Multi-faces Location in Color Image

Shixiong Li

MAJOR REPORT
IN
THE DEPARTMENT
OF
COMPUTER SCIENCE

PRESENTED IN PARTIAL FULFILLMENT
OF THE REQUIREMNT FOR THE DEGREE OF
MASTEROF COMPUTER SCIENCE

CONCORDIA UNIVERSITY
MONTREAL, QUEBEC, CANADA
AUGUEST 2002

Abstract

A practical algorithm for face detection is presented to locate multiple faces in a color scenery image. An image is first classified by skin/non-skin algorithm. Then, morphology algorithm is used to clean the skin-likes image map by a pre-defined face element structure. Next, the image is clustered and region is generated according to processed pixels left in the image. After region separation and coarse region detection, we introduce an elliptic face mode by using eigen vector to locate a candidate human face. Finally, a face verification process is implemented to detect a real human face for all candidate face regions. A face center and pose is detected and modified by rotation. The main technique in facing location are color, shape detection and noise cleaning. In this implementation, there is no size limitation for color scenery image and human face inside. Experimental results cover different phases. Critical problems in face location still need to be explored. This project is implemented in Java.

Keywords: Face location; skin-color classification; image morphology; Clustering, Region generation, eigen center and face verification.

Acknowledgements

I would like to sincerely thank my supervisor Prof. Adam Krzyzak, for introducing me to facial recognition. His insight and advice help me to complete this major report. I also want to thank Prof. Mudur Sudhir P, who exams and gives me lots of useful advice in this project.

.

I dearly thank my family and friends. My wife, Lei Fang, always gave me her support and understanding during my whole study.

Table Of Contents

1. Introduction:

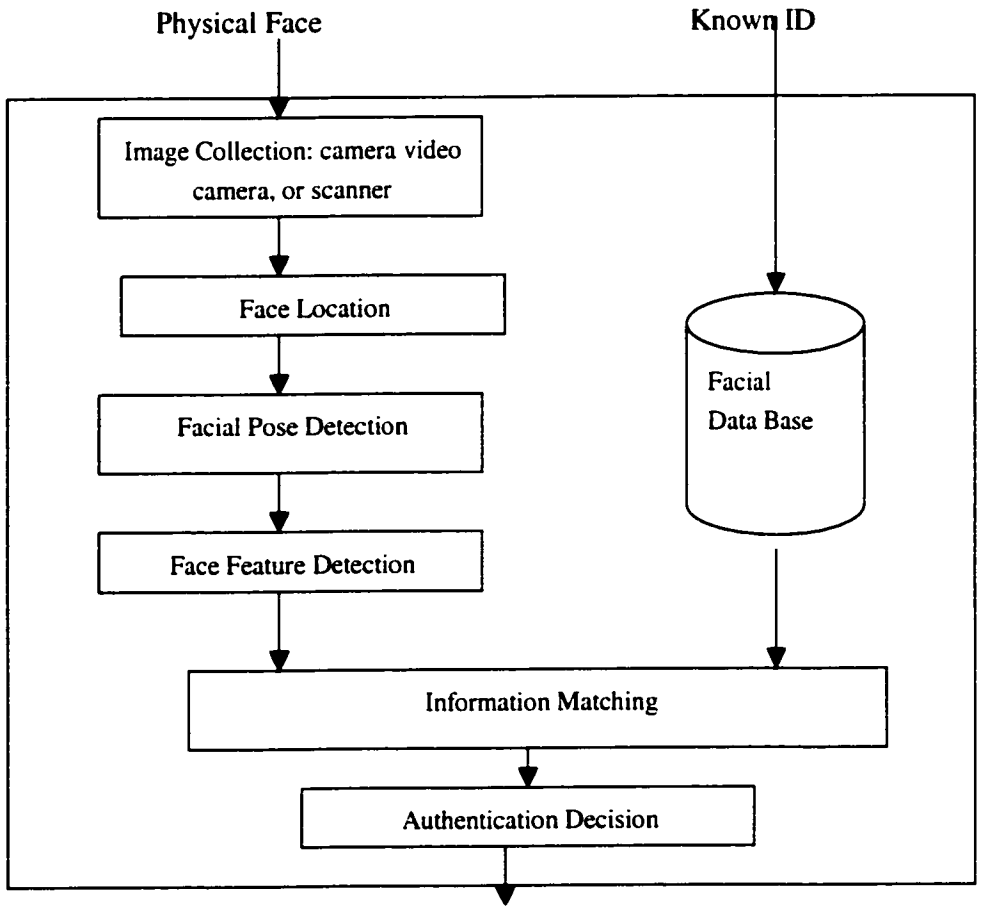
An automatic face recognition system is composed of three main components: human face detection, face pose detection and face recognition. An image can be collected from any sources, a scanner, a video camera or even a photo. After an image is fed into the system, recognition results can be output from the system, and the information can also be linked to a data system so all information can be retrieved according to this recognized results, such as personal information. As face is the most common and difficult part of human beings to be caught without disturbing, so it has more practical use than traditional biometrics technologies, such as finger print and handwriting recognition. Work on each phase of face recognition is still growing due to two main obstacles: the first obstacle is that even today people do not know how humans recognizes an object. Some of this knowledge is acquired in different fields including psychology. Another obstacle is that there is no proven theory of face recognition. Although lots of work has been done in the third phase, very little work existed on face location and pose detection of black and white face location, let alone color image. Color image can offer more information than black and white image, but this also adds the overhead difficulty of color information processing.

1.1 The use of facial recognition for Person authentication

An automatic facial recognition system for person authentication can compare a face with a known face and testify if it is the expected person. For example, when people standing in front of a facial key guard door system, the scanner can take a picture of the

person and compare it with its host name in a list of faces. If one of them matches a host face, the door will be opened. In fact, a lot of fingerprint locker systems use similar authentication theory, but there are two big differences. The first is that taking a picture does not disturb the tested people, and untouched snap can be taken even when people are not perceptible. The second is that facial image has less distortion than fingerprint. When finger are wet and has some mugs , or sometimes finger scanners are not clean, it will get an incorrect fingerprint. On the other hand, face is always among clean parts in human body. So the accuracy will be higher than in case of finger print.

The block diagram for facial authentication system is as follow [23]:



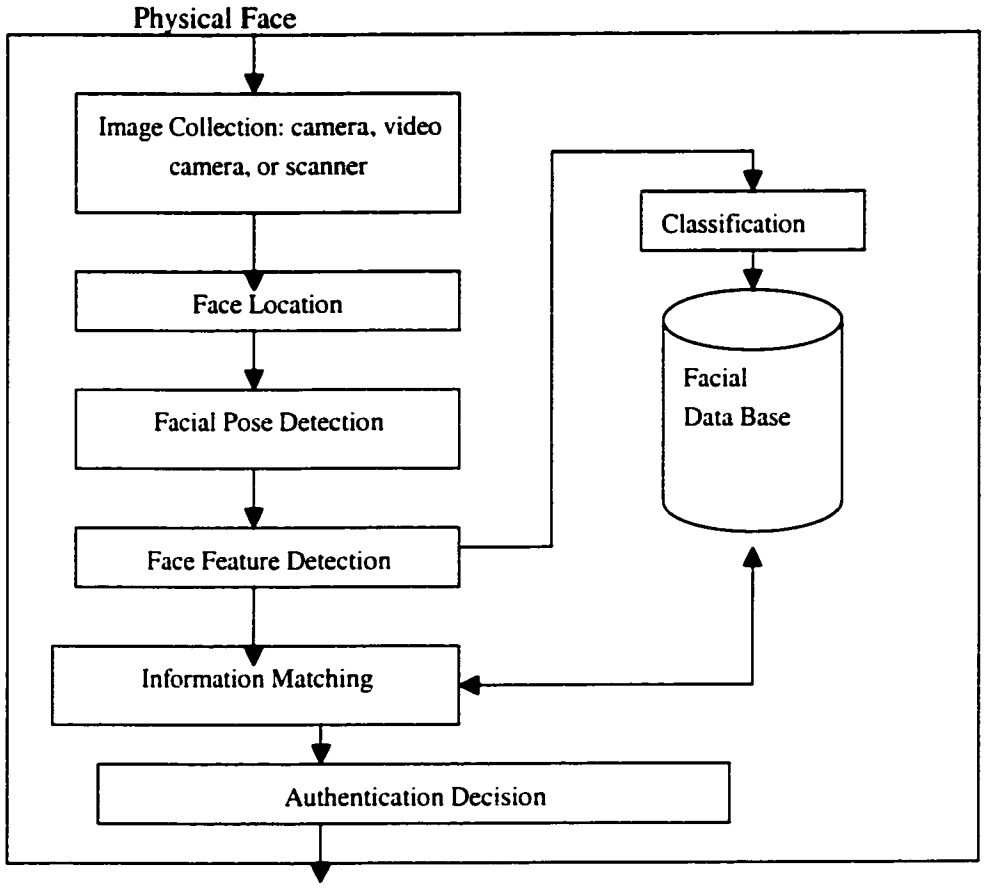
An Automatic Face Recognition System

After being snapped by a camera or a camcorder, the face will be located in the scene image, then the pose will be detected and facial features will be extracted. Extracted face matched with a known face, and a threshold will be used to evaluate if it meet the minimum acceptance requirement. Otherwise, the face will be rejected.

1.2 The use of face recognition for Person Identification

The difference in authentication and identification is that authentication answers the question of " are you she/he", and identification determines "who are you". They also have different practical use. Identification is used for social safety, such as criminal identification or police work. Authentication is used for identifying in private security systems, such as door guard system. Authentication system will give us a positive or negative answer. On the other hand, identification gives a list of possible faces and lets an expert make the final decision. A perfect identification should have the ability to give us an exclusive answer, but may also increase error identification. If an input face similarity is less than a minimum threshold, the system rejects a face.

The block diagram for facial identification system is as follows:



An Automatic Face Identification System

This report explores issues related to face location to the first step for designing an automatic face recognition system. Face location has been studied be many people, but none of studies can produce a satisfactory solution. Ing-sgeeb Hsieh [2] uses HIS to identify face and concludes that intensity of less than 40 came to non-oriental face. The disadvantage is that intensity changes according to luminance, even for an oriental face. Furthermore, the algorithm can only identify an oriental face. Rein-Liens Hsu [8] use

specification value of Cr and Cb between eyes and lip to distinguish face location. The problem is that most people wear glasses and some women frequently use color lip-sticks, which decreases its reliability. Doglus Chai etc. [9] use luminance regularization to identify face on the basis of non-face region having uniform brightness, but it fails when the background is complex.

Our algorithm is based on color specification. We select different human's skin color specification to locate human face, such as white, yellow, and black. In order to eliminate the luminance affection, we use H and S values in HIS color space, or Cr and Cb from YCrCb color space as these values are stable under different luminance conditions. One advantage of the YCrCb color space is that it is used in broadcasting system, so our system can get image information from a video camera without additional signal processing. Our Algorithm is composed of three steps: color classification, region-clustering and candidate face location and face verification.

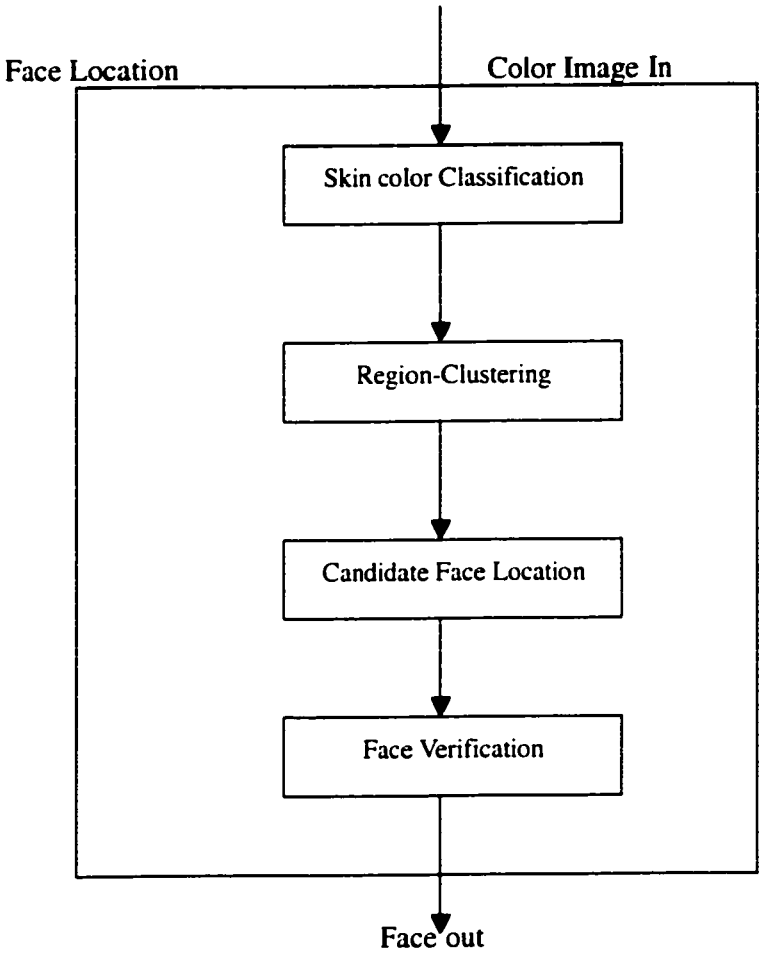
Color Classification will get useful image information according to the human's skin color range, then morphology will be used to eliminate any small region smaller than our expected element structure.

Region –clustering will cluster processed image pixel to some region, as human face is a region. So any connected and big enough regions can be possible human faces.

Candidate face location will create an ellipse region for any candidate face, and this candidate face must include some possible basic face information, such as eyes, nose, and mouth.

Face verification will verify that a candidate is indeed a human face. The verification

has detailed relational specifications, such as the ratio of eyes, distance to face width, two eyes at most and one mouth should lie below eyes etc.



Face Location Diagram

1.3 Face Detection Review

Face processing includes face recognition, face tracking, pose estimation, and expression recognition. Face detection is the first step of all automatic face processing systems. The goal of face detection is to determine if one or more faces are exist in an image and to locate face locations in images. Face detection techniques can be classified to four categories: knowledge-based method, feature invariant approaches,

template matching methods, and appearance-based method [25]. A typical top-down method used by Yang and Huang is based on hierarchical knowledge-based method [26]. Yang and Huang used three levels to detect a face from a white and black image. The first level is to detect a candidate face region by using some rules from an input image. The lowest level is to detect facial features from a candidate face. The mechanism they used is image resolution. The first level separates an input image to several big grids, and likely grids will be selected out for next level's detection. High level separates grids selected by prior level to thin grid and resolution detection is used for more sensitive facial features detection, such as eyes and mouth. The basic rule is face resolution as different scenes have different average pixels value, such as eyes' resolution has a different resolution of mouth in a face region. This method is computation time saving as a coarse-to-fine strategy is used. Feature-based methods can be further classified to three kinds of features: facial features, texture, and skin color. Yow and Cipolla [27] presented a feature-based method that uses a large amount of evidences from visual images and their contextual evidences. Augusteijn and Skufca [28] presented a method by using human faces distinct textures, such as skin and hair, to separate human faces from other different objects in an image. Human skin color has been proved to be an effective feature for face detection. Several color spaces have been utilized to label pixels as skin, such as RGB, normalized RGB, HSV, HIS, YIQ, CIE, XYZ, and CIE LUV [25]. Kjedsen and Kender [29] defined a color predicate in HSV color space to separate skin regions from images. Mckenna, Raja, and Gong [30] applied an adaptive color mixture model to track faces under varying illumination

condition. However, this method can not be applied to detect faces in a single image. There are two kinds of template matching methods: predefined templates and deformable templates. Sakai , Nago, and Fujibayashi [31] used several subtemplates for eyes, nose, mouth, and face contour to model a face, and each subtemplate is defined in terms of line segments. Yuille, Hallinan and Cohen [32] used derformable templates to model facial features that fit a priori elastic model to facial features, such as eyes, and they described facial features by parameterized templates. Template matching used predefined "templates", However, appearance-based methods rely on techniques from statistic analysis and machine learning to find the relevant characteristics of face and non-face images. Many appearance-based methods can be understood in a probabilistic framework, but finding a discriminant function between face and non-face class is another method. [25]. Appearance-based methods can be separated to nine branch methods: Eigenfaces, Distribution-based method, Neural Networks, Support Vector Machines, Sparse Network of Winnows, Naive Bayes Classfier, Hidden Markov Mode, and Inductive Learning. Turk and Pentland [33] applied principal component analysis to face recognition and detection. Principal Component analysis on a training set of face image is performed to generate eigenfaces which span an eigenspace of the image space. Distance from face space is applied as a measure of "faceness" , and the result of calculating the distance from face space is a "face map". A face can be detected from the local minimum of the face map. Sung and Peggio [34] developed a distribution-based system by demonstrating how the distribution of images patterns from one objects class can be learned from positive and negative examples of that class.

Feraud and Bernier [35] used auto-associative neural networks which are based on five layers to perform a non-liner principal component analysis. Here we only discuss the frequently used appearance-based methods, but the other 6 methods can also be found in other transactions and have their advantages [25].

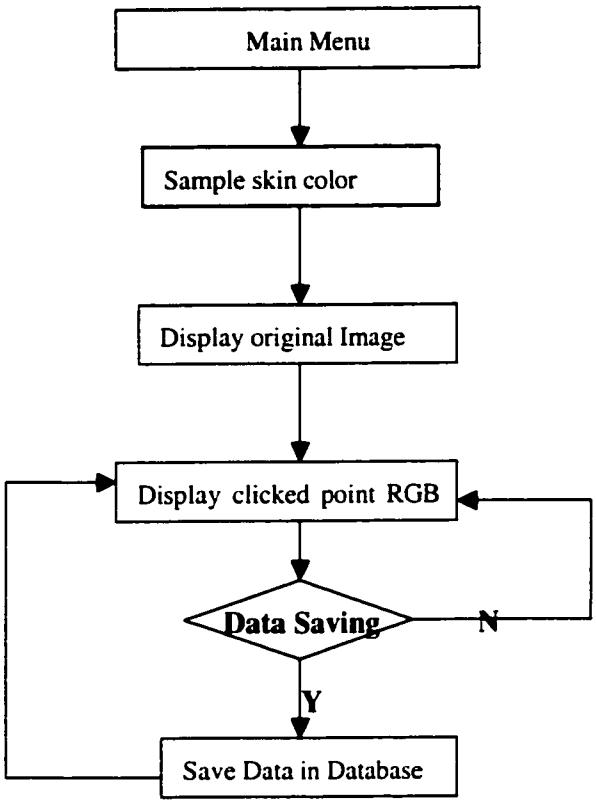
There are also some methods can not be classified into these four categories. However, these four major classes categorize most methods sufficiently and appropriately.

2. Skin Color Classification:

2.1 Sample Skin color parameters:

Skin-color is the foundation of our face location algorithm. We can set a threshold. Any pixel within this threshold is skin color. Otherwise, the pixel is discarded as non-skin color pixel. We can get this from other study 'statistic or sample by ourselves. Ing-Sheen Hsieh etc [2] identify skin color range threshold as: H $\in$ [1-28; 309-360], S $\in$ [13, 110]. The oriental face intensity of I is less than 40. Douglas Chai and N. Ngan [9] identify the threshold tested in YcrCb color system as: Rcr=[133,173] and Rcb=[77,127].

This range can also be tested and sampled in our study. We can sample the processed image skin color in our system and get empirically data from an image, and possible to use this empirical data in our future research to set the threshold rather than use other people's study mentioned above.



Skin color Pixel Sampling

2.2 Lighting compensation:

2.2.1 Luminance Compensation

We regard pixels with top 5 percent of the luminance values as the reference white if the number of the reference-white pixels is larger than 1000. The R, G, and B components of a color image are also adjusted by the same way as these reference-white pixels are scaled to the gray level of 255.[8]

As the luminance for a pure white is 255, 255, 255 of a real color image of R, G, and B, we will compare this pure white color value with the reference white in the image. We select the maximum value from the value of Rr, Gr, and Br. If the maximum value is smaller than 255, which is used as the maximum luminance value in this image, we will calculate the ratio between 255 and the maximum value, and we apply this ratio to the

reference white value so as to the get the luminance of reference white.

Rr=Rs*255/Max ;

Gr=Gs*255/Max ;

Br=Bs*255/Max ;

Rs, Gs, and Bs are the reference white of image, the Max is the maximum value of Rs, Gs, and Bs.

For example, if the reference white of R, G and B are 150, 150, 100, as the maximum value is 150, and the ratio is 255/150. The reference whit value will be as follows:

Rr=150*255/150=255;

Gr=150*255/150=255;

Br=100*255/150=170;

2.2.2 Source Light Detection and Elimination

After the image color luminance is scaled, the source light can also affect the color in an image. For example, a white paper appears as green under green light, and it will be red under red light. So we must detect the source light in the image and eliminate this effect as skin colors' R, G and B will change under different source light. As we use skin color to locate human face, the value range we used is the value under pure white source light, so source light detection and transfer must be the first process step as all following steps are based on the results of this process.

We assume that 5 percent highest luminance is the reflection of white background in an image. By using white background to get the source light, we assume the reference white in the image is Rr, Gr, Br, and the pure white shining on the white blackboard is

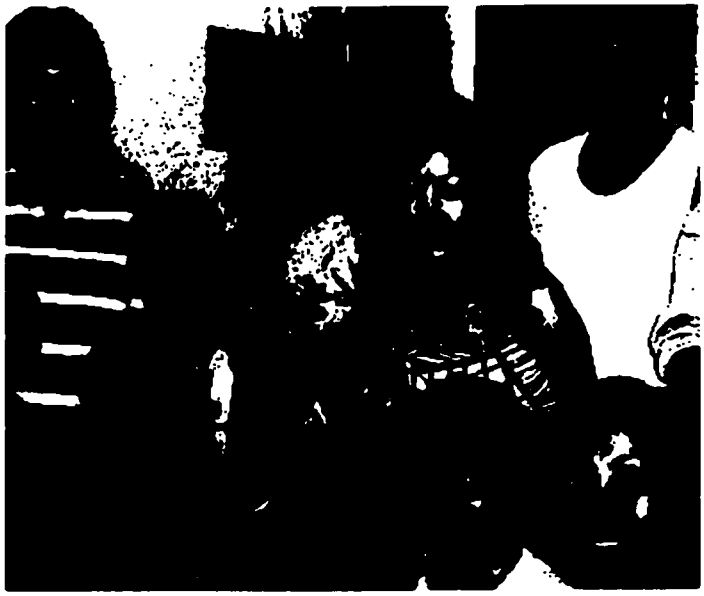
Rp, Gp and Bp. If the image pixels color value is r, g, b, we can approximately calculate the image pixel values under the pure white light is :

R=(r/Rr)*Rp

G=(g/Gr)*Gp

B=(b/Br)*Gb

Here, we can set Rp=255, Gp=255 , and Gp=255 as pure white light.



Skin color processed image before lighting compensation

Skin color processed image after lighting compensation

2.3 Skin–Color Color Range

Color can be analyzed in different color space, and the common color spaces used are RGB, HIS and YcrCb. Although RGB is the mostly widely used color space be used, we can not directly use it as it includes luminance information. As skin color should be detected under different light intensity, so HIS and YCrCr can give us this information as HS and CrCb do not change when the luminance has changed. I and Y are the luminance information in color space of RGB and YcrCb, and these color space values can be easily converted.

HIS color system is adopted to design the color classification algorithm because it is stable for skin color in different lighting condition[2].

$I1=(R+G+B)/3;$

$I2=(R-B)/2;$

$I3=(2G-R-B)/4;$

$I=I1,\ S=(I2*I2+I3*I3)^{1/2},\ H=\tan^{-1}(I3/I2);$

H,S , and I are the hue, saturation, and intensity components of HIS color system.

Color space : YcbCr:

YcrCb is a subset of YUV that scales and shifts the chrominance value into the range of 0..1; the advantage is that it is used in video coding, and can avoid the extra computation required in conversion. The second is that it remains effective regardless of skin color variation

Face Value map: Rcr=[133,173], Rcb=[77,127] for all human skin color values.[9]

Y=0.299R+0.587G+0.114B

Cr=((B-Y)/2)+0.5;

Cb=((R-Y)/1.6)+0.5

A more accurate formulation is follows:

RGB to YCbCr Conversion[10]

YcbCr (256 levels) can be computed directly from 8-bits RGB as follows:

Y = 0.299R+0.587G+0.114B

Cb = -0.1687R-0.3313G+0.5B+128;

Cr = 0.5R-0.4187G-0.0813B+128

YcbCr to RGB Conversion

R = Y+1.402(Cr-128)

G = Y-0.34414 (Cb-128)-0.71414 (Cr-128)

B = Y + 1.772(Cb-128)

The following is a color image classified with threshold of Rcr=[133,173] and Rcb=[77,127]. All pixels in this range do not change. Black color pixels are pixels that do not fit into this range and have been discarded.

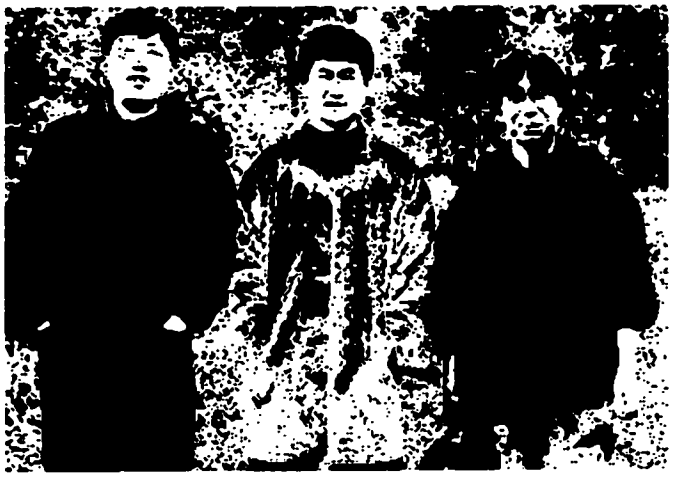
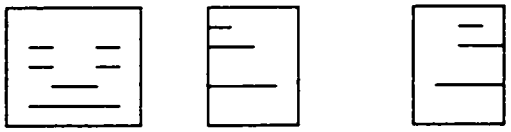


Image after skin color classification

2.4 Morphology

Small isolated regions (like holes) will be filled by morphological methods.



Face left profile right profile

Minimum face area:

width: 5 pixels, eye: at least 2 pixels, eye-brow: 2 pixels; nose; height: 10 pixels.

2.4.1 Noise Cleaning: Morphology

"There are two important morphological operations: opening and closing. Opening

generally smoothes the contour of an image, breaks narrow isthmuses, and eliminates thin protrusions. Closing also tends to smooth section of contours, but as opposed to opening, it generally fuses narrow breaks and long thin gulfs, eliminating small holes, and filling gaps in the contour."[7]

Opening is defined as :

$$A \circ B = (A \ominus B) \oplus B$$

Closing is defined as :

$$A \bullet B = (A \oplus B) \ominus B$$

Using opening and closing can create significant noise cleaning effects. One of the uses in the image can be seen from the following image processing examples [7]

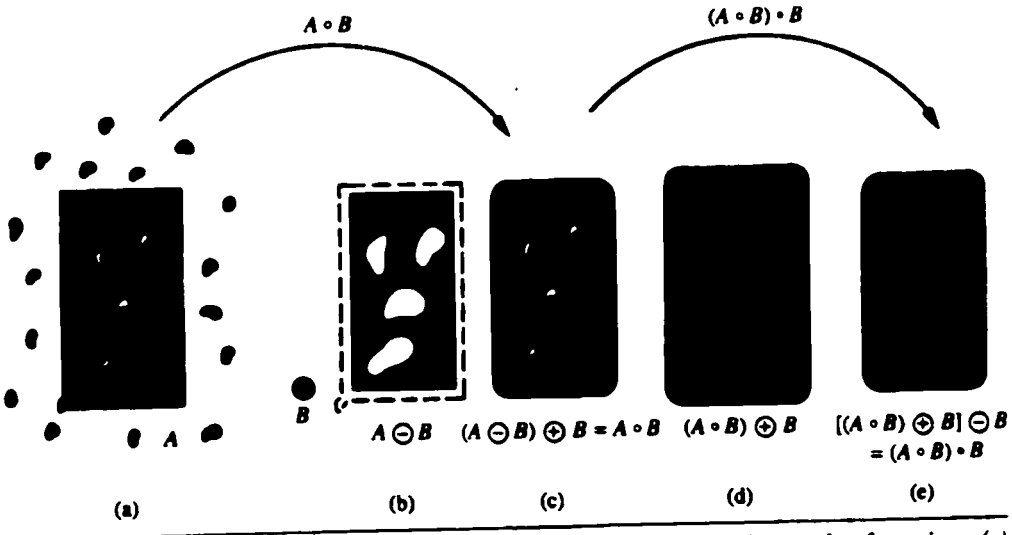
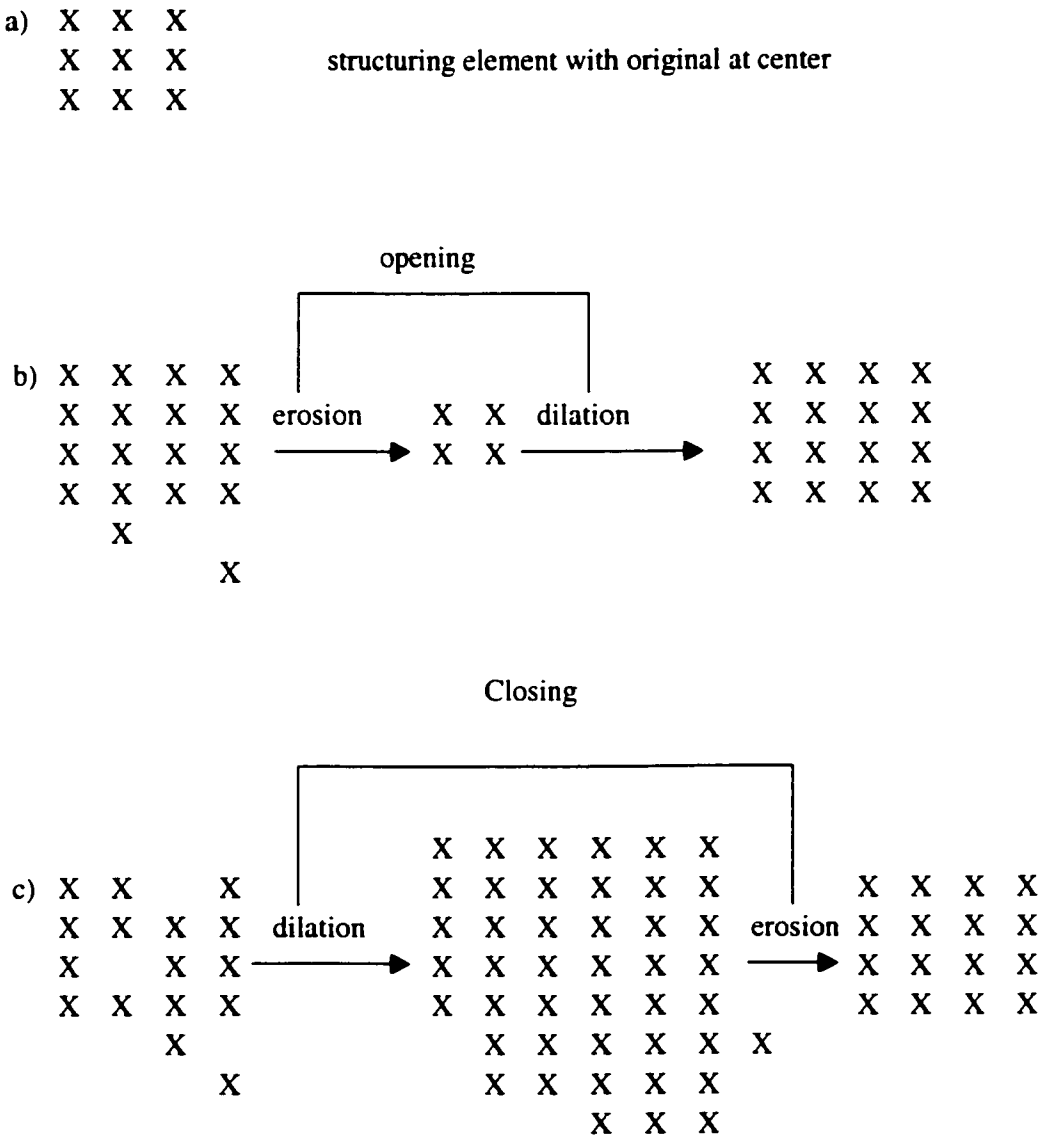


Figure 8.31 *Morphological filtering: (a) original, noisy image; (b) result of erosion; (c) opening of A; (d) result of performing dilation on the opening; (e) final result showing the closing of the opening. (Adapted from Giardina and Dougherty [1988].)*

We can transfer this complex mathematical formulation into a simple pixel calculation for image processing.



The structuring element in (a) is centered on each pixel in the image, and pixel values are changed as follows. For erosion, an ON-valued pixel is turned off if the image based on this pixel as a center of the structuring element is not a structuring element, other wise, it will keep ON value. For dilation, On-value pixel will be expend to a structuring

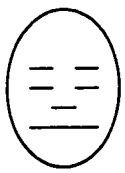
element based on this value as a center point of this structuring element. These rules also can be used for closing which have two processing: dilation followed by erosion [12].

From the above example b), we can see in the process of opening, two side pixels are removed out. In example c), gaps inside the image are filled, side and isolated pixels are connected by adding some bridge pixels.

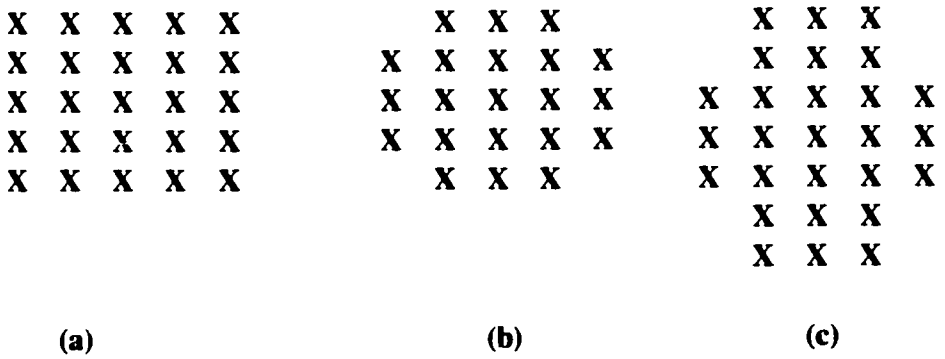
In our face location processing, we assume a minimum face must at least be bigger than an 5*5 square area. That means at least one eye and a boundary must exist in an image. Any color less than this length, for example of 5 pixels, will be eliminated as it is a non-face area. The face squares will be regarded as our structuring element for the morphology using in our image processing. In fact, in order to make our assumption more reasonable, we can erase the corner pixels of a square area to let the structuring element to look alike.

Square Face

Circle Face

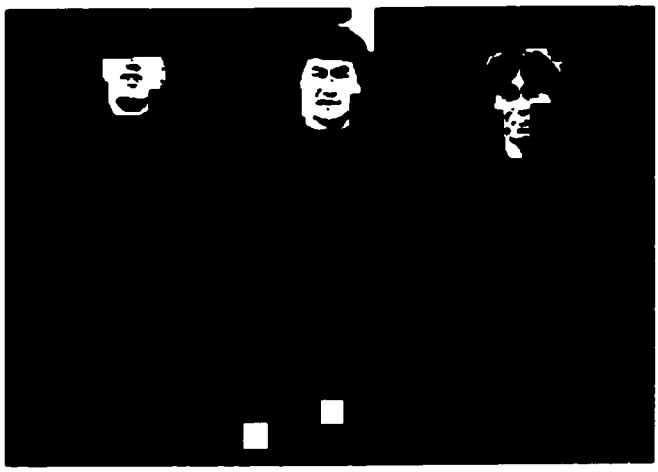


Ellipse Face



Face Structuring Element Model

Morphology can separate skin-color regions and non-skin color region if they are loosely connected. But if the connected pixels are bigger than structuring element, it does not work. So selecting an appropriate size of structuring element model is very important. Small size model keeps too much useless pixels in the image, but big size model eliminates useful pixel information, and sometimes this can cause face location to fail.



Morphologies by an element structure as 13x13 pixels

Morphologies by an element structure as 5x5 pixels

3. Region Clustering

3.1 Region Cluster Preprocessing

An original image may have lots of abrupt color variations and noise. If we cluster this unprocessed image, it will create lots of small color region, including a noise color region. These tiny images are useless for image identification. In fact, they will confuse our identification algorithm and make them it more complex. For example, if cheek and forehead belong to different region because of small color or light difference, it will confuse our identification.

A 5x5 window for minimizing the Gaussian noise is used in our smoothing algorithm. A pixel Fp(x, y) , p∈(R, G, B) in the image is generated according to following equation[2]:

$$Fp(x,y) = \sum_{m=-2}^{2}\sum_{n=-2}^{2} W(m,n) \times Fp(i+m, j+n)$$

$0 \le i \le I,\ 0 \le j \le i,$

Where I and J are the limits of the image boundary of Fp(i,j).

$W(m,n)=W'(m) \times W'(n),\quad m, n \in [-2, 2].$

$$\sum_{m=-2}^{c} W'(m) = 1$$

W'(m)=W'(-m)

Here , we set W'(0)=0.3

W'(-1)=w'(1)=0.25

W'(2)=W'(-2)=0.1.

before Gaussian noise cleaning

after Gaussian noise cleaning

The above two images give us the results of Gaussian smoothing. We can find that our algorithm suppresses abrupt pixel changing. The effects can be seen more significant in next color classification step.

3.2 Clustering Algorithm Selection

The goal of clustering is to create sets of clusters and partitions that group data into similar groups. Close values are assumed to be similar and the goal of the partitioned clustering is to group similar value pixels together. As we have very different HS parameter pairs in HIS image, in order to group some similar color pixels into one color class, we use cluster-seeking algorithms. There are many algorithms, but we can group them into three categories[14]. The first one is "when we know the desired cluster centers". K-menas and Forgy algorithms belong to this category[6]. The second one is "unknown clusters algorithm", simple Cluster-Seeking Algorithms and

Maximum-Distance algorithms belong to this category[6]. The third one is "unclear clusters algorithm when people do not know the clusters numbers in the beginning but have some specification to limit clusters numbers", such as Isodata Algorithm [6].

The properties of K-menas and Forgy algorithms are that the number of clusters to be constructed is specified. Maximum-Distance algorithms can create an unknown clusters before, but the problem is that this algorithm is too rough and it is difficult of select an appropriate threshold. Furthermore, it may create too many clusters with few elements in one cluster. The best algorithm is Isodata clustering algorithm. People can define the maximum and minimum clusters and data in one cluster, merge, splitting ,merging condition and iterations numbers. This algorithm can give us very exact clustering results, but the disadvantage is that it needs to set more parameters for accurate clustering. It is also very time-consuming. Tilton reported that isodata algorithm required sever hours of computing time on a VAX-11/780 to cluster the gray levels in a 512x512 image in 16 clusters [14]. In our image processing, a 1024x768 image is very common, and pixels are real color. A gray level only has 256 values, but a real color has 256x256x256, 16 millions, values. So this algorithm is difficult to be implemented in our work.

Selecting an appropriate algorithm is very important. For example, in numeral classification system, as there are only 10 numbers, so we know we have 10 cluster centers for a numeral system, so K-means is more effective than Maximum-Distance algorithm. However, in color image classification system, one true color image picture can have 16777216 colors for one pixel in R, G, B color system. It is difficult to assign

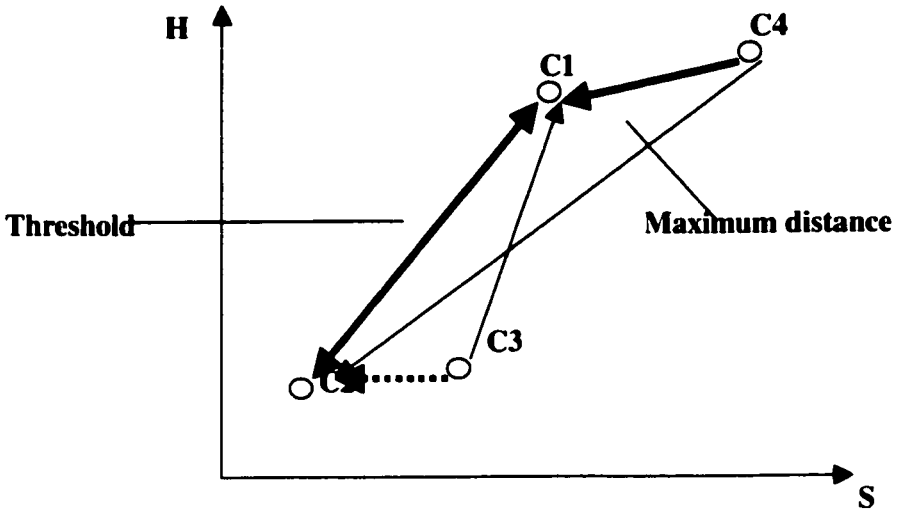
these cluster centers; even 1% can create huge computer problems. Furthermore, we do not know the exact number of clusters in our image, so Maximum-Distance is the only realistic algorithm that can be used in our project.

