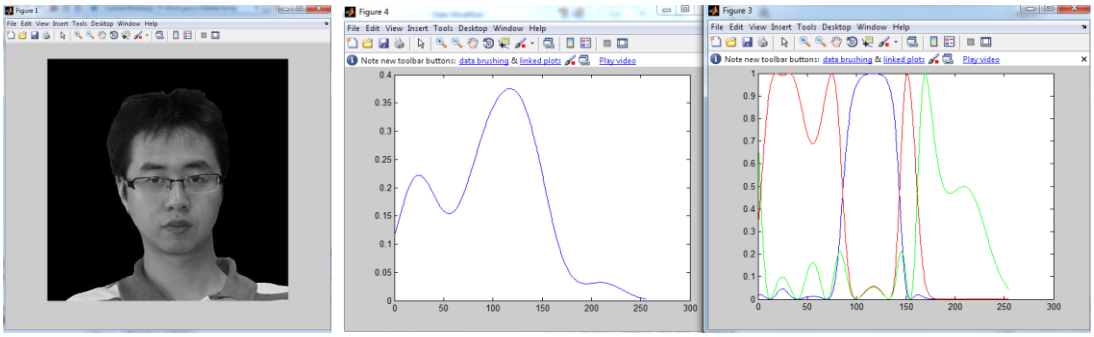
Step 4) If ($E < \varepsilon$) stop. Iterator Step2)

The FCM clustering technique is applied on the histograms of the foreground and the background components in color and infrared images. Each histogram is decomposed into a number of clusters, which are described by a fuzzy membership function. Figures 16-(a) and 16-(b) show the histograms and the resulting fuzzy memberships.

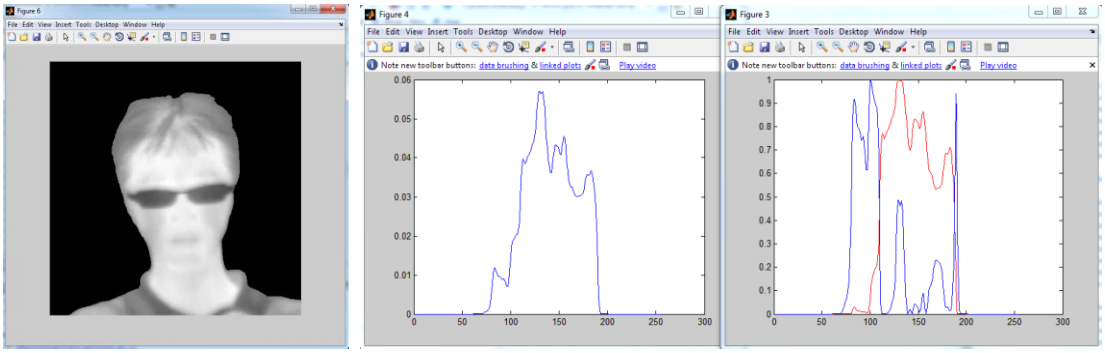
In Figure16-(a), we can see clearly that the hair part in the KDE histogram, which is the part with gray level below 50, has a low probability. But we know this part belongs to the foreground. It is supposed to have higher probability value to reflect its membership to the foreground (see Figure 16-(c)). This problem is caused by the KDE which derives the *p.d.f.*

using the prior probability that is related to the size of population. Indeed, the hair component has a small population, compared to skin and clothes components in the foreground region.

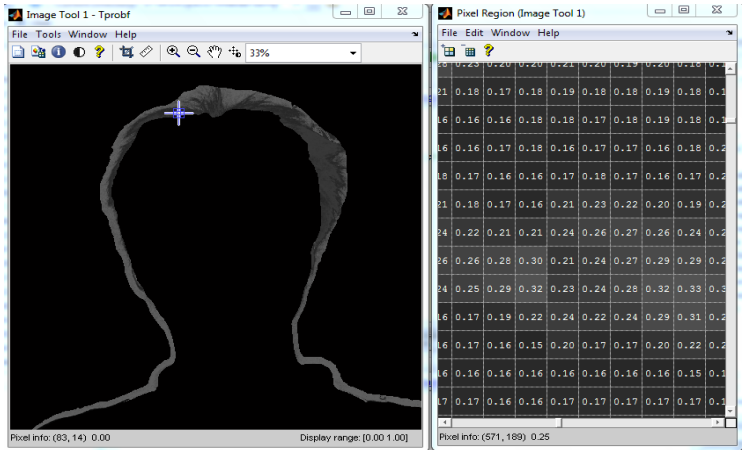
The fuzzy clustering-means (FCM) is applied to decompose the KDE histogram (see Figure 16-(d)). The number of clusters depends on the test image. In this case, we chose 3 clusters for color, and 2 clusters for the infrared image.



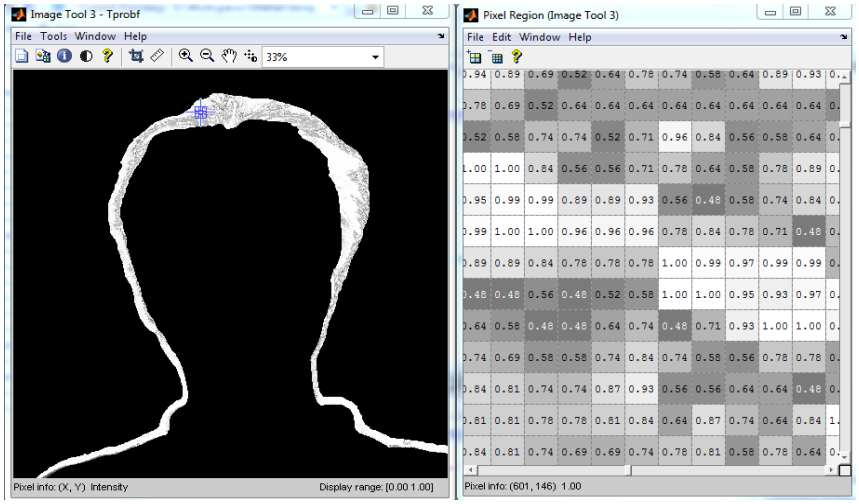
(a) Original RGB image-foreground and the memberships is compare with the KDE



(b) Original infrared image-foreground and the memberships is compare with the KDE



(c) The computed foreground probability of unknown region with the KDE



(d) The computed foreground probability of unknown region with fuzzy memberships

Figure 16: The Resulting Fuzzy Memberships

For a given $x \in \Omega_u$, $\mu_{F,i}^{C}(x)$ and $\mu_{B,i}^{C}(x)$ respectively refer to the ***i***th fuzzy membership function of the foreground and the background component in the color image. Similarly, we denote $\mu_{F,i}^{IR}(x)$ and $\mu_{B,i}^{IR}(x)$ as the ***i***th fuzzy memberships of the foreground and the background components of the infrared image.

