I. Introduction
Face recognition techniques have attracted much attention over the years and many algorithms have been developed. Since it has many potential applications in computer vision and automatic access control system, its research has rapidly expanded by not only engineers but also neuroscientists. Especially, face segmentation is an essential step of face recognition system since most face classification techniques tend to only work with face images. Therefore face segmentation has to correctly extract only face part of given large image. However, because of lots of variations of image appearance, such as pose variation, occlusion, image orientation, illuminating condition and facial expression, it generates difficulties for implementing such algorithm.
In this work, we assumed that image will always have at least one face and size of face can be as small as 50 x 50 pixel wise. With this assumption, we’ve tried to use both feature-based and image-based approaches to maximize the probability of segmentation. Specifically, the process begins by converting a color image into a binary image of skin-likelihood regions. This binary image is further processed using morphological operations in order to optimize it for face segmentation. Template matching is subsequently employed in order to find face candidates in the image. Overall, the face segmentation algorithm presented here uses these various stages to maximize the number of faces segmented in a target image.

II. Skin Color Analysis
A. Color Space
There are many ways to represent digital images in color space model.
- RGB: This model is an additive model in which red, green, and blue are combined in various ways to reproduce other colors. The name of the model and the abbreviation ‘RGB’ come from the three primary colors, red, green, and blue.
- HSV(Hue, Saturation, Value): Hue represent color type such as red, blue, or yellow. Saturation represents the vibrancy of the color. And value represent brightness of the color.
- YCrCb: In this model, luminance and chrominance components of the image are separated. Y is the luma component, and Cb and Cr are the blue and red chroma components.
Not all color models are available for image processing, but it is important to choose the appropriate color space for certain application. In this work, we choose to work with YCrCb color model because of its effectiveness in our application. In this space, chrominance of the image can provide effective information for human skin color than any of two color space described above.

**B. Analysis of YCrCb color space on human skin color**

Any RGB digital image can be converted into YCrCB color space using following equation:

\[
\begin{align*}
Y &= 0.299R + 0.587G + 0.114B \\
Cb &= -0.169R - 0.332G + 0.500B \\
Cr &= 0.500R - 0.419G - 0.081B
\end{align*}
\]  

Using Georgia Tech’s front face images of 50 unique people, YCrCb color space of human face color is analyzed. There are 15 images per each person and each image is converted to YCrCb space. Figure 1 and 2 displays histograms of Cb and Cr of 750 images.

![Fig. 1. Histogram of Chrominance Blue](image1.png)  
![Fig. 2. Histogram of Chrominance Red](image2.png)

As it can be seen from these two histograms human face skin color of Cb ranges from –30 to 18. And for chrominance red, it ranges over -9 to 30. From these frontal face images, means and standard deviations of Cr and Cb are found. Later, these values are used to set maximum and minimum Cr and Cb to discriminate between face part and background. Finally, Figure 3 shows distribution of Cb given Cr.
C. Limitation of color space
Advantage of using skin color map is in its simple implementation and running time. Once prior knowledge on skin color is analyzed, one can easily discriminate skin part of image and non-skin part. Also because each pixel is looked only once to check if its intensity value falls into skin color, it is reasonably fast. However, it doesn't provide accurate and reliable results if there is low contrast between skin color and those of the background objects. In other words, if image contains many objects which have color similar to skin, it tends to detect those objects as face. There are many techniques to overcome this issue such as prior knowledge of a stationary background, setting regularization process and shape analysis. In this work, we’ve tried to apply regularization process as well as template matching for shape analysis.

II. Skin Color Map and Regularization
A. Skin Color Filter
As stated in previous section, means and standard deviations of Cr and Cb of 750 images are calculated. These values are used to set min and max allowance of given pixel intensity value. Means and standard deviations calculated are following:

\[
\begin{align*}
Cb \_ mean & = -11.098 \\
Cb \_ STD & = 4.265 \\
Cr \_ mean & = 21.927 \\
Cr \_ STD & = 4.143
\end{align*}
\]  

(2)

With these numbers, min and max are calculated as following:
\[
Cb_{\text{min}} = Cb_{\text{mean}} - Cb_{\text{STD}} \times 2 \\
Cb_{\text{max}} = Cb_{\text{mean}} + Cb_{\text{STD}} \times 2 \\
Cr_{\text{min}} = Cr_{\text{mean}} + Cr_{\text{STD}} \times 2 \\
Cr_{\text{max}} = Cr_{\text{mean}} + Cr_{\text{STD}} \times 2
\] (3)

Addition of standard deviation to mean maximizes boundary conditions. After min and max are obtained, skin color filter can be easily created by simple operation. Consider an input image of M x N pixels. Since only Cr and Cb values are considered in this step, the output filter is the binary matrix of M/2 x N/2 size.

\[
\text{init} \_ \text{filter}(x, y) = \begin{cases} 
1, & \text{if } [Cr(x, y) \in R_{Cr}] \cap [Cb(x, y) \in R_{Cb}] \\
0, & \text{otherwise}
\end{cases}
\] (4)

where,

\[
R_{Cr} = [Cr_{\text{min}} Cr_{\text{max}}] \\
R_{Cb} = [Cb_{\text{min}} Cb_{\text{max}}]
\] (5)

Figure 4 and Figure 5 displays first step of our implementation. As it can be seen from figure 5, it correctly spotted face part of image, but it also extracted other skin parts (neck and shoulder) of person and background noises. Also parts of face (eyes, eyebrows, and mouth) were not exactly. Finally, it can be noticed that her hair separated right ear from whole face because hair color doesn’t belong to skin color space. Therefore, further regularization processing must be applied to filter removing unnecessary noisy spots.

