IT 08 006
Examensarbete 30 hp
Februari 2008

# Finding and Segmenting Human Faces

Qing Gu

Institutionen för informationsteknologi
*Department of Information Technology*

UPPSALA
UNIVERSITET

**Teknisk- naturvetenskaplig fakultet**
**UTH-enheten**

Besöksadress:
Ångströmlaboratoriet
Lägerhyddsvägen 1
Hus 4, Plan 0

Postadress:
Box 536
751 21 Uppsala

Telefon:
018 – 471 30 03

Telefax:
018 – 471 30 00

Hemsida:
http://www.teknat.uu.se/student

# Abstract

## Finding and Segmenting Human Faces

*Qing Gu*

Human face and facial feature detection have attracted a lot of attention because of their wide applications, such as face recognition, face image database management and human-computer interaction. So it is of interest to develop a fast and robust algorithm to detect the human face and facial features. This paper is about a study of finding faces within images and segmenting the face into numbered regions which are the face-, mouth-, eyes- and hair regions respectively. In the last few years, many face detection methods have been proposed based on different specific conditions. The detection system in this master thesis project is implemented using color images with complex backgrounds under various lighting conditions. Each input image is dominated by the upper half of a single person. The algorithm presented in the paper uses a combined algorithm to detect the face. First, a skin color detection algorithm is applied to detect the four possible face regions. Second, the algorithm locates eyes within the candidate regions, and then the region with eyes becomes the face region. Finally, the algorithm locates the mouth and hair from the eyes and face regions.

Handledare: Bo Nordin
Ämnesgranskare: Ewert Bengtsson
Examinator: Anders Jansson
IT 08 006
Tryckt av: Reprocentralen ITC

# 1. Introduction

## 1.1 Face processing systems

With the advancement of information technology, the field of Human Computer Interaction (HCI) has been developed greatly. People have become more and more interested in the new interaction which is less dependent on the traditional methods? In the vision technology area, researchers have started to investigate and develop human face processing systems.

Human face research, as an independent research topic, has attracted a lot of attention in the last 30 years. Human face research mainly includes human face recognition and human face detection. Human face recognition is defined as identifying or verifying a person from a digital still image or video image, while human face detection is defined as determining the locations and sizes of human faces in images. In the past, people gave a lot of attention to human face recognition. Nevertheless, human face recognition is a very complicated problem. In order to do face recognition, people first need to build a database which includes all kinds of human face images. To verify an image from the image database, the system needs to detect the human face within the image and then analyze all the similarity factors among all the images within the database. And up until now, no one has developed a very mature human face recognition system with high accuracy and speed. But, in any human face processing system, the first step is to detect human faces. So face detection actually began as an independent topic itself because of the requirement of human face recognition system. Due to the complexity of face recognition, detecting a human face and its facial features without identifying the person is of interest. And this is the focus of this project.

This project, finding and segmenting human faces, actually includes two parts: human face detection and facial feature segmentations. To give more detail, the goal is to find the head region of a person in an image that is dominated by the upper half of a single person, and to segment this head region into four parts: the skin region, eyes region, mouth region and hair region.

## 1.2 Face detection challenges

There are many challenges in face detection. The typical challenges in human face detection research are:

(1) Pose.
In the image, people may pose differently. And the angles of the faces can be very different as well.

(2) Facial expression
People can have very complicated facial expressions, such as having the eyes and mouth closed or open.
(3) Face occlusion
Face might be covered or obstructed. For example, people might be wearing something such as glasses or caps covering part of their faces.
(4) Imaging conditions
Lighting and camera characteristics may affect the appearance of the image greatly.

## 1.3 Face detection and facial features methods

In response to all the challenges, there have been many detection methods proposed, including some early works such as Template Matching [12], Eigen Faces [11] and Neural Network [9]. M. Turk and A. Pentland's Eigen Faces method [11] had the advantage of being easy to implement but it was quite limited under certain lighting conditions and in various sizes. Rowley's ANN method [9] and Yuille's deformable templates [12] resulted in very high accuracy. But it's very time-consuming due to its numerous amounts of required calculations. With the development of the technology, real-time face detection systems have become more and more important. In recent years, Combined Facial features methods in [3], [4], [8] achieved relatively high accuracy in real-time face detection systems.

Accuracy and speed are a trade-off most of the time. The Combined Facial Features methods apply to real-time systems. In this project, I adopt the Combined Facial Features method which is relatively easy to implement as well as high in accuracy.

Human face detection normally involves detecting more than one human face in an image, but according to the requirements of this project, the images to be processed must have only one person in them. In this paper, the main contributions are developments in skin color region detection, eyes region detection and mouth region detection.

### 1.3.1 Skin color detection

Skin color is a very useful and important facial feature for human face detection. Many human face detection methods are based on skin area detection. But it is also challenging for many reasons, including skin color differences among people from different ethnic backgrounds, camera characteristics, and lighting conditions. Due to these challenges, it's necessary to use a reliable skin color model. Skin color model is the mathematical model that describes the distribution of skin colors. In a successful skin color model, skin color should distribute compactly, which helps the skin color segmentation greatly.