We define the *p.d.f.s* of the ith components of the foreground component in the color and IR images as follows (see figure16):

$$f_{F,i}^{C}(x) = \mu_{F,i}^{C}(x) \times f_{F}^{C}(x) \tag{4.14}$$

$$f_{F,i}^{IR}(x) = \mu_{F,i}^{IR}(x) \times f_{F}^{IR}(x) \tag{4.15}$$

where $\mu_{F,i}^{C}(x)$ and $\mu_{F,i}^{IR}(x)$ are used to extract the ith cluster from the color and IR *p.d.f.s*. Once this step is done, $f_{F,i}^{C}(x)$ and $f_{F,i}^{IR}(x)$ can be normalized according to their priors (population size) as follows:

$$f_{F,i}^{C}(x) = \frac{f_{F,i}^{C}(x)}{\sum_{\forall x} f_{F,i}^{C}(x)} \tag{4.16}$$

$$f_{F,i}^{IR}(x) = \frac{f_{F,i}^{IR}(x)}{\sum_{\forall x} f_{F,i}^{IR}(x)} \tag{4.17}$$

Equations (4.16) and (4.17) measure the membership of x to the ith component of the foreground. Therefore, the membership to the whole foreground can be computed by:

$$M_{F}^{C}(x) = \max_{i}\left(f_{F,i}^{C}(x)\right) \tag{4.18}$$

$$M_{F}^{IR}(x) = \max_{i}\left(f_{F,i}^{IR}(x)\right) \tag{4.19}$$

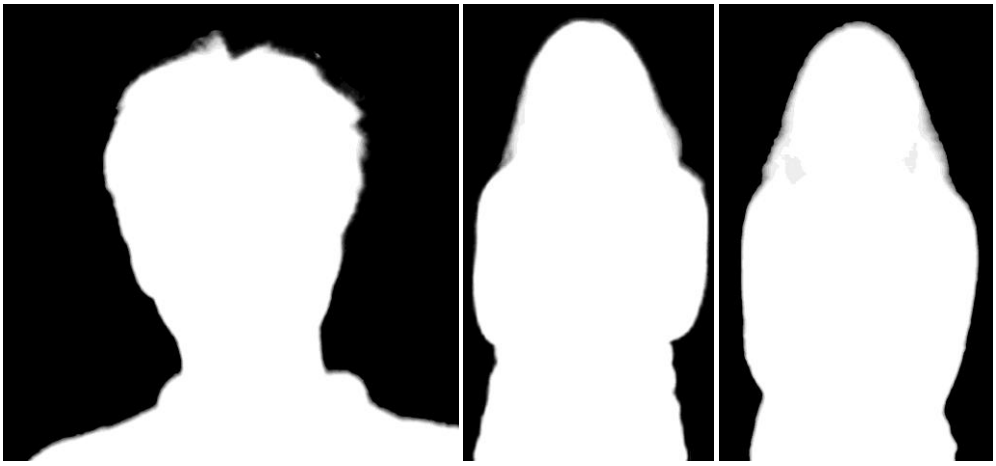
Finally, the α-matte from the color and the IR images can be derived as follows:

$$\alpha_{C}(x) = \frac{M_{F}^{C}(x)}{M_{F}^{C}(x) + f_{B}^{C}(x)} \tag{4.20}$$

$$\alpha_{IR}(x) = \frac{M_{F}^{IR}(x)}{M_{F}^{IR}(x) + f_{B}^{IR}(x)} \tag{4.21}$$

Figure 17 shows α-mattes computed with KDE and the fuzzy membership functions in (4.20) and (4.21).

(a) Matting results of RGB with fuzzy membership improved KDE



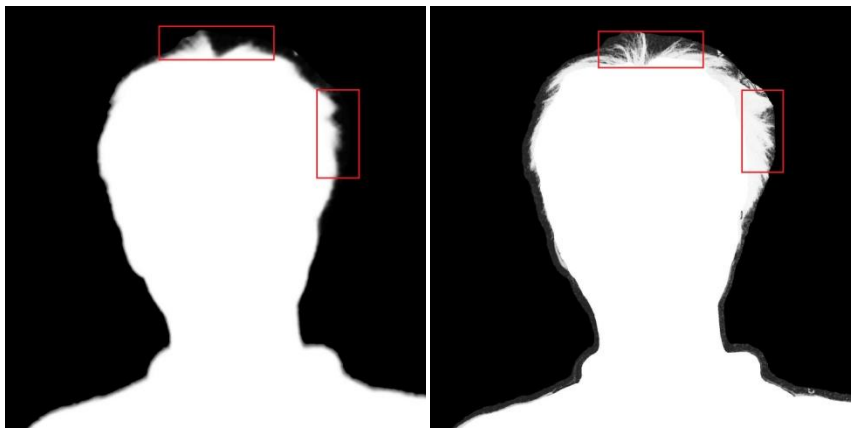
(b) Matting results of IR with fuzzy membership improved KDE

Figure 17: Matting Results

In comparison with the results obtained with the *p.d.f.s*, the use of fuzzy membership function improves significantly the matting for both IR and color image. In the rest of this chapter, we will propose different approaches to combine the color and IR cues to build a unified α-matte that integrates the advantages of both imaging modalities.

4.2 Combination Using Decision Rules

The simplest way to combine the α-mattes obtained from the IR and the color inputs is to use a set of decision rules. These rules must take into consideration the cases where both inputs agree or disagree on the membership of a given pixel. Ambiguity between IR and color could be seen in Figure 18, where α_{IR} is much larger than α_C for the same pixels.



α_{IR} α_C

Figure 18: A Case of Ambiguity Between IR and Color Sensors

For a given $x \in \Omega_U$, $\alpha_{IR}(x)$ and $\alpha_C(x)$, the first two rules could be written as follows:

R1: if $\alpha_{IR}(x) = \alpha_C(x) = 0$ then $\alpha(x) = 0$

R2: if $\alpha_{IR}(x) = \alpha_C(x) = 1$ then $\alpha(x) = 1$

R1 and *R2* address the case where both IR and color agree that $x \in \Omega_B$ or $x \in \Omega_F$. For mixed pixels, the following rule is used:

R3: if $\alpha_{IR}(x) \neq \alpha_C(x)$ then $\alpha(x) = \dfrac{2\alpha_{IR}(x) \cdot \alpha_C(x)}{\alpha_{IR}(x) + \alpha_C(x)}$

In *R3*, if $\alpha_{IR} \approx \alpha_C \approx \alpha$, then $\alpha(x) \approx \alpha$. Thus, *R3* handles well the case where color and IR give the same assessment regarding membership of *x*. If the ambiguity between the two sensors increase, then $|\alpha_{IR}(x) - \alpha_C(x)| \approx 1$. In this case, $\alpha(x) = 0$. This means that *R3* sets $\alpha(x)$ to zero if there is a significant contradiction between color and infrared. This situation appears in two cases: 1) a pixel in the background has the same color as the foreground but there is no emitted heat; 2) a pixel from the background emits heat but has a different color from the foreground.

For the intermediate cases, *R3* provides $\alpha(x) \in [0,1]$ that behaves as shown in Figure 19.