Fig. 4. Original Image          Fig. 5. Skin Color Filter
B. Regularization Process – Dilation and Erosion

The filter obtained from skin color analysis can be corrupted by noise. The noise may appear as small holes on the facial region due to undetected facial features (eyes, mouth, or eyebrows), or background objects may appear as skin regions. Therefore, simple morphological operations such as dilation to fill in any small holes in the facial area and erosion to remove any small objects in the background area.

To dilate and erode input image, the density distribution of initial filter has to be calculate.

\[
D(x, y) = \sum_{i=0}^{3} \sum_{j=0}^{3} \text{init}_\text{filter}(4x + i, 4y + j)
\]  

(6)

According to the density value, each pixel in input image can be classified into three types: zero (D = 0), intermediate (0 < D < 16), and full (D = 16). After density distribution of initial filter is calculated, density regularization process begins.

1) Init - Set boundaries of density map to zero.
2) Dilate - If any point of either zero or intermediate density has two or more full-density point in its local 3 x 3 neighborhood, set that point equal to 16.
3) Erode - If any full density point is surrounded by less than five other full-density points in its local 3 x 3 neighborhood, set that point equal to zero

Then this density map can again be converted to second filter.

\[
\text{second}_\text{filter}(x, y) = \begin{cases} 
1, & \text{if} D(x, y) = 16 \\
0, & \text{otherwise}
\end{cases}
\]

(7)

Fig. 6. Filled small hole   Fig. 7. Eroded after filling.

Already the map has reduced lots of noise initial filter contained. However, this step only removed small noises in either background and within face region. The filter still can’t separate neck from face region.

C. Regularization Process – Edge Detection
Sometimes skin color theory can extract too much information from input image if original image already contains high number of skin color pixels. In this case, edge detection on original image can be used to separate face region from connected skin region of person. In this work, we’ve used Roberts Cross Edge Detection algorithm to find edges of original image. Reference to this algorithm can be found in reference section. Basic idea of edge detection is following

\[
\begin{align*}
tmp1 &= \text{image}(x, y) - \text{image}(x + 1, y + 1) \\
tmp2 &= \text{image}(x + 1, y) - \text{image}(x, y + 1) \\
edge(x, y) &= |tmp1| + |tmp2|
\end{align*}
\]

This edge is then applied to filter with & operator.

![Fig. 8. Edges of original image](image1)

![Fig. 9. Edge and Filtered combined](image2)

Again combined filter must be eroded to maximize thin edge lines and small holes created from this step are removed by dilate operation.

![Fig. 10. Eroded image of figure 9](image3)

![Fig. 11. Dilated image of figure 10](image4)

Clearly, edge detection along with density regularization has improved initial filter to segment only skin part of input image. However, it is noticed that edge detection layer
may create additional noise. In case of figure 11, we see that noise reoccurred in eye and eyebrows. Therefore, further studies are required to decide when to use edge detection. Also one should discuss how to remove holes large enough that dilation cannot detect.

### III. Template Matching

Assuming input image will have at least single person’s face and its size is as small as 50 x 50, we’ve set our template as oval shape of 100 by 100. Figure 12 shows template used in this work.

![Oval shape Template (100 x 100)](image)

Given skin color filter and the template, template convolves with original filter and mean square errors are calculated at each index. Index of minimum error will be top left corner of the filter. Mean square errors are calculated as follow.

\[
\min \sum_{x,y \text{in ter}} | I_1(x, y) - I_2(x + u, y + u) |^2
\]

where \( I_1 \) is the filter and \( I_2 \) is the template

Equation (9) can be expanded to

\[
\min \sum_{x,y \text{in ter}} I_1^2(x, y) - 2I_1(x, y)I_2(x + u, y + v) + I_2^2(x + u, y + v)
\]

Since filter and template have different size and we only want to match part of template in the filter, first part of equation (10) is basically following.

\[
\sum_{x,y} I_1^2(x, y) \text{mask}(x + u, y + v)
\]

where \( \text{mask} \) is binary matrix with ones in size of template and zeros elsewhere.