In order to get a reliable skin color model, color space choice is the first step. In the last decade, several color spaces have been proposed, such as RGB, XYZ, CIE-Lab, HSV or YCbCr. In [1], K. Sobottka and I. Pitas used the HSV color space for skin color detection. Hue and saturation domains were utilized for classification of skin colors. Certain thresholds for two domains were set. The hue and saturation values of each pixel in the image were tested, and if they were within the interval formed by the thresholds, the pixel was identified as a skin pixel. But if the values were outside the interval, the pixels as not identified as a skin pixel. In [3], [4], color space YCbCr was used. They set thresholds for Cb, Cr, using the same method to detect skin pixels from non-skin pixels. In [2], using both normalized RGB and HSV color spaces, they proposed Gaussian distribution to solve the problem. All these tests showed good results, but, the problem was that the detection result was still quite sensitive to the lighting conditions. The detected face region sometimes didn't include the real face region because it was too bright or too dark. In addition, it often included the clothing area and the background area, because the domain values of the non-skin region sometimes were within the interval they set.

Due to the problem described above, in [5], they found a solution which was quite independent of lighting condition. First, they created CrCgCb color space based on YCbCr following a certain formula, then used fuzzy cluster algorithm to segment the skin pixels within the same cluster. It produced a good result, but the problem was that it included many calculations and the system was quite time-consuming. In [6], Chen proposed a new way to eliminate the clothes area from the face area. Based on the R, G, B values of each pixel, they created a formula to calculate a variance named Var. They found the Var values of clothes area pixels are often 10 times larger than those of face area pixels. With this formula, it is indeed possible to eliminate some clothes area and background area.

### 1.3.2 Eyes and Mouth detections

Eyes and mouth detections are the main facial feature detections. So they have been researched for a long time. Early on, Yuille [12] proposed the deformable templates method for both eyes and mouth, which was a big improvement because of its flexibility in templates. This method achieved very high accuracy but included numerous calculations which resulted in slow processing. Sobottka and Pitas [1] determined the positions of eyes and mouth by evaluating the minimum of the topographic gray level relief of the face region. It has the advantage of being fast and simple to implement. Yet its mouth and detection is highly dependent on the gray level relief of the face region. So the accuracy is quite limited.

In more recent years, more real-time methods came out. A.Berbar, M. Kelash and A. Kandeel [3] came up with a method that computed the location of

features according to the facial feature textures and then made horizontal and vertical histograms to show the regions in order to verify and ensure that features detected were facial features. In [4], Chan and Abu-Bakar tried to detect eyes regions based on the chrominance components in YCbCr color space. Their eyes detection algorithm performed wavelet transform and made use of multilevel shareholding. According to Frakas and Munro's golden ratio theory for ideal faces, they can calculate the positions of all the other facial features. Ideas in [8] were quite similar to those in [4], but instead of using the multilevel shareholding algorithm whose threshold was quite empirically dependent, their algorithm was clearer and more precise.

In this paper, section 2 mainly describes the implementations based on the information above.

# 2. Implementation

In this section, I will describe the details of implementation of the algorithm used in this project. Matlab is used as the development tool in the implementation section. The implementation mainly includes four parts: 2.1 Skin color detection, 2.2 Eyes detection, 2.4 Mouth detection and 2.5 Hair detection.

The algorithm implemented in this project is only for images with a single person and mainly for the front-face images in which both eyes and mouth can be seen. In the images, the different facial regions for people with glasses covering the eyes or beards covering the mouth are hard to detect. In this project, every picture can be represented within coordinate. Coordinate (0, 0) starts from up-left. The X coordinate is from left to right and Y coordinate is from up to bottom. The output picture of the program is always 200 x 250 pixels.

## 2.1 Skin color detection

In this project, human skin color detection is the basic step for the human face region detection. Human face regions can be selected from the detected skin areas combined with the eyes positions. The eyes and mouth are all within the skin areas, so eyes and mouth detection can be based on the detected skin areas.

Human skin color detection depends on color spaces. Color spaces can be used to classify pixels in an image into skin and non-skin. There were many color spaces proposed in the past. But, HSV, YCbCr and normalized RGB are considered to be very suitable for human skin color detection. In this paper, an algorithm combining the use of three color spaces is implemented.

Suppose the input image as a sequence of M*N of pixels, an image is a map of M*N pixels. Whether a pixel is within skin areas or not is decided by the threshold for the specific color space. Details about the implementations in each color space are described below.

### 2.1.1 Variance method

At first, I did some tests with color space YCbCr. In color space YCbCr, Cr represents the difference between the red component and a reference value. Based on the algorithm in [13], the threshold for Cr is Rcr = [133,173]. The formula for Map Cr is:

Map Cr = { 255  Cr □ Rcr
{ 0  otherwise

But I found a problem in which the clothing and background areas were often included into skin areas. In [6], according to the authors, the distributions of components of R, G and B are different between skin-color and clothes-color. So a variance statistic is proposed for each pixel in the image. The formula to calculate the variance value is:

$$Var(X_{x,y}) = E(C^2) - E(C)^2 = 1/3 \sum (Ci - avg(Ci))^2 \quad (i=0-2)$$

$X_{x,y}$ denotes the GRB value of the pixel(x, y) in the image. C includes C0, C1and C2. These three values are R, G, B values respectively. According to my testing, I found that the variance values of the clothing pixels are usually bigger than 2500 while the variance values of the skin pixels are usually less than 150. In Figure 1, it can be easily seen that in Map Cr which only adopts the Rcr threshold, many clothing pixels are included in the skin areas while in Map Var, the clothing pixels are excluded from the skin areas.