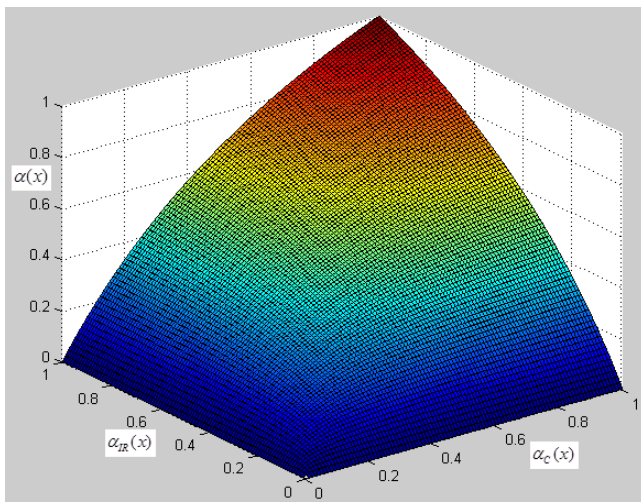


Figure 19 Combination Rule R3

The decision list formed by *R1*, *R2* and *R3* has been applied on the test image pairs. The results are given in Figure 20. $\alpha_C(x)$ and $\alpha_{IR}(x)$ in *R1*, *R2* and *R3* were derived using equations (4.20) and (4.21). As we can see from these results our decision rules combine efficiently the input α-mattes. The area with false α-values in the color α-matte is removed; however, the result lacks some smoothness. This is due to the shape of the combination function in *R3* which has a pseudo binary shape that lacks intermediate values. Additional

rules could be added to improve the results. This will be dealt with in the next section where a fuzzy inference system is used to combine $\alpha_C(x)$ and $\alpha_{IR}(x)$.

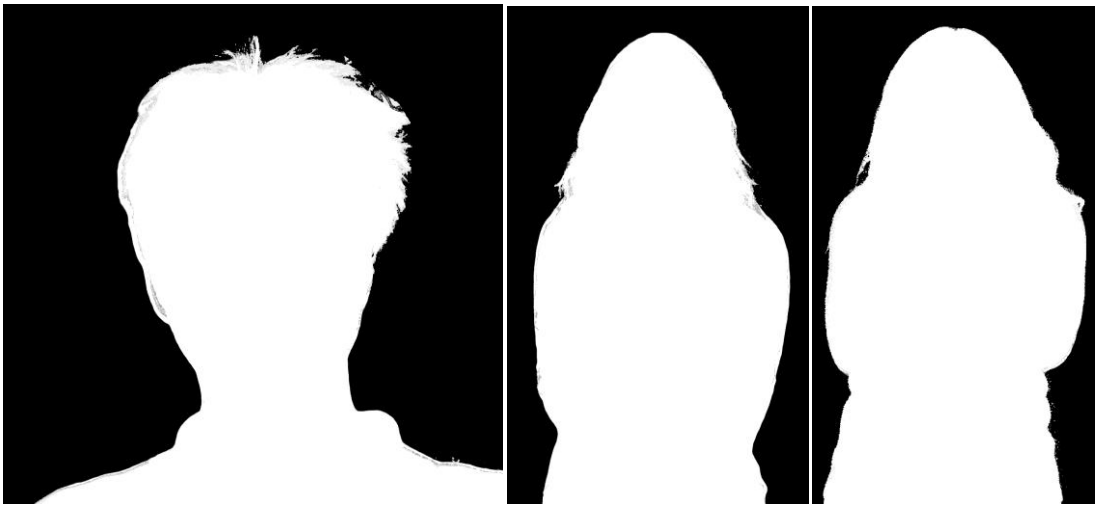


Figure 20: Matting Results Using A Decision List

4.3 Combination Using a Fuzzy Inference System

4.3.1 Fuzzy Logic

Fuzzy logic [27, 28, 29] starts with the concept of a fuzzy set. A fuzzy set is different from a classical set, which has a crisp, clearly defined boundary. In a classical crisp set, membership or non-membership of element ***x*** in set *A* is described by a membership function:

$$\mu_A(x) = \begin{cases} 1, & if \quad x \in A \\ 0, & if \quad x \notin A \end{cases} \tag{4.22}$$

A fuzzy set *A* on a universe of discourse *U* is characterized by a membership function $\mu_A(x)$ that takes values in the interval [0, 1]. Fuzzy sets represent commonsense linguistic labels like *slow*, *fast*, *small*, *large*, *heavy*, *low*, *medium*, *high*, *tall*, etc. A given element can be a member of more than one fuzzy set at a time. A fuzzy set *A* in *U* may be represented as a set

of ordered pairs. Each pair consists of a generic element *x* and its grade of membership function; that is:

$$A = \{(x, \mu_A(x)) \mid x \in U\} \tag{4.23}$$

where *x* is called a support value if $\mu_A(x) > 0$. A linguistic variable ***x*** in the universe of discourse *U* is characterized by $T(x) = \{T_x^1, T_x^2, ..., T_x^k\}$ and $\mu(x) = \{\mu_x^1, \mu_x^2, ..., \mu_x^k\}$, where *T(x)* is the term set of ***x***—that is, the set of names of linguistic values of *x*, with each T_x^i being a fuzzy number with membership function μ_x^i defined on *U*. For example, if ***x*** indicates height, then *T(x)* may refer to sets such as *short*, *medium*, or *tall*. A membership function is essentially a curve that defines how each point in the input space is mapped to a membership value (ro degree of memebership) between 0 and 1.

As an example of a fuzzy set is the set of the tall people. In this case, the universe of discourse is all potential heights, let's say from 3 feet to 9 feet, and the word tall would correspond to a curve that defines the degree to which any person is tall. If the set of tall people is given the well-defined (crisp) boundary of a classical set, the statement might be like all people taller than 6 feet are officially considered tall. It may make sense to consider the set of all real numbers greater than 6 because numbers belong on an abstract plane, but when we want to talk about real people, it is unreasonable to call one person short and another one tall when they differ in height by width of their hair.