Equation (11) can be simply transformed into Fourier domain and it becomes simple convolution in Fourier domain.
\[ I_1^2(w) \ast \text{mask}(w) \]  \hspace{1cm} (12)

Middle part of equation (10) can also be transformed into Fourier domain and it becomes following convolution.

\[ 2I_1(w) \ast I_2(w) \]  \hspace{1cm} (13)

Therefore, in Fourier domain, equation (10) becomes

\[ 2I_1(w) \ast I_2(w) + I_1^2(w) \ast \text{mask}(w) + I_2^2(x, y) \]  \hspace{1cm} (14)

Once index which minimizes error is found, we are only interested in that region out of many regions extracted in the filter. To set binary filter to one in the region found by template matching, first different contour regions in the filter map must be labeled. And label closest to index found by template matching technique will stay alive and all other pixels in the filter are set to zero.

**IV. Segmentation and Results**

Once final map is found through many steps discussed in previous sections, it can be simply augmented with original image to segment desired face part of input image. Here are some of successful segmentation of face part of image. It displays final filter on the right and filtered image on the left.
Still some of segmented images contain noise such as black spot in hair region. However, segmented human faces are recognizable. Some of unsuccessful segmentations are displayed next.

Figure 14 demonstrates issue of skin color model. Since original images were large enough, implementation couldn’t fill the holes of eye, eyebrows, mustache, and mouth.
parts. Further face shape analysis may remove this noise.

![Fig. 15. Better Edge detection to separate face and neck](image1)

In figure 15, it is noticed that edge detection tried to separate connected skin regions. Both filter images above tried to close jaw as its boundary between face and neck, but failed to close.

![Fig. 16. Better dilation method can remove features inside face](image2)

Figure 16 correctly separated face from neck using edge detection, but edge detection created additional noise. Therefore, it failed to segment eye and eyebrows of person’s face.

V. Conclusion

In this work, we have analyzed how skin color model can be used to segment face of input image. Then we’ve tried several methods for regularizing noises which were produced by the model. Once skin color map is obtained, template matching found face part from many regions. Skin color model works very well under certain circumstances.

1) Person’s face and background colors are significantly different
2) Under constraint lighting environment
3) Within face features are relatively small to fill holes
4) Input image has only single person

In all other cases, further processing must be done to specifically locate face image. Based on density distribution of filter, dilation and erosion can remove significant number of noises. In addition, edge detection can separate parts of skin (face, neck, arms, and etc). However, if face is relatively small comparing to whole image, edge detection might produce additional noise. Therefore, careful consideration is required when using edge detection techniques. Overall, skin color map is efficient algorithm to locate face. It is fast and simple. However, refinements are needed when image contains various noises.

VI. Future work
In this work, simple oval shape binary template was used to locate face region of given filter. Since our assumption was that face image will be at least 50 by 50 matrix, our oval shape template size of 100 by 100 was reasonable. To improve this scale issue, better template matching algorithm can be implemented such as resizing template and minimizing error rate. Also instead of using simple binary template, more complex template like eigenface can be applied. Once face region is detected using template, region can be analyzed further to merge close enough neighbor regions, to expand itself, and to remove any noise within selected region. If small regions are clustered within some distance apart, it may merge those regions to create bigger region. Also if region is not totally oval shape, it can expand itself to be more like oval shape.
Skin color model sometimes can cause trouble producing unwanted noises in the filter. In this case, prior knowledge on background objects can be analyzed first to remove those regions out of filter. If the system is used in public area for security purpose, pictures will be taken at same location. Therefore, color analysis at that place can be done initially and this information can locate background objects.

Division of labor
Kyungsog Oh - Skin Color Analysis, Acquiring skin color filter, refining filter using dilation and erosion based on density probability map, Template Matching.

Reference: Roberts Cross Edge Detection Algorithm
The Roberts Cross operator performs a simple, quick to compute, 2-D spatial gradient measurement on an image. It thus highlights regions of high spatial frequency which
often correspond to edges. In its most common usage, the input to the operator is grayscale image, as is the output. Pixel values at each point in the output represent the estimated absolute magnitude of the spatial gradient of the input image at that point. In theory, the operator consists of 2x2 convolution kernels as shown in Figure 1. One kernel is simply the other rotated by 90°.

![Figure 1. Roberts Cross Convolution Kernels](image)

These kernels are designed to respond maximally to edges at 45° to the pixel grid, one kernel for each of the two perpendicular orientations. The kernels can be applied separately to the input image, to produce separate measurements of the gradient component in each orientation (call these $G_x$ and $G_y$). These can then be combined together to find the absolute magnitude of the gradient at each point and the orientation of that gradient. The gradient magnitude is given by:

$$ |G| = \sqrt{G_x^2 + G_y^2} $$

Although typically, an approximate magnitude is computed using:

$$ |G| = |G_x| + |G_y| $$

which is much faster to compute.

The angle of orientation of the edge giving rise to the spatial gradient is given by:

$$ \theta = \arctan\left(\frac{G_y}{G_x}\right) - \frac{3\pi}{4} $$

In this case, orientation 0 is taken to mean that the direction of maximum contrast from black to white runs from left to right on the image, and other angles are measured clockwise from this.
References:

