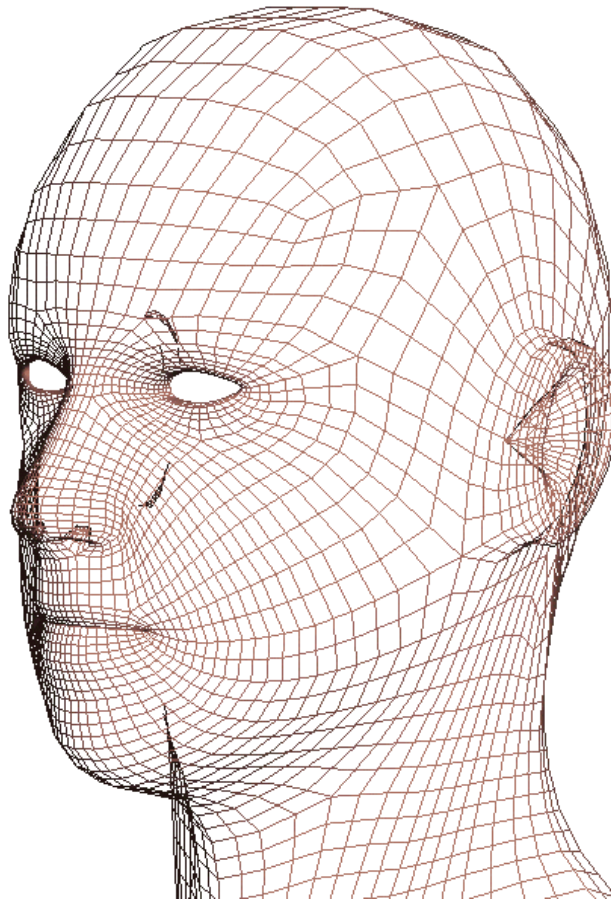


# Training Course

Field: Image Processing

Facial Features Detection For 3D Face Modeling



Supervised by Doc. Ing. Gregor ROZINAJ  
FEI-KTL - Bratislava

MISTRAL François-Louhenvel - Polytech'Nantes, SEII3  
January - June 2003

## Resume

The algorithm developed under VC++ is a part of a bigger multimedia project of 3D speech animation.

The main goal of this algorithm is to localize accurately some specific points of facial features, to allow a 3D head creation. The program treats two photos, one from front view and the other from profile. It permits to obtain the three coordinates for each point localized on the face. A normalization and storage step is realized before the reused of these data for the 3D model. The algorithm was tested on different morphologies and ethnics subject under various light conditions. Moreover a protocol on the photo shooting step was determined to have optimal results.

## Résumé

L'algorithme qui a été développé sous VC++ s'inscrit dans le contexte d'un projet multimédia de modélisation faciale en trois dimensions utilisant OpenGL.

Ce programme a pour but de localiser précisément une liste de points particuliers nécessaires à une modélisation en trois dimensions d'un visage. Le programme traite les photos de face et de profile d'un même sujet afin d'obtenir le triplet de coordonnées de chaque points. Les coordonnées sont ensuite normalisées avant d'être utilisées pour la création du model 3D. L'algorithme a été testé et amélioré à l'aide de photos présentant des sujets de morphologies et d'ethnies différentes, sous des conditions d'éclairage variées. De plus un protocole pour la prise de photos a été établi pour obtenir des résultats optimaux.

## Summary

I-	Presentation of FEI and KTL Department .....	4
II-	Presentation of the project environment .....	6
III-	Description of the subject .....	10
IV-	Project organization .....	12
<b>PART 1: Frontal view algorithm .....</b>		<b>15</b>
I-	Noise reduction .....	16
II-	Skin detection .....	17
III-	Choice of the face area .....	23
IV-	Eyes localization .....	26
V-	Symmetrical Axe determination.....	39
VI-	Mouth localization.....	43
VII-	Nose Localization .....	48
<b>PART 2: Side view algorithm .....</b>		<b>54</b>
I-	Facial edges detection .....	55
II-	Coarse to fine ( eyes and mouth areas) .....	60
III-	Results .....	65
<b>PART 3: Data and protocol .....</b>		<b>66</b>
I-	Data normalization and storage.....	67
II-	Shooting protocol definition .....	69
<b>CONCLUSION .....</b>		<b>71</b>
<b>REFERENCES .....</b>		<b>72</b>
<b>ANNEXES .....</b>		<b>73</b>

## I- Presentation of FEI and KTL Department

This internship of end of engineer studies occurred in Bratislava, capital of the Slovak Republic, in the Slovak Technical University.

### THE COUNTRY

On August 26, 1992, the leaders of Czecho-Slovakia agreed to the peaceful division of the country into two independent nations, scheduled to take effect January 1, 1993.

### THE FACULTY

The Faculty of Electrical Engineering and Information Technology of Slovak University of Technology in Bratislava (STU) is a diverse, cosmopolitan community. While 90% of the 2500 students come from Slovakia, students also come from 40 countries beyond the borders of Slovakia.

### ROLE AND MISSION

The Faculty of Electrical Engineering and Information Technology as a segment of STU is responsible for offering a wide range of undergraduate and graduate programs, including those leading to the doctorate.

Faculty of Electrical Engineering and Information Technology (since 1994)



In the FEI STU, about 4600 students are studying, and 400 teachers teaching. We can find several branches of study:

- Automation
- Electromaterial Engineering
- Electronics
- Power Electrical Engineering
- Information Technology

More precisely I worked in the Department of Telecommunication (KTL) which is one of the 18 departments of the FEI and the laboratory was the one of Digital Image and Speech processing.

- Laboratory of Telecommunications Management Network
- Laboratory of Telecommunications Technology (Optical SDH, ATM, RWS)
- Laboratory of Digital Signal Processing
- Laboratory of Digital Switching Systems and ISDN
- Laboratory of Optocommunications Systems
- Laboratory of Data Transmission
- **Laboratory of Digital Image and Speech Processing**
- Laboratory of Intelligent Networks and Services

In this department of telecommunication, international projects are led.

#### Examples of research projects

- Broadband Telecommunications Networks and Services, VTP 315/2000, I. Baroňák
- Multiservices Networks, No. č. 42 0000 15 80, Research Project (2000), I. Baroňák
- COST 257 - Methods for Performance, Evaluation and Design of Broadband Multiservice Networks - Impact of New Services on the Architecture and Performance of Broadband Services, P. Podhradský
- TEMPUS TELEEDUCA (1998-2001), P. Podhradský
- LEONARDO-TRICTSME (1999-2001), P. Podhradský
- Methods and Algorithms of Speech and Image Signals, VEGA 1/7627/20, P. Podhradský
- New coding methods of information and protocols for communications systems and services of next generation, VEGA 1/7615/20, P. Farkaš
- Communications Technologies, The British Council Academic Link, P. Farkaš

#### International Cooperations:

- |                                                  |                                          |
|--------------------------------------------------|------------------------------------------|
| • Telenor, Norway                                | • Rechenzentrum der RWTH Aachen, Germany |
| • Ericsson, Sweden                               | • Siemens Vienna, Austria                |
| • Institut National des Télécommunications (INT) | • UPC Barcelona, Spain                   |
| • TU Budapest, Hungary                           | • Politecnico di Torino, Italy           |
| • TU Zagreb, Croatia                             | • University of Luton, United Kingdom    |
| • TU Budapest, Hungary                           | • Kingston University, United Kingdom    |
| • TU Wien, Austria                               | • Alcatel SEL, Stuttgart, Germany        |
| • UOC Barcelona, Spain                           | • Nortel, Vienna, Austria                |
| • Lancaster University, United Kingdom           |                                          |
| • TU Maribor, Slovenia                           |                                          |

## II- Presentation of the project environment

The project, I had to work on was called "Talking Head", and consist in creating a model of a human head in 3D which can speaks and shows emotions.

This part of project was well started when I arrived; in fact a model was already implemented under visual C++ and OpenGL. One student, Viktor POZGAY, was working on this project since one year and the essential of correlation between speech and face animation was done. A second student Miroslav Vajas was working on the speech and emotional part.

An executable was already operational, where you can make the face speaking whatever combination of phonemes you want, but without the speech associated.

### a) Facial detection environment

#### What is Visual C++ .NET?

This is the latest version of Microsoft Visual C++, it allows new possibilities like internet applications, but in our use it just allows us to develop a program of facial features detection; this choice come from the desire of unifier of our project, indeed the existing files of "Talking Head" project were developed with this software. It is more efficient than the previous version (VC++ 6) thanks to new library and a professional compiler and debugger. This software requires Windows XP. (Visual C++ .NET overview is available in **Annex 1**).

### b) 3D model synthesis environment

#### What is OpenGL?

OpenGL is 3D graphical library evolutionary and portable; it provides solutions for the creation of 3D engine for video games for example. It uses the hardware capacity available with the 3D graphical card.

#### How does it work?

The user just gives order to trace graphical primitives directly in 3D, camera position, lights, and textures to apply on surfaces and so on...

OpenGL is divided into 3 engines:

- *Geometric engine* computes the 3D part: polygons triangulation, changes of repair, management of perspective projection on the screen, clipping (cut the parts bigger than the screen).
- *Raster engine* compute the 2D part: it changes triangles to pixels line by line (called *fragments* before displaying), computes colors and texture's coordinates interpolations, and eliminates the hidden parts. This is the OpenGL basic principle but it can provide other mechanisms and means of control.
- *Raster manager*, the pixel is given to this engine to have or not additional treatments and to be displayed on screen after.

OpenGL doesn't manage the displaying part; it just gives the required information to your system. This is the reason why OpenGL is portable. (OpenGL overview is available in **Annex 2**)

Let see more concretely the possible result of such a library.

### c) Existing executable: MORPHY

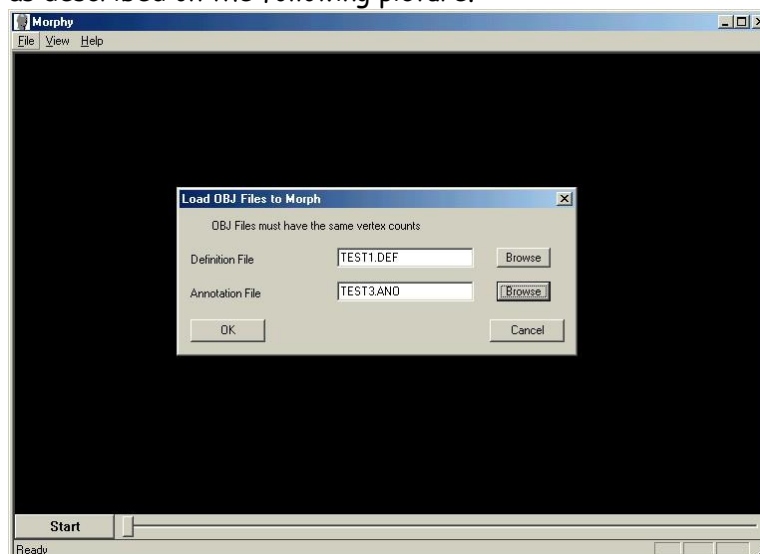
This part consists in presenting what still exists in this face animation project. This project was developed in C++ and use OpenGL for the 3D animation.



Morphy.exe

#### - The interface

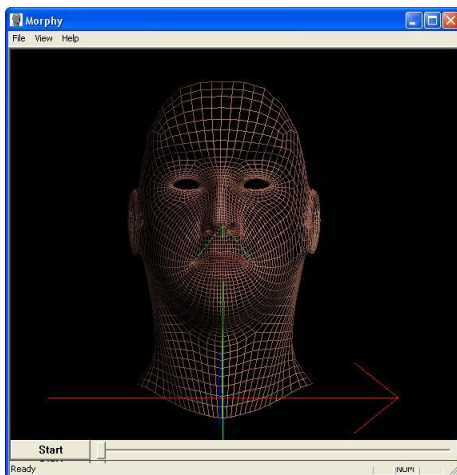
First launch the Morphy executable and browse the '.ano' and '.def' files as described on the following picture.



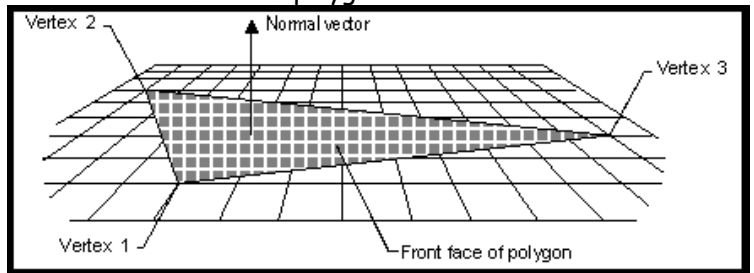
## - The standard 3D Model

This model is in fact composed of 11 different faces (11 files) each face corresponds to a phoneme group. Indeed the shape's mouth can be similar for different letters. The transition between the face at the instant  $t$  and  $t+1$  is computed thanks to a Spline Transformation which provides best result than linear transformation.

Let's have a look to the model structure.



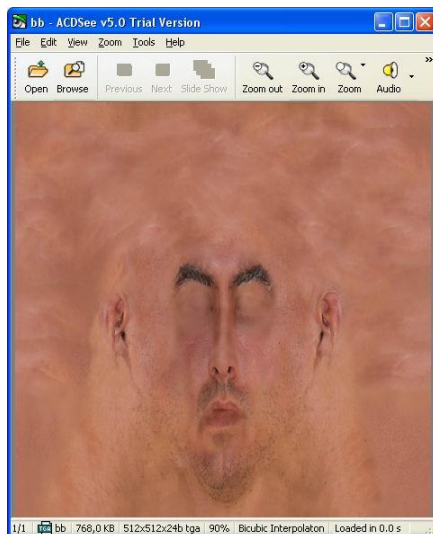
You can notice that this model is composed by more than 4600 polygons. This is just the skeleton structure for a neutral face. These polygons define the vertices in the 3D space. A vertices is defined by its coordinates in 3D space (vertex) and normal vectors to the face of polygons.



The next step will show how the model's texture is applied.

## - The texture

First of all you need a photo which will fit with the skeleton, this photo has to be compute manually (for not real subject like in video games) or thanks to a head scanner (for human) indeed you can see the whole head projection on the same photo. (A description of existing system is available in next section)



Considering this example you can notice that hair could be more difficult to represent as a bald head.



### -The whole standard model

After the two previous steps, you can have a 3D face model which seems more human. The speech part is already well advanced and works thanks to the slider on the bottom part.



### Accurate description:

At the starting point it's possible to make the model speaking, you just need:

- the 11 '.obj' files which contain the vertices for each face for each phonemes
- the 11 '.mtl' files for the ambience parameters
- the '.ano' file for the speech (phoneme's list)
- the '.def' file for the link between the phoneme's list and the associated face
- the '.tga' file for the photo to fit on the model skeleton

Note that this presentation corresponds to the existing project at the beginning of my internship, now you can find in **Annex 3** the latest description and explanations about this part of the project.

### III- Description of the subject

#### What is asked?

First of all the main goal is to adapt whatever real head to a 3D structure corresponding morphologically. More precisely it means that with photos from front and profile, we can determine what the characteristics of this head are. (Length of nose and so on) And in function of these characteristics we will adjust the skeleton of the existing 3D model, and finally apply the face on this skeleton. It's clear that each person has his own morphology and each face can't fit with another skeleton.

#### a) Existing system

This type of 3D modeling already exists but requires specific and expensive equipment. This system is based on a scanning system which turns all over your head and generates both of the structure and texture, according a data transmission from the scanner to the graphical workstation.

#### HEAD & FACECOLOR 3D SCANNER

Gyberware®



The scanning process captures an array of digitized points, with each point represented by x, y, and z coordinates for shape and 24-bit RGB coordinates for color. Data are transfer via a SCSI interface to a graphics workstation for immediate viewing and modification.

This system is designed to scan the head and face of live subjects quickly, comfortably, and safely.

It also serves well for personal portrait sculpture, in which a person's head is digitized, and then reproduced on an automated milling machine or rapid prototyping system with remarkable fidelity.



Structure



Cylinder projection of the texture



Association

**Price: \$72,600**

In the video games world we can find another way closer to ours which consists to design and change manually the characteristics of this 3D model, but this relieve more of artistic domain and can't permit a wide and adaptable use.

## b) Our approach

- First we need to take some photos from front and side of the subject to know precisely their dimensions. Then these photos are treated and the face characteristics are stored before to be reused for the 3D step.
- This next step consists in changing the existing characteristics of the standard 3D model to a personalized one. It is essentially changing coordinates of point in the 3D space and to do it recurrently to modify the whole structure.
- In fact to resume, I'm asked to localize and transmit the necessary data to turn the existing program into an adaptable one.