Cr

Var

**Figure 1**

### 2.1.2 HSV threshold method

After using the algorithm above, the skin detection is improved, but the result still isn't adequate because the skin area often includes the background. In Figure 2, pictures Var didn't show a very satisfactory result, as many background areas are included in the skin area. In [7], the authors proposed some strategies to eliminate the extra pixels based on the existing algorithm. HSV color space was proposed to help as well. And the threshold for H is RH = [0, 32°]. The formula for Map HSV is:

Map HSV= { 255  H □ RH
{ 0  otherwise

### 2.1.3 Normalized RGB threshold method

Map HSV shows the result of the algorithm. It helps to eliminate some background noise pixels. At the same time, after testing, it can be seen that the detection results are very sensitive to the lighting conditions. Because of sometimes the change of lighting conditions, the background pixels are easily segmented into skin area. In [6], the authors described a robust way to eliminate this kind of background noise. Actually, even though skin pixels and the background noise pixels are in the same range in the Cr component, they are quite different in RGB color space. According to statistical results, normalized RGB color space minimizes the dependence on the luminance values, so the red and blue histograms for skin pixels are distributed in a narrow area. Then thresholds for normalized B represented by Pb and normalized B represented by Pr are proposed. The formula for Pb, Pr and Map PbPr is:

Pb=B/(R+G+B)
Pr=R/(R+G+B)
Map PbPr= { 255  Pb □[0,0.12]  Pr □[0.5,1]
{ 0  otherwise

### 2.1.4 Combined method

After getting the result that the PbPr produced, I combined the results of all the algorithms described above, and I got a Map: Var ∩ HSV ∩ PbPr. It shows a relatively satisfactory result which eliminates most of the background, and none of the skin area is eliminated.

Original

Var

HSV

PbPr

Var ∩ HSV ∩ PbPr

**Figure 2**

### 2.1.5 Gray-Scale Elimination

However, it's still quite obvious that in the Map Var ∩ HSV ∩ PbPr, some hair area and some dark background still remained. So I considered getting some ideas from the gray-scale image. After testing, I found that no matter which ethnicity the person in the image is, under good lighting conditions, the gray values for the skin color pixels are usually larger than 75. However, this is experimental and not always the case. Sometimes pictures are taken under very dark lighting conditions. So the gray values of skin color pixels can be very dark and smaller than 75. Then the detected skin area would be very small. In this project, we assume that if a separate skin area is smaller than 3% of the whole picture size, it is considered to be a noise area. So if the detected skin area is too small, it will be automatically recognized as noise area. When this happens, I adopt an adaptive way to solve this problem. If the program cannot find any qualified skin areas, the threshold 75 will be reduced by 20 and the skin detection program will run again. If it still cannot find any skin area, the threshold will be reduced by 20 again and the program will run again. So threshold will be reduced by 20 repeatedly until it finds the right area, or until the threshold is less than 20. If the threshold is less than 20, face detection will show failure.

After combining all the detection algorithms together, I got the result shown in Figure 3, which is quite accurate.

Gray-scale

Result

**Figure 3**

## 2.2 Eyes detection

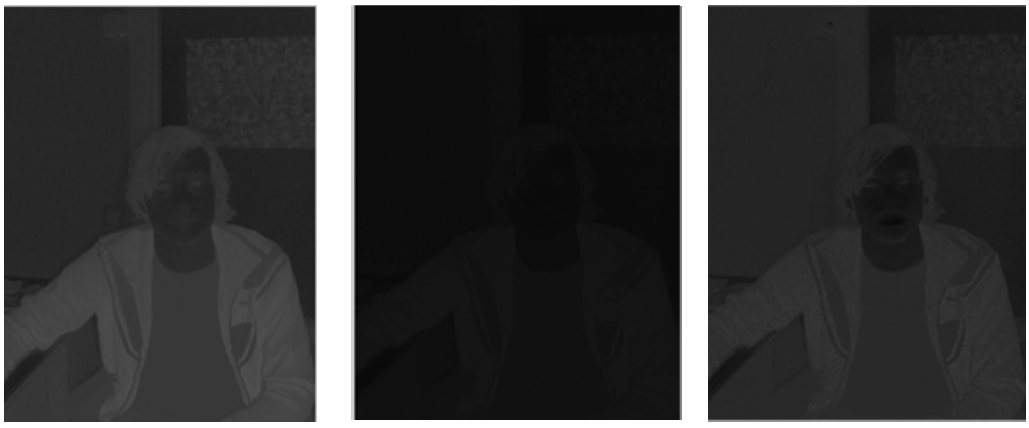
Eyes detection is a very important part of this project, because it is very independent and all the other detections depend on it. In this section, I first describe how to build the eyes map Emap and then locate the eyes within Emap.

### 2.2.1 Eyes Map Building

Eye detection is based on the YCbCr color space. According to the algorithms shown in [8], First, I build two eyes maps, which are EMC and EML. EMC is built from chrominance components while EML is built from luminance components.

EMC is built based on the normalized $Cb^2$, Cbr/Cr indicated by Cbr and negative $Cr^2$ indicated by $N\_Cr^2$. Actually in YCbCr color space, Cb and Cr and within the range [16, 240]. But we need to normalize $Cb^2$, Cbr/Cr and negative $Cr^2$ into the range [0, 255]. The formulas to normalize $Cb^2$, Cbr/Cr and negative $Cr^2$ are as below:

$Cb^2 = Cb^2/ (240^2– 16^2) * 256^2$
$Cbr= ( (Cb– 16)/(Cr– 16) )/(240/16 – 16/240) * 256$
$N\_Cr^2 = (240– Cr– 16)^2/ (240^2– 16^2) * 256^2$



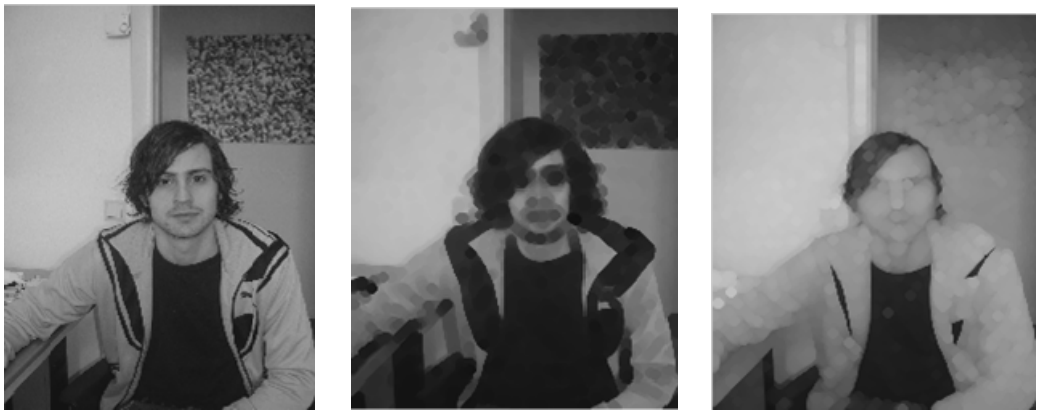
$N\_Cb^2$ Cbr $N\_Cr^2$

EMC

$$EMC = N\_Cb^2 + Cbr + N\_Cr^2 \quad (1)$$

**Figure 4**

Based on the maps built in Figure 4 and follow the formula (1), I get the EMC map. Then I use the Y component, which results in both Erosion and Dilation Maps shown in Figure 5. Following the formula (2), EML map results.



Y Erosion Dilation

EML

$$EML = 2 * Dilation - 2 * Erosion \quad (2)$$

Emap
Emap =8100 * EML * EML * EMC^9 (3)

**Figure 5**

At this point, I have both EMC and EML. Using formula (3), Emap is built. The formulas (1), (2), (3) are based on the algorithms in [8]. But based on my testing, I set the new coefficients. After getting the Emap, it's not hard to see the eyes here and they are very obvious. So now the problem becomes how to eliminate all the other areas.

### 2.2.2 Eyes Localization

After Emap is built, the eyes regions are very obvious as well as some noise regions. There are basically four steps required to get the final result. They are Filtering, single elimination, pair elimination and adaptation. The whole algorithm for eyes localization is based on testing experience and all the numerical parameters for eyes localization are set by testing.

#### 2.2.2.1 Step1 Filtering

Emap is converted to binary image BEMap. Based on BEMap, Median filtering using 2-by-2 neighborhoods is adopted. The Median filtering is a nonlinear operation often used in image processing to reduce "salt and pepper" noise. The purpose of this filtering used here is to eliminate some noise and make the image smoother. After filtering, a number of candidate eyes regions result. After filtering, BEMap is converted to Map Filtering shown in Figure 6.

#### 2.2.2.2 Step2 Single Elimination

First, according to requirements of this thesis, the test pictures are dominated by the upper half of a single person. So it makes sense to cut 30% of the lower part of a picture for this reason, and because of people's foreheads take some space too. So I cut 10% of

the highest part of a picture.

Second, the output of the program is always 200 x 250 pixels, which means the area for the whole output picture is 50000 pixels. So the eyes region areas should not have too man pixels. And I choose the areas which have less than 315 pixels.

Third, the orientation of the areas has limitations too. Orientation here means the angle (in degrees) between the x-axis and the major axis of the ellipse that has the same second-moments as the region. Because people's eyes orientations should not be vertical or even close to. And I set the orientation to be within the range [-50, 50].

Fourth, all the algorithms for eyes detection described above are based on the whole image. But actually, the eyes should be within the skin region, and I have already done the skin detection in the first part. So I can certainly eliminate the pairs of eyes in which one or both are outside the detected skin region. However, in some cases, the number of skin detection regions is only one, which means the skin region detected is the face region. For these cases, I can also set one more restriction based on the distances between the eyes and the face region center. To give more detail, the distance between eye1 and the face region center is dis1 while distance between eye2 and the face region center is dis2. Then I set dis1+dis2<150.

Finally, it's the eccentricity of the ellipse that has the same second-moments as the region. The eccentricity is the ratio of the distance between the foci of the ellipse and its major axis length. The value is between 0 and 1. An ellipse whose eccentricity is 0 is actually a circle, while an ellipse whose eccentricity is 1 is a line segment. People's eyes regions should not be too thin. So the eccentricity for eyes areas here are set to be less than 0.945.

After the descriptions of the qualified single areas for eyes, I start to focus on the relationships between pairs of eyes. Regarding areas which are all candidates for the eye areas, I need to consider every pair of them. Now every two areas remaining are candidates for the real eyes. So suppose I have m qualified eyes areas, then the first one has to be paired with the second, the third ..... and the mth, and second one has to be paired with the third, the fourth…. and the mth. So they are all together (m-1)*m/2 pairs.

### 2.2.2.3 Step3 Pair Elimination

There are (m-1)*m/2 pairs of areas. For each pair of areas, I first get two centroids which are the centers of the two areas. Then I set the restrictions below for every pair of areas:

#### 2.2.2.3.1 Slope

Slope of the line that connects two centroids (x1, y1), (x2, y2) is calculated as:

Slope = (y2-y1)/(x2-x1)

But for a pair of eyes, the slope of the line connecting two eyes centers should not be too big or too small. I set it between [-0.7, 0.7].