Normally, we select average distance as the clustering criterion in Maximum-Distance Clustering algorithm. We try to limit the number of clustering to 20 color classes. So we can adjust the clustering criterion if it creates too much or too few color class.

For a class space, we can use different kinds of color space, such as HIS or YCrCb. No matter which color space we use, selecting 2D parameters are very important, and this 2D parameter must be stable in luminance intensity. According to Color space definition, we know that both HS and CrCb meet these requirements.

In our image color classification algorithm, we use Maximum-Distance Algorithm [6]. In the first step, we randomly fetch one pixel from the remaining pixels after morphological algorithm, and put this pixel color as the first color, say C1, and this pixel's H and S value will be regarded as the center of color C1. Then, we calculate the farthest pixel from the HS coordinator by calculating Euclidean distance between the remaining pixels and the first pixel. The farthest pixel is set as the second color class, say C2, and its H and S value will be also regarded as the center of C2. Then, we calculate the remaining pixels' distance to the center of C1 and C2. For each pair of the

Euclidean distance we save the minimum distance. Then, we select the maximum distance from these minimum distances.



Example of Maximum-Distance Algorithm

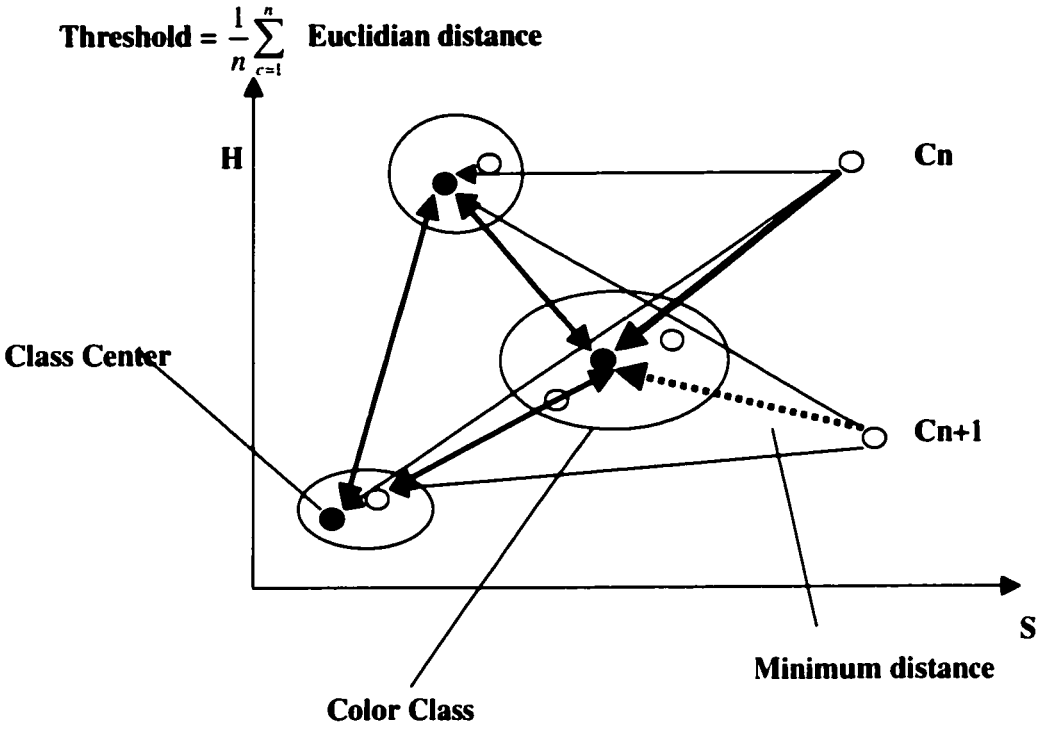
Now we need to decide if the minimum distance can be regarded as a new color class or it belongs to color class C1 and C2. Normally, we compare this distance with the exit color class distance, and use a threshold to decide if a new color region should be created. For example, if we use half distance of C1 and C2 as a threshold, then if the maximum minimum distance is larger than this value, we create a new color class, say C3. Otherwise, we will put this pixel into the nearest color class, such as C4 will belong to class C1.

Threshold = (Euclidian distance of C1 and C2)/2

Euclidian distance of C1 and C2 =($(Hc1-Hc2)^2+(Hc1-Hc2)^2$)$^{1/2}$

When we have more than two color classes, the threshold will be calculated as the average Euclidean distance of all color class center distance. Then we

repeat a similar process to calculated the remaining pixels and compare the distance to decide what should be done for this pixel: creating a new color class or assigning it to an existing color class.



Example of Multiple color class in HS coordinator

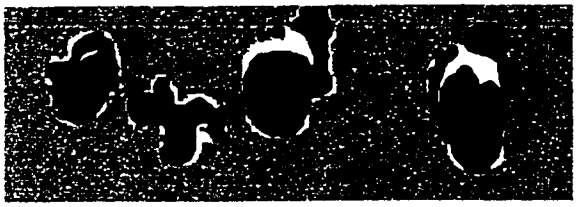
3.3 Post-Processing

Clustering Algorithms assign similar color pixel to one color region, but there are still some "noise " that need to be deleted. For example, a spot in a big pure domain region can be thought as a "noise" even though this spot is not a real noise. So a noise filtering procedure is used to move a noise or spurious point inside an uniform region in a clustered image. Before further discussion of removing, we must give definition of some of the terms.

	Definition	Processing
Noise Point	Its clustered color is different from its neighbors which all neighbors share the same clustered color.	Replace its clustered color by its neighbor clustered color.
Spurious Point	Its clustered color is different from its neighbors , but its neighbors share more than one clustered color	Replace its clustered color by the major color of its neighbor.

The following two images show that some black "noise " has been removed out and replaced by the domain color.

Clustered image after clustering



Clustered image after Noise cleaning

Pixels with the same color means that they belong to same color region. Two images denoted that their neighbors have replaced some small clustered points, and results in image with more color integrity.

4. Region Generation

4.1 Region Creating:

A region consists of all connected pixels that belong to same color class. Although we have classified similar color pixels in one color class, these pixels may be located in different area of a image, so distinguishing this region is very important as we assume that one region can be only one object and human face can be only located in one area. We can separate this into following steps:

Step (1): Applying region algorithm to find all regions in one color class.

Step (2): Using structuring element to morphology the separated region, and any pixels left will be regarded the next candidate region.

Step (3): Return to step 1 and generate the next color class until all color classes are generated.

Step (4): Record all generated regions.

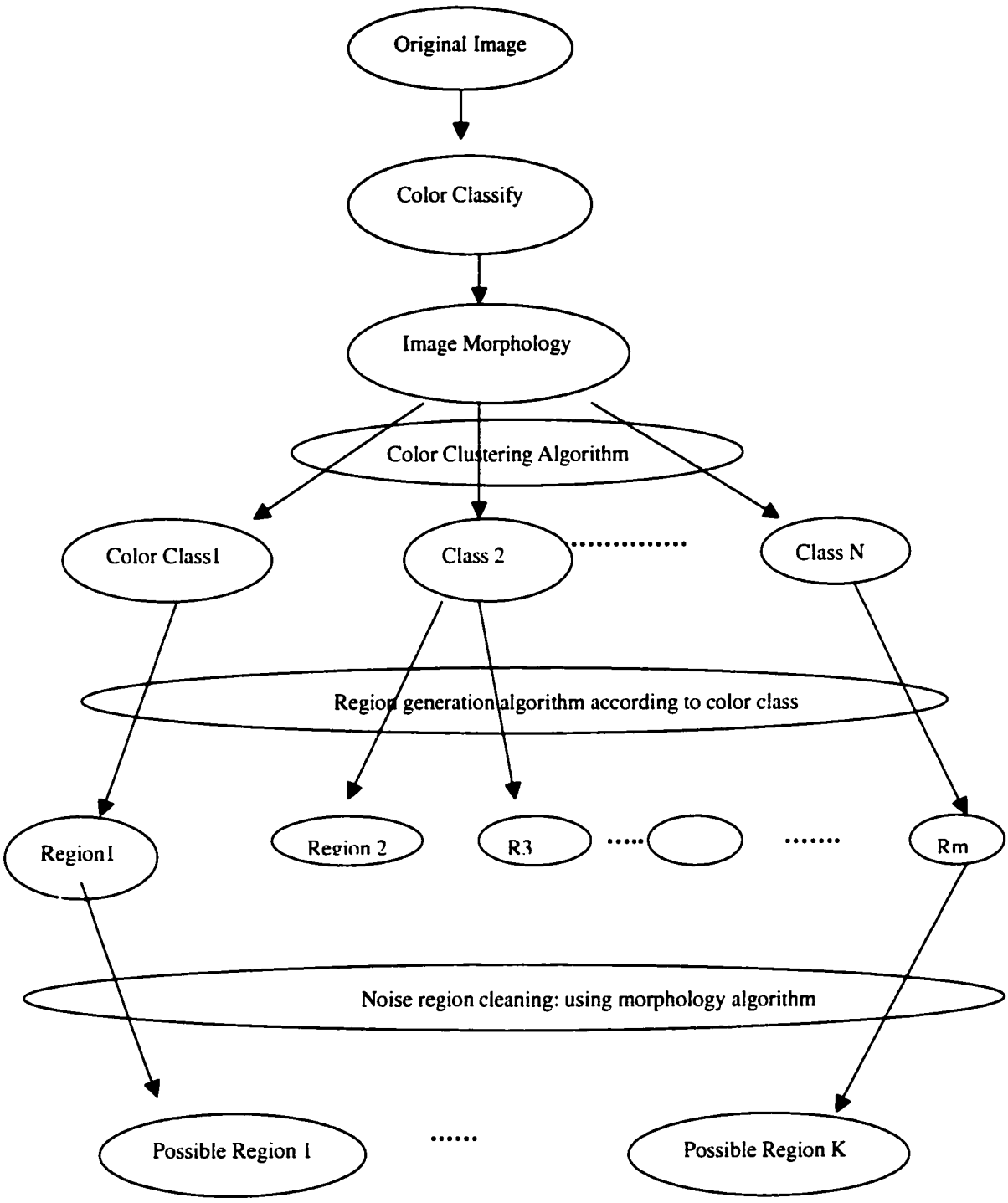
Region algorithm Pseudo code:

1. Select one color class from color classes
2. Take one pixel from the color class, and put it in one region as the first pixel of this region class, then delete this pixel from the color class.
3. Take another pixel from the color class, and try to find if there is any neighbor existing in the region class.
4. If neighbor exists, then put this pixel in region class and delete this pixel from the color class.

5. If neighbor does not exist, take another pixel and try the above testing again.
6. If no pixel has neighbors in the region class after scanning total remaining pixels inside the color class, then one region is generated.
7. If the color class is blank, the one region is generated and this color class has been finished.
8. Try remaining color class until all color class are scanned.

Now step 3 above algorithm time-consuming computation if we do not select a good algorithm. The simplest algorithm is to check if one pixel in the checking list is a neighbor pixel inside the region. This computation scale will increase very quickly. For example, suppose that we want to generate a color class with 10,000 pixels. In the first step, we only compare 9,9999 pixels in the color class with one pixel in the color region, and the computation is 99,999*1=99,999 for one round. Now if we have 5000 pixels in the color class with 5000 pixels in the color region, one round will become to 5,000*5,000=25,0000,000. So finding an appropriate method is necessary.

A possible method is to use boundary checking instead of all pixels checking. That means we only check the region's boundary pixels with the color class pixels for any possible neighbor pixel check. Now we encounter another problem: any time when one region merges with another pixels, the region boundary will also be changed.