### List of the needed points:

#### Eyes:

- Corners
- High and low borders

#### Nose:

- Nostrils
- Top of the nose
- End of the nose

#### Mouth:

- Corners
- Limits of the
- Lips

#### Other:

- Shin
- Center of the cheeks
- Beginning of the hair



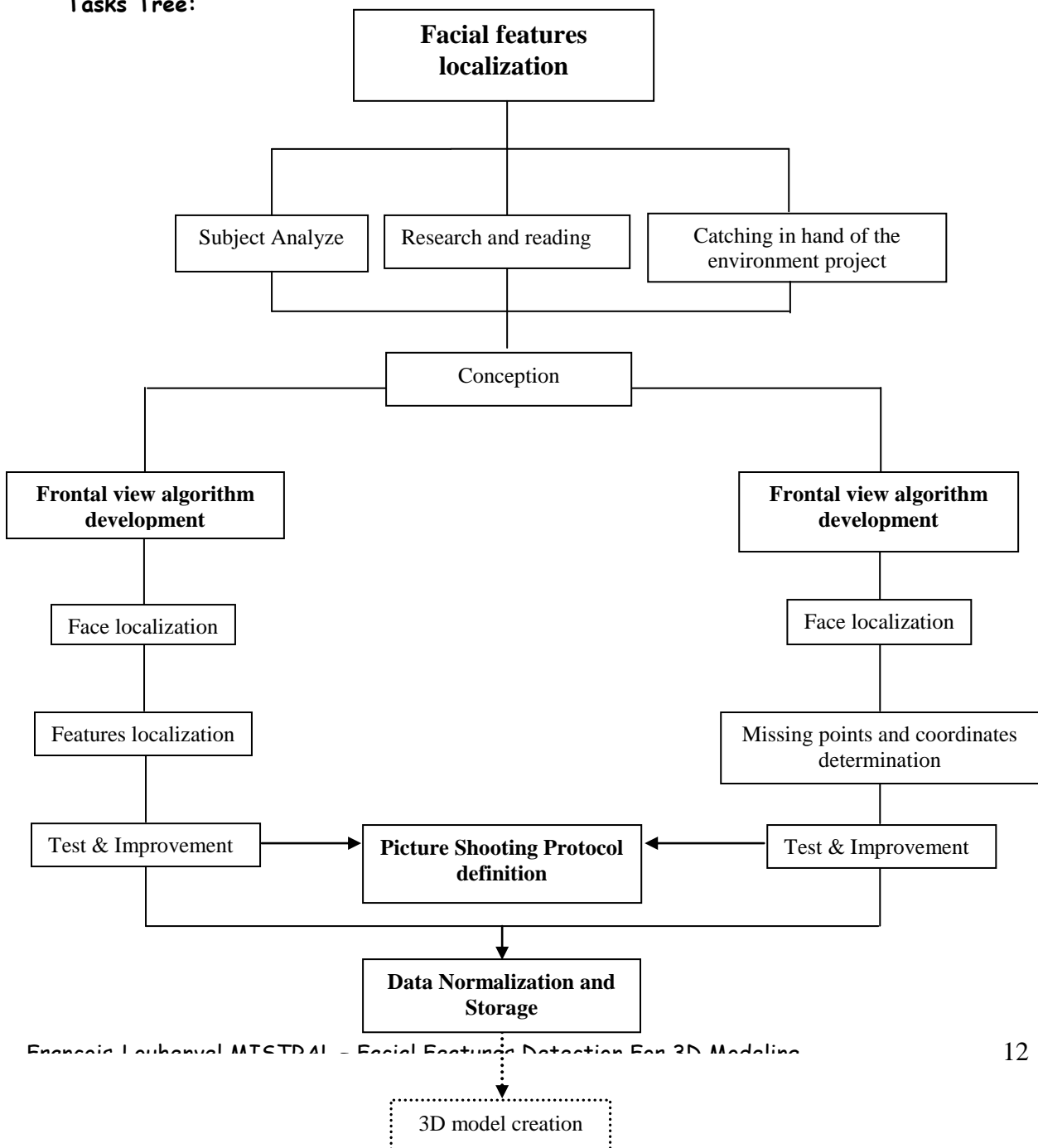
## IV- Project Organization

The detail of the organization of this project is exposed below. First you will find a decomposition of the work by tasks thanks to the Tasks Tree. Then a simple evaluation of these tasks is given. Finally a Gant diagram is available to compare the estimated and real plannings.

### Aim of the facial features detection program:

- Detect the face on a colored photo.
- Localize the important facial features.
- Normalize and store the data (coordinates of these features).

### Tasks Tree:



### Tasks evaluation

1. Research and reading about face detection and its features ( **2-3 weeks** )
2. Implement the algorithm for the front view, localization of the needed points to prepare the 3D face model. ( **3-4 months** )
3. Test the algorithm on a set of faces and improvement of the algorithm. ( **2-3 weeks** )
4. Implementation of the algorithm for the profile view to localize the missing points and coordinates to prepare the 3D face model. ( **1-2 months** )
5. Test the algorithm on a set of faces and improve the algorithm. ( **2-3 weeks** )
6. Data normalization and storage. ( **1 weeks** )
7. Design the methodology of a testing place for shooting an image of real faces for the models. ( **few days** )

Note that you can't see on the following Gant diagram the control meetings. Every ten days a meeting was planed with the project supervisor to eventually change the way of the algorithm development and to control the situation for the deadlines.

## Project Development Plannings



# PART 1: Frontal view algorithm

## I- Noise reduction

Before all treatment on picture, it's better to reduce the eventual noise. It will permit to obtain better result for the future color based facial detection.

### a) Specification

We need to realize a low-pass filter, which applied on each of the picture's pixel allows a standardization of the area considering the neighborhood of this pixel.

### b) Conception

The impulse response of the low-pass filter is given by:

$$1/9 \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

The weight of each neighbor is '1', the sum of the 9 RGB pixel's values is computed and divided by the sum of the weights (9\*1=9) and then attributed to the central pixel.

### c) Results

We obtain a less precise resolution picture, which seems a bit blur:



Now the picture is ready to be treated, and the color segmentation will provides better results.



## II- Skin detection

The first step of face detection is the face localization in a background. It is quite easy to find this characteristically area in a photo for a human, but for an artificial intelligence it's much more difficult.

### a) Specification (pixel based approach)

Different approaches exist for face detection, simple and less simple. The complex ones like neural networks are very efficient but difficult to implement. We don't need such a good method, this is the reason why simple color based detection was considered. To segment human skin regions from non-skin regions based on color, we need a reliable skin color model that is adaptable to people of different skin colors and to different lighting conditions.

### b) Conception

It exists different types of colorimetric domains, the most used is the RGB one. But this domain presents some difficulties, it is not suitable for characterizing skin-color. Indeed the color of a pixel is defined by 3 components (the Red, Green and Blue components). The problem of this representation is the luminance which is defined into this triplet; you can't detect easily a dark skin and a clear one at the same time, and the luminance vary across a person's face due to the ambient lighting. Although skin colors of different people appear to vary over a wide range, they differ much less in color than in brightness. So it is judicious to use the YCbCr (Luminance, Blue Chrominance and Red Chrominance) domain because of the Y component which is dedicated to the luminance. We can thus define simply a specific area of skin color by just using the two chrominances.

Schema:

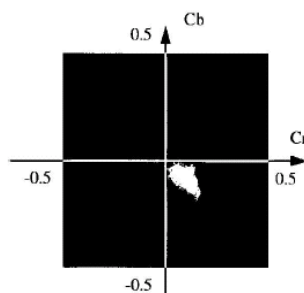


Figure 1. skin tone color distribution in the chrominance place

This place is normalized. We need to denormalize it.

So:

$$Cb = (Cb_{norm} * 256) + 128 \quad \text{and} \quad Cr = (Cr_{norm} * 256) + 128$$

- **Principe**

The idea is to mix two different methods to obtain better results in approximating the skin color area.

- The first one consists in defining four thresholds to approximate the theoretical skin color area valid for a wide range of photos.
- The second one is based on the probability for a pixel to belong or not to a skin area defined by samples of skin's color.

***First technique:***

In fact this skin color area is not square but for an easier use we just have to determine the maximal and minimal thresholds of blue and red chrominances. So we approximate this area by a square or more precisely by a rectangle.

Each RGB value of each pixel is converted linearly in YCbCr value. Then this new values are compared to the predefined thresholds and if this pixel belong to the predefined area, it is detected as belonging to a potential face. The following transformation permits to pass from the RGB space to the YCbCr one.

$$\begin{aligned}
 Y &= 0.299 * R + 0.587 * G + 0.114 * B; \\
 Cb &= -0.1687 * R - 0.3313 * G + 0.5 * B + 128; \\
 Cr &= 0.5 * R - 0.41874 * G - 0.08135 * B + 128;
 \end{aligned}$$

This transformation is linear, so you don't loose any information.

***Second technique:***

This time the technique is based on the study of samples of skin color. We just compute the probability for each pixel of being a skin color pixel. This technique also uses the YCbCr domain. And it allows to define a threshold in function of this probability.

In fact to determine the color distribution of a human face in chromatic color space we used samples of skin color from different ethnicities (African, Asiatic, and European).



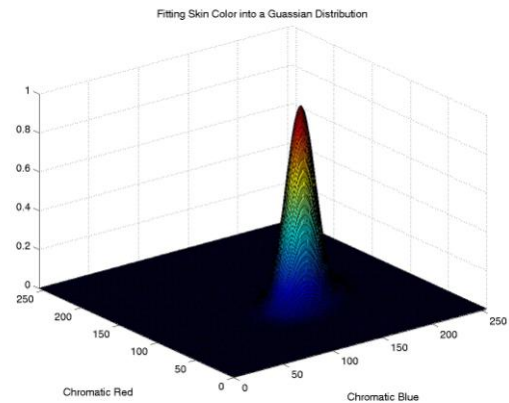
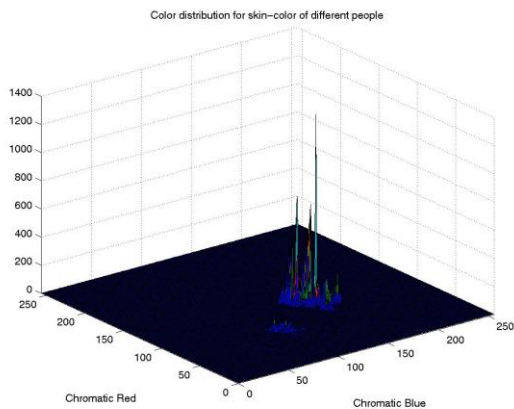
***Samples of skin (From the left to the right: European, African, African, and Asiatic)***

These skin samples were filtered using a low-pass filter to reduce the effect of noise in the samples. (Detail of this filtering step is given in previous chapter)

The skin color distribution can be represented by a Gaussian model  $N(m, C)$ , where:

Mean:  $m = E \{ x \}$  where  $x = (Cr \ Cb)^T$

Covariance:  $C = E \{(x - m)(x - m)^T\}$



*Color distribution for skin-color of different people. Fitting skin color into a Gaussian distribution.*

We compute the mean of each chrominances of each pixel of the color skin's samples and then the covariance matrix of these two mean chrominances. (realized under Matlab)

The likelihood of skin for this pixel can then be computed as follows:

$$\text{Likelihood} = P(Cr, Cb) = \exp[-0.5(x-m)^T C^{-1} (x-m)]$$

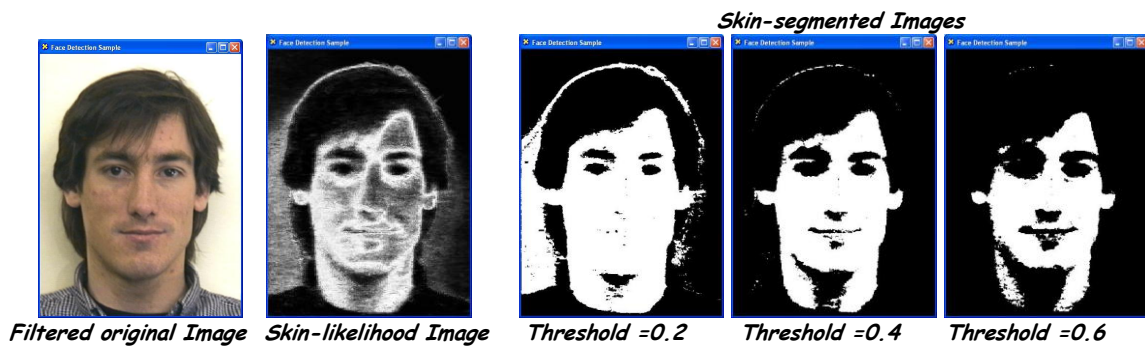
Where  $x = (Cr \ Cb)^T$

This second method comports two advantages:

- An approximation based on real wide samples of skins.
- A possible tresholding step on the likelihood allows determining the potential skin pixels.

### c) Result of skin segmentation

The skin-likelihood image is a gray-scale image whose gray values represent the likelihood of the pixel belonging to skin. With appropriate threshold, the gray scale images can then be further transformed to a binary image showing skin regions and non-skin regions. The first technique leads to the same type of binary image. Next section is dedicated to the combination and results of these two techniques.



Not all detected skin regions contain faces. Some correspond to the hands and arms and other exposed part of the body, while some corresponds to objects with colors similar to those of the skin. The important point here is that this process can reliably point out regions that do not have the color of the skin and such regions would not need to be considered anymore in the face detecting process.

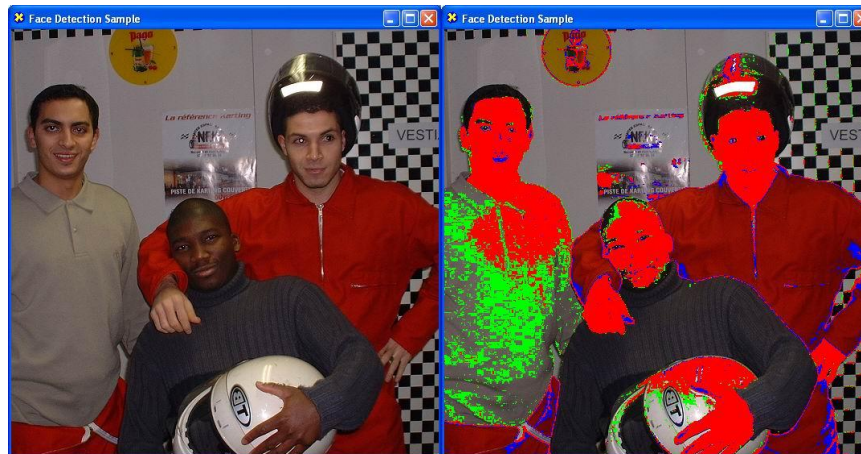
Later a test of dimension of this connected components (all the potential faces) is used to determine if the component is able or not to be a face by its size in function of the whole picture size.

#### d) Tests for skin segmentation

We need to create a face database more consequently to test our method, and these results are just some explicative samples about the two techniques of skin detection.

The samples:

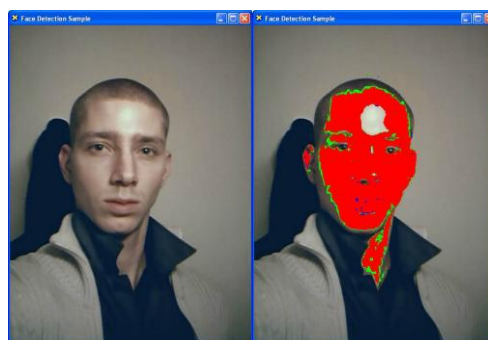
- █ Intersection of the two methods
- █ Other pixels detected by (1)
- █ Other pixels detected by (2)



*The interest is detection of clored skin.*



*Once again non white face is detected.*



*Saturation can make the detection more difficult.*

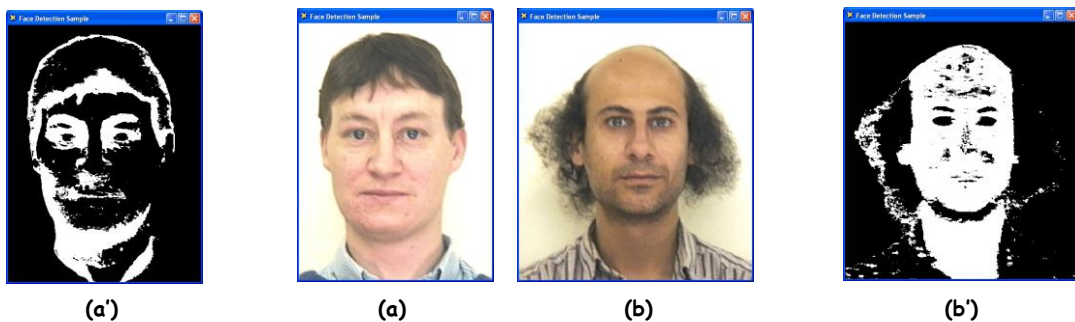
#### Remarks:

- With an appropriate background, false alarms can be reduced.
- Sometimes (2) Permits a first approach in mouth localization.
- Face localization is quite well localized just keeping the **intersection** of the two methods.

### e) Improvement and adaptation

We saw before that the skin color can be approximated without taking care of the brightness, however the threshold on the likelihood of being skinny pixel or not has to be adapted to the condition of brightness.

Indeed sometime if the brightness is too important (photo (a)), the probability threshold is too high and we can't find an only connected component standing for the most potential face area (photo (a')). We can see that there is less saturation phenomenon on the (b) photo and leads to one big skinny area.



The improvement of the algorithm is based on the fact that we go through the skin detection function not only once with a predetermined threshold but until obtaining enough potential skinny pixels. "Enough" means that 1/3 of the total number of pixels are considered like skinny area. We chose 1/3 because of the quite big size of the face required for the precision of our treatment and because the eyes areas have to belong to the main connected component standing for the face (Next Chapter).

So the probability threshold is decreasing in each new lap in the function, the **Step** chosen is  $-0.02$  from  $Pr = 0.6$  until  $NumPixel > 1/3 (NumPixelTotal)$ .

Indeed we noticed that if we consider to high probability the number of skinny pixels is really low and we lost in computational time. So to reduce the number of way through the loop, we chose  $Pr = 0.6$  instead of  $Pr = 1$  to start, logically the **step** has to be quite reasonable too. We made a compromise between speed and adaptability.

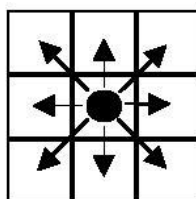
The risk of detecting non skinny pixel is avoided thanks to the fact that we cross the two methods, we still have our 4 thresholds on the chrominances to limit the colorimetric domain. It allows us to not detect non skinny pixels when the probability threshold is decreasing. So if the number of pixels is still inferior to the required threshold the algorithm will run until the end of the loop, it means when  $Pr = 0$  and the considered pixels will be only the cross between two method, no more.

### III- Choice of the face area

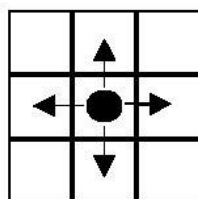
After this first step of face localization, we need to know which of these areas is supposed to be the face. This step of our process will determine which one of all the color skin areas will be treated as the potential face.

#### What is a connected component?

A connected component is a group of pixels which are connected and they have the same value. It means that each of these pixels has at least a neighbor (with the same value) in his 8 or 4-connexity environment. Let's see an example:



8-connexity



4-connexity

0	0	0	1
0	0	1	1
1	1	0	0
0	1	0	0

0	0	0	1
0	0	1	1
1	1	0	0
0	1	0	0

Considering 4-connexity we obtain 2 patterns.  
Considering 8-connexity we obtain 1 pattern.

#### a) Specification

First each connected component of our binary picture has to be labeled; it means that each independent area has a number, so each pixel of this area is identifiable. When all these areas are identifiable we need to keep the most potential to be a face.

#### b) Conception

The main hypothesis of this process is that we treat some portrait photos, so if there are two faces on the same picture, the biggest head will be selected. It reduces the risk of finding hands or other body parts. To eliminate the less potential face areas, the idea is to use the size of the area in function of the size of the photo. So the biggest area is normally the face. But before eliminating areas, we need to fill and label it.

- **Labeling step**

When a picture is displayed, the picture is generally covered from the top left corner to the right and line by line until the low right corner. The algorithm's aim is to determine all connected areas. The value considered is first "0" (black pixels on the binary picture) to determine the background; the same function will be reused to determine the most potential face area considering pixel's value "1" (white pixels on binary picture).

So the algorithm works as it's described below:

**1<sup>st</sup> Step:**

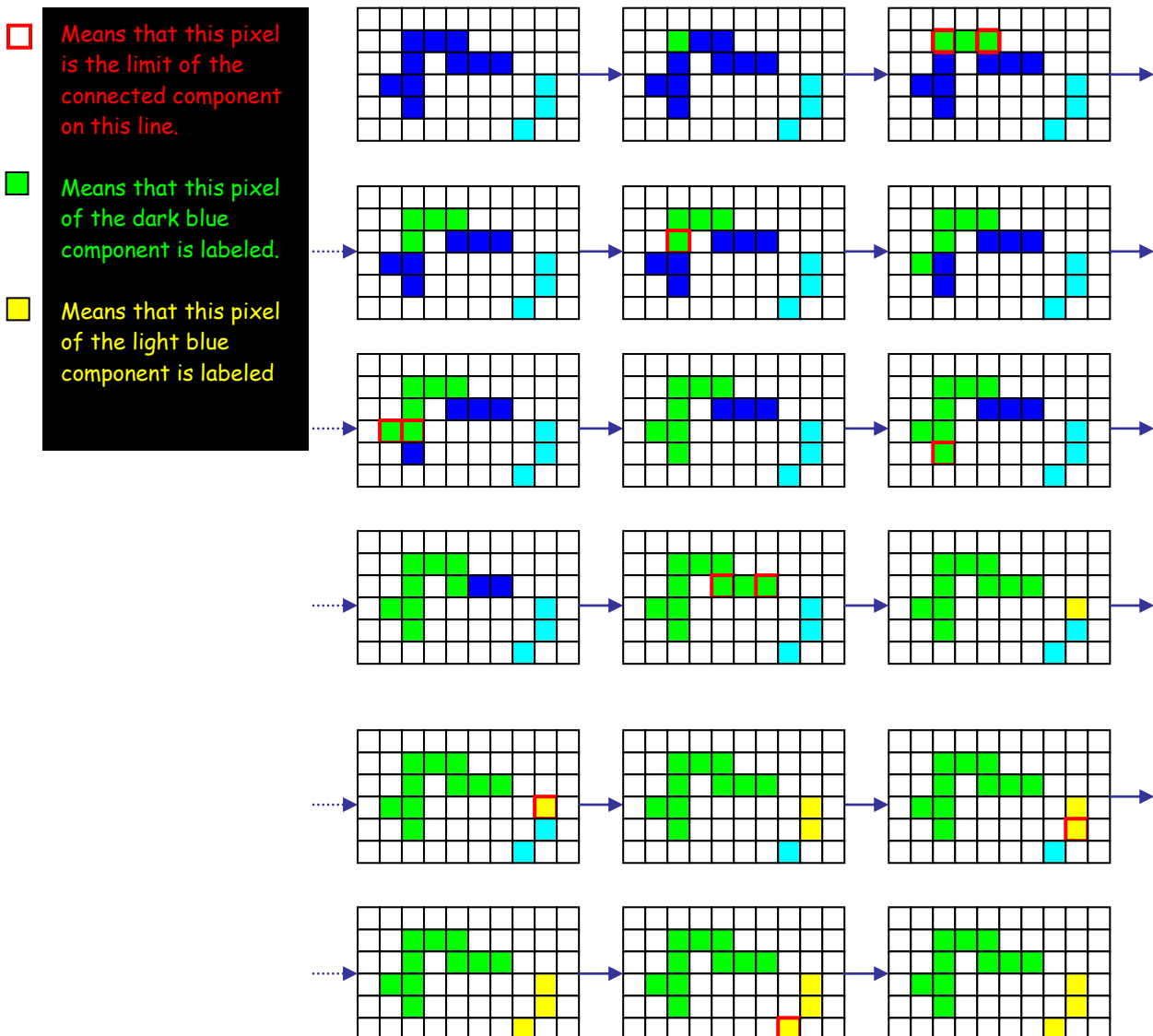
Find the first pixel with value "0", if next one has the same value it belongs to the same connected component if not the previous one was the last one of this area for this line. So we find for each line and each connected component a Xmin and a Xmax delimiting independents areas.

**2<sup>nd</sup> Step:**

Then next row is covered from the Xmin-1 pixel to the Xmax+1 pixel if some of these pixels have the same value as Xmin or Xmax, they belong to the connected area.

**3<sup>rd</sup> Step:**

During this particular sweeping, each new pixel belonging to the connected component is labeled. When no more pixels can be added to the connected component, we start a new lap in the loop increasing the label by 1 to determine the next component.

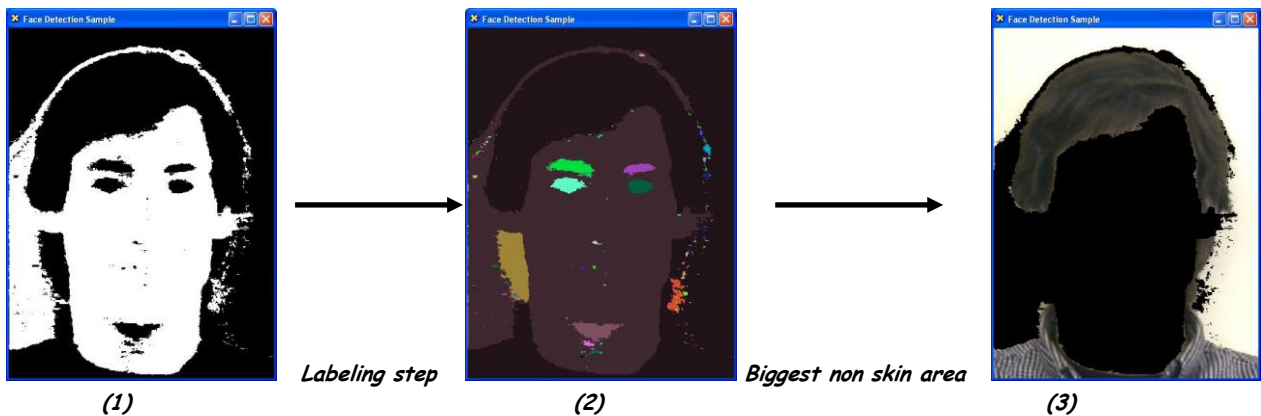




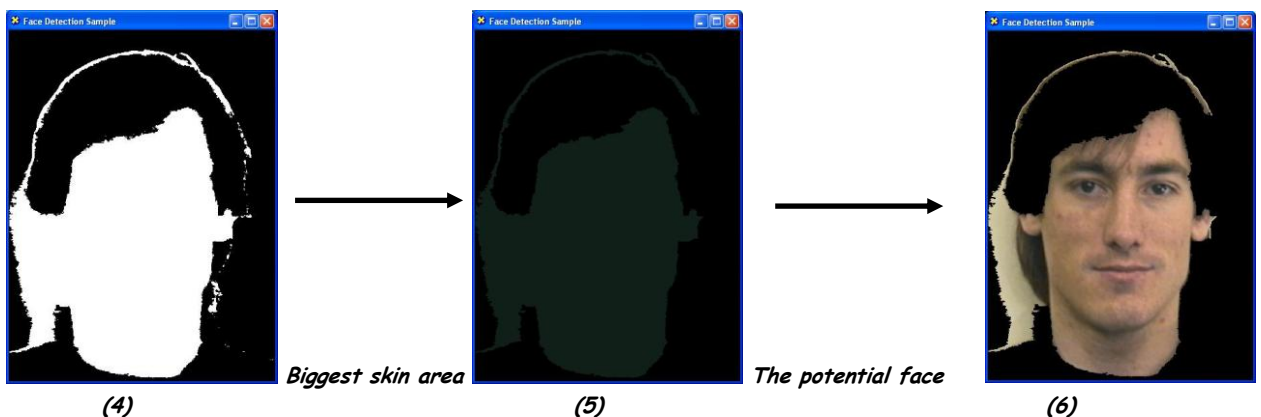
- **Filled face area**

We can notice that when we obtained the binary picture, there were some holes in the potential face areas. For best result we need to fill these areas, it will allow us to compute the centroid in the next step for eyes detection. The method consists in considering the black areas confined in white areas, it means all the black areas except the background (the biggest black one). So after finding this biggest area all the other are set to the white value and we obtain the filled skin area. We restart the previous process to label this time the white areas it permits to determine the biggest white area, which is supposed to be the face area. We finally display the filtered picture in this area and set all the other to the black value.

- **Results**



- (1) to (2) all the connected components black and white are labeled.  
 (2) to (3) all the black and white connected components are black except the biggest non white area (the background).  
 (3) to (4) background is set to black value and other area to white one, we obtain the filled color skin areas.



- (4) to (5) the white connected components are labeled and only the biggest one is kept.  
 (5) to (6) this biggest component is displayed with the filtered picture.  
 (6) to (7) the face area is delimited by a rectangle thanks to (Xmin, Xmax, Ymin, Ymax of the potential face area).

## IV- Eyes localization

By their own nature, eyes are very complex areas. Considering this statement, it is easier to start the feature detection with eyes localization than mouth or nose one.

### a) Specification

The aim of this eyes detection is important because the next feature localization will be based on this first step of eyes localization. So we need to implement a robust algorithm which determines accurately where the eyes in the precedent picture of potential face are.

### b) Conception

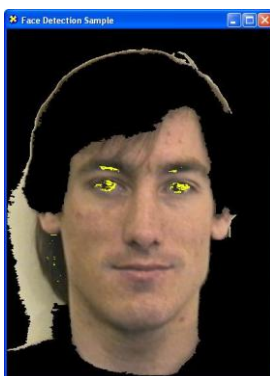
- **Eyes latitude**

To determine the vertical position of the eyes in the face, we use the observation that eye-regions correspond to regions of high intensity gradients. We created a vertical gradient-map, from the estimated region of the face using edge-detection. We then do a horizontal projection by summing the pixels along the horizontal lines. Both eyes are likely to be positioned at the same line; the sum will have a strong peak on that line. However, in order to reduce the risk of error, color observation was reused. The vertical gradient map is obtained thanks to the fact that vertical gradient is important for eyes but also for mouth. However mouth is a skin colored area and can be ignored.



Indeed during skin detection step, on the binary picture, eyes are never detected while mouth is.

- **Results**



Vertical gradient-map



Sum of the pixel for each line



Vertical gradient-map



Sum of the pixel for each line

- **Remark:** Some errors can appear due to the hair area.

### c) Improvement and adaptation

This step of the algorithm is one of the most important; an error in eyes localization can't lead to good results in determining other facial features location. Then the eyes region has to be emphasized and separated from the other pixels detected like being potential part of eyes. The main goal of this part is to underline only the eyes pixels. This is not possible and we always detect some false eyes pixels, the algorithm improvement consists in selecting the most accurately as possible the real peak standing for the eyes latitude after the horizontal sum.

#### 1<sup>st</sup> Step:

The important horizontal gradient area are underlined thanks to horizontal edge detection, however the conditions of brightness and the different eye's color can sometimes lead to a bad characterization of the eyes area.

#### 2<sup>nd</sup> Step:

We made an adaptable loop to determine enough pixels to characterize the eyes region and avoid the brightness problem. So the gradient map is realized and validated if at least a number **N** of pixels are susceptible to belong to eyes part in each side of the photo (it supposes that we treat portrait photo). If not, the gradient threshold is decremented and a new lap in the loop is started. When the condition is validated, the loop stops and the gradient map is obtained.

The decision on each pixel uses the vertical neighborhood of the treated pixel. So if the pixel doesn't belong to a skinny area, the upper neighbor either and the gradient between the blue chrominances, red chrominances and the brightness of these two pixels is superior to a threshold, we can consider this pixel like being a potential eye's part.

#### 3<sup>rd</sup> Step:

If after reducing threshold until the limit value, the number of pixels is still inferior to **N**, we use the other characterizing method, which consist to find the darkest part in the face to detect pupils (presented p.33). The threshold on the brightness will increase in a new loop, if not enough pixel are detected. This latest technique is used generally for picture with important saturation, which causes some troubles for eye's characterization.

#### 4<sup>th</sup> Step:

Then we need to distinguish a big area like hair confined area like eye, before summing the pixel by line, we make a dilatation (cf. **Annex 4** for more details about morphological treatments), which allow us to connect the closest pixels and permits to create a bigger eyes area and finally we erase all the isolated pixel. Indeed the sum which leads to some errors in case of hair is

generally due to a lot of isolated pixels instead the sum of eye's area is due to a compact pattern corresponding to eyes. The dilatation is computed even if **N** is not reached in the previous step. However the final step which consists in deactivating the isolated pixels can be started only if **N** was reached. Indeed if the previous loop stopped because of the ending value of the loop, the isolated pixels are not deactivated; it would be too risky because of the number of pixels inferior to **N**.

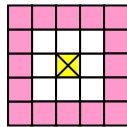
Detail of the morphologic treatment computed on the vertical gradient map:

We consider an 8-connexity neighborhood and realize a dilatation. If the central pixel is not yellow ( $R=255, G=255, B=0$ ) and at least one of the 8 neighbors is, the central pixel's value is set to ( $R=255, G=0, B=0$ ), it will be displayed in red. This treatment allows to connect very close areas. In confined areas like eyes, we have a higher probability to connect some detected pixels than in wide area like hair.



8-connexity

So the next step of treatment consists in eliminating the isolated pixels of the vertical gradient map. This time we consider the 16 neighbors of the previous structuring element.

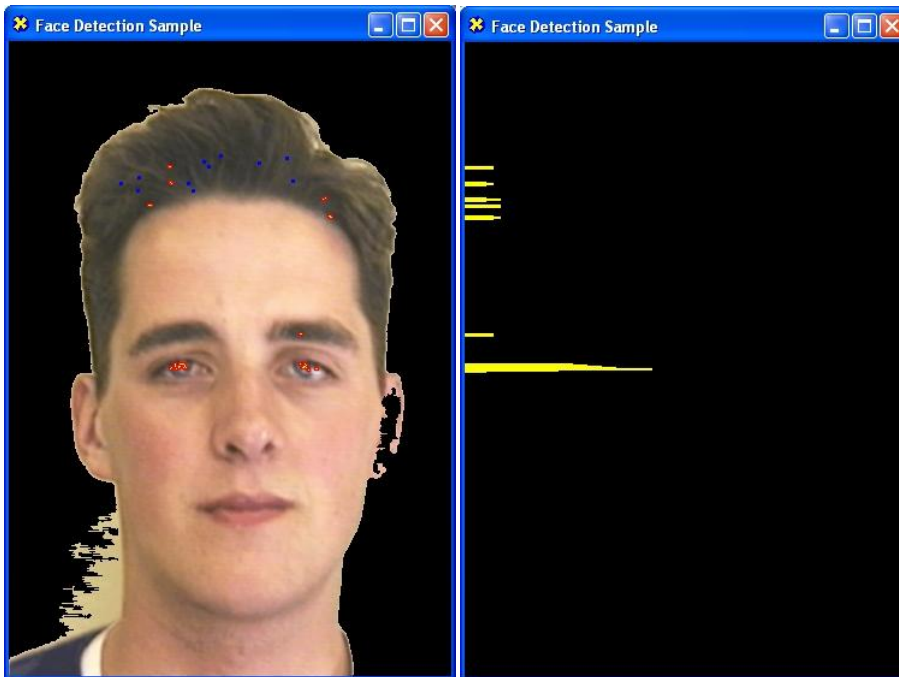


If the central pixel is yellow and at least one of the 16 extended neighbors is red, we keep the central pixel as detected on vertical gradient map, otherwise the central pixel's value is set to ( $R=0, G=0, B=255$ ). Indeed in the next step of the algorithm we just sum the pixel whose R value is 255 and  $B=0$ , it means only red and yellow pixels. Blue pixels won't be considered.

Example of vertical gradient map obtained after morphological treatment:



The horizontal sum will be realized on the treated gradient map.



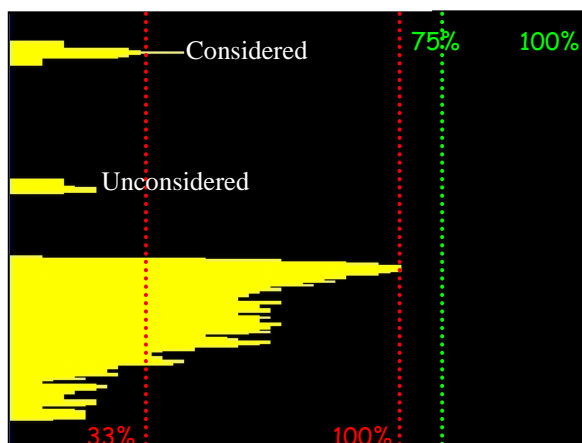
The summing step allows us to obtain a histogram showing the regions of highest vertical gradient.

This histogram thus obtained has to be analyzed to then determine the most probable peak(s) for eyes latitude. We saw that we are supposed to find an important peak on the eyes latitude due to the fact that we have two eyes on the same

latitude. Another characteristic is that this peak can't be really wide because eyes are confined inside skinny area, instead hair can lead to wider peaks because of the importance of the hair area.

### 1<sup>st</sup> Step:

Found the maximum peak of the histogram, if the maximum is higher to 75% of the picture's width, the maximum peak control identifier is set to 66% of this width otherwise we keep the real value. This last detail allows to consider later (during the 3<sup>rd</sup> Step) other peaks. Note that the maximum peak value can be higher than the picture width because of one coefficient used for better view on the histogram. This justifies the set of the control peak identifier to the maximum value 66% of the picture width.

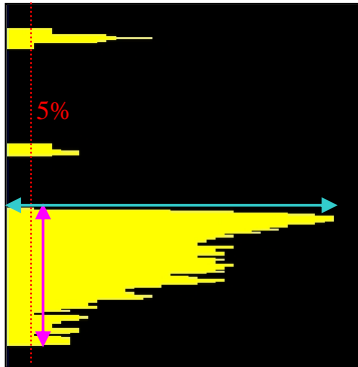


### 2<sup>nd</sup> Step:

We determine a minimum threshold on the maximum peak value; it is given by 33% of the maximum obtained previously (in 1<sup>st</sup> Step). Only the peaks higher or equal to this minimum will be considered.

### 3<sup>rd</sup> Step:

We start to sweep vertically the histogram on the row defined by 5% of the maximum of the considered peak and search for the width of this peak on this column, we then compute:



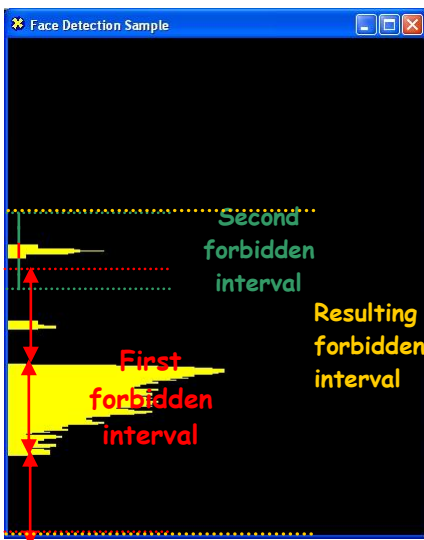
$$\text{(Maximum Peak Identifier)} / \text{(Width Peak)}$$

If the result is inferior to a threshold, the peak can't be representative of eye's latitude and we search the next maximum of the histogram, if the result is superior we keep this latitude and search a second potential one. Note that if only one peak is available it is considered as eye's latitude even if the result of the

division is inferior to the threshold.

### 4<sup>th</sup> Step:

Maximum searching area, it's important to not consider twice the same peak, so some limits are defined around the last peak treated. We can find two cases, first the peak is not susceptible to be eyes latitude, and it means that the width of the peak is too elevated, so the next maximum will be searched outside of the following interval:



$$[ Y\text{Starting Peak} - \text{Width peak}; Y\text{Finishing Peak} + \text{Width peak} ]$$

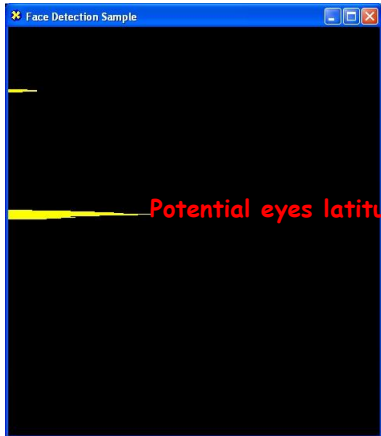
Second case if we just finished the treatment of one susceptible peak, the interval is given by:

$$[ Y\text{Starting Peak} - 2 * \text{Width peak}; Y\text{Finishing Peak} + 2 * \text{Width peak} ]$$

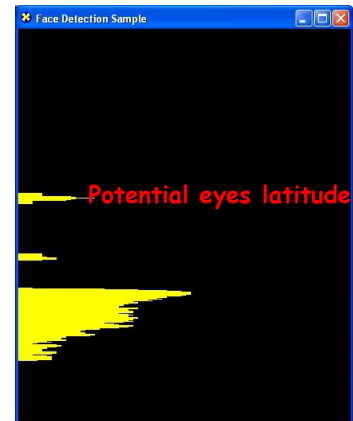
This forbidden interval is considered for each new lap in the loop to prevent finding twice the same potential latitude. If two forbidden interval are crossed, they are turned into one only forbidden interval.

Then the algorithm starts a new loop in the 4 steps, determining a new maximum inferior to previous one. The loop stops when two potential latitudes are found or when no more new peaks can be considered outside the forbidden interval(s).

Examples of encountered cases:



The smallest peak is inferior to 33% of the maximum, so it is unconsidered.



The highest peak is too wide to be considered.



The 2 highest peaks are distant enough and the division result is superior to the threshold. Both are considered.

- **Eyes longitude (old method)**

The hypothesis of front face detection permits to consider the symmetrical characteristic of a face or more widely face neck and shoulder together. For this reason each eye is in a particular side of the face. The first idea was to compute the cross correlation (presentation in next chapter) between the potential face area and its mirror, the symmetrical axe of the face would be find accurately, but this method is really heavy in time consuming. So we decided to compute the centroid of the region which is not so accurate but the gain of time is incomparable. Details are shown bellow:

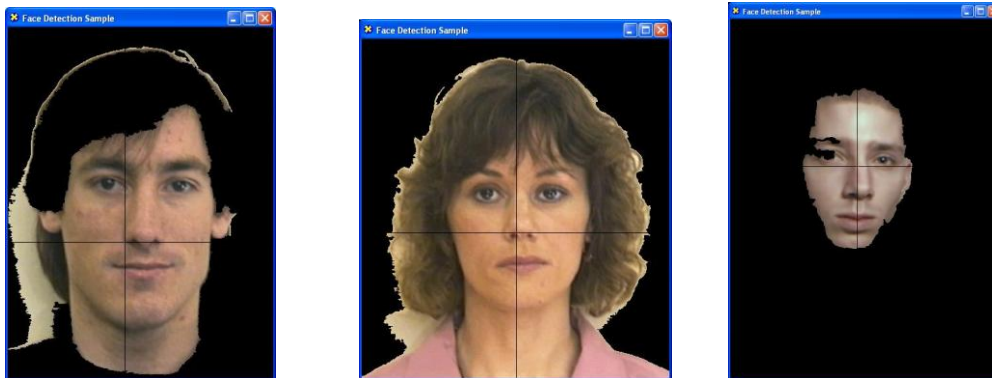
$$\bar{x} = \frac{1}{A} \sum_{i=1}^n \sum_{j=1}^m jB[i, j]$$

Where: **B** is the group of pixels which represents the facial region.  
**A** is the size in pixels of the facial region.

$$\bar{y} = \frac{1}{A} \sum_{i=1}^n \sum_{j=1}^m iB[i, j]$$

Note that for this computation, we are also considering the holes that the face region has. This justifies our filling function presented previously.

**- Results**



*Potential faces and centroid computation*

The weakness of this method is the face rotation, moreover background detection can shift the centroid to one side. However in the major of the cases the X axe of centroid coordinate is always between the two eyes. Note that the cross correlation's method is also disturbed by face rotation.

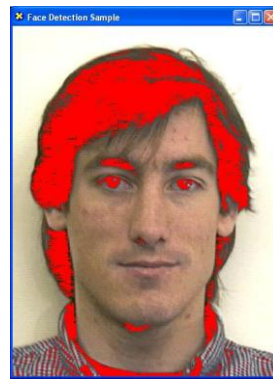


- **Criteria of eyes detection and old method**

We used previously the fact that pupils are the darkest part of the face; we made a threshold on the brightness for the eye characterization (actual algorithm). In the oldest method, the areas found were then labeled and classed by their size. However the pupils are not always well detected. We need two thresholds on the pupil's size for eyes detection.

The eyes are very complex areas, but the pupils are very easy to identify with color segmentation. Indeed this is the darkest part of any face.

**Tests of color based eyes detection**



Darkest parts of picture  
(Eyes in face area)

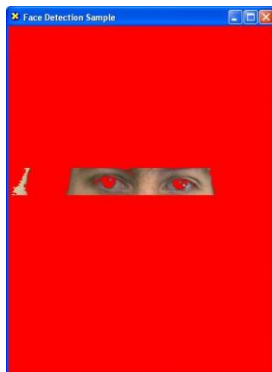
We

*can see quite good results in eyes detection but sometimes eyebrows are also detected.*

- **Principle of the decision's algorithm for the old method**

When the darkest parts of the pictures are shown, we need to determine which one are eyes. A first test is on the size of the dark spots that we will consider. Too big are eliminated and too small too. The first and the last spot belonging to this interval are detected. Then if one spot is isolated horizontally, it will be ignored. More precisely it means that there's no other spot on the same line (more or less 25% for face inclination). Then each of the two potential eyes must be in a respected side of the X-axis of centroid. The previous step of determining eye's latitude provides a faster computation, thanks to the limitation of the research area. Sometimes dark hair could make some detection errors.

**- Results:**



*Restricted search area      First and final darkest spot of this area*

- **Eyes longitude improvement**

We finally obtained a selection of the two best representative peaks for eye's latitude. The facial symmetrical axe will be introduced in the next chapter to determine other facial features; it will be determined more accurately than with the mass center technique and is not used anymore in eyes longitude determination.

After the horizontal sum, we proceeded with the same histogram method by summing this time vertically the yellow and red pixels for the two most potential eye's latitude on the previous vertical gradient map. The vertical sum is realized only for the (two) latitude(s) considered. The next step of the algorithm is about determining which one of the two possible latitude is the most susceptible to be the eyes one.

**1<sup>st</sup> Step:**

We determine the width of the considered peaks on the previous histogram for 20% of the **peak's maximum**.

**2<sup>nd</sup> Step:**

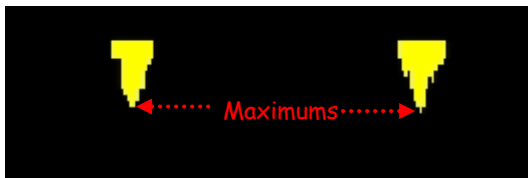
We sum all the yellow and red pixels located between:

**[peak's maximum latitude - 1.5\* peak's width; peak's maximum latitude + 1.5\* peak's width]**

This interval allows us to obtain the pixels representing the two eyes even if a small inclination of the head occurred.

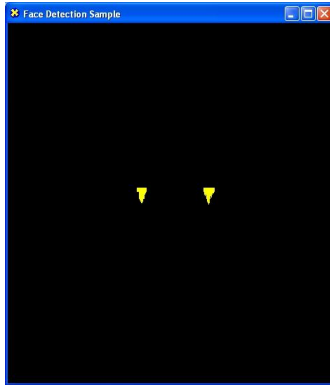
**3<sup>rd</sup> Step:**

Then we need to find the maximums of this sum for the left and right eyes. So we determine two maximum on each side of the symmetrical axe of the photo. The hypothesis of portrait photo is important; it means that eyes are located on different sides of the symmetrical axe.

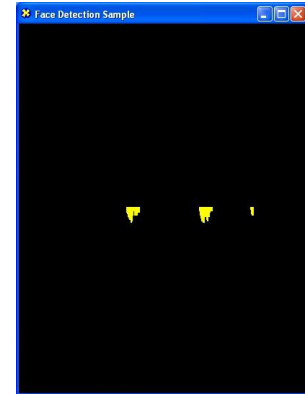


#### 4<sup>th</sup> Step:

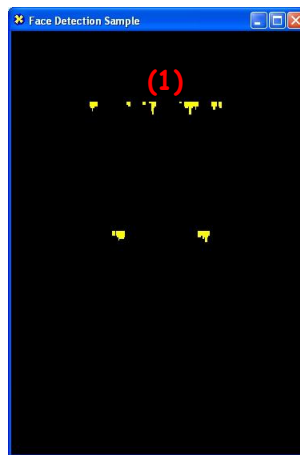
How to decide which one of the two latitudes is the right one considering the different representations?



This is the easier case, only one latitude is available. And the representation is typical the one of eyes area.

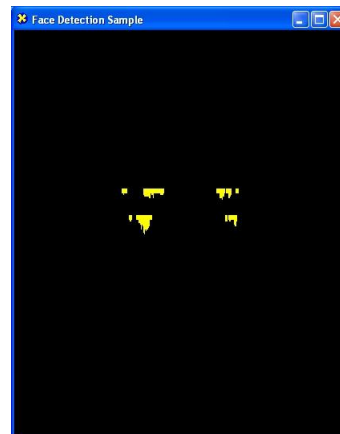


This choice is based on observations, by testing the previous histogram method on wide range of photos; we met generally really typical kinds of pixel sum. We can consider three distinct sorts:

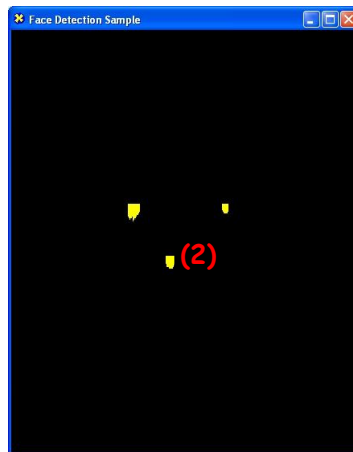


- Due to hair's area, we obtain a quite discontinued long line (1).

- Due to eye's area, we obtain two specific spots, eyebrows, opened mouth or nostrils can lead to a similar kind of representation.



- Due to errors detection we can find isolated spot (2).



We can then determine simple rules to eliminate some of these representations. First we can eliminate (2) if there's spot only on one side of the vertical symmetrical axe of the picture, then to eliminate (1) we need to determine that there are too many yellow pixels on this latitude to be eye's latitude.

Others conditions must be considered to avoid errors.

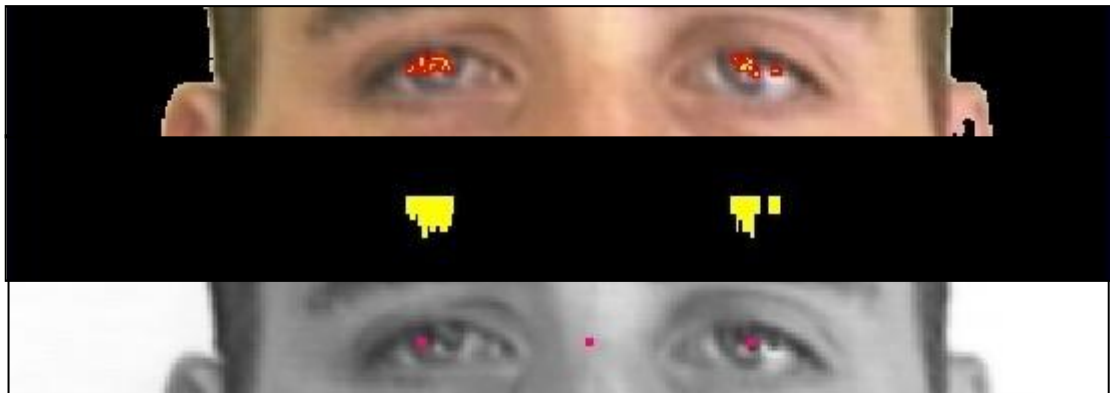
- The maximums must be located on each side of the symmetrical axe.
- The proportion of yellow pixels between the two maximums must be inferior to 50% of the total number of pixel between the two maximums.
- The distance between the two maximums must be inferior to 50% of the picture's width. We will see later that this condition is important to determine geometrically the potential mouth area

If after these tests the two potential latitudes are able to be the good one, we can test if the two latitudes are really close, if yes it means that we have in one hand real eye's latitude and in the other eyebrow's latitude. So the lower is kept.

Then the maximums corresponding to the eyes are stored and reused to find on the vertical gradient map the middle of this yellow and red area. The middle might be in the center of the eye. We obtain like this independent coordinates for each eyes. And head inclination problems are avoided.

The head inclination could lead to this kind of problem:



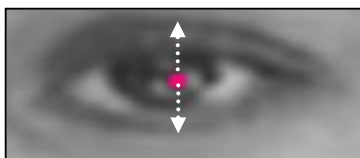


Results obtained after eyes longitude and latitude determination.

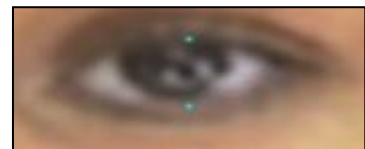
The eyes centers are not directly needed for the 3D modeling but they are essential to find the eyes borders and corners which them are necessary to the 3D model creation.

### 1<sup>st</sup> Step:

We start from the eye middle point that we found before and go up until we find a skinny pixel logically placed on the top border of the eye. The problem of different colors of eyes can be encountered sometimes. The same method is used for the bottom border.



Result after this specific edge detection we obtained the top and bottom borders:



The corners can't be found so easily, because of the white part of the eye which is sometimes similar to our skinny model. Moreover everybody has different eye's shape, the corners aren't on the same latitude, we can't generalize the cases, and the solution should be adaptable. We decided to follow the top and bottom borders to find finally the corners latitude.

### 2<sup>nd</sup> Step:

We start the edge detection on the lower border from the previous determined point to the right or to the left. The choice of the lower border is justified by the fact that fewer lashes are present and this area is less dark than the top border.

**3<sup>rd</sup> Step:**

In fact to avoid the errors of detection we make a mean on seven values of determined pixels. Moreover this mean should be inferior or equal to the precedent pixel latitude. Indeed the eye's shape can't allow finding a lower point. This computation is done until three times the distance between the top and bottom border from the eye center.



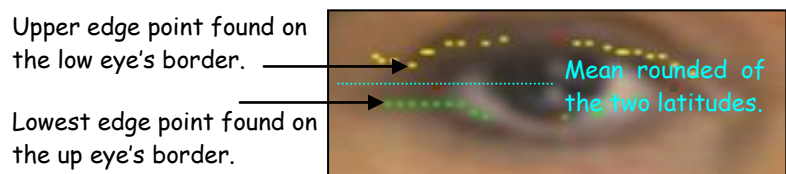
**4<sup>th</sup> Step:**

Then the same treatment is started but not from the top border, from the maximum found on the lower edge to the top border. The mean system is reused and for each new x increasing, the next edge can't be located on upper place.



**5<sup>th</sup> Step:**

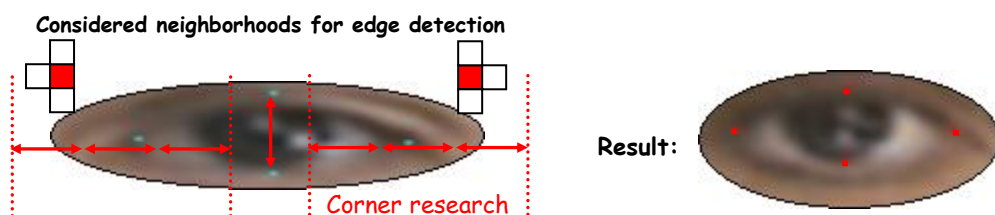
Now if the two edges are not connected, we compute the mean of the upper edge point found on the lower border and the lower edge point found on the top border. This should be the latitude of the corner.



This five steps treatment is realized for both of two eye's sides and for each eye. We now know the corner's latitudes, the longitude must be found.

**6<sup>th</sup> Step:**

Specific edge detection with 3 neighbors is started from center of eyes plus half of the distance between the two borders to avoid the edge detection due to the pupil and the white part of the eye. This treatment is realized in the two ways for both of the two eyes.



## V- Symmetrical Axe Determination

This chapter presents the method of cross correlation which is really suitable in patterns recognition. However this method require a certain computing time.

### a) Specification

Our previous eyes localization was the first step of feature localization, to detect the other facial features; a really useful tool is the facial symmetrical axe. The aim of this function is to determine the most accurately as possible this specific axe. The mass center method was drawn aside because of the risk to consider some parts of the background or clothes. The choice of more sure method led us to the Normalized Cross Correlation Method.

### b) Theoretical notion: What is the normalized cross correlation?

#### Simple definition

The cross-correlation of two images  $I_1$  and  $I_2$  is defined as the product:

$$\sum_{p1 \in w1} \sum_{p2 \in w2} p1 \otimes p2$$

Where  $p_1$  is the pixel index running over the domain of interest  $w_1$  in the image  $I_1$ , and similarly  $p_2$  a running 2-dimensional index over the domain of interest  $w_2$  in the image  $I_2$ .

The cross-correlation product denoted by  $\otimes$  can be defined by several possible functions. Common product definitions are:

- Sum of squared differences: 
$$p1 \otimes p2 = \sum_{w1, w2} (p1 - p2)^2$$

- Sum of products: 
$$p1 \otimes p2 = \sum_{w1, w2} (p1 p2)$$

The first criterion describes a measure of difference between the two interest regions of the images. For a pattern-matching application, this criterion is the quantity to be minimized to find the best matching point.

On the opposite, the second criterion describes a measure of resemblance between the two interest regions; it is a quantity which has to be maximized in order to find the best correlating point.

### c) Conception

Our approach to identifying a pattern within an image uses cross correlation of the image with a suitable mask. Where the mask and the pattern being sought are similar the cross correlation will be high. The mask is itself an image which needs to have the same functional appearance as the pattern to be found.

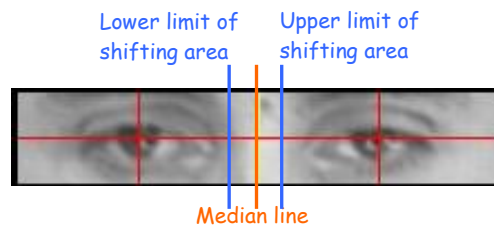
Our use is really specific; we consider the left eye as the image and the mirror of the right one as the mask. Moreover the size of the image is the same as the mask one.

In many image identification processes the mask may need to be rotated and/or scaled at each position. Thanks to the fact that we consider the two eyes of a same face, scale problem can't be encountered; just a too important inclination of the head can disturb the process.

Thanks to the previous step which consisted in localize the eyes coordinates we can reduce the area of the image that should be covered by the mask.

#### 1st Step:

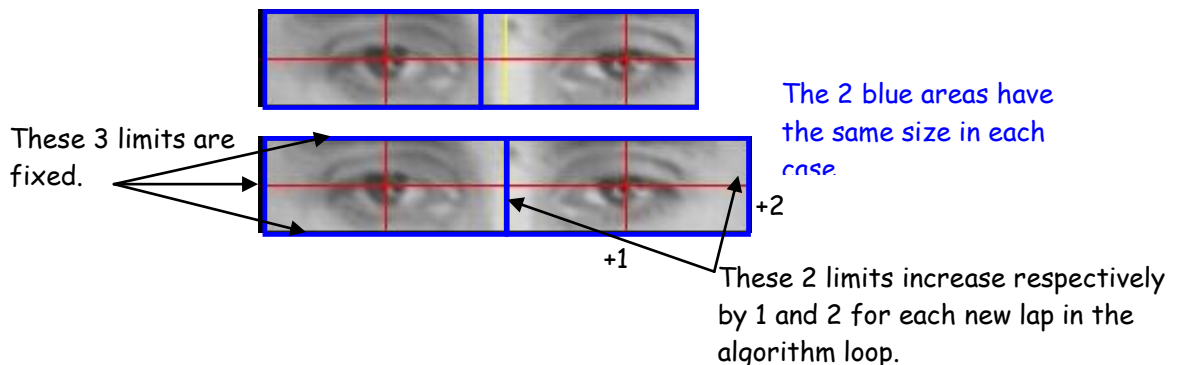
We determine the median line between the two eye's longitude, and the inferior and superior limits of shifting area. It means that the maximum of Normalized Cross Correlation will be found for the most potential symmetrical axe between these two limits around the median line.



#### 2nd Step:

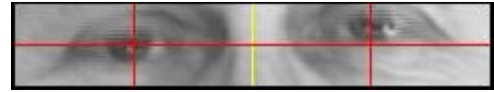
We determine the size of the two areas between which we will compute the cross correlation. The size is not fixed; it depends on the considered potential symmetrical axe. That's why we will compute the **Normalized** Cross correlation function. It will allow us to compare results of different size areas computation.

Let's see two examples:

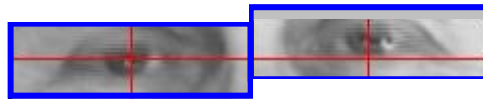




In some case the eyes are not in the same latitude:



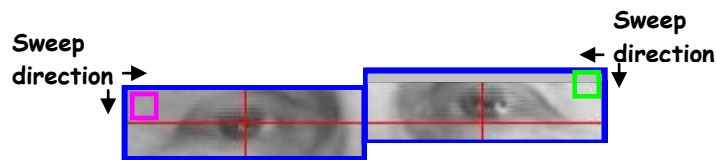
The two blue areas will be shifted of the difference between the two eyes latitude. We will obtain something like that:



It allows to avoid a bad computation due to the head inclination.

### 3<sup>rd</sup> Step:

We sweep the two areas in opposite directions. And compute the normalized cross correlation function. Store the result for each new potential symmetrical axe tested.



$$X_{image} = (R+G+B)/3$$

$$X_{mask} = (R+G+B)/3$$

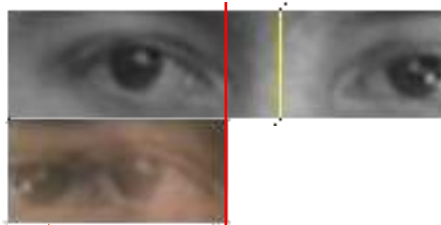
In our case,  $X_{image}$  and  $X_{mask}$  represent the mean of the color photo RGB values of each pixel for every (i, j) of the blue areas. So the normalized cross correlation is computed on gray scale levels.

### Normalized Cross Correlation Formula:

$$\frac{\sum (X_{image} * X_{mask})}{\sqrt{(\sum X_{image}^2 * \sum X_{mask}^2)}}$$

For each value between the inferior and superior limit of shifting area, the 2<sup>nd</sup> and 3<sup>rd</sup> steps are restarted. When the superior limit is reached, the loop stops and the maximum of the cross correlation function is determined.

**Results:**



Superposition of the two pictures  
(Case of low value for the cross  
correlation function)



Facial symmetrical axis,  
after cross correlation  
computation.

Superposition of the two pictures  
(Maximum of the cross correlation  
function)

Let's see other results:



Case of inclined head, without correction



Case of inclined head, with correction

The results provided by this function allow now to start an easier localization of all the others facial features thanks to morphological considerations.

## VI- Mouth Localization

Mouth is one of the most difficult facial features to detect, because of its various possibilities of shape and size changes.

### a) Specification

We need to approximate the mouth area; the two corners and the lips limits are required for the 3D model creation.

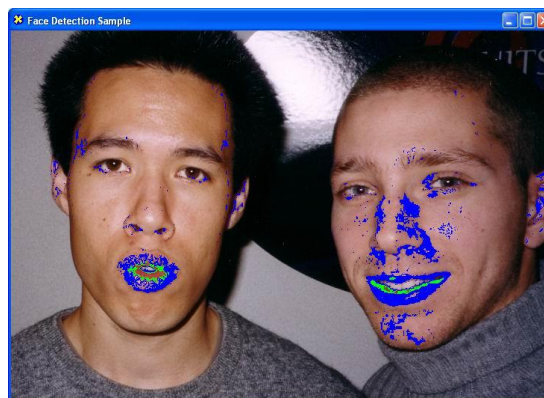
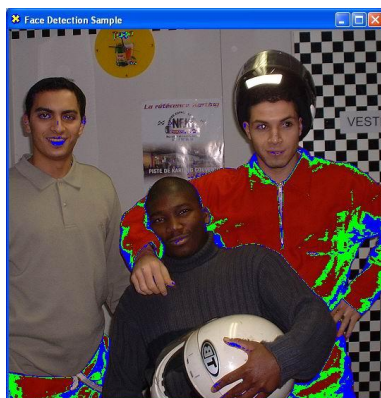
### b) Conception

The first idea was to use a color segmentation to find mouth in the potential face area. Then crossing this method with a based edge detection one, it would permit a precise localization of this ROI.

- **Color considerations**

This time we met some problems to determine a specific colorimetric area due to the redundancy with the skin color area; because of it we needed to use some edge detection method.

This color segmentation task is the less significant by its result. We can't determine a colorimetric area for mouths by the way of color based detection.



Are 2 different thresholds for the mouth color segmentation.

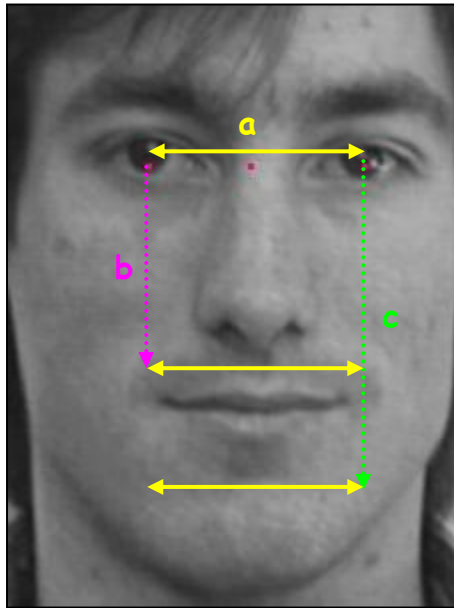
### Remarks:

These results were obtained thanks to the first method which consists in approximating a color area by rectangle, with 4 thresholds. We tried with some samples of lips to compute the probability of belonging to a mouth part. However after trying this probability method, the weakness of this technique was an evidence, in fact color of lips are so close to skin color that it wasn't possible to use it like a good criteria of segmentation.

- **Geometrical and morphological considerations**

When you know where the eyes are, you are supposed to know approximately the mouth position.

**Schema:**



Statistically eyes are always at the same distance from mouth. And we can define a potential mouth area thanks to this consideration; however eyes must be precisely localized. If the person doesn't show any special emotion, the corners of the mouth are approximately on the same latitude as the pupils.

**a** is the distance between the two pupils, determined before.

$$b = 0.8 * a \quad \text{and} \quad c = 1.5 * a$$

- **Edge characterization**

Previously we determined the potential mouth area shape and location, now we need to characterize the mouth itself. However the mouth characterization has been turned into an adaptable one to avoid the problems due to the difference of brightness from one picture to another.

**1<sup>st</sup> Step:**

A threshold is fixed on the number of pixel detected as belonging to the mouth inside the potential mouth area. The minimum required is 10% of the whole mouth area. If the number is inferior to this threshold, a new lap in the loop is started with a decision's threshold decreased from 10 to 1 by step of 1. When the minimum number of pixels required is obtained, the next step can be started.

Vertical edge detection:

- We passed from RGB domain to YCbCr one.
- Test on the chrominances (Cb, Cr) and (Cb<sub>line+1</sub>, Cr<sub>line+1</sub>).
- Test on the luminance (Y) and (Y<sub>line+1</sub>) because the mouth area is darker as cheek for example.

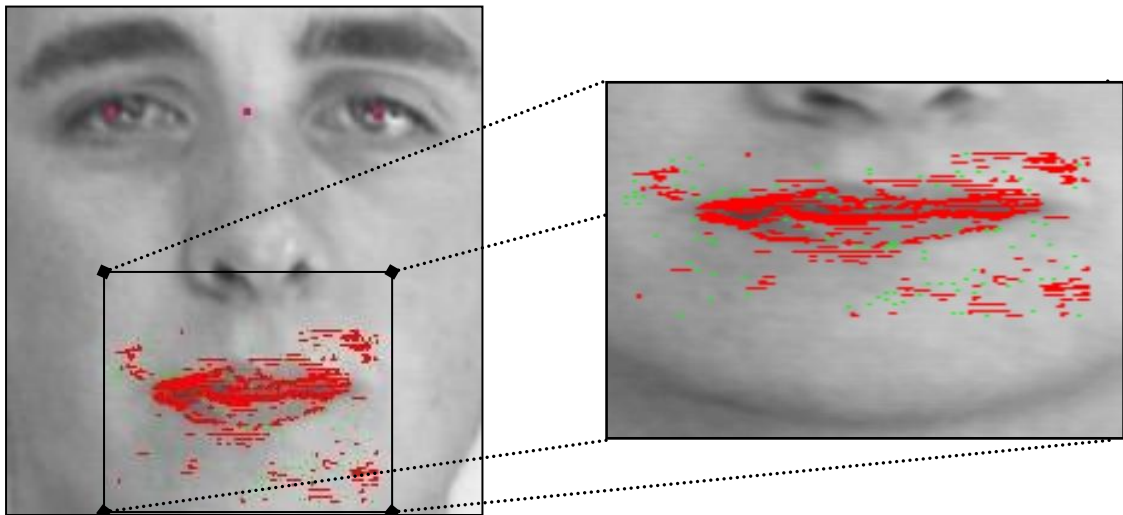
"Test" means that we compare the variation between these variables and the threshold to obtain the vertical gradient. We start with a high value of threshold and it will decrease until satisfying the condition of 10%.