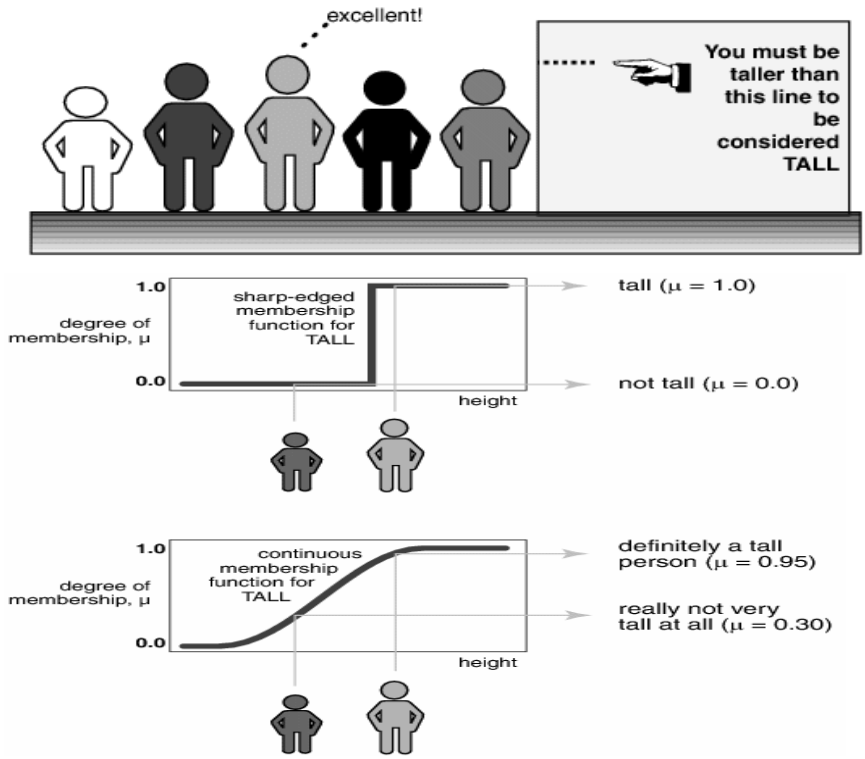


Figure 21: Fuzzy Set Example With A Smoothly Varying Curve

In the fuzzy logic, the truth of any statement becomes a matter of degree. It describes vague concepts and admits the possibility of partial membership in it. The fuzzy system can contain elements with only a partial degree of membership which is defined by the membership function. The only condition a membership function must really satisfy is that it must vary between 0 and 1. The function itself can be an arbitrary curve whose shape we can define as a function that suits us from the point of view of simplicity, convenience and efficiency.

In the above example, there is only one condition as input in the fuzzy system. Let's consider other more complex examples, like: what season is it right now, or our matting problem etc., there are multiple input conditions need to be considered. How to resolve the

multiple input conditions or the multiple input fuzzy sets? The answer is using the logical operations, like the classic logical operators: *AND*, *OR* and *NOT* on the truth table, the fuzzy logical operations use minima, maximum and 1-*A* (A is the real number between 0 and 1) to define what are known as the fuzzy intersection or conjunction (AND), fuzzy union or disjunction (OR), and fuzzy complement (NOT):

$$\begin{cases} AND: \mu_{A\cup B}(x) = \max\left[\mu_A(x), \mu_B(x)\right] \\ OR: \ \mu_{A\cap B}(x) = \min\left[\mu_A(x), \mu_B(x)\right] \\ NOT: \qquad \mu_{\bar{A}}(x) = 1 - \mu_A(x) \end{cases} \qquad (4.24)$$

Zadeh (1965) defined fuzzy ***union*** and fuzzy ***intersection*** as in Equation (4.25):

$$\mu_{A\cup B}(x) = \mu_A(x) + \mu_B(x) - \mu_A(x)\mu_B(x)$$

$$\mu_{A\cap B}(x) = \mu_A(x)\mu_B(x) \qquad (3.25)$$

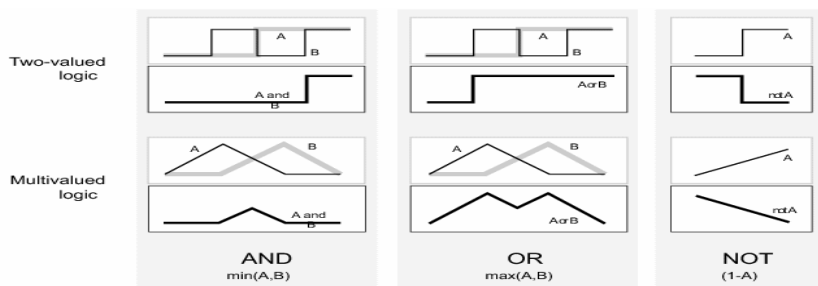


Figure 22: Constructions Using The Fuzzy Sets And Logical Operations

Given these three functions, you can resolve any construction using the fuzzy sets and the fuzzy logical operations.

4.3.2 Fuzzy Rules

Fuzzy sets and fuzzy operators are the subjects and verbs of fuzzy logic. These *if—then* rule statements are used to formulate the conditional statements that comprise

fuzzy logic. A fuzzy rule base contains a set of fuzzy rules *R*. A single *if—then* rule assumes the form "if *x* is T_x then ***y*** is T_y." An example of a rule might be "*if education is high and experience is high, then salary is very high*." For a multi-input, multi-output system,

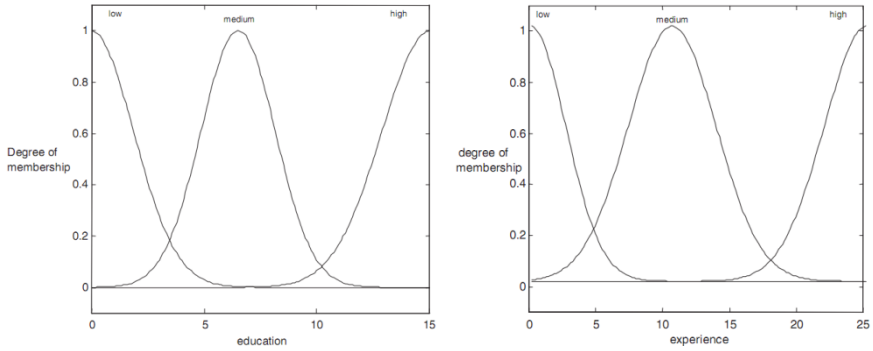
$$R = \{R_1, R_2, ..., R_n\} \qquad (4.26)$$

where the ***i***th fuzzy rule is

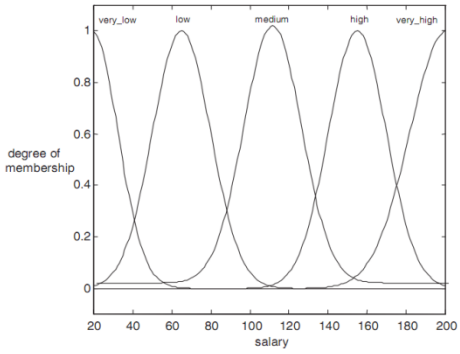
$$R_i = if\,(x_1 is T_{x_1}, and..., x_p is T_{x_p})then(y_1 \text{ in } T_{y_1}, and..., Y_q in T_{y_q}) \qquad (4.27)$$

The *p* preconditions of R_i form a fuzzy set, and the consequent is the union of *q* independent outputs. If we consider a multi-input and single-output system, then the consequent reduces to (y_1 is T_{y_1}). Consider a following multi-input, multi-output example. Let $X = (x_1, x_2, ..., x_n)^T$ be the input vector and $Y = (y_1, y_2, ..., y_n)^T$ be the output vector. The linguistic variable $\boldsymbol{x_i}$ in the universe of discourse ***U*** is characterized by $T(x) = \{T_x^1, T_x^2, ..., T_x^k\}$ and $\mu(x) = \{\mu_x^1, \mu_x^2, ..., \mu_x^k\}$ where *T(x)* is a term set of *x*; that is, it is the set of names of linguistic values of *x*, with each T_x^i being a fuzzy membership and the membership function μ_x^i defined on *U*. As a illustration, consider a fuzzy inference system with two inputs (*n=2*) and one output (*m=1*). Let the two inputs represent the number of years of education and the number of years of experience, and let the output of the system be salary. Let $\boldsymbol{x_1}$ indicate the number of years of education, $T(x_1)$ repersent its term set {*low*, *medium*, *high*}, and the universe of discourse be [0-15]. Let x_2 indicate the number of years of experience, the universe of discourse be [0-30], and the corresponding term set be {*low*, *medium*, *high*}. Similarly, linguistic variable *y* in the universe of discourse *V* is characterized by