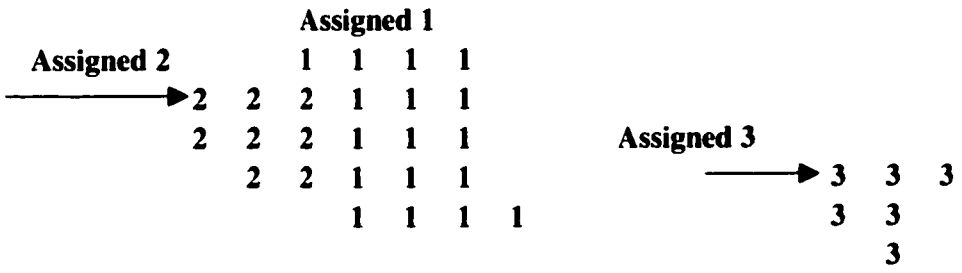
a structure diagram of Region generation

4.2 Effective Region Detection:

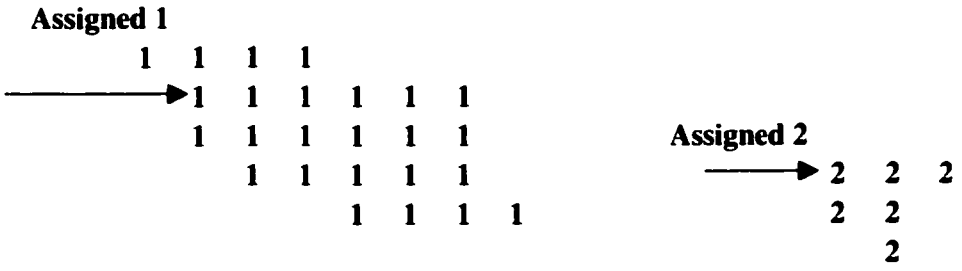
4.2.1 Region Generation Algorithm

There are lots of region detection methods, one effective method to create regions is to perform connected-component labeling (or region coloring) which results in each

region of an image being assigned a different label (or color)[13]. The method involves two-pass process where pixels are individually labeled. First, the image is scanned in raster order, and then each pixel is examined. For each ON-valued pixel, the previously labeled pixels to the left and above is examined. If none is On, then the pixel is set to a new label value. If one of these is On, the current pixel is given the same label as that pixel. If more than one are ON, then the pixel is set to one of the label, and all label are put in an equivalence class (that is, thy are stored for later merging). At the end of this pass, the number of connected components is the number of equivalence classes, plus the number of labels not in an equivalence class. The second pass consists of first merging all labels in each equivalence class to be the same label and then reassigning all labeled pixels in the image to these final labels.[12]



Region detection Algorithm: step 1



Region detection Algorithm: step 2 (merge region)

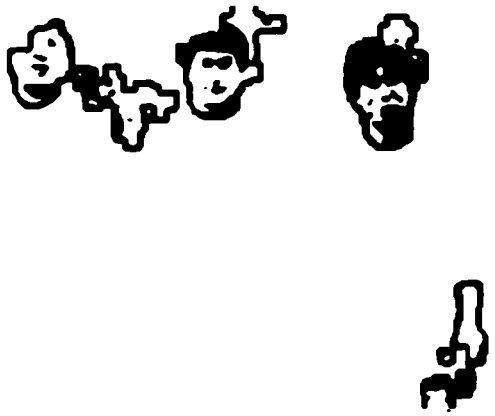
This region detection has been used in our face location algorithm. In the following

processed image, all black regions are detected regions.

Detected Region

4.2.2 Face Region Detection

The detected regions may be any size, but human face must have minimum size for recognition. In other words, the face recognized must bigger than a predefined size. Further more, face region's length and width size must meet some condition. For example, a face 's ratio or length or width must less than 2. A lot of possible face limitation can be used to detect if an isolated region is a face region. The following left isolated regions are regions that meet our face conditions, but this does not mean they are face regions.



Possible face region

4.3 Region Separation

As we use skin color to detect face region, when the background has the similar color values, it will also be regarded as possible face region. If the fraud region is an isolated region, we can use next steps to detect if it is background or real human face. Normally, we will check if it has eyes or mouth feature inside the possible region. In order to focus our searching range, we always try to find the center of this region and try to find human features around the center. So the first problem we need to solve is to find the center of a region. An effective and common used method is eigen vector.[2] We will discuss this later. Now we encounter another problem. If a region is almost a human face or background region, the use of this center searching method can also help in face feature detecting. If a region is a combined region, that means face-likely background are merged with face region, center obtained by eigen vector will distort the face region. So distinguishing big background region and face region is very useful in future face processing.

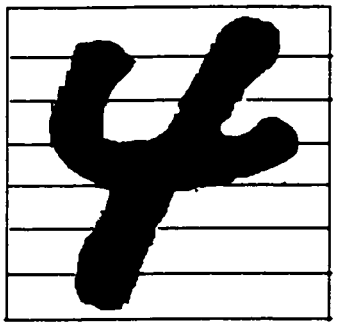
A common feature of merged regions is that they are connected by a thin linker region compare with main face color and background. The reason is that human face is normally a similar big size, for example, a circle or broad ellipse. That means the width of pixels from one side to another side must bigger than a minimum length which can be detected. In fact, a too small face region is no use in face location, as we could not separate necessary information from a tiny face, so in our system, we define a minimum face size. If we define the minimum size for a face is 30*30 pixels, we know that any horizontal or vertical pixels numbers will bigger than 30 for a frontal face. In order to

reduce this deviation for a profile, we can define the value as 10 or other reasonable value.

Now let us consider the background region. A background region is a random region. For example, its outline is random, and its horizontal or vertical pixels are longer than minimum face horizontal or vertical pixels. Now according to our assumption, any region will be connected by less than minimum pixel linker is not a face. This feature can help us separate a region to more regions. So we can separate merged region to several regions, and usually, one separated region must include a face if the merged face has a face inside.

The above image shows than a face is merged with a background region. Now we use histogram to seperate this merged region. Histogram method is to calculate the horizontal or vertical pixels inside a picture.

Histogram Feature



Horizontal Histogram feature

Vertical Histogram feature

The formula of Histogram feature is:

Horizontal Histogram:

$$HH1(i,j) = \sum_{x=i*n}^{i*n+n} \sum_{y=0}^{m} p(x, y)$$

Vertical Histogram:

$$VH1(i,j) = \sum_{y=i*m}^{y=i*m+m} \sum_{x=0}^{m} p(x, y)$$

$$VH(I,j) = \frac{VH1(i, j)}{MaxVH1(u, v)} \qquad for\ all\ u,v$$

for i=0,1,.....,(N -1)

j=0,1,....., (M-1)

The following are the results of using the histogram method to separate the above mentioned merged region into two regions, face region with background region. If a background is merged strongly with a face that means the link width is equal or larger than a face width. This method cannot solve strong merger. We do not need to worry about this condition because a random background seldom have a width feature like a human face.

A face region is separated with another noise background region

4.4 Region Identification

4.4.1 Region Cleaning by Isolated Region

As we have generated and separated region from an image, the next step is to process a face possible region. If we think any isolated region is a face region, this will make our computation incredible large. So identifying a region is also important. A very simple rule for identification is its size specification. For example, a circle with 10 pixels radius is not a face if we defined the minimum radius for a face is 30 pixels. Now we identify a region according to the following conditions:

Assumption: the minimum face width or height must be larger than 30 pixels.

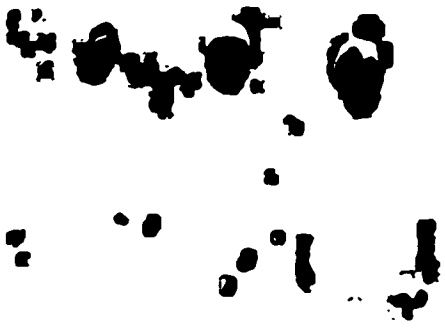
Condition:

1: Any region less than 900 pixels is not a human face region.

2: Any region with a width or height less than 30 is not a human face region.

3: As human face or head is an ellipse, and the ration of major and minor axis for head or face must be in a range.

In our region identification, we use the above mentioned three basic rules. These rules can be adjusted according to different situations.

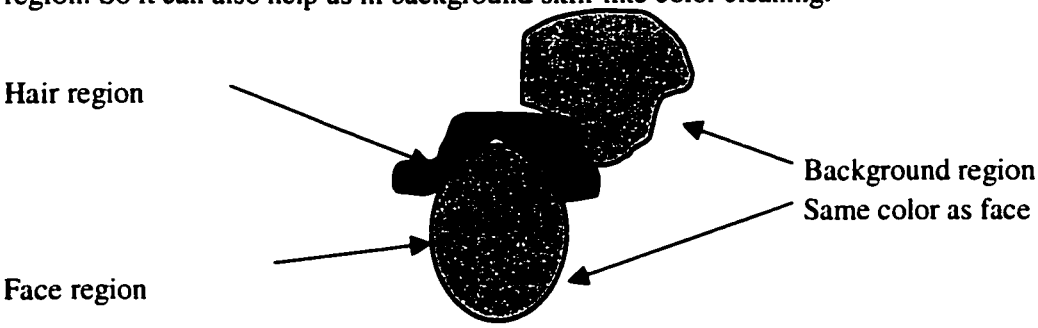


Regions before region identification

Regions after region Identification

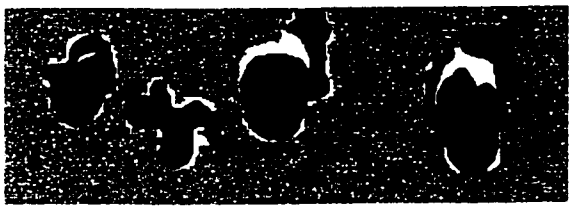
4.4.2 Region Cleaning by Clustered Region

To clean a classified and clustered image is also necessary. Because one big region can be comprised of several different regions, and each region is composed of similar pixels, so an integrated region can be analyzed to consist of several regions. This concept is very useful in face region identification. Two big distinguishing and different regions in a face are hair and face, and what we want to get is face not hair. Although people have moustache and eyebrow, this feature is included in face structure, and this feature can also be cleaned out as hair. We can restore all these features after we get the outline of face. As we can take hair out, so region without hair is more like a face if it is a face. Here we will use face detection to identify any clustered as a possible face or not by its size and outline. Another information hair can give us is that some noise regions which have the same color of face will be separated out when we take out the hair clustered region. So it can also help us in background skin-like color cleaning.

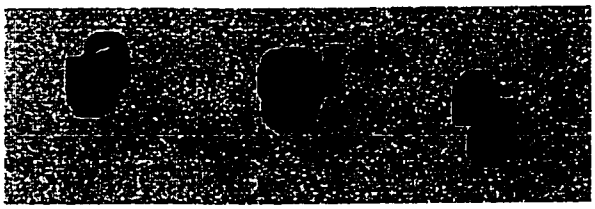


A face image comprised of three region

The processed results can be seen as follow:



Clustered region

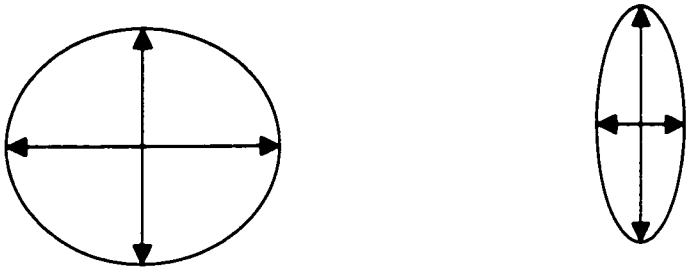


Cleaning after clustered region

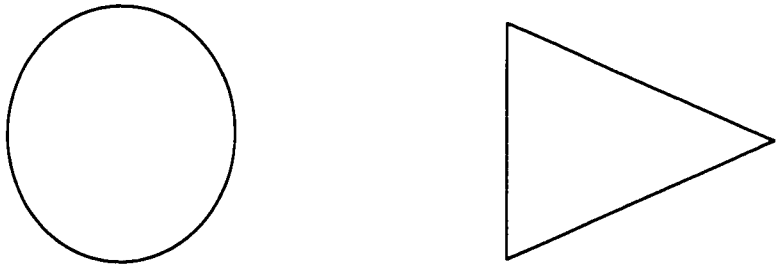
4.5 Shape Detection:

Moment is a simple and effective shape detection method. The first moment is the average of (x,y) location for all pixels in the region. This x and y moments are $m_x^{(1)}=(1/n)\sum X_i$ and $m_y^{(1)}=(1/n)\sum Y_i$, where i=1,..., n is the number of ON pixels. The first moment describes the average location of an object. The second moment is the average of the squares of x and y locations, and so on. The central moments normalize the measurement with respect to the first moment to make it independent of location. The second central moments in x and y are $m_x^{(2)}=(1/n)\sum (Xi - m_x^{(1)})^2$and are $m_y^{(2)}=(1/n)\sum (Yi - m_y^{(1)})^2$. These moments can be used to indicate the non-roundness, eccentricity, or elongation of the shape; for example, if the second moment in x and y is similar in value, then the object is more round than otherwise. The third central moment gives a measure of the lack of symmetry of this eccentricity. Besides these central moments, moments of different

orders can also be combined into functions that are invariant to other geometric transformations, such as rotation and scaling; these function are called moment invariant [12].



Compactness-ratio of area to square of contour length, ratio of major to minor axes, or second central moment.



Asymmetry-third central moment. The circle has a value of zero indicating perfect symmetry , and the triangle will have a non-zero value.

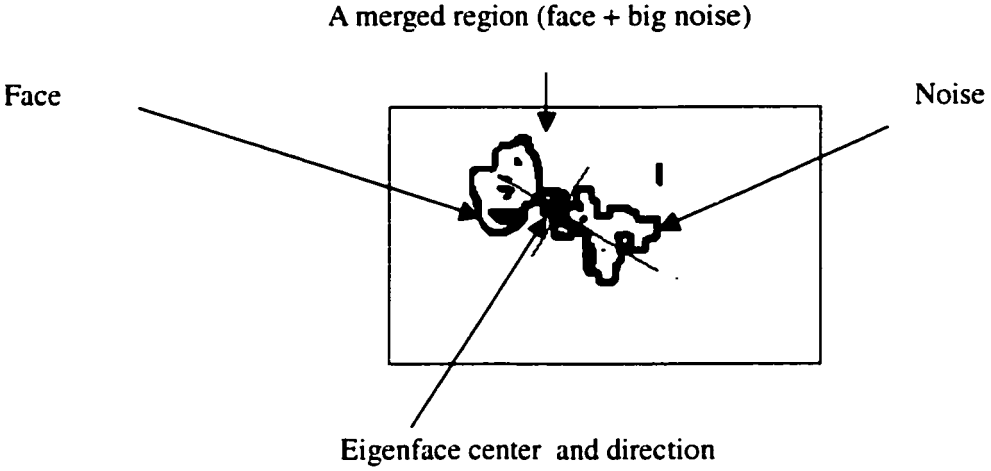
4.6 Ellipse Detection

4.6.1 Random Hough Transformer (RHT)

After skin color detection and region generation, we can get some isolated regions. These regions can be categorized to three kinds of regions: face region, background region, and combined region. A combined region includes face and background image. Eigenface[2] has broadly used in face location as they assume all region must belong to face or non-face region. If eyes, nose, or mouth features can be found in this region,

then this region is a candidate face region. Otherwise, it is a non-face region.