#### 2.2.2.3.2 Size

The size of one eye area shouldn't be too much larger or smaller than the other one. So for any pair of candidate areas A and B, the difference I accept between them is A<2.2B and B<2.2A.

#### 2.2.2.3.3 Distance

I treat each candidate area as an ellipse, and then define the length (in pixels) of the major axis as MajorAxisLength and the length (in pixels) of the minor axis as MinorAxisLength. Now, consider the distance between the two area centroids. The distance should not be too long. I set it as:

1.8* min MajorAxisLength <Distance< 6* max MajorAxisLength

The min MajorAxisLength is the smaller MajorAxisLength of the two ellipses while the max MajorAxisLength is the larger MajorAxisLength of the two ellipses.

### 2.2.2.4 Step4 Adaptation

According to the restrictions specified above, candidate pairs which don't satisfy the restrictions are eliminated. And three possible results may show:

1 Only one pair left

Then this pair of regions is the real eyes.

2 None are left

This means the Emap has problems. I set two coefficients 8100 and 9 in formula (3) which are used to produce Emap. These two coefficients are appropriate for many pictures, but not for some of them. When they are not appropriate, it's usually because the Emap is too dark. Emap is still a gray image. Even though the eyes are always lighter than the surrounding areas, when Emap converted to binary image BEMap, the real eyes areas might be set too dark due to the noisy areas being too light. So I use adaptive algorithm to solve this problem. If no eyes are found, multiply the Emap by 2 and go back to step 1 to repeat all the steps again. However, there is a counter to count

how many times it repeats because it might not be able to find the right eyes for some other reasons. So I set an upper limit 5 for the counter, which means it will repeat all the steps 5 times at most. But if the system runs 5 times and still no pairs are found, it means the eyes detection failed. And then the eyes detection section shows failure.

3 More than one pair left

This indicates the real eyes and some noise pairs exist. Among all these pairs, only one of them is the right one, so I have to compare all of them in order to choose. Generally, in these cases, it becomes quite complicated to choose the right pair. The basic algorithm relies on the distances since real eyes regions are quite close to the face region center.

1 Only One skin region is detected.

In these cases, since only one skin area is detected, I can calculate the center of the skin region which actually is the face region. What often become the noise regions are the eyebrow regions. However, eyes are usually close to the face region center, so the distances between the centers of candidate regions and the center of the face region can be measured to eliminate the noisy pairs.

(1) Two pairs have one region in common and only one skin region is detected

Dis represents the distance between the shared region of the two pairs and the center of the face region.
Edis1 means the distance between the other region of pair 1 and the center of the face region.
Edis2 means the distance between the other region of pair 2 and the center of the face region.
Dis1=Edis1+Dis.
Dis2=Edis2+Dis.

Compare Dis1 and Dis2. If Dis1 is the larger one, and if Dis1/Dis2>1.2, then it shows Edis1 to be much larger than Edis2. Since both eyes should be close to the face center, if one pair is quite far from the face center, then it has a good chance of being a noise region. So Pair2 should be chosen. But if 1<Dis1/Dis2<1.2, it shows both pairs are good concerning the distance. So the pair with larger combined area is chosen.

(2) Two pairs have no region in common and only one skin region is detected

EdisL1 means the distance between one region of pair 1 and the center of the face region.
EdisR1 means the distance between the other region of pair 1 and the center of the face region.
EdisL2 means the distance between one region of pair 2 and the center of the face region.

EdisR2 means the distance between the other region of pair 2 and the center of the face region.
Dis1= EdisL1 + EdisR1.
Dis2= EdisL2 + EdisR2.

Compare Dis1 and Dis2. Suppose Dis1 is the larger one, and if Dis1/Dis2>2, it shows Edis1 is much smaller than Edis2. So Pair2 should be chosen. If 1<Dis1/Dis2<2, it shows both pairs are good concerning the distance. So the pair with the larger combined area is chosen.

2 More than One skin regions are detected.

In this case, it's very hard to distinguish which pair is better. But according to probability, the pair which has larger area has much better chance of being the real eyes pair, because smaller areas often are noise areas.

Filtering

Result

**Figure 6**

In figure 6, the result of the detected eyes is shown as Map result.

## 2.3 Face Region Localization

In section 2.1 and 2.2, I successfully detected skin regions and eyes regions. The detected skin regions can be only the face region but can also include other parts of the body as well. In order to locate the real face region, detected eyes regions can be very helpful because of their independence.

The algorithm first locates the centroids of all the skin regions and eyes regions. Centroids for the left eye and right eye are (EyeLx, EyeLy) and (EyeRx, EyeRx). The center of the eyes is C (Cx, Cy). By following the formulas below I can get the Cx, Cy.
Cx= (EyeLx+ EyeRx)/2
Cy= (EyeLy+ EyeRy)/2

Second, I can get the distances between C (Cx, Cy) and centroids of all the skin regions. Among all the skin regions, the skin region with the shortest distance to C (Cx, Cy) should be the real face region.

## 2.4 Mouth Detection

Mouth detection includes two parts which are Mouth Map Building and Mouth Localization.