## 2<sup>nd</sup> Step:

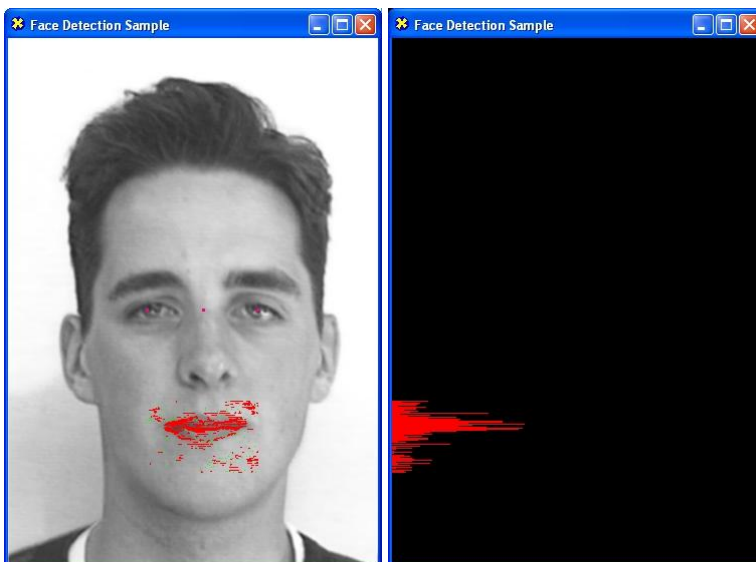
A morphologic treatment allows to delete all the isolated pixels (in green on the picture below) which are supposed to be some errors of detection and don't belong to the mouth.

The structuring element used is: 

If none of the neighbors is red, the central pixel is turned from red value to green one.



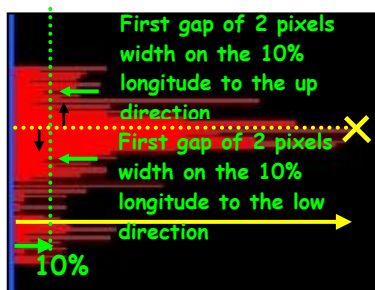
After the color characterization we use one more time the histogram method to determine the location and size of the mouth.



**1<sup>st</sup> Step:** The first thing consists in summing horizontally the potential mouth pixels from the previous treated picture.

**2<sup>nd</sup> Step:** We search the maximal peak; it normally corresponds to the mouth latitude.

### 3<sup>rd</sup> Step:



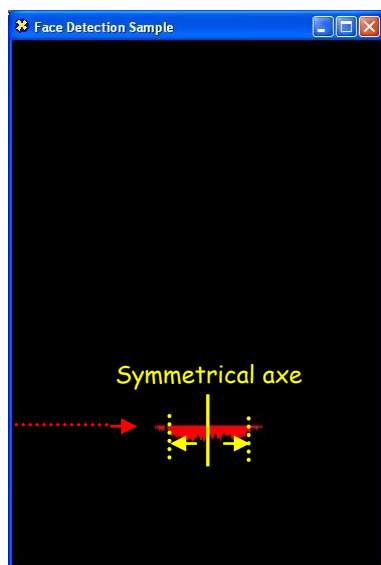
Then the top and the bottom of the mouth are determined considering the width of the peak for 10% of the maximum. Two new latitudes are determined like this, the superior and the inferior one. However if the difference between the middle mouth latitude and upper one is superior to the difference between the lower mouth latitude and the middle one, we set the lower as necessary to obtain the same variation. Indeed the lower lip is always more than or as wide as the upper one and its color is really close to the skin color.

Sometimes a gap of 2 pixels width can't be encountered except on the end of histogram, only 1 pixel width can be found. To avoid errors due to this kind of configuration, after 3 times finding a gap of 1 pixel width the next one is considered as being the one standing for the lip latitude.

Then we search the width of the mouth.

### 1<sup>st</sup> Step:

This time we sum vertically the potential mouth pixels included between the inferior and superior mouth latitude determined previously. If these two latitudes are too close, a predefined summing area is used.



### 2<sup>nd</sup> Step:

Then we start to sweep the histogram on the line of the middle mouth latitude. The total number  $N$  of red pixels on this latitude is counted and a rate  $T$  is determined for the next step.

$$T = 0.7 * (N/2)$$

### 3<sup>rd</sup> Step:

We start to sweep the histogram on the row of the middle mouth latitude from the symmetrical facial axe to the left and right direction when the first black pixel is found and the  $x$  coordinate of this pixel is superior to the  $X$  coordinate of the facial symmetrical axe by  $T$ , it means that the corner of the mouth is reached. This is the same to the left direction when the limit  $X-T$  is reached.

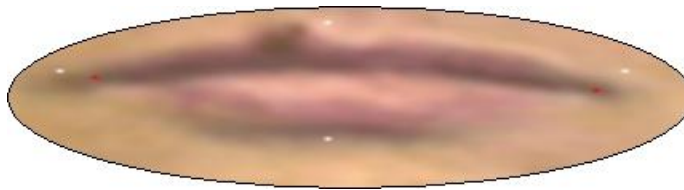
#### 4<sup>th</sup> Step:

This last step consists in finding more accurately the two mouth corners. A small area is swept and allows us to find the darkest pixel of this area. Indeed the lips are darker than normal skin area and we always notice a shadow between the two lips. Considering the following result, now the two corners are localized more precisely and each corner has his specific latitude if an inclination is eventually encountered.



- **Results**

On the following results the two white points standing for the corners are the points before the fourth step, the red ones are determined thanks to this fourth step. This last step provides some better results and is able to compensate the head inclination.



We can see that the upper and lower lips limits are quite well approximated.



We never encountered some corners inside the mouth but some times the corner can be badly determined and the fourth step can't rectify the localization error in every cases.



## VII- Nose Localization

Now the next facial feature which will be treated is the nose, this time the deformation problems don't exist; only some difficulties due to the brightness can be encountered.

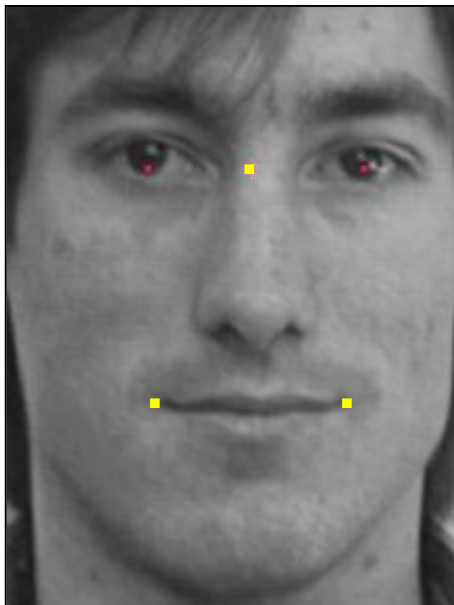
### a) Specification

We need to determine some strategic point for the future 3D modeling, like the nostrils, the end of the nose and the top of the nose. This last point was already found during the cross correlation step.

### b) Conception

The use of color segmentation can't be envisaged because of the uniform color of nose and cheeks.

- **Morphological considerations**



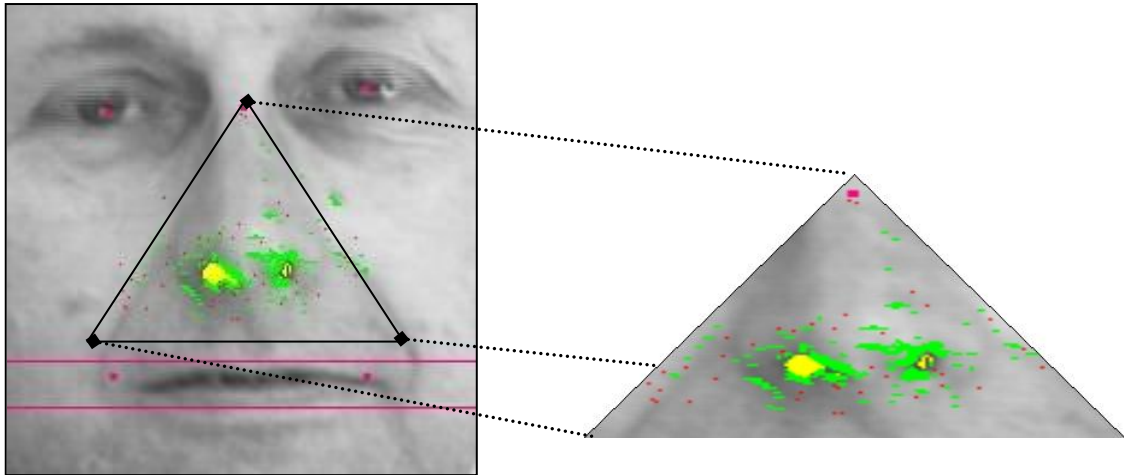
Now thanks to the previous steps we know enough coordinates to intent a potential nose area definition. The three yellow points on the picture are able to provide a potential area containing the whole nose.

Indeed the nose is normally less wide as the mouth and situated between the eyes and mouth latitudes. Even if the head is a bit inclined, these statements are still valid.



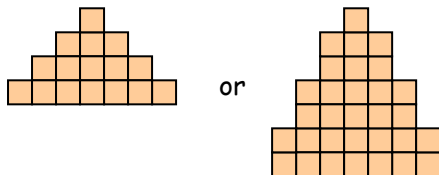
- **Potential nose area definition**

The ideal case would be like on the following pictures, a triangular area fitting on the nose. However the problem encountered comes from the picture structure itself.



In fact if we want to cover a wide range of different morphologies, the triangle must be wide enough and not too wide either to limit the error detection.

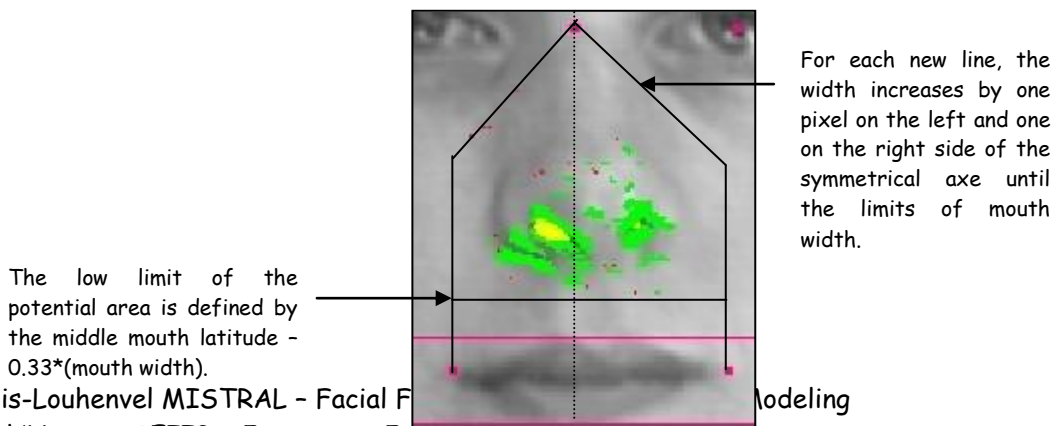
The picture structure is defined by the unity of pixel. So we can't consider half of pixel; for this reason the triangle shape is limited to this kind of configuration:



In one case the risk of too wide triangle could lead to treat some unexpected part of the face, in the other case a too thin triangle couldn't contain the whole nose.

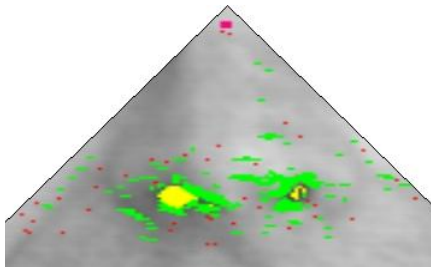
**1<sup>st</sup> Step:**

Define a potential search area for nose; we know precisely where the eyes and the mouth are so an area with specific shape can be defined. The potential area is defined like described below.




### 2<sup>nd</sup> Step:

We noticed that noses always present some horizontal edges on their low part and the brightness is really low for the nostrils. We created a vertical gradient map to determine the width of the nose on the lower part. Moreover the nostrils must be found for the 3D model creation. So one more time an adaptable loop allows us to characterize it in most of the cases by increasing the brightness in the interval [20; 100] with a step of 2. The condition to stop this loop is to obtain at least 10 yellow pixels on each side of the facial symmetrical axe inside the predefined area. Otherwise the loop will stop when the limit of brightness will be reached. This limit is fixed to 100 on 255 levels (quite dark).

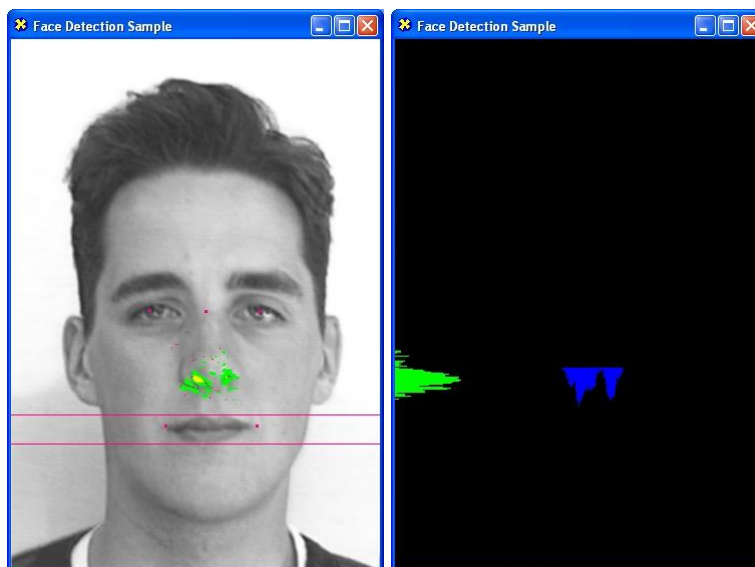


The yellow pixels represent the nostrils and the green one the vertical edges. One more time the isolated pixels are set to red value thanks to the following structuring element sweeping the area.

Structuring element: 

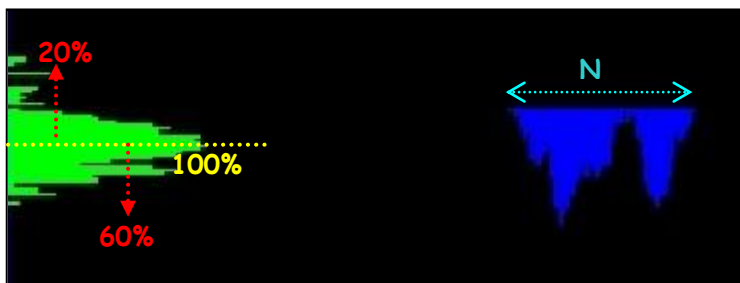
### 3<sup>rd</sup> Step:

We summed horizontally all the underlined (only the yellow and green one, ( $G = 255$ )) pixels to find the latitude of the lower part of the nose. This latitude corresponds to the maximum peak on the following histogram.



#### 4<sup>th</sup> Step:

When the latitude is determined a vertical sum is started from the upper limit determined for 20% of the maximum peak when the first black pixel is encountered to the lower limit determined by the same way, but in the other direction and for 60% of the maximum peak.



The number  $N$  of pixels on the first line of this blue pattern is calculated. And a rate  $T$  is determined:

$$T = 0.7*(N/2)$$

#### 5<sup>th</sup> Step:

The blue pattern obtained is swept from the symmetrical axe to the left and to the right directions, if the  $x$  coordinate of the running pixel is not such as  $X_r - X_{sym} > T$  and a black pixel is found, this black pixel is ignored. However if the condition on  $T$  is satisfied, the blue pixel before the first black pixel found defines the limit of the right side of the nose. This operation is realized in the other direction to find the left side. Using this rate we can avoid some detection errors due to case like shown below:



#### 6<sup>th</sup> Step:

It lasts now to find the nostrils coordinates, on the blue pattern resulting of the vertical sum; it seems pretty easy to determine the maximum of the peak to determine the  $x$ -coordinates. However we chose another technique which can lead directly to the two coordinates of each nostril. In fact we compute the mass centers of the darkest parts of the nose on each side of the symmetrical axe and each of this two mass centers represent the nostrils. This step is computed only if the darkest parts are detected; sometimes it can happen that just one nostril is underlined by the adaptable method. In this case the second one is determined finding the symmetrical point of the first one.

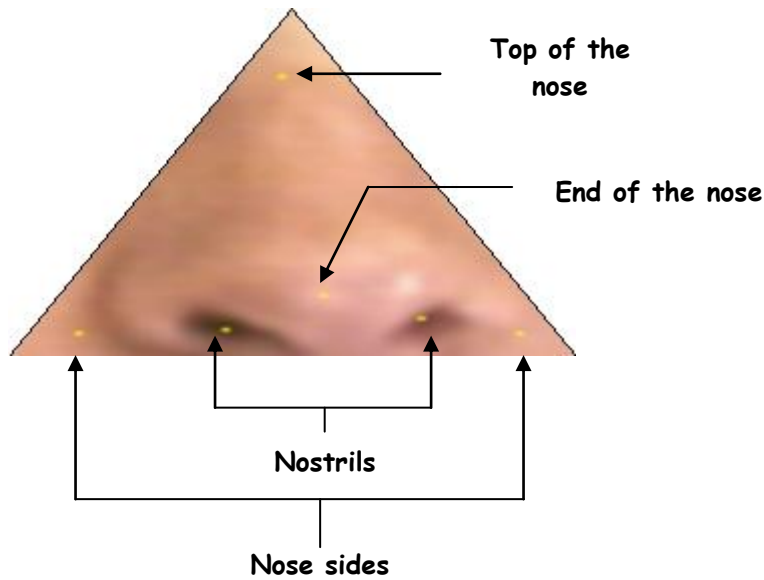
#### 7<sup>th</sup> Step:

The ultimate needed point is the end of the nose. This last step finally gives us its location, the  $x$ -coordinate is the mean of the two  $x$ -coordinates of

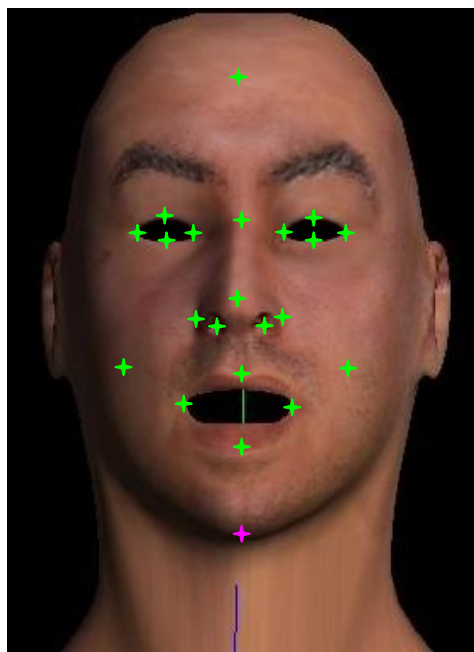
the nostrils. The y-coordinate is search from the lower latitude of the two nostrils until this latitude minus 1.5 times the distance between the nostrils x-coordinates.

A test on the brightness is realized and the most saturated pixel is standing for the end of the nose. Indeed the end of the nose is the nose part the most close to the camera and the maximum of saturation is always observed on this specific place.

Results:



As you can see on the schema below we localized all the needed points except the shin. The cheeks points were determined geometrically, between the nose and the mouth. The point standing for the beginning of the hair was determined by a vertical sweeping on the facial symmetrical axe until finding a non skinny area (case of dark hair) or a big gap on the luminance (case of hair detected as skinny area).



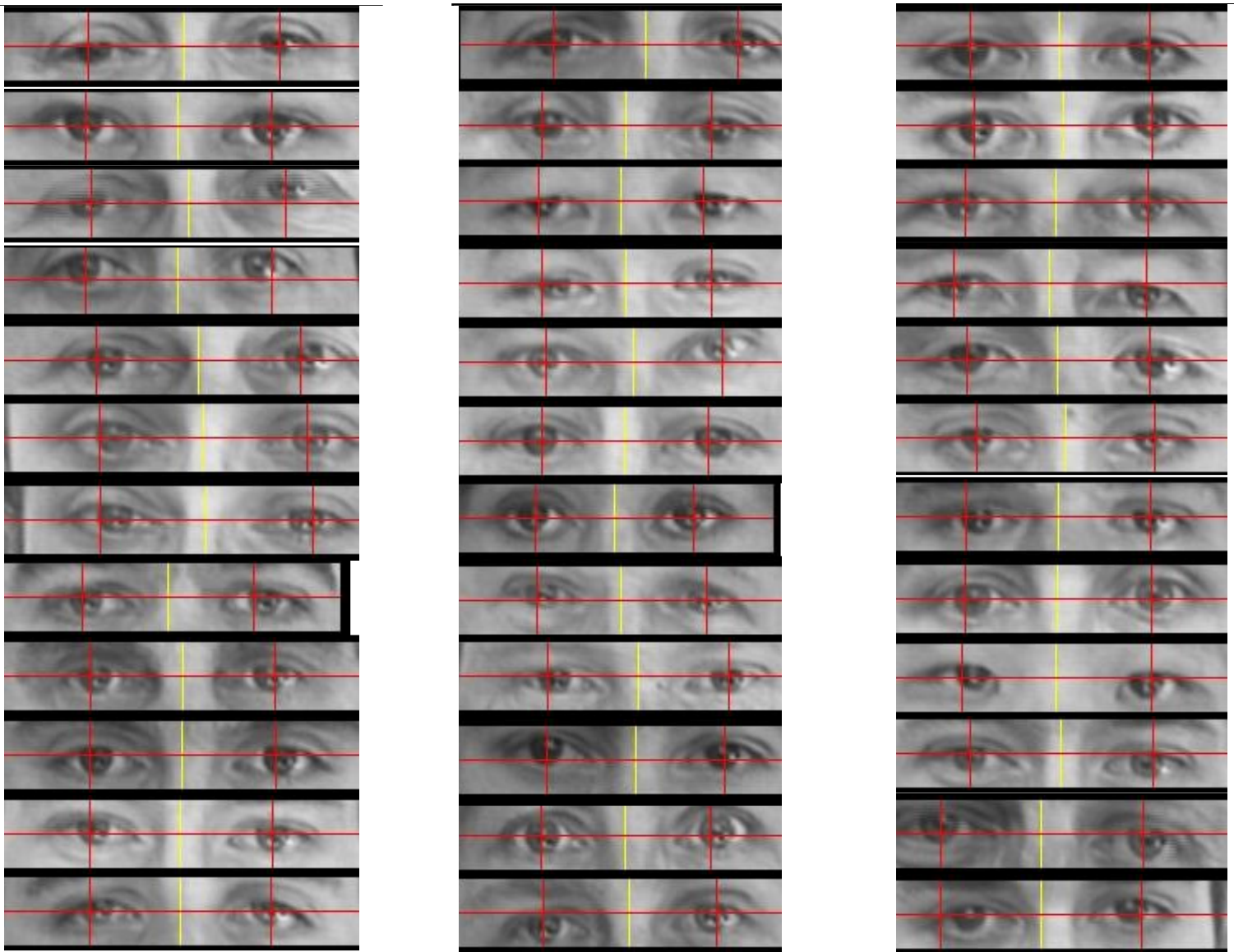
The shin and the third coordinate of the needed points will be determined thanks to the profile algorithm presented in the second part of this report.

## VIII- Conclusion on frontal algorithm

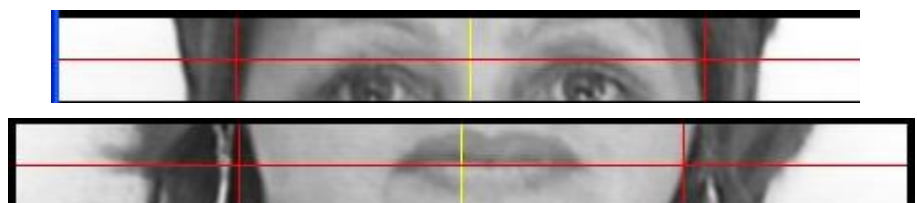
The developed algorithm is based on eyes localization, morphological considerations and generalizations, after testing this algorithm on a face data base (about 50 photos from the University of Stirling Face Data Base); we obtained about 88% of successful eyes localization. The other facial features are then more or less well localized. The main disturbing factor is the high saturation.

Our current implementation is limited to the detection of frontal human faces. An extension is presented in the next part to treat profile photo to obtain the missing points and coordinates.

### *Examples of eye localization results*



Some errors encountered:

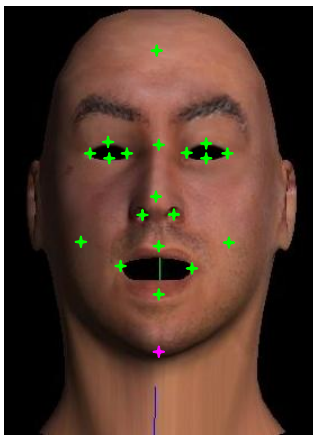


## PART 2: Profile view algorithm



## I- Facial edges detection

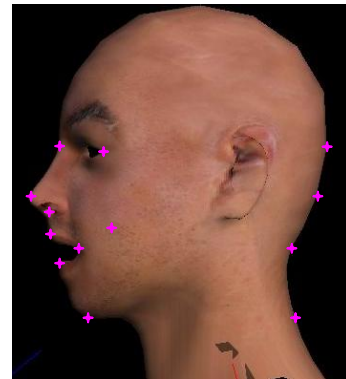
This new part will allow us to obtain the third coordinate of the previous detected points. This is necessary to create a 3D model. Moreover some useful points like the shin weren't detectable due to the uniform color of certain facial part. The profile view will solve this problem.

### a) Specification



Considering the list of needed and obtained points, this new algorithm will permit to find the missing points or coordinates.

-  Needed points
-  Detected points



### b) Conception

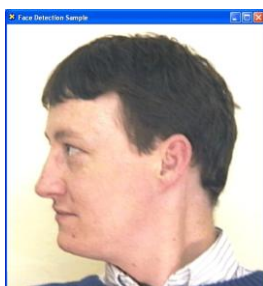
The algorithm is based on a ROI characterisation. And then some coarse to fine steps allow to target the needed points. Note that some basic functions of the previous part have been reused.

- **Noise reduction (cf. p.16)**

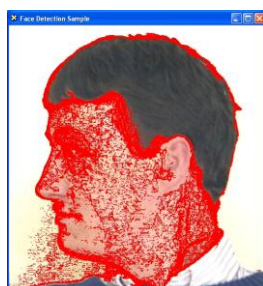
This function is strictly identic to the one used for the frontal view algorithm.

- **Edge detection**

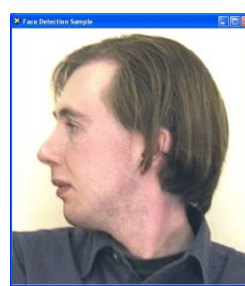
First thing was about underlining the head edges. We defined a wide edge detection based on the skin color gradient. Because of the background's color knowledge, it was easier to realize this part.



*Filtered picture*



*Treated picture*



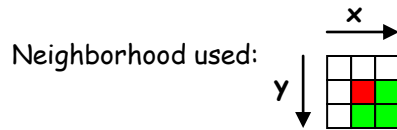
*Filtered picture*



*Treated picture*

We can see that many pixels are detected, and they don't represent evident edges. However this is not a problem, indeed we need to find the face edges in most of the cases with different brightness conditions.

#### Detail of this edge detection:



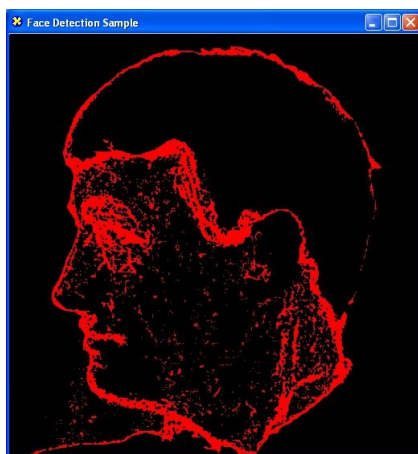
This specific neighborhood allows detecting the vertical, horizontal and diagonal edges. The condition on the running pixel is that it should be a skinny one (red pixel). Moreover the gradient on the luminance should superior to 1; it means that the running pixel is detected if its luminance ( $lu$ ) and the three other luminances ( $lu_1$ ,  $lu_2$  and  $lu_3$ ) satisfy the conditions below:

$$\text{abs}(lu-lu_1)>1 \ || \ \text{abs}(lu-lu_2)>1 \ || \ \text{abs}(lu-lu_3)>1$$

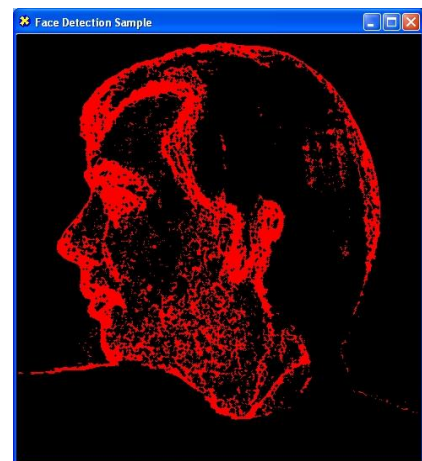
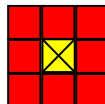
The absolute value allows detecting edges from dark to light or light to dark parts.

- **Morphological treatment**

Then we started specific morphological treatments to fill the red areas obtained previously. It will permit to obtain a potential bigger connected component.



Structuring element used:



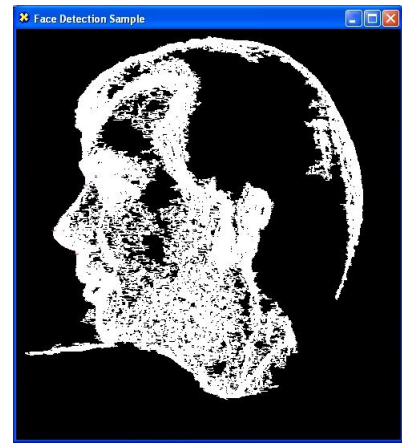
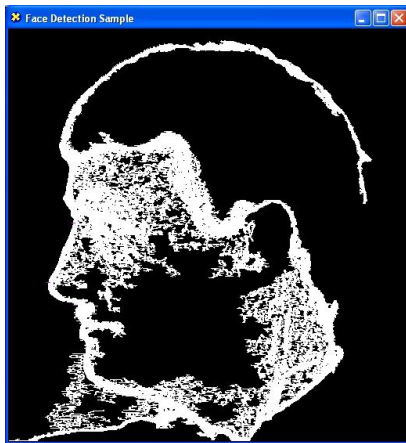
The condition to set the central pixel to the red value is to have at least two red pixels on the eight neighbors when the central pixel is black. This condition is not too difficult to satisfy. The isolated pixels or small groups of pixels will be deleted on the next step, but not thanks to morphological treatment.



- **Connected component labeling step**

Instead of treating one more time the pictures with morphological treatments, we select the biggest connected component. And all the other are erased. We obtain by this way the head's edges. After the two previous treatments, the biggest connected component should contain the facial features edges.

**Examples of results:**



This function was already implemented and used in the front view algorithm. For more details, see the page 23.

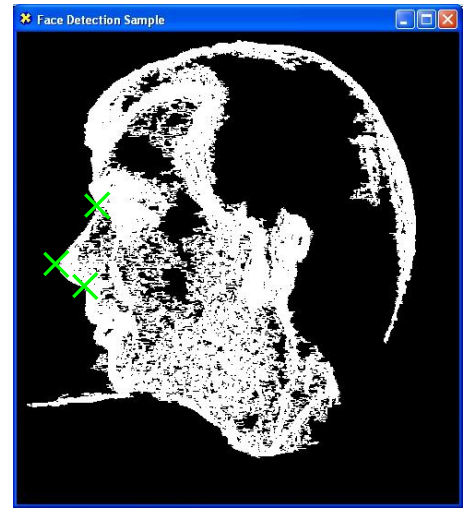
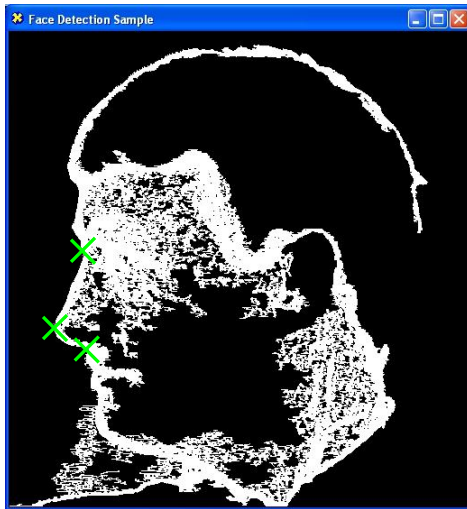
- **Easy points determination**

The human profile is really specific, so on this part we start the detection thanks to simple observations and generalizations about human morphology.

**Observations:**

- The end of the nose is the closest facial part to the picture vertical border. Except if we consider the shoulder or hair for different hair cut.
- The top and base of the nose are the places where the facial edges are quite verticals if we compare to the nose itself.

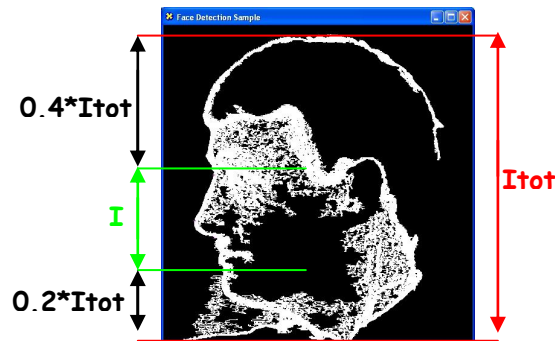
Thanks to these considerations we searched for the easiest points to find. They are shown below.



**1<sup>st</sup> Step:**

We determine a vertical interval where the end of the nose is supposed to be.

This interval **I** is determined as it follows:

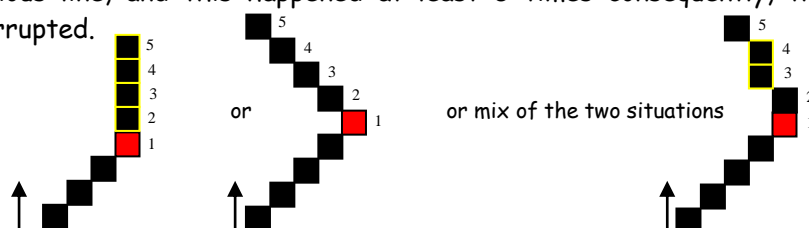


**2<sup>nd</sup> Step:**

We sweep the **I** interval line by line until finding the minimum of longitude. This minimum is the end of the nose.

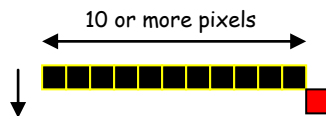
**3<sup>rd</sup> Step:**

The top of the nose is found by sweeping line by line the picture from the end of the nose latitude to the up direction. When the longitude of the first white pixel encountered from the left is superior or equal to the one of the previous line, and this happened at least 5 times consequently, the search is interrupted.



#### 4<sup>th</sup> Step:

The same method is used for the down direction to find the nose base. But this time if the running latitude of the first white pixel is superior by 10 pixels to the previous line, the search can be interrupted. It allows finding some different shapes of nose.



We can now determine new areas of search, based on the previous results, to get some new points. So two new areas are defined, one to find mouth corners, shin and upper and lower lips limits; the other one is dedicated to eye's corners localization. Each of these areas is treated separately, to underline these points.

## II- Coarse to fine (eyes and mouth areas)

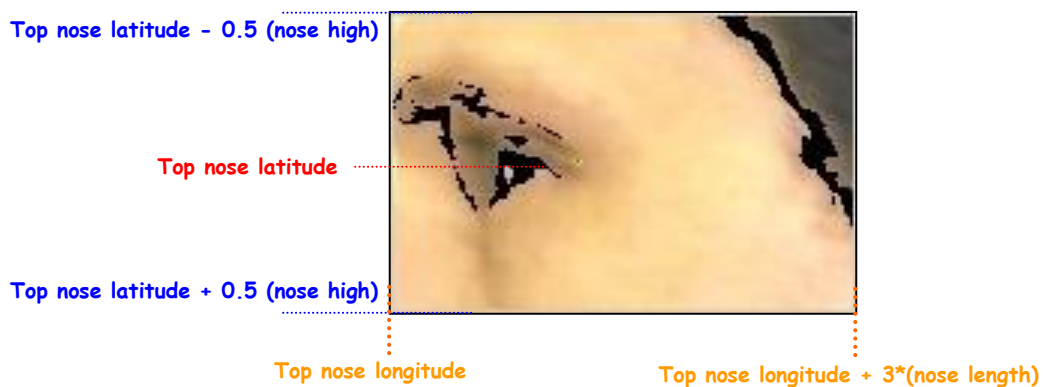
Previously we determined three points which will allow us to define some new search areas closer and closer of the needed points. Some morphological observations allowed us to define these areas; these observations are suitable only on normal face.

The new areas:



We can see the two new areas, and the result of the characterization of each of these areas. The details of the areas construction and pixel's characterization is given on the next page. Let's start with the eye area.

Eye's area definition:



### Eye's area treatment:

The following threshold allows to underline the specific area of eyes. It's based on the fact that too high saturation areas aren't considered and too dark areas either. This color segmentation was tested on a range of photos and provided good result in eyes characterization.

$$200 > R > 100 \ \&\& \ 200 > G > 100 \ \&\& \ 200 > B > 100$$

### Research algorithm:

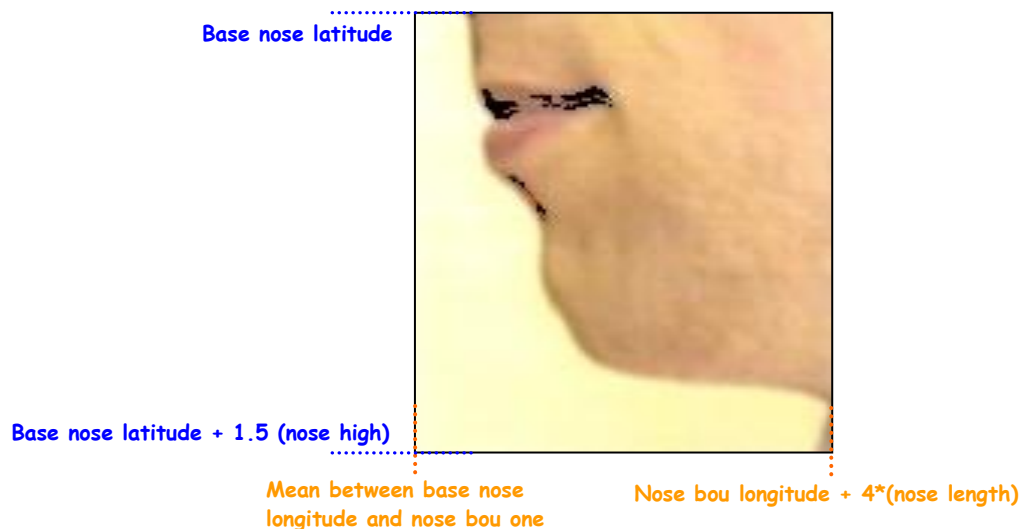
The searching area is one more time reduced and we start to sweep this new area from the left to the right, line by line. The last black pixel found on one line stands for the eye corner after each new line swept, we verify if a new pixel is not on the same longitude as the previous one or more on the right.



Point standing for  
the eye corner

One more condition is needed, to valid the pixel chosen. On its latitude at least 10 black pixels should have been finding before, it permits to avoid considering isolated pixel, without doing a morphological treatments.

### Mouth area definition:



### Mouth area treatment:

Horizontal edge detection in skinny area is realized to underline the mouth. It means that a pixel is detected if:

$$\text{abs}(l_{u1}-l_u) > 3 \ \&\& \ l_u < 128$$

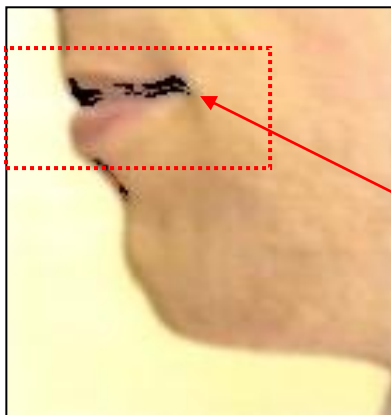
The potential area considered is not too light because of the condition on the running pixel luminance. And the gradient between the brightness of the running pixel on the line  $j$  and the pixel on the line  $j+1$  is computed. The threshold 3 is quite elevated, it allows to consider only the mouth area which is the darkest place in this part of skinny area.

To limits the possible errors in the next step of the algorithm a morphological treatment is realized with the following structuring element:



It allows us not considering isolated pixels anymore.

### Research algorithm:



Like in the research of eye corner the area is reduced and the same kind of sweeping is realized.

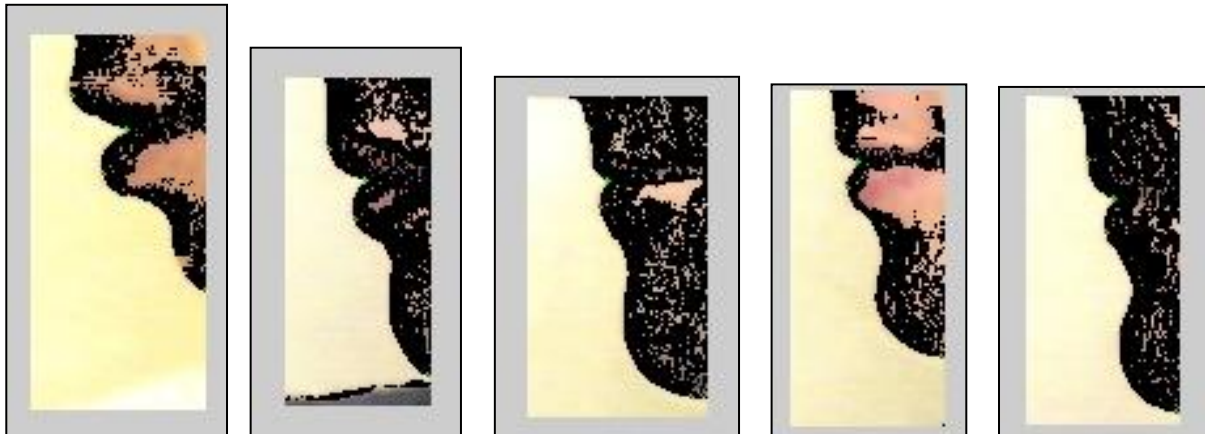
**Point standing for the mouth corner**

**III- Coarse**

**to fine (lips and shin determination)**

Previously we determined one important point which will allow us to define some new search area closer and closer of the new needed points. This time we want to localize the top and bottom of the lips. Some general morphological considerations are the base of this determination.

The new area:



Now we know the mouth corner coordinates, we can then limit the next area of research.

#### Area treatment:

This time a vertical edge detection is realized, and some conditions on the brightness and being or not skinny pixel are considered to obtain the result here above.

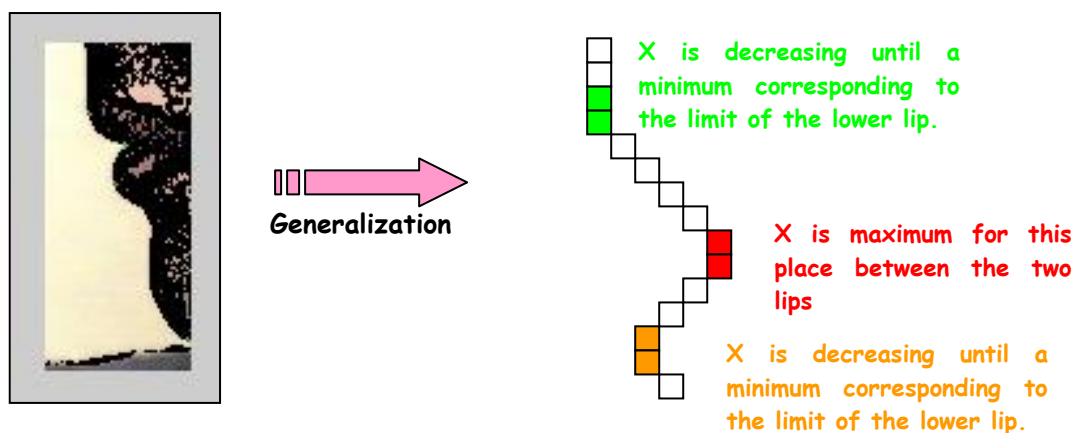
$$(lu_1 - lu) > 0 \ \&\& \ lu < 200$$

$lu$  is the running pixel luminance on the coordinates  $(i; j)$ , and  $lu_1$  the luminance for the pixel located in  $(i+1; j)$ . Moreover the chrominances of the running pixel have to be included in the interval defined by our skin model. The knowledge of the background color leads to the 200 threshold on the running pixel luminance. Indeed a white background presents the highest saturation.

#### Research algorithm:

##### 1<sup>st</sup> Step:

We noticed that the lips edges are really particular, and by generalizing our observation, we decided to find the point which stands for the middle of the mouth between the two lips.

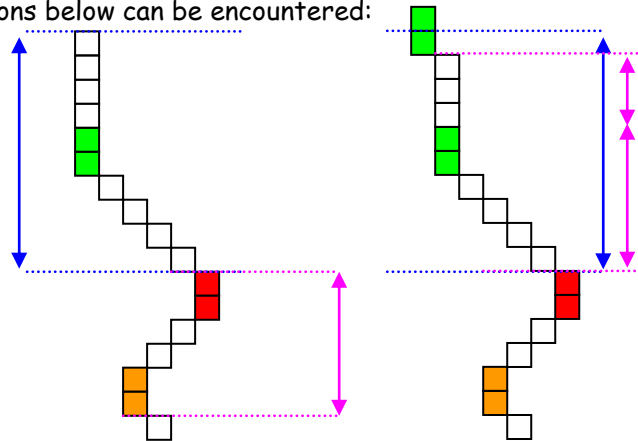


## 2<sup>nd</sup> Step:

We are still closer and closer of the two needed points, we started with the lower lip because the human morphology makes it easier to detect with our method. So we swept the edge from the maximum (in red on the previous schema) to the minimum (orange pixels). The sweeping interval doesn't need to be so restricted because the next minimum is rarely obtained for shin and is generally obtained for the shoulder. However concerning the upper lip, the problem is to determine a restricted area.

## 3<sup>rd</sup> Step:

Finally we now search as precisely as possible the upper lip limit. Considering some different morphologies, sometimes this point is more or less well detected. It depends on our searching interval. Sometimes the two kinds of configurations below can be encountered:



In these two configurations with the same interval, only the first case provides good results. The research interval should be more restricted to obtain good result in most of the cases. Actually the new interval is defined thanks to the distance between the red and orange pixels. Indeed generally the lower lip is at least as wide as the upper one.



### III- Results



Thanks to this profile algorithm we obtained good results in determining the needed points however the main disturbing factor is again the important saturation. This algorithm part

was tested on fewer photos than the frontal one. The pictures come from the University of Stirling Face Data Base.

To develop this algorithm two steps appeared clearly, the first one is targeting reference point to allow to search from coarse to fine the needed feature, the second step is more a generalization one, indeed all the human are different but present the same general morphology. A work of standardization of the facial characteristics was needed to obtain a good percentage of localization. This human aspect of my work was really interesting. It wasn't just a programming work; it was before everything an observation work.



## PART 3: Data and Protocol

## I- Data Normalization and Storage

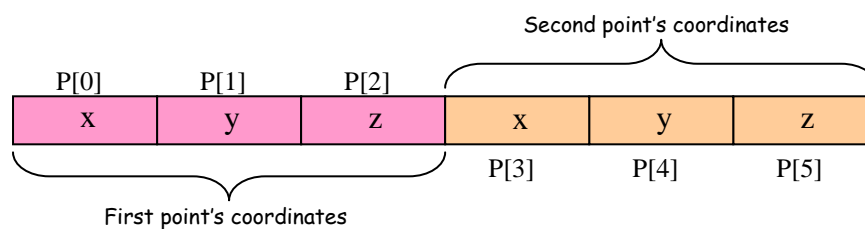
In the same time the other parts of this project were developed and should be merged in only one program. A commune interface has been designed permitting the communication between the different parts of the project.

The 3 parts:

- Facial feature localization.
- Adaptable 3D model.
- Facial animation of the model.

- **Data Storage**

The storage is done thanks to one vector not with a 3D matrix. For example, the two first point's coordinates are stored like it follows:



The coordinates of determined points are stored in one vector, it allows to use less arguments in each function of the localization program, only the vector is passed like argument to have access to all coordinates whenever it's needed.

- **Data Normalization**

We treated two pictures, one from front view and the other from profile. The distance between the camera and the subject is logically the same but a normalization step is needed, to use the facial feature coordinates.

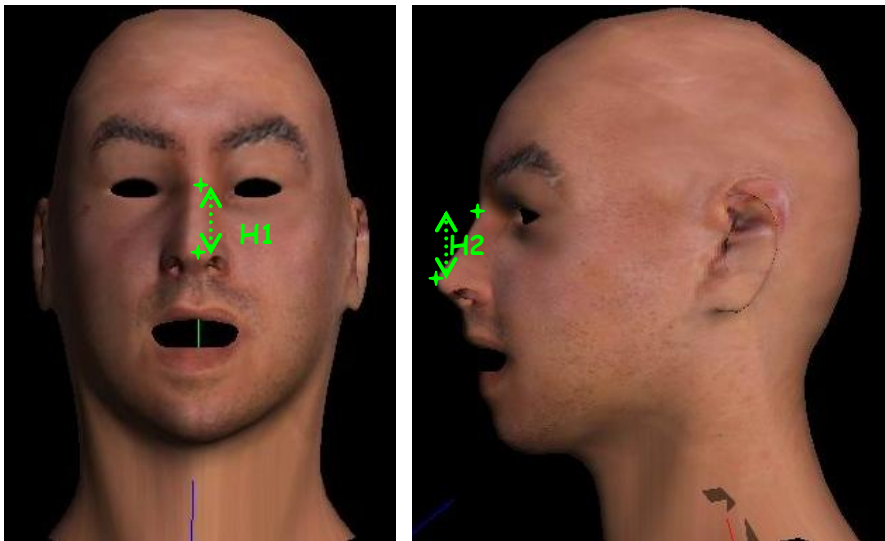
This normalization step will permit to obtain the three coordinates for each point. So we need two reference points. The best localized points are the end and the top of the nose. So we decided to consider it like the reference distance.

Indeed these two points are most of the time precisely determined on the two pictures. Before transmitting any coordinates, a coefficient must be calculated.

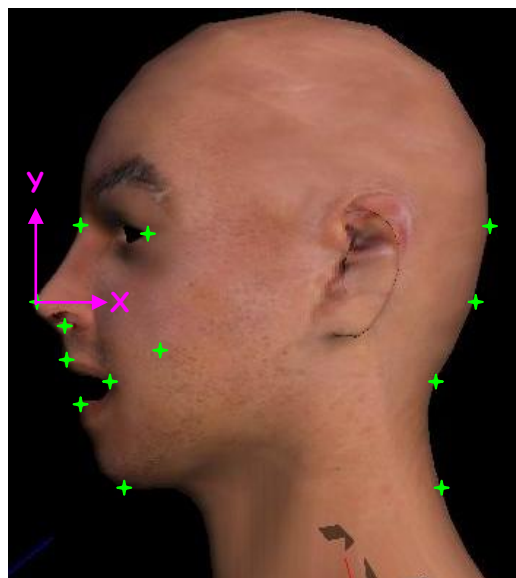
Coefficient:  $H1/H2$

Then we have to define the origin of the new referential for the 3D model, for example, we decided that the end of the nose has the coordinates (0; 0; 0) so all the coordinates of the needed points have to be changed to define the facial feature location in this new referential.

Moreover, we found the major part of the points coordinates on the front view so all these coordinates will be kept like that. However the considered values from the profile picture must be multiplied with the previously determined coefficient.



Reference distances for coefficient computation



New referential for Z-coordinates determination

## II- Pictures Shooting Protocol

To obtain optimal results on the facial feature detection, a protocol for the photo's shot is described below. This protocol describes how to manage the different conditions to obtain pictures providing good results during the Facial Features Localization.

- **List of the variables**
  - Distance between the digital camera and the subject
  - Center of the picture
  - Light conditions
  - Background (color)
  - Format of the pictures files
  
- **Frontal view**

### *Subject position:*

The subject must avoid inclining his head, and closing his mouth. A neutral expression is recommended.

### *Distance:*

The distance between the subject and the camera must be determined to obtain portrait pictures which present the whole head (hair and neck) of the subject in the proportion 75% of the whole picture.

### *Center:*

The center of the picture should be centered as well as possible on the facial symmetrical axe