Eigenface can detect a region if it is a face likely region. If a face has big background beside, or a face is merged with a big skin-color liked region, an Eigenface method will give us a wrong center and direction. If we try to find a face feature based on this incorrect face center, we would miss face feature and get false conclusion.



Detect ellipse shape from the region can distinguish face and non-face even though they are merged in one region. A face is approximately ellipse, and the center of the face ellipse can be used as the original to detect the other features in face. Feature number may be different for frontal face or profile face, for example, front face has two symmetry eyes and one mouth, but for a profile face, sometimes one eye and mouth can be found in the image. Center of ellipse can do the same work as face location, but it can avoid any incorrect center when using eigenface method.

Hough transform is a very useful method to detect lines and circles. But standard Hough transform is time consuming and unrealistic. For example, the parameters for lines detection are ρ and σ.

Hough transform for line: $\rho = x \cos\theta + y \sin\theta$.

In order to detect all possible lines in an image, we have to try all possible ρ and θ [0-360] . This detection is time consuming. For a line, we only have 2 parameters. If this is an ellipse, they have 5 parameters.[5]

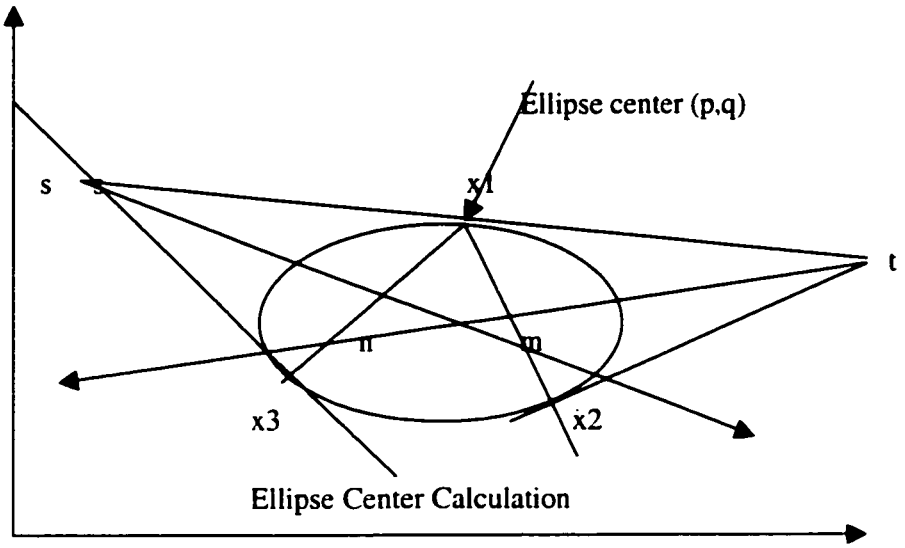
Ellipse formulation: $a(x-p)^2+2b(x-p)(y-q)+c(y-q)^2=1$

With the restriction that $ac-b^2>0$ in this equation.

Now we need to calculate 5 parameters: p,q, a,b ,c. This is a non-liner equation and solving for these 5 parameters impractical. The convenient Hough transform cannot be used any more. A variation named Variant Randomized Hough transform has been invented to detect ellipse and used in our ellipse detection [15].

In our algorithm, we randomly choose three pixels and obtaining estimates tangent at each pixel. The three pixels and two tangents can help us to find the center of the ellipse (p,q) and the remaining three parameters: a, b, and c.

To find the ellipse center, we use a feature of ellipse geometry noted by Yuen et al. (1989)[18]. Take two points on the ellipse and find their midpoint m, and the intersection of their tangent t. The ellipse center will lie on the lien of tm. Now we get another line by using the same method above named sn. As ellipse center lie on line of tm and sn, so the intersection of tm and sn will be the center.



After we get the center point of ellipse, we get two parameter values out of five. Now we need to calculate the remaining three. First, we need to translate the ellipse to the origin. So the ellipse equation is reduced to the following:

$ax^2+2bxy+cy^2=1$

Substituting in the coordinates of the three pixels (X1=(x1, y1), X2=(x2,y2), X3=(x3,y3)), we obtain the following system of three simultaneous linear equations:

$$\begin{bmatrix} x1*x1 & 2x1x2 & y1*y1 \\ x2*x2 & 2x2y2 & y2*y2 \\ x3*x3 & 2x3*y3 & y3*y3 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

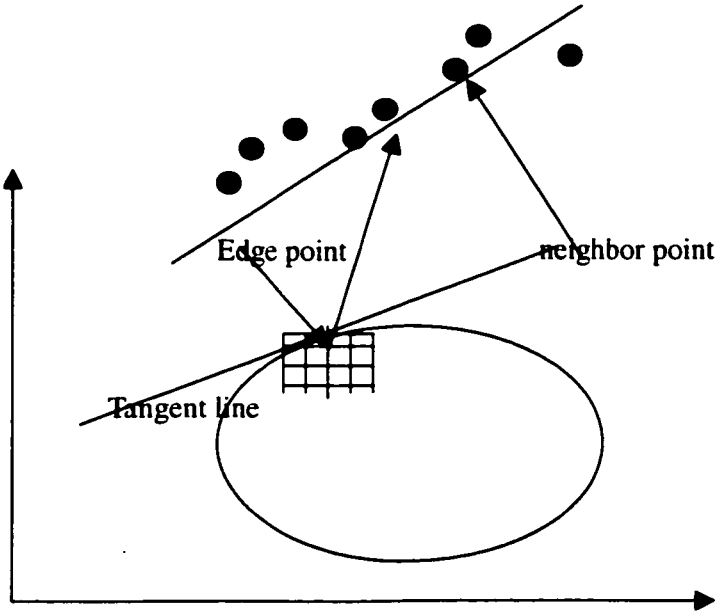
From this matrix, we calculate the value of a, b, and c. Now we need to check if x1, x2 and x3 belong to same ellipse. If $ac-b^2<=0$, then these three pixels do not belong to one ellipse, so we need to try other three pixels.

The great problem affecting the RHT is that its performance is poor when the image is noisy and complex. For example, the three pixels whatever we used to detect an ellipse may include two noisy points inside. If three points are very close, then this possibility will be high. So we need to modify and improve the performance of RHT. The simplest

modification is RHT with distance criterion named RHT_D. The first point will be sampled random but the other points are sampled from all points that are within certain distance (greater than dmin and less than dmax)[19]. The If we want to find an ellipse, three random pixels must be sampled from the edge set. In order to avoid three pixels that are too close, three pixels must have at least one unit distance. So what we used here is an optimal randomized Hough transformer.

4.6.2 Least Square Line

In the above section, we use tangent line to get the ellipse center. Now we need to find a way to calculate the tangent line in an ellipse. As there are some noise points beside an ellipse point, so getting an accurate tangent line is very important, as our next steps depend on this tangent line. The method we used here is least square method. This is done by defining a small neighbor around the pixel and find the line of best fit to those pixels within its neighbors [15].



The least-squares line uses a straight line

y=a+bx

to approximate the given set of data, (x_1,y_1), (x_2,y_2), ..., (x_n,y_n), where $n \geq 2$. The best fitting curve $f(x)$ has the least square error, i.e.,

$$\Pi = \sum_{i=1}^{n} [y_i - f(x_i)]^2 = \sum_{i=1}^{n} [y_i - (a + bx_i)]^2 = \min.$$

Please note that a and b are unknown coefficients while all x_i and y_i are given. To obtain the least square error, the unknown coefficients a and b must yield zero first derivatives.

$$\begin{cases} \dfrac{\partial \Pi}{\partial a} = 2\sum_{i=1}^{n} [y_i - (a + bx_i)] = 0 \\ \dfrac{\partial \Pi}{\partial b} = 2\sum_{i=1}^{n} x_i [y_i - (a + bx_i)] = 0 \end{cases}$$

Expanding the above equations, we have:

$$\begin{cases} \sum_{i=1}^{n} y_i = a\sum_{i=1}^{n} 1 + b\sum_{i=1}^{n} x_i \\ \sum_{i=1}^{n} x_i y_i = a\sum_{i=1}^{n} x_i + b\sum_{i=1}^{n} x_i^2 \end{cases}$$

The unknown coefficients a and b can therefore be obtained:

$$\begin{cases} a = \dfrac{\left(\sum y\right)\left(\sum x^2\right) - \left(\sum x\right)\left(\sum xy\right)}{n\sum x^2 - \left(\sum x\right)^2} \\ b = \dfrac{n\sum xy - \left(\sum x\right)\left(\sum y\right)}{n\sum x^2 - \left(\sum x\right)^2} \end{cases}$$

where $\sum \cdots$ stands for $\sum_{i=1}^{n} \cdots i$. [17]

Suppose we get two edge point (x1, y1) and (x2, y2), and the tangent line is: y=a+bx.

Now we have two equations:

Suppose two tangent lines are:

y=a1+b1x

y=a2+b2x

The intersection point for two tangent lines is:

tx=(a1-a2)/b2-b1;

ty=a1+b1(a1-a2)/(b2-b1)

The midpoint of two edge points is:

mx=(x1+x2)/2

my(y1+y2)/2 ;

By using the position of intersection point and midpoint, we can construct line1's equations that pass through the center point of the ellipse. We can get line2's by using the same way, and the intersection point is the position of ellipse center.

4.6.3 Edge detection

Now we know how to find ellipse from an image. But before our starting, we need an edge set to choose random edge points. Finding the edge lines or points from an image is the first step to find ellipse.

An edge is the boundary between two regions with relatively distinct gray-level or color-level properties. Our next analysis will base on gray-level discontinuities region that can be distinguished. Basically, the method used here is based on the computation of local derivative operator.

The most common method of differentiation in image processing application is the gradient [20].The gradient of an image f(x,y) at the location of (x,y) is the vector of

$$\nabla f = \begin{bmatrix} Gx \\ Gy \end{bmatrix} = \begin{bmatrix} \partial f / \partial x \\ \partial f / \partial y \end{bmatrix}$$

Approximate formulation is as follows:

$$\nabla f \cong |Gx| + |Gy|$$

The direction of the gradient vector is also an important quantity. Let $\alpha(x, y)$ represent the direction angle of the vector ∇f at (x,y). Then from the vector analysis,

$\alpha(x, y) = \tan-1\left(\frac{Gy}{Gx}\right)$ where the angle is measured with respect to the axis[7].

Derivative can be implemented in digital form in several ways. However, the sobel operator has the advantage of providing both a differencing and a smoothing effects. Because derivative enhance noise, the smoothing effect is a particularly attractive feature of the sobel operators.[7]

The masks are 3*3 masks.

M=

Z1	Z2	Z3
Z4	Z5	Z6
Z7	Z8	Z9

Mx=

-1	-2	-1
0	0	0
1	2	1

Mx=

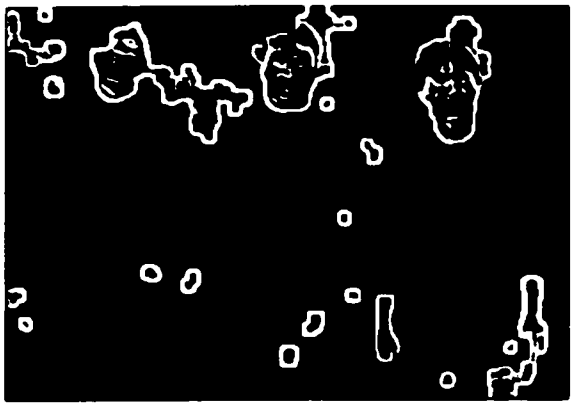
-1	0	1
-2	0	2
-1	0	1

Gx=(Z7+2Z8+z9)-(z1+2Z2+z3)

Gy=(Z3+2Z6+Z9)-(Z1+2Z4+Z7)

The gradient(x,y) = |Mx*I|+|My*I|, I is the image

The direction of the edge , $\alpha(x, y) = \tan -1\left(\frac{Mx * I}{My * I}\right)$



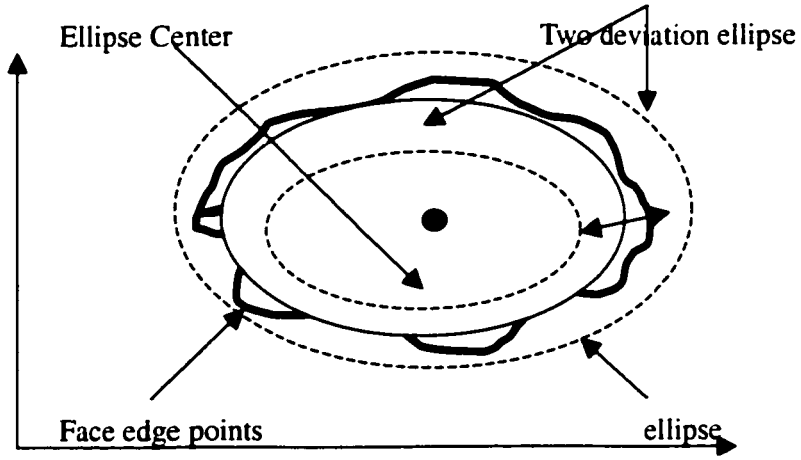
Edge detection based on unclustered Image

The above edge is the result of edge detection from original image. We can find there are a lot of noises inside the image and these noises will strongly increase our computation time and reduce our accuracy. An optical method to improve this is detecting edge after color clustering and region cleaning. The following image has less useless edge information than the previous one.

Edge detection based on clustered Image

4.6.4 Maximum Probability Face Ellipse

As face's edges are not perfect ellipse, the ellipse which can math the most edge points in the image will be regarded as the face structure. So we will give a deviation for our ellipse. Any edge point that falls into this deviation will be regarded as part of the ellipse point. In other words, we can think that an ellipse has a very thick edge line.



The formulation for an ellipse center at the origin is :

$$\frac{(x-p)^2}{a^2}+\frac{(y-q)^2}{b^2}=1$$

If p=0, q=0, then it is an ellipse with its center at the origin, so the formulation will be :

$$\frac{(x)^2}{a^2}+\frac{(y)^2}{b^2}=1 \qquad (1)$$

If the ellipse is rotated by an angle θ, we know than

$x=x\cos\theta-y\sin\theta$ (2)

$y=-x\sin\theta+y\cos\theta$ (3)

substitute 2 and 3 into formulation 1, we get the formulation 4:

$$x^2\left(\frac{\cos^2\theta}{a^2}+\frac{\sin^2\theta}{b^2}\right)+xy\left(\frac{2\cos\theta\sin\theta}{b^2}-\frac{2\sin\theta\cos\theta}{a^2}\right)+y^2\left(\frac{\sin^2\theta}{a}+\frac{\cos^2\theta}{b^2}\right)=1$$

(4)

From 4.6.1, we know the equation of an ellipse center at the origin is:

$$Ax^2+2Bxy+Cy^2=1 \qquad (5)$$

As equation 4 and 5 denote the same ellipse, so we can write:

$$\sin\theta\cos\theta\left(\frac{1}{b^2}-\frac{1}{a^2}\right)=B \qquad (6)$$

$$\left(\frac{\cos^2\theta}{a^2}+\frac{\sin^2\theta}{b^2}\right)=A \qquad (7)$$

$$\left(\frac{\cos^2\theta}{b^2}+\frac{\sin^2\theta}{a^2}\right)=C \qquad (8)$$

Plus 7 and 8 , we have

$$\left(\frac{1}{b^2}+\frac{1}{a^2}\right)=A+C \qquad (9)$$

Square equation 6, we get

$$\sin^2\theta\ \cos^2\theta\left(\frac{1}{b^2}-\frac{1^2}{a^2}\right)^2=B^2 \Rightarrow \sin^2\theta\ (1-\sin^2\theta\ )\left(\frac{1}{b^2}-\frac{1}{a^2}\right)^2=B^2 \qquad (10)$$

From equation 7, we get

$$A=\frac{1}{a^2}+\sin^2\theta\left(\frac{1}{b^2}-\frac{1}{a^2}\right)$$

$$\sin^2\theta=\frac{A-\frac{1}{a^2}}{\frac{1}{b^2}-\frac{1}{a^2}} \qquad (11)$$

Using equation 11 to substitute for 10, we get

$$(A-\frac{1}{a^2})(\frac{1}{b^2}-A)=B^2 \qquad (12)$$

From 9 and 12 , we can get the value of a and b

Let $x=\frac{1}{a^2}, y=\frac{1}{b^2}$

We get, x>=0, y>=0

$$y^2-y(A+C)+(AC-B^2)=0$$

$$x=A+C-y \qquad (13)$$

As Y>=0, so

$$Y=\frac{(A+C)}{2}\pm\sqrt{\left(\frac{A+C}{2}\right)^2-(AC-B^2)} \qquad (14)$$

When we get values of a and b, from equation 11, we can get θ

5. Face location:

5.1 Eyes Detection

Face region verification will determine if any possible face region has face features inside the region. The most salient feature in a face is eyes feature. Although we have other significant geometry features, such as nose, nose hole, and mouth, these features can be blurred under light shinning direction, face direction, and lift styles. A good example is mouth which is thought as a stable feature. In paper [8], it says "the mouth region contains more red component and smaller blue component than other facial regions. Hence, the chrominance component Cr is greater than Cb near the mouth areas ". The rule seems correct to distinguish mouth from a face region. Now the question is, today, many people use lipstick for making up, and most of them use

special color, such as blue or yellow. If people have yellow lipstick in their lips, this rule will be incorrect, and lipstick is widly used almost everyday by most ladies. No matter what color they are using, people can immediately recognize a person face, but a strict condition will not work under this tiny situation change.

A steady feature is eyes. We all know that eye's color is darker than other parts of a face. Eyes are seldom decorated except sunglasses or a near sight glasses is wearied. Normally, the color of sunglasses is always dark, and near sight glasses is always transparent. So with or without glasses, eyes parts color is always darker than other parts of a face. Furthermore, no matter people are opening or closing their eyes, this parts color is also dark. An opening eye has dark iris inside the eye, and a closing eye has a dark line. So dilating eyes dark color feature can help us to find eye feature in a face image.

5.1.1 Morphological Dynamic Link Architecture

"An approach that exploit both the grey-level information and the geomerical one is called dynamic link architecture (DLA)"[21]. Dynamic based on multiscale morphological dilation-erosion is called morphological dynamic link architecture (MDLA).

The formulation for a gray-dilation image is as follows:

Dilation

Gray-scale dilation of f by b , denoted as $f \oplus b$, is defined as

$$f \oplus b\,(s,t)=\max\{f(s-t, t-y)+b(x,y) \mid (s-x), (t-y)\in D_f ; (x,y)\in D_b\}$$

Erosion

Gray-scale erosion, denoted as $f \ominus b$, is defined as

$f \Theta\ b\ (s,t)=\min\{f(s+x, t+y)-b(x,y)|(s+x), (t+y)\in D_f\ ;\ (x,y)\in D_b\}$[7]

MDLA can be used to substitute the time consuming Gabor-based feature vector that rely on floating point architecture.(i.e. FFT's or convolution).'[21] Let R and Z denote the set of real and integer number . Give an image $f(x):D\subseteq Z^*Z\rightarrow R$ and a structuring function $g(x) : \xi\subseteq Z^*Z\rightarrow R$, the dilation of the image f(x) by g(x) is denoted by $(f\oplus g)(x)$. Its complementary operation, is denoted by $(f\Theta g)(x)$. The multiscale dilation-erosion of the image f(x) by $g_\delta(x)$ is defined by [22]:

$$(f * g_\sigma)(x) = \begin{cases} (f \oplus g_\sigma)(x), if\sigma > 0; \\ (f(x), if\sigma = 0; \\ (f \oplus g_\sigma)(x), if\sigma < 0) \end{cases}$$

when $\delta<0$, dilation is erosion as $\delta>0$, so we derive this formulation as:

$$(f * g_\sigma)(x) = \begin{cases} (f \oplus g_\sigma)(x), if\sigma > 0; \\ (f(x), if\sigma = 0; \\ (f \Xi g_{|\sigma|})(x), if\sigma < 0) \end{cases}$$

Where δ denotes the scale parameter of the structuring function. Several structuring functions can be chosen. Typical scaled structuring functions includes[22]:

- "flat " structuring functions $g\delta(x) =0\ x\in G$ particularly scaled disks where $G=\{x||\leq|\delta|\}$,
- spheres $g\delta(x) =|\delta|\ ((1-\|x/\delta\|2)1/2-1)\|x\|\leq\delta$,
- circular poweroids $g\delta(x) =-|\delta|\ \|x/\delta\|\alpha)\alpha>1$, particularly the circular paraboloids $\alpha=2$.

We use flat structuring function, and a scale-recursive method to calculate the relevant values.

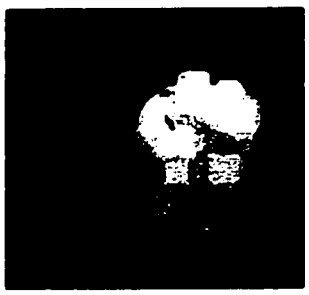
$$(f \oplus g_{\sigma+1})(x_1, x_2) = \max\left\{\begin{array}{l}(f \oplus g_{\sigma})(x_1, x_2) \\ \max\limits_{(z1,z2)\in \Delta G(\sigma+1)} \{f(x_1 + z1, x_2 + Z2)\} \\ \max\{f(x_1 \pm (\sigma+1), x_2 \pm (\sigma+1))\}\end{array}\right\}$$

Where $\Delta G(\sigma+1)=\{(z1,z2)\in Z*Z:(z1*z1+z2*z2)>\sigma^2,\ (z1*z1+z2*z2)<(\sigma+1)^2,\ |z1|\leq\sigma,\ |z2|\leq\sigma\}$[21].

The following image is the result of erosion in a face image. From the image, we can see that two eyes inside the face are two black holes surrounded by light area. We also find that mouth is also blacked by erosion because mouth has a darker color than faces other parts. As we mentioned before, eyes are more stable features than mouth. But if we can find eyes and mouth around the center of a face, the probability of being a face region will be raised.

Erosion in face region after MDAL.

Now can we apply dilation to white parts that is mentioned in [22]? We tried it and found that it is very unstable. Dilation can enhance the white part in a face, and pixel around eyeball is really white, why does not it work? It dilates white eyes to white hole if under a perfect image, which must have only white pixel in the eyes parts. But in the real environment, light can shine most parts of a face, sometime this parts are whiter than eyes, especially when we have a photo taken under strong light condition. So MDAL dilation cannot give us more accurate information than erosion



Several parts been dilated (white parts)

Original image (after binary) **Dilated imaged**

5.2 Region Center Detection

5.2.1 Eigenvector Detection

A region center can be located by using eigenvector and eigen values. It is also called as hotelling transform, principal component or discrete Karhuinen-Loeve transform. In order to get the eigen center, the first step is to calculate its mean vector Mx and covariance matrix Cx.

If we define a population of random vectors forms as:

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_n \end{bmatrix}$$

The mean vector of the population is defined as

$$M_x = E\{x\} = \frac{1}{M}\sum_{k=1}^{M} x_k$$

$$C_x = E\{(x - m_x)(x - m_x)\} = \frac{1}{M}\sum_{k=1}^{M} x_k x_k^T - m_x m_x^T$$

Where M is the pixel number of region, and Xk is the coordinate vector of the kth pixel, and T means a transpose of a vector. The square root of the eigen value is the major and

minor axis value of ellipse. The eigenvector is the direction of the ellipse and the intersection of the eigenvector is the center of ellipse.

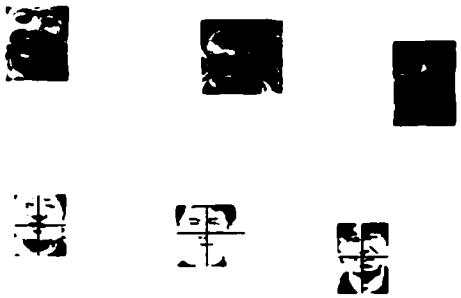
Use eigenvector to detect the candidate face region.

6. Face verification:

6.1 Local threshold:

Face features always have different colors compared with face skin. For example, iris, eyebrow, nose hole and mouth are always darker than skin color. This darker feature can be extract from a face so that we can verify a face by checking if it includes these features. A color image will be first degraded to a white-black for feature extraction. Then, face feature will be retained and skin will be eliminated in the same time. A common method is to set a threshold and compare the binary value of the photo pixels on the photo. Any pixel with value less than the threshold will be retained as face features pixel value, and its value is smaller than normal skin pixel value. Now the question is how to set the threshold, and can one fixed threshold be used in all photos? As varied intensity distribution in a photo is very common, such as a face with different light on each side, a fixed threshold is not suitable in all lighting conditions. The second question is how to calculate this threshold? Because we have cropped possible face region form the photo, so the average binary value of this possible region can help us calculating the threshold. Then we select a percent value of average value as threshold

and let pixel value less than this will be set to black point (0), and all other points will be set to white (255 in a 8-bites binary white-black color system). The following images show the results of this idea.

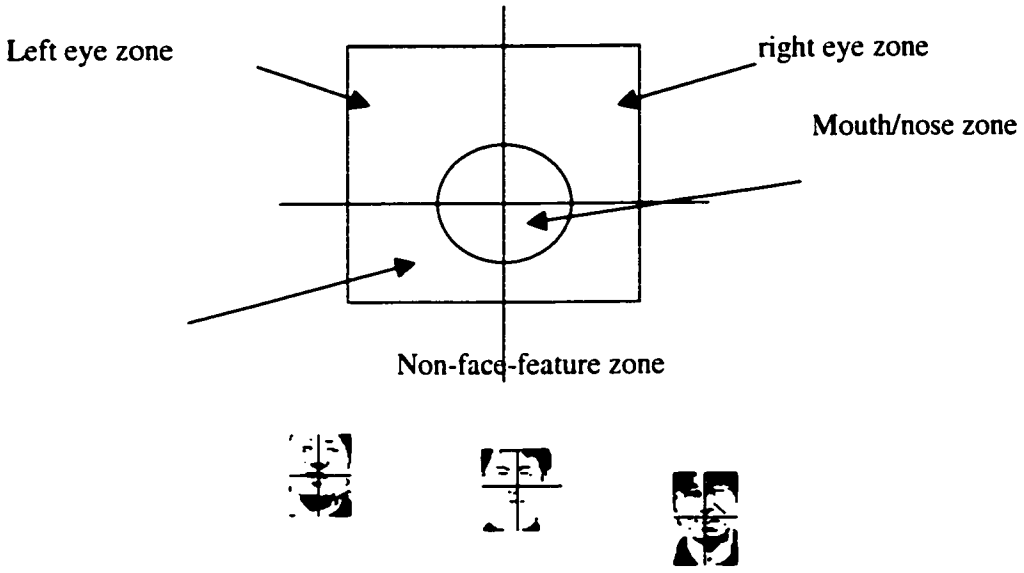


Original Image **Image after processed by a threshold**

6.2 Facial feature extraction

Eyes and mouth are the blackest points in a binary threshold image. If we can extract accurate face and mouth, we can verity it is a face. The position of eyes and mouth can help us to distinguish if a black component is a eye or mouth. If we know the center position of a face, we can define any component in the up-left of the center point is left eye, and any component in the up-right is a possible right eye, and any component lower than the center point is mouth. As nose and mouth are adjacent organs on the face and they are located in the center of a face, so mouth and face will depend on the location of center. In other words, their location might be higher or lower than its center point. In order to avoid the puzzle caused by the center point, we locate the center point of a face and define a circle around center point. Any component with its middle point inside this circle will be considered as mouth or nose. As eyes are relatively far, so if the radius of circle can be selected suitably, the middle of eyes will not fall into this mouth-nose zone. If we consider a rectangle zone show below, the middle circle, is a

mouth/nose zone, the up outside will be eyes zone, and down outside is non-face feature zone. That means if a component is located in a non-face feature zone, such as a chin, it will not be regarded as face feature.



A processed image with its eigen center is as follows:

6.3 Non-face-feature cleaning

In the above image, we can see some noise can still be found in a face. For example, hair, chin and neck. As we know that eye, nose and mouth are separate organs in a face, and they are separate black components in face. Any components of these features have a maximum ratio in the face. For example, a mouth cannot cover more than 10% of face zone in the normal condition, and eye and nose are smaller than a mouth. According to this analysis, any black component with over than 10% of the rectangle face zone will not be a face feature and should be cleaned out. This can help us to reduce the possible face feature identification.

The following image shows that some big black components, such as hair, chin, or neck components are moved out.

6.4 Face feature Identification

Left eye: middle point of a component in the upper-left outside the center circle, component size is bigger than 2 pixels, and its width is less than 1/3 of a face width plus height is less then 1/6.

Right eye: middle point of a component in the upper-right outside the center circle, component size is bigger than 2 pixels, and its width is less than 1/3 of the face width plus height is less then 1/6.

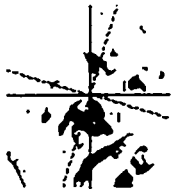
Mouth/Nose: middle point of component in the center circle, component size is bigger than 4 pixels, and its width is less than ½ of the face width.

Non-feature: Any component does not meet the above condition will be discarded.

6.5 Face verification

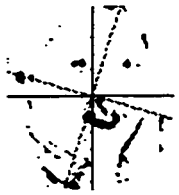
Face: at least one eye and one mouth/nose. As a face has maximum six features (two eyes, two eyebrows, one mouth as well as one nose), we will plus four additional features as well as six real features for any feature likely component. If a region has more than 10 features, it is not a face.

Original Image



Located Face

Face Rotation



Face Location before MDLK

After MDLK

Original Image

Final Located Face Image

In order to verify a rotated face has two eyes, we must rotate the face to the standard X-Y coordinate system. Then we use above mentioned face feature zone to verify eyes exist in the face image. We will use the eigen direction to rotate the image. This eigen direction is the direction of long axis of this eigen value (eigen ellipse). Eigen direction is not accurate when the face region are too small, as small image deviation can cause big distortion of eigen direction.

Face center can also be modified to calculate the middle position of verified two eyes. If the horizontal coordinator is not the same as the eigen center, the eigen horizontal coordinator will be replaced by the middle horizontal value.

6.6 Pose detection and Modification

Face pose detection will be detected by eigen direction. Then, we rotate the face and check if the two eyes are on the same horizontal level. If two eyes are on the same horizontal level, that means the eigen direction is just the pose direction. If their horizontal has difference, the angle of two eyes will be the deviation of the eigen direction. The pose direction will be modified by using eigen direction deduct the deviation angle.

7. Future work:

7.1 Face Location

7.1.1 Computational-saving Method

In order to extract human face from a scene picture, we use skin-color as the extract

tools. But after skin-color segmentation, there are still many image pixels that are left in the image. As image processing is computational time consuming, and lots of computation is wasted in unnecessary processing. Some good algorithms, including neuron network, have been used in the facial recognition, but hours or days of computation for one image processing make this algorithm unrealistic even though they have a high recognition rate.

In our project, we find clustering is the most computing consuming step in processing stages. As any pixel will calculate its Euclidan distance with all pixels in the image, the number of pixels is very important in time saving. Although we use morphology and separate method to delete useless pixels or regions from the picture, the remaining pixel number is still very large.

A lot of methods have been used for time saving, but motion seems more simple and convenient. Because humans are moving objects, and any still object is not interested in this project, especially if we snap image from a camcorder. "In imaging application, motion arisen form a relative displacement between the sensing system and the scene being view"[7]. One of the simplest approaches for detecting changes between two image frames f(x,y, ti) and f(x,y, tj) taken at time ti and tj, respectively, is to compare the two images one by one. [7]. If we select one image as reference image, the following formulation can help us to detect a moving object:

$$d_{i,j}(x,y) = \begin{cases} 1 if \mid f(x,y,t_i) - f(x,y,t_j) \mid > \theta \\ 0 otherwise \end{cases}$$

Where θ is the threshold.

Small moving object can not be detected if our image sampling frequency is not quick

enough. There are three methods for two images without pixel difference. The first is increasing sampling frequency. The second is abandon this pair images, sample again until we get a pair image with object movement. The last way is considering this image as still image and processing it like a picture. No matter which method we select, the worst condition is what we are using now.

7.1.2 Edge Separation

It is common to find people's faces overlapping in a 2-D dimension photo taken from a 3-D world. Using of skin-color can recognize overlapping faces as one face region, but we all know that an edge exists between two overlapping images. We will detect shapes inside a face region to make sure there is no overlapping edge that exists in a face region. Edge can also be used to separate neck from a face region.

7.2 Pose Detection:

Eigen vector gives us a coarse angle of a face region. If we can confirm that a region is a face region, using a template can help us to refine the face angle. Beymer introduces a way by detecting the face region by different size and angle of face feature [24], such as eye and mouth, and the nearest template will be considered as the angle of the face. In order to increase the accuracy of comparing, a large number of templates must be saved and compared with different angle or size of face feature. "Because of large number of template, the computation takes around 10-15 minuts for Sun sparc 2, Using fewer exemplars decrease the running time but also reduces system flexibility and recognition performance "[24].

In stead of saving large amount of templates, we consider to save some characteristic

angle or size of face feature and using rotation, scaling and translation to get expected feature. Furthermore, if we can identify the position of eyes or mouth, by calculating the outline or by using template and geometry, these features can help us identify the angle of a face.

Another possible way is using eigenface to compared with the sets of all different poses human faces (we can keep different face pose sets, such as front, side views or up and down) to detect the possible pose of this human faces[1]. The nearest one will be considered as the angle of a face.

7.3 Face Detection:

There are two kinds of face detection, nearest or neuron network. A typical algorithm for nearest is Eigen face detection. Eigenfaces can be used to compare with known faces eigenvector inside the database, any DFFS (distance-from-face-space) which is over the threshold will be output as be recognized face. Otherwise, it will output as an unrecognized notation.

Neuron network has more accuracy than Eigenface if we could not locate a face in a center position. As we train the system with known people face feature, such as eye, eyebrow, nose, mouth, when we input the tested face features, it will give us the possible result from the trained faces.

8. Conclusion

This project introduces a face location algorithm based on skin color detection. We first use lighting compensation to compensate chrominance value to standard level. Next, skin color will retain pixels that their values fall into the skin range. Then, morphology

is used to eliminate small regions and return big regions. After that, A region separation technology is used to delete irregular or non-face like region. Following that, a classification is used to cluster the main color regions. A clustered region is checked again to eliminate any non-clustered region. Eigen face is used to locate the center of a face's possible region. Finally, face features are detected in the possible face region to verify the face region.

In addition to face location, we also give a structure of automatic facial authentication and identification system. Future work includes effective image processing and future stages are also discussed.

The weakness of this project is that we only tested color races, including yellow and white. We are not sure if skin color concept can also be used for black people. The computational consuming for pixel clustering is another problem when an image has too many skin color pixels.

Reference:

[1] Fan lixin , Welcome to Face Detector Home Page

http://www.comp.nus.edu.sg/~fanlx/face_detector/face_detector.html#eigenface May 30, 2002

[2] Ing-sheen Hsieh, Kuo-Chin Fan , Chiunhsiun Liu, " A statistic approach to the detection of human faces in color nature scene". Pattern recognition , 2002

[3] Kin choong Yow and Robert Cipolla. "Feature-Based Human Face Detection" Image vision computer 15 (1997) 713-735

[4] D.W. Purnell , C. Nieuwoudt, E.C. Botha, Automatic face recognition in a heterogeous population. Pattern recognition letter 19(1998) 1067-1075

[5] E.Saber, A.M.Tekalp, Front0-view face detection and facial feature extraction using color, shape, symmetry based cost function, Pattern Recognition Letter 19(1998) 669-680

[6] J.T.Tou, R.C.Gonzalez, Pattern Recognition Principles, Addison-Wesley, Reading, MA, 1974.

[7]Rafel C.Gonzalez and Richard E.Wood Digital Image Processing, Addison-Wesley, 1992

[8] Rein-Lien Hsu, Mohamed Abdel-mottaleb, and Anil K.Jain. "Face Detection in Color Images", http://www.cse.msu.edu./~hsurein/facloc/, 5/31/2002

[9] D. Chai, K.N. Ngan, Locating facial region of a head-and-shoulders color images. Int. Conf. Automat. Face Gesture Recognition PDI (1998) 124-129

[10] Eric Hamilton "JPEG File Interchange Format " version 1.02.

http://www.dcs.ed.ac.uk/home/mxr/gfx/2d/jpeg.txt 6/11/2002

[11] DaubNet File Format Collection "Windows Bitmap –defined by Microsoft" http://www.daubnet.com/formats/BMP.html

[12] Lawrence O'Gorman and Rangachar Kasturi "Document Image Analysis" 1995 by Institute of Electrics Engineering, Inc.

[13] Ronse, C., and P.A. Divijver, Connected component in Binary Images: The Detection problem, Research Studies Press Ltd., Lethworth, England, 1984.

[14] Earl Gose, Richard Johnsonbaugh, Steve Jost. "Pattern recognition & image analysis ". 1996 by Prentice Hall PTR Prentice Hall Inc.

[15] Robert A. Mclaughlin. "Randomized Hough Transformer: Improved ellipse detection with comparision" Pattern recognition Letter. 1998. 299-305.

[16] Lei Xu, Erkki OJA and Pekka KULTANEN "a new curve detection method: Randomized Hough Transformer (RHT)" Pattern recognition Letter 1990 331-338

[17] efunda "least square line" http://www.efunda.com/math/leastsquares/leastsquares.cfm 16/07/2002

[18]Yuen , H.K,. Illingworth . J., Kittler. J., 1989. Detection partially occluded ellipse using the Hough transformer. Image and vision Computer. 7(1). 31-37

[19] Daniel Walsh, Adrian E. Raftery Accurate and efficient Curve detection in Images : The Importance Sampling Hough transform. www.stat.washington.edu/raftery. 16/07/2002.

[20] V.F. Leavers "Shape detection in computer vision using the Hough transform" Springer-Verlag London Limited 1992

[21]Constatine Kotropoulous, Anastasios Tefas and Ioannis Pitas. "Frontal face Authentication Using Morphological Elastic Graph Matching" IEEE Trans. Image Processing , vol 9, pp555-560 April 2000.

[22]Paul T. Jackway and Mohamed Deriche. "Scale-Space Properties of the Multiscale Morphological Dilation-Erosion" IEEE Transactions on Pattern Analysis and Machine Intelligence, vol 18, No.1 January 1996.

[23] L.C. Jain, U.Halici, I.Hayashi, S.B. Lee and S.Tsutsui. "Intelligent Biometric Techniques in FingerPrint and Face Recognition" 1999 by CRC

[24] Beymer, D.J. (1994) "Face recgonition under varing pose" Proceeding of IEEE Conference on Computer Vision & Pattern Recognition, pp.756-761, Seattle, WA. IEEE Computer Society Press

[25] Ming-Hsuan Yang, David J.Kriegman, and Narendra Ahuja "Detecting Faces in Images: A Survey" IEEE Transaction on Pattern Analysis and Machine Intellegience. Vol.24. No.1. January 2002.

[26] G. Yang and T. S. Huang , "Human Face Detection in Complex Background," Pattern Recognition, vol. 27, no.1. pp 53-63, 1994

[27] K.C. Yow and R.Cipolla , "Feature-Based Human Face detection," Image and Vision Computing, vol.15, no.9, pp713-735, 1997.

[28] M.F. Augusteijn and T.L. Skujca, "Identification of Human Faces throgh Texture-base Feature Recognition and Neural Network Technology," Proc. IEEE

Conf. Neutral Netwroks, pp.392-398, 1993

[29] R.Kjeldsen and J. Kender, "Finding Skin in Color Images," Proc. Second Int'l Conf. Automatic Face and Gesture Recognition, pp. 312-317, 1996

[30] S.Mckenna, Y.Raja, and S. Gong, "Tracking Color Objects Using Adaptive Mixture Models," Image and Vision Computing, vol.17, nos. 3/4, pp.223-229, 1998.

[31] T. Sakai, M. Nago, and S. Fujibayashi, "Line Detection and Pattern Detection in a Photograph," Pattern Recognition , vol. 1, pp233-248, 1969

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

[33] M.Turk and A. Pentland, "Eigenfaces for Recognition," J.Cognitive Neuroscience , vol. 3, no.1, pp.71-86, 1991

[34] K.-K. Sung and T.poggio, "Example-based Learning for View-Based Human Face Detection," IEEE trans. Pattern Analysis and Machine Intelligence, vol. 20. No.1, pp 39-51, Jan, 1998.

[35] R.Feraud and O. J. Bernier, "Ensemble and Modular Approaches for Face Detection: A Comparison , " Advanced in Neural Information Processing System 10. MIT Press. 1998