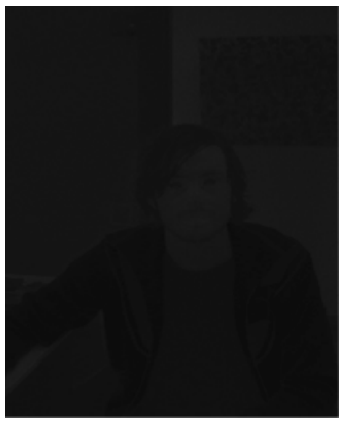
### 2.4.1 Mouth Map Building

Mouth detection is also based on the YCbCr color. In order to locate the mouth, what we need to do first is to build the Mouth Map (Mmap).

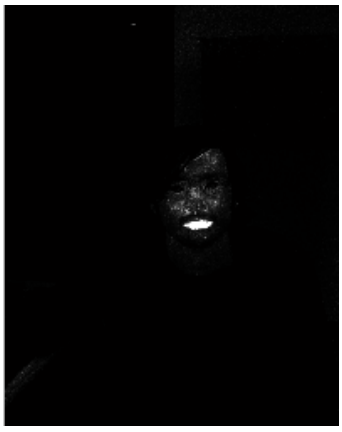
Mmap is built based on the normalized $Cr^2$, Cr/Cb is indicated by Crb. Mmap is built based on the algorithm in [8], but coefficients in formula (4) is created based on my experiments. The formulas for normalizing $Cr^2$and Crb into range [0, 255] are shown below:

$Cr^2 = Cr^2/ (240^2– 16^2) * 256^2$
$Crb= ((Cr– 16)/ (Cb– 16))/ (240/16 – 16/240) * 256$

$Cr^2$



Crb



Mmap

Mmap =4800000*($Cr^2$-Crb) ^15 (4)

**Figure 7**

### 2.4.2 Mouth Localization

After Mmap is built, the mouth regions are very obvious in the Mmap as well as some noise regions, but I need to locate real mouth regions. There are basically three steps necessary to get the final result. They are Filtering, elimination and adaptation. The whole algorithm in moth localization is based on testing experience all the numerical parameters for mouth localization are set by testing.

#### 2.4.2.1 Step1 Filtering

Mmap is converted to binary image BMMap. Using BMMap, Median filtering using 2-by-2 neighborhoods is adopted. As mentioned in the discussion of eyes localization in section 2.2.2.1, the Median filtering is a nonlinear operation often used in image processing to reduce "salt and pepper" noise. The purpose of the filtering here is to eliminate some noise and make the image smoother. After filtering, a number of candidate mouth regions result. All these candidate regions form a Candidate Region Set, or CRS.

#### 2.4.2.2 Step2 Elimination

In this step, I basically try to eliminate the non-mouth regions. People have two lips, but when they take pictures, sometimes their mouths are closed and sometimes they are open. When the mouth is open, the number of detected mouth regions is usually two, while when mouth is closed; the number of detected regions is usually one. So within the CRS, not all of the regions are real lip regions. Some rules are needed to eliminate some regions.

1 Area
The mouth region areas should be within a range of values. We set this range as [20, 800]

2 Orientation
The orientation of mouth region should be within some a range also. The range for the orientation is within [-65, 45].

3 Position
With the formulas below; I can set some restrictions for Mouthx and Mouthy.
Mouthx should be within the range:
[EyeLx-1/8* EyeDis, EyeRx+1/8* EyeDis]
Mouthy should be within the range:
[Y+0.7* EyeDis, Y+ 1.35* EyeDis]

Mouthx is the x-coordinate for any mouth region
Mouthy is the y-coordinate for any mouth region
EyeLx is the x-coordinate for the left eye
EyeRx is the x-coordinate for the right eye
EyeLy is the y-coordinate for the left eye
EyeRy is the y-coordinate for the right eye
X= (EyeLx+ EyeRx)/2
Y= (EyeLy+ EyeRy)/2
EyeDis = EyeR – EyeL

### 2.4.2.3 Step3 Adaptation

In the mouth detection section, there are also two coefficients 4800000 and 15 set in the formula (4) above. These two coefficients used to build Mmap are quite good for some pictures but can also result in wrong detection. The two possible wrong results are:
1 Failing to find any mouth regions
2 Mouth region areas are much bigger than the real mouth should be.

These problems are actually caused by the different lighting conditions of the pictures. The coefficients 4800000 and 15 are only good for certain lighting conditions. For those lighting conditions which are not suitable, new coefficients are needed. Actually, changing the Mmap is enough.

In order to deal with these two problems, I try to use adaptive ways to detect the mouth. Algorithms for dealing with three different situations are below:

1 Failing to find any regions
Failing to detect any mouth regions can result from two conditions:
(1) Image is too bright. The detected mouth regions don't follow the three rules described above.
(2) Image is too dark and no mouth region can be found.

Within the CRS (candidate region set), find the region with the largest area. If the largest area is more than 3000, Mmap is divided by 5. If the largest area is between 1500 and 3000, Mmap is divided by 2. In these cases, Mmap is too bright. If the largest area is less than 1500, it means that the failure to find faces results from the Mmap being too dark. So Mmap should be multiplied by 3. Then go back to step1 filtering and produce a new BMMap, and repeat all the steps.

2 Being able to find one region
If only one mouth region is found, the reason could be that in the image, the person's mouth is closed, so only one region can be found. But it could also be that the Mmap is

not bright enough. In this case, the detected region is real the mouth region, but the problem is that the region is just a small part of the mouth. And it's possible to improve the detection result. So if the only detected region area is less than 25, multiply Mmap by 2, go back to step 1 and repeat all the steps above.