$T(y) = \left\{T_y^1, T_y^2, ..., T_y^I\right\}$, where *T(y)* is a term set of ***y***; that is, *T* is the set of names of linguistic values of *y*, with each T_y^i being a fuzzy membership function μ_y^i defined on *V*. If the variable *y* represents *salary*, then *T(y)* represents a term set {*very low*, *low*, *medium*, *high*, *very high*}, and the universe of discourse is [20-200], which represents the minimum and maximum in thousands of dollars—that is, 20,000, and 200,000, repectively. In order to map input variables $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ to output ***y***, it is necessary that we first define the corresponding fuzzy sets. The membership function for the inputs and output variables are shown in Figure 23.



(a) Fuzzy membership functions for input1and input2



R_1: *if education is low and experience is low, then salary is very low*
R_2: *if education is low and experience is medium, then salary is low*
R_3: *if education is low and experience is high, then salary is medium*
R_4: *if education is medium and experience is low, then salary is low*
R_5: *if education is medium and experience is medium, then salary is medium*
R_6: *if education is medium and experience is high, then salary is high*
R_7: *if education is high and experience is low, then salary is medium*
R_8: *if education is high and experience is medium, then salary is high*
R_9: *if education is high and experience is high, then salary is very high*

(b) Fuzzy membership functions for output and if-then rules

Figure 23: The Input and Output Membership Functions, and If-Then Rules

Interpreting an *if–then* rule is a three part process:

a) Resolve all fuzzy statements in the antecedent to a degree of membership between 0 and 1;

b) if there are multiple parts to the antecedent, apply fuzzy logic operators and resolve the antecedent to a single number between 0 and 1, is the result being the degree of support for the rule;

c) apply the implication method,using the degree of support for the entire rule to shape the output fuzzy set. If the rule has more than one antecedent, the fuzzy operator is applied to obtain one number that represents the result of applying that rule.

For example, consider an ***i***th rule R_i: if $x_1 is T_{x_1}$ and $x_2 is T_{x_2}$ then $y \text{ is } T_{y_1}$. Then the membership of the rule can be denoted as

$$\alpha_i = \min(\mu^i_{x_1}(x_1), \mu^i_{x_2}(x_2)) \text{ Or } \alpha_i = \mu^i_{x_1}(x_1)\mu^i_{x_2}(x_2) \quad (4.28)$$

Equation (4.28) represents fuzzy intersection with the minimum or product operators. Each fuzzy rule yields a single number that represents the firing strength of that rule. The firing strength is then used to shape the output fuzzy set that represents the consequent part of the rule. The implication method is defined as the shaping of the consequent (the output fuzzy set), based on the antecedent. The input for the implication process is a single number given by the antecedent, and the output is a fuzzy set. Two methods are commonly used: the minimum and the product methods represented, respectively, by Equations (4.29) and (4.30):

$$\mu^i_y(w)' = \min(\alpha_i, \mu^i_y(w)) \qquad (4.29)$$

$$\mu^i_y(w)' = \alpha_i \mu^i_y(w) \qquad (4.30)$$

where w is the variable that represents the support value of the membership function. For given example, if assume that *education* equals 10 years and *experience* equals 18.6 years, then it can be seen in Figure 18 that rules R_5 and R_6 fire.

After the firing strengths of the rules have been obtained, to combine the corresponding output fuzzy sets into one composite fuzzy set is needed. The process of combining output fuzzy sets into a single set is called aggregation, a process that unifies the outputs of all the

rules. Essentially, aggregation takes all fuzzy sets that represent the output for each rule and combines them into a single fuzzy set that is used as the input to the defuzzification process. Aggregation occurs only once for each output variable. The inputs to the aggregation process are truncated or modified output fuzzy sets obtained as the output of the implication process. The output of the aggregation process is a single fuzzy set that represents the output variable. Since the aggregation method is commutative, the order in which the rules are executed is not important. The commonly used aggregation method is the *max* method. If we have two rules with output fuzzy sets represented by two fuzzy sets $\mu_y^1(w)$ and $\mu_y^2(w)$, then, combining the two sets will obtain the output decision:

$$\mu_y(w) = \max(\mu_y^1(w), \mu_y^2(w)) \qquad (4.31)$$

4.3.3 Defuzzification

Notice that the last result is a membership curve. In order to get a crisp value for output ***y***, we need a defuzzification process. The input to the defuzzification process is a fuzzy set (the aggregate output fuzzy set), and the output of the defuzzification process is a sing crisp value.

Many defuzzification techniques have been proposed in the literature. The most commonly used method is the centroid. Other methods include the maximum, the means of maxima, height, and modified height method. Here we just give the *centroid defuzzification method*: the defuzzifier determines the center of gravity (centroid) of *B* and uses that value as the output of the FLS. For a continuous aggregated fuzzy set, the centroid is given by

$$y' = \frac{\int_s y_i \mu_B(y) dy}{\int_s \mu_B(y) dy} \qquad (4.32)$$

where S denotes the support of $\mu_B(y)$. Often, discretized variables are used so that y' can be approximated as shown in Equation (4.33), which uses summations instead of integration.

$$y' = \frac{\sum_{i=1}^{n} y_i \mu_B(y_i)}{\sum_{i=1}^{n} \mu_B(y_i)} \qquad (4.33)$$

The centroid defuzzification method finds the "*balance*" point of the solution fuzzy region by calculating the weighted mean of the output fuzzy region. It is the most widely used technique because, when it is used, the defuzzified values tend to move smoothly around the output fuzzy region.

4.3.4 Fuzzy inference system (FIS)

The inference process can be described completely in five steps (as shown in Figure 19):

1) **Fuzzy Inputs**: The first step is to take inputs and determine the degree to which they belong to each of the appropriate fuzzy sets via membership functions;
2) **Apply Fuzzy Operators**: Once the inputs have been fuzzified, we know the degree to which each part of the antecedent has been satisfied for each rule. If a given rule has more than one part, the fuzzy logical operators are applied to evaluate the composite firing strength of the rule;
3) **Apply the Implication Method**: The implication method is defined as the shaping of the output membership functions on the basis of the firing strength of the rule. The input for the implication process is a single number given by the antecedent, and the output is a fuzzy set. Two commonly used methods of implication are the minimum and the product;

4) **Aggregate all Outputs**: Aggregation is a process whereby the outputs of each rule are unified. Aggregation occurs only once for each output variable. The input to the aggregation process is the truncated output fuzzy sets returned by the implication process for each rule. The output of the aggregation process is the combined output fuzzy set;

5) **Defuzzify**: The input for the defuzzification process is a fuzzy set (the aggregated output fuzzy set), and the output of the defuzzification process is a crisp value obtained by using some defuzzification method such as the centroid, height, or maximum.