3 Being able to find more than one region

The detected regions should cover the real mouth regions, but the region areas might be much larger than the real mouth region area is. How can I know if the area is too large? The limits for the detected mouth regions are:

(1) The area of largest region should be less than 10 times of the area of eyes regions.

(2) The Eccentricities of the regions should be less 0.8.

Actually the rules I set above are not strict, and the ranges are usually more than enough. But it's helpful to eliminate a lot of noise regions. If any of the rules in (3) above are not obeyed, this means the regions detected are not the real mouth regions. What we need to do is to divide Mmap by 2, go back to step 1 and repeat all the steps above.

There is also a counter in the mouth detection section, and the upper limit for this counter is also 5. So if the counter reaches 5 and no satisfactory result is found, then mouth detection will show failure.

Result

**Figure 8**

Figure 8 shows the result of the mouth detection, which is quite obvious.

## 2.5 Hair Detection

Hair detection is most challenging section of this project. There haven't been any very successful and mature hair detection systems that have come out yet. People always have one face, two eyes and one mouth. However, in contrast to face, eyes and mouth, people may have hair but also may be bald. People's hair colors and hair styles can be quite different as well. So it's quite difficult to follow some specific rules to detect people's hair region very precisely. Therefore, locating the hair region only by its own characteristics is a very complicated task in this project.

In this project, it's too complicated to detect all kinds of hair. So I decided to ignore the two difficult conditions:
1 People with different hair color regions.
2 People who are partly bald

Basically, my algorithm locates a reference pixel in the hair region and finds pixels that have closest Hc value (shown in formula (1)) to it in the RGB color space. And the pixels found are from the hair region. The choice reference pixel becomes quite important because if it's outside the hair region, then the detection will fail.

In order to locate a reliable reference pixel, I decide to do the hair region detection relying on the face region. Because no matter how different or strange the hair is, if there is hair, it should be connected to the face region. So I first locate the point with smallest y value represented as I (x, y) of the face region. If there is hair, I (x, y) should be the point that connects the hair region and face region. However, the face region detection is not 100 percent accurate most of the time. And due to the lighting conditions being too dark or too light in the upper area of the face, the following errors could result:
1 A small part of the real face region is not detected
2 Some non-face region is detected as face region.

If the upper part of the face region is not detected correctly, I (x, y) shouldn't be the point that connects the face region and hair region. If error (1) happens, one way to help reduce the chances of failure is to choose the reference pixel I1 (x1, y1) to be higher than I(x, y). But how much higher it should be? There should be a measurement. Among all the detections above, eyes detection is the most accurate and the distance between eyes is relatively independent. In addition, this measurement is not a fixed value but varies with the size of the face. So I use the eyes distance as a measurement. So the reference point in hair I1 (x1, y1),

$$x1 = x$$
$$y1 = y - 0.2 * EyeDis$$
$$EyeDis = EyeR - EyeL$$

Ir, Ig and Ib denote the R, G, B values for point I. For any point P in the image, Pr, Pg and Pb represent the R, G, B values for point P.

$Hc=Sqrt( (Pr –Ir)^2+ (Pg–Ig)^2+ (Pb –Ib)^2)<25$ (1)

The candidate regions are formed by the points that satisfy the formula (1). Regarding these candidates, the real hair region should not be too large. And because I ignore the bald cases, the area should not be too small either.
I set it to be no larger than 200 and no smaller than 9600. If the area is out of the range, this system would show failure. And if error (2) happens, I1 might be within the hair region but it might not. I1 within hair region can help to detect the right hair region. If not within the hair region, the detected area should be the background region. Normally failure can be noticed because of its big area.

The distance of from the center of the detected hair region and the center of the face region area should be considered as well. Because if the distance is too long, the detected region has a good chance of being a noise region. The distance should be less than 1.7* EyeDis. If it's larger than this value, system would show failure.

Result

**Figure 9**

# 3. Conclusion and Results

## 3.1 Conclusion

In this paper, I present an implementation of face location and segmentation. A Combined Facial Features method is introduced in this project. Detections for face region, eyes regions, mouth regions and hair region are introduced separately.

In face detection, a combined algorithm based on different three-color spaces is adopted in skin detection first. In previous research, fixed thresholds systems normally are very sensitive under different lighting conditions. The combined algorithm in this project is not only based on the fixed thresholds in the color spaces but is also based on different lighting effects. In order to make the system stronger under different lighting conditions, a few methods based on the fixed thresholds of YCbCr, HSV, Normalized RGB color space and gray-scale map are combined to eliminate the non-skin noises. This combined method successfully decreases the influence of many lighting conditions that reduce the accuracy of the skin detection. Skin region detection shows a satisfactory result. Skin region detection only helps to find the skin areas, but more work needs to be done to locate the face region among the skin regions. In this project, eyes detection is the most independent of all the detections. Because eyes should be within the face region, the detected eyes regions can be used to locate the face region among all the skin regions.

The basic algorithms for Eyes and Mouth regions detections are quite similar. The first step is to build Emap and Mmap according to the algorithms from [8]. Pictures are separated into three spaces: Y, Cb and Cr. These color spaces are processed by following certain formulas and are combined together. Emap and Mmap are built based on these processed color spaces. The second step involves locating the real eyes and mouth regions. This step is quite challenging. A number of measures are introduced to eliminate the noise regions. Some of these measures are produced according to basic knowledge about the positions of the eyes and mouth and their relationship, such as the symmetry of two eyes and the horizontal mouth in-between the two eyes, while some of them are produced experimentally according to the testing database. Eyes detection is quite independent of other detections. However, to some extent, mouth region localization is based on the position of detected eyes locations.

Hair detection is the most challenging task in the project. It's quite difficult because of variability in hair color, style and size. There haven't been any papers showing successful hair detection. In this project, I actually produce an algorithm which chooses a reference pixel within the hair region and finds all the hair pixels based on the reference pixel. The reference pixel is actually chosen by finding the edge of the forehead which comes from the detected face region. So, hair detection is dependent on

face region detection and needs to be investigated further.

## 3.2 Results

The performance of the face processing system produced by this project is tested by a database containing 100 pictures in which pictures were downloaded randomly from internet. According to the specific requirements of this project, each picture has upper half of a single person, but the pictures have quite different backgrounds and different lighting conditions. In addition, this database includes both males and females of different ages with various backgrounds. The accuracy of the face region detection, eyes detection, mouth detection and hair region detection were tested on this database.

The test results for all the detections are shown in Table 1.The detection results can be either failure, wrong detection or correct detection. Failure means the system fails to detect any region while wrong detection means the system succeeds at detecting some regions but the regions are not the correct ones. Both the Failure and wrong detection results are not satisfactory results. In contrast to the unsatisfactory results, correct detection represents the satisfactory and correct results. The percentages of each result are shown in Table 1.

| Region | Failure (%) | Wrong Detection (%) | Correct Detection (%) |
|---|---|---|---|
| Face Detection | 1% | 19% | 80% |
| Eyes Detection | 9% | 2% | 89% |
| Mouth Detection | 7% | 5% | 88% |
| Hair Detection | 43% | 12% | 45% |

**Table 1**

## 3.3 Results Evaluation

According to Table 1, it can be easily seen that eyes detection and mouth detection achieved a high level of accuracy. In eyes and mouth detection, the Emap and Mmap buildings are quite successful, but the localizations of the eyes and mouth regions are quite challenging. With the adaptive methods used in this project, the system achieved quite satisfactory results. However, due to some fixed ranges set, wrong detections happened. Skin detection results are satisfactory. The only problem is that lighting conditions still affect the results a lot. On pictures which are extremely bright or dark, the system often shows wrong detections. In other words, part of the face region is undetected or part of the non-face region is over detected. It would improve the system greatly if the correction for the lighting conditions led to some improvements. Table 1 clearly shows that the hair detection results are not satisfactory in this project. The algorithm I produced is good under quite limited conditions and shows the most accurate results only for those pictures in which people have dark hair and simple background. Based on the testing database which includes various cases, the algorithm shows less than 50 percent accuracy for hair detection and needs to be investigated further.

# 4. References

[1] K. Sobottka and I. Pitas, "Face Localization and Feature Extraction Based on Shape and Color Information," Proc. IEEE Int'l Conf. Image Processing, pp. 483-486, 1996.

[2] Wanzeng Kong and Shan'an Zhu, "A New Method of Single Color Face Detection Based on Skin Model and Gaussian distribution", Proceedings of the 6th World Congress on Intelligent Control and Automation, vol.1, pp. 261-265, June 21 - 23, 2006.

[3] Mohamed A. Berbar, Hamdy M. Kelash, and Amany A. Kandeel, "Faces and Facial Features Detection in Color Images", Proceedings of the Geometric Modeling and Imaging, pp. 209 - 214, July 05-06, 2006.

[4] Y.H. Chan and S.A.R. Abu-Bakar, "Face Detection System Based on Feature-Based Chrominance Color Information", Proceedings of the International Conference on Computer Graphics, Imaging and Visualization (CGIV'04), pp. 153-158, 2004

[5] Kong Xiao, Liu Danghui, and Shen Lansun, "Segmentation of skin color regions based on fuzzy cluster", Proceedings of 2004 International Symposium on Intelligent Multimedia, Video and Speech Processing, pp. 125- 128, October 20-22, 2004.

[6] Mei-Juan Chen, Ming-Chieh Chi, Ching-Ting Hsu and Jeng-Wei Chen, "ROI Video Coding Based on H.263+ with Robust Skin-Color Detection Technique", IEEE Transactions on Consumer Electronics, Vol. 49, No. 3, August 2003

[7] Teerayoot Sawangsri, Vorapoj Patanavijit, and Somchai Jitapunkul, "Face Segmentation Using Novel Skin-Color Map and Morphological Technique", Transactions on engineering, computing and Technology, Vol. 2, December 2004

[8] Rein-Lien Hsu, Mohamed Abdel-Mottaleb, and Anil K.Jain , "Face detection in color image", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.24, no.5, May 2002.

[9] H. A. Rowley, S. Baluja, and T. Kanade, "Neural Network Based Face Detection", IEEE Transactions on Pattern Analysis and Machine Intelligence, 20, January, pp. 23-38, 1998.

[10] Ming-Hsuan Yang, David J. Kriegman, and Narendra Ahuja, "Detecting Faces in Images: A Survey", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.24, no.1, January 2002.

[11] M. Turk and A. Pentland, "Eigen Faces for Recognition", Journal of Cognitive. Neuroscience, 3(1), 1991.

[12] A. Yuille, P. Hallinan, and D. Cohen, "Feature Extraction from Faces Using Deformable Templates," Int'l J. Computer Vision, vol. 8, no. 2, pp. 99-111, 1992.

[13] D. Chai and K. N. Ngan, “Face segmentation using skin-color map in videophone applications”, IEEE Trans. On Circuits and Systems for Video Technology, vol. 9, No. 4, pp.551-564, 1999.