Our fuzzy inference system is built with *Matlab* fuzzy logic tools interface. The *Matlab* toolbox includes 11 build-in membership function types. In our case, we use the piece-wise linear functions as the input membership functions. The FIS inputs are α_{IR} and α_C from equations (4.20) and (4.21). The overview of the basic process structure of the FIS is shown in Figure 24.

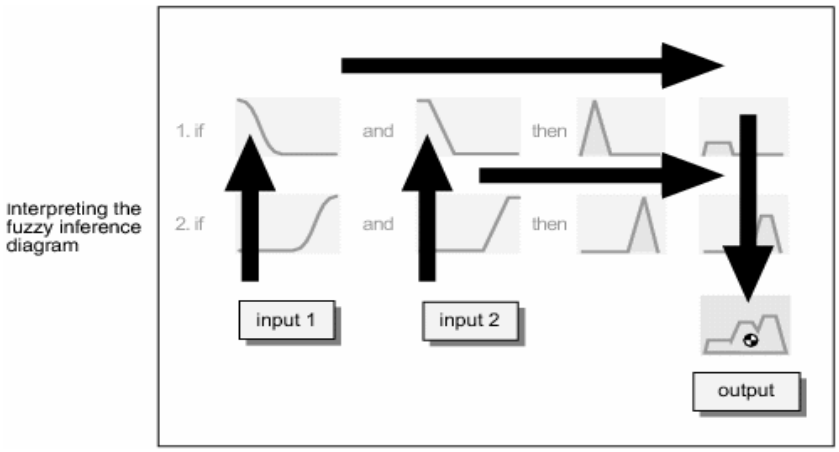


Figure 24: Overview Of A Fuzzy Inference System

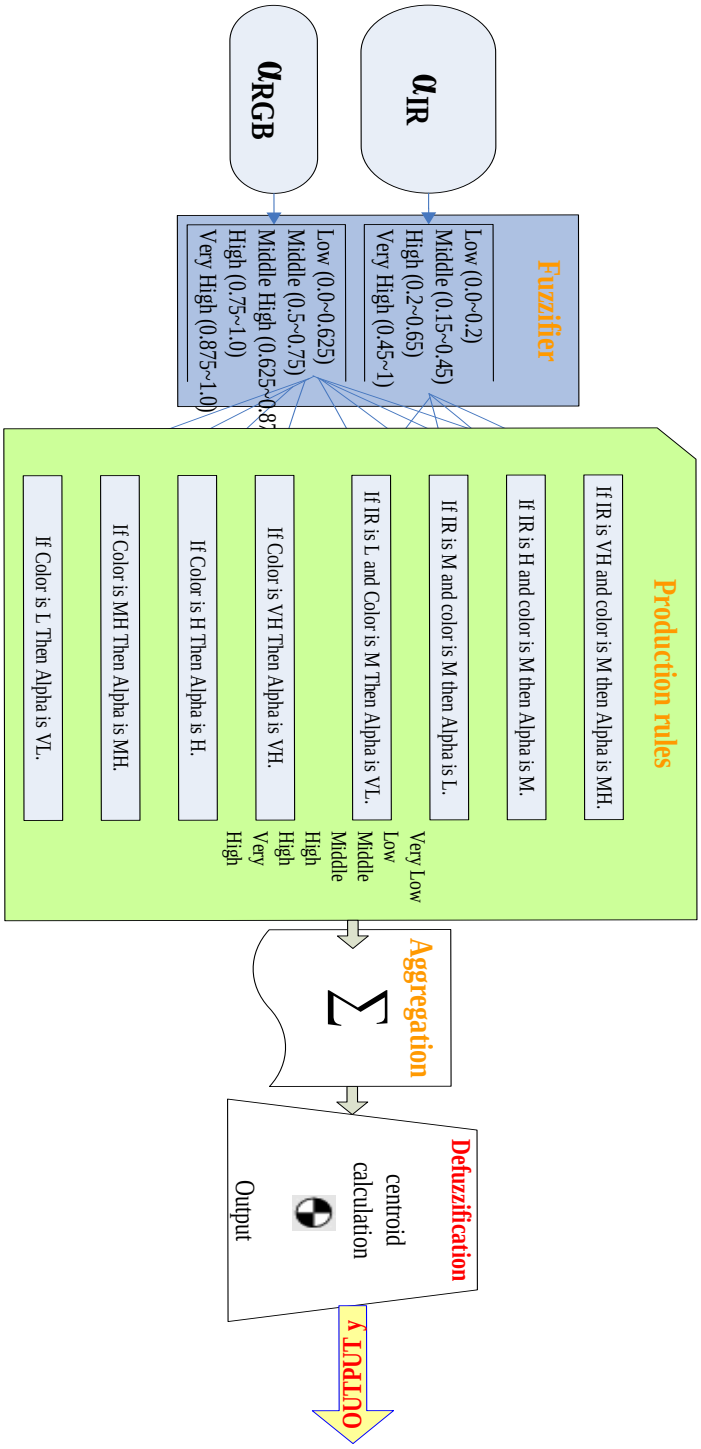
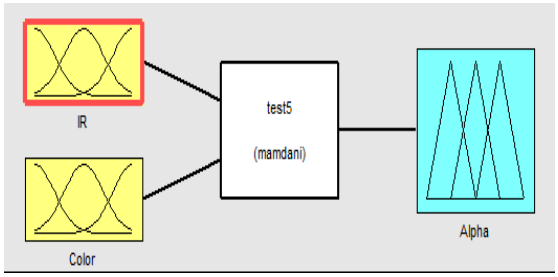
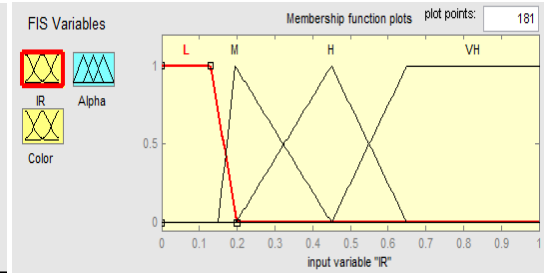


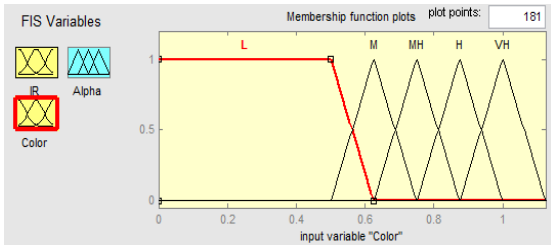
Figure 25: Overview of the Structure of our fuzzy inference process



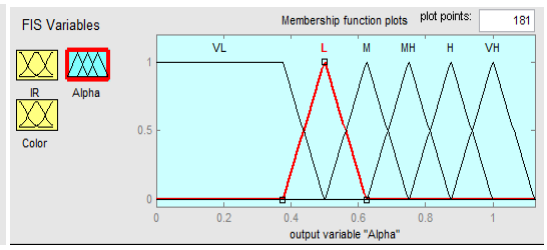
(a) Fuzzy Interface System



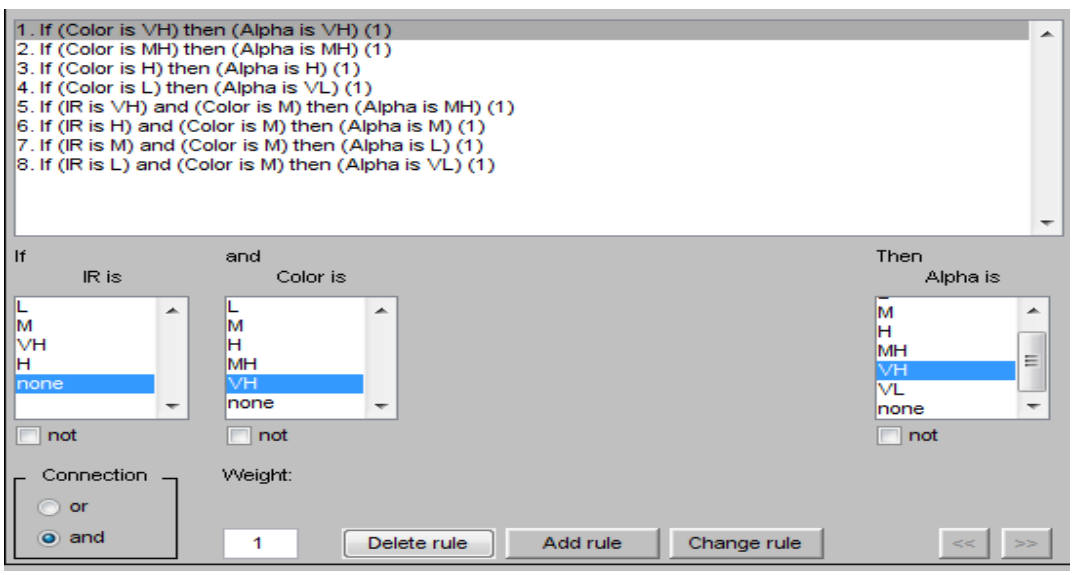
(b) α_{IR} **input and membership functions**



(c) α_{C} **input and membership functions**



(d) Output levels and membership functions



(d) Fuzzy rules

Figure 26: Fuzzy System Components

The first step is to take the inputs and determine the degree to which they belong to each of the appropriate fuzzy sets via membership functions. We divide the α-matte to different

levels with *triangular* and *trapezoidal* membership function, as shown Figures 26-(c), (d), (e). We define 4 levels for α_{IR}: L (*Low*), M (*Middle*), H (*High*), VH (*Very High*), and define 5 levels for α_C: L (*Low*), M (*Middle*), MH (*Middle High*), H (*High*), and VH (*Very High*). We define the final α-matte using 6 levels: VL (*very Low*) L (*Low*), M (*Middle*), MH (*Middle High*), H (*High*), VH (*Very High*).

In the second step, the fuzzy inputs are combined using the fuzzy rules in Figure 24-(f). The output is a single truth value that is used to shape the output fuzzy sets.

Thirdly, we use the two built-in methods of matlab which are the *AND* method that truncates the output fuzzy set, and *PROD* (product), which scales the output fuzzy set. In the all our experiments we used *prod* function, because it gives us better results.

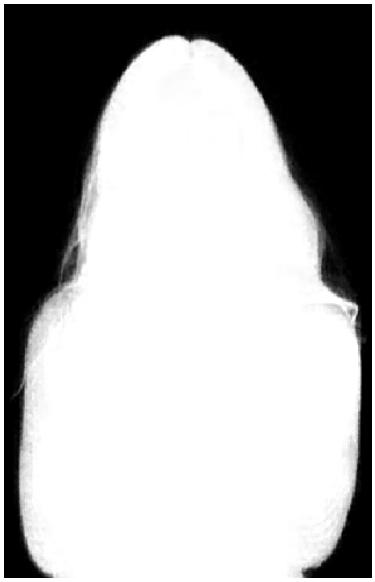
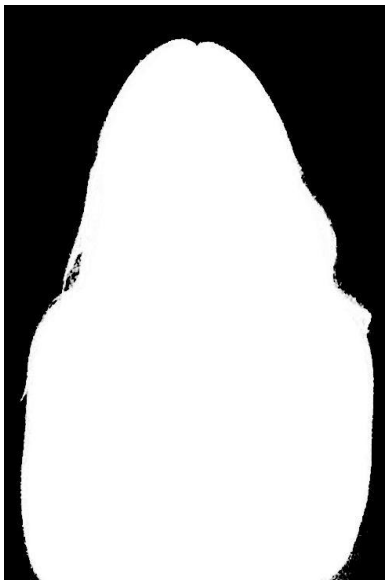
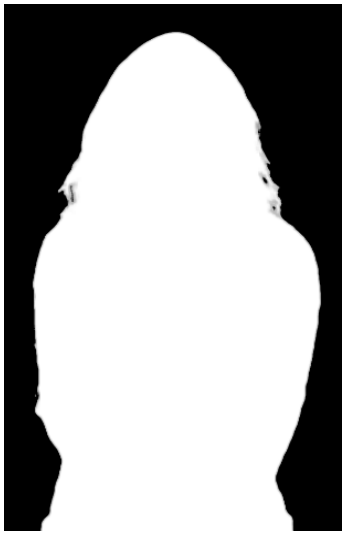
The next step is aggregating all outputs. Because decisions are based on the testing of all of the rules in a FIS, the rules must be combined in some manner in order to make a decision. Aggregation is the process by which the fuzzy sets that represent the outputs of each rule are combined into a single fuzzy set. Aggregation only occurs once for each output variable, just prior to the fifth and final step, defuzzification. The input of the aggregation process is the list of truncated output functions returned by the implication process for each rule. The output of the aggregation process is one fuzzy set for each output variable.

For example, one pixel can belong to one or more levels at the same time in both inputs. So, it could belong to more than one fuzzy set of the output. As shown in Figure 23, the end of each row gives one output fuzzy set. To make a decision, aggregation is needed to combine all these end of rows into a single fuzzy set.

So far, a fuzzy set is still what we have, which means the α matte value of the pixel still need to be calculated. This is done by defuzzification the aggregation result. The input of the defuzzification process is a fuzzy set (the aggregate output fuzzy set) and the output is a single number. As much as fuzziness helps the rule evaluation during the intermediate steps, the final desired output for each variable is generally a single number. However, the aggregate of a fuzzy set encompasses a range of output values, and so must be defuzzified in order to resolve a single output value from the set. The most popular defuzzification method is the centroid calculation, which returns the center of area under the curve. There are other methods such as: the centroid, bisector, middle of maximum (the average of the maximum value of the output set), largest of maximum, and smallest of maximum.

4.3.5 Results Using Fuzzy System

The final results are shown in Figure 25. In comparison with the results obtained with the closed-form technique, the results α-matte using fuzzy inference system has less smooth details in the hear area; but, it provides sharp edges for the border pixels between the body and the background. Also our approach's computational cost is very low in comparison with the closed form matting. This makes our technique more suitable for video-matting where real-time or fast computation is required.

	Closed-form	fuzzy inference system
1		
Size	**250*500**	**500*1000**
Running time	**1 min 21Second**	**9 Second**
2		
Size	**300*500**	**600*1000**
Running-time	**1 min 29Second**	**11 Second**

3

Size	**420*420**	**1400*1400**
Running time	**57 Second**	**16 Second**

Figure 27: Comparison of matting results: left column with closed form matting, right column with a fuzzy inference system.

Conclusion and Future Work

In this thesis, we have addressed the problem of image and video matting. We have presented, implemented and discussed some of the most cited approaches in the literature such as the closed form matting and the Bayesian matting. We have introduced matting using multiple views which allow adding constraints to the ill-posed problem of matting. We have also developed new techniques for matting which combine the use of infrared and color cues. The infrared camera allows a rapid extraction of a human actor from a background containing non-living objects; whereas, the color information adds fine details to the extracted map. Multiple solutions for the combination of color and infrared cues were proposed, implemented and tested. These solutions are based on the concepts of fuzzy logic and fuzzy inference systems.

It is well known that existing video matting systems either are computationally expensive, or have too many assumptions which may or may not hold in practice. In comparison with exiting matting systems, our solution is computationally efficient and provides enough accuracy for video matting applications. The next step task may consist of implementing our matting algorithm on a Graphical Processing Unit to test in the context of real-time video matting [31].

Reference

[1] J. Wang and M. F. Cohen, "*Image and Video Matting: A Survey*" Adobe Systems Incorporated, 801 North 34th Street, Seattle, WA 98103, Microsoft Research, One Microsoft Way, Redmond, WA 98052, USA.

[2] T. Porter and T. Duff, "*Compositing digital images,*" in Proc. of ACM SIGGRAPH, pp. 253 – 259, July 1984.

[3] R. Fielding. "*The Technique of Special Effects Cinematography*," Focal/Hastings House, London, *3rd* Edition, 1972.

[4] Y.-Y. Chuang, B. Curless, D. H. Salesin, and R. Szeliski, "*A bayesian approach to digital matting*," in Proceedings of IEEE CVPR, p., 2001.

[5] A. Levin, D. Lischinski, and Y. Weiss, "*A closed form solution to natural image matting*," in Proceedings of IEEE CVPR, p., 2006.

[6] A. Levin, D. Lischinski, and Y. Weiss, "*A closed form solution to natural image matting*," in IEEE Transactions on Pattern Analysis & Machine Intelligence, FEB, 2008.

[7] M. McGuire, W. Matusik, H. Pfister, J. F. Hughes, and F. Durand, "*Defocus video matting*, " in *Proceedings of ACM SIGGRAPH*, pp.567-576, 2005.

[8] N. Joshi, W. Matusik, and S. Avidan, "*Natural video matting using camera arrays*," in Proceedings of ACM IGGRAPH, pp. 779-786, 2006.

[9] M. A. Ruzon and C. Tomasi. "*Alpha estimation in natural images*." In CVPR 2000, pages 18–25, June 2000.

[10] M. T. Orchard and C. A. Bouman. "*Color Quantization of Images*." IEEE Transactions on Signal Processing, 39(12):2677–2690, December 1991.

[11] J. Wang and M. Cohen, "*Optimized color sampling for robust matting*," in Proc. of IEEE CVPR, 2007.

[12] A. Zomet and S. Peleg. "*Multi-sensor super resolution*," In Proc. of the IEEE Workshop on Applications of Computer Vision, 2002.

[13] J. Wang and M. Cohen. "*An iterative optimization approach for unified image segmentation and matting*," In Proc. IEEE Intl. Conf. on Computer Vision, 2005.

[14]A. Smith and J. Blinn, "*Blue screen matting*," in Proceedings of ACM SIGGRAPH, pp.259–268, 1996.

[15] J. Sun, Y. Li, S.-B. Kang, and H.-Y. Shum, "*Flash matting*," in *Proceedings of ACM SIGGRAPH*, 2006.

[16]P. FAVARO, S. SOATTO, "*Seeing beyond occlusions (and othermarvels of a finite lens aperture)*," In IEEE CVPR, 579–586, 2003.

[17]Y. WEXLER, A. FITZGIBBON AND A. ZISSERMAN, "*Bayesian estimation of layers from multiple images*," In Proc. of 7th European Conference on Computer Vision (ECCV), vol. III, 487-501. 2002.

[18] Rafael C. Gonzalez, E. Woods, Steven L. Eddins, "*Digital Image Processing Using MATLAB*," Pearson Education.

[19]H.S. Alhichri,M. Kamel, "*Virtual circles: a new set of features for fast image registration*," Pattern Recognition Letters 2003.

[20]B. Zitova, J. Flusser, "*Image registration methods: a survey*," Academy of Sciences of the Czech Republic, Image and vision computing, 2003.

[21]J. Ashburner, Karl J. Friston, "*Spatial Transformation of Images*" , Institute of Neurology, London.

[22]H. Zhang, "*Bayesian Matting Using Thermal Vision,"* pp.28-36, 2010.

[23] C. Yang, "*Improved Fast Gauss Transform and Efficient Kernel Density Estimation*," in Proc. Of the Ninth IEEE International Conference on Computer Vision, 2003

[24] J. C. Bezdek: "*Pattern Recognition with Fuzzy Objective Function Algorithm*", Plenum Press, New York 1981

[25] J. C. Bezdek,"*A convergence theorem for fuzzy isodata clustering algorithm*," *IEEE Trans, Pattern Analysis and Machine intelligence*, vol. 2, pp. 130-142, 1980.

[26]A. Boudraa and P. Clarysse, "*Fast fuzzy grey level image segmentation method*," *Medical Biological Engineering Computing*, vol. 35, Suppl. Part. 2, pp. 804, 1997.

[27] Zadeh, L. A., "*Fuzzy sets*." Information and Control, vol, 8, pp, 338–353. 1965.

[28] L.A. Zadeh, Outline of A New Approach to the Analysis of Complex Systems and Decision Processes, 1973

[29] L.A. Zadeh, "*Fuzzy algorithms*," Info. & Ctl. , Vol. 12, pp. 94-102,1968.

[30] C. Chien Lee, "*Fuzzy Logic in Control System: Fuzzy Logic Controller-Part I*," IEEE Tran. On Systems. Man, And Cybernetics, 1990.

[31]L. Grady, T. Schiwietz, S. Aharon, and R. Westermann, "*Random walks for interactive alpha-matting*," in Proceedings of VIIP 2005, pp. 423–429, 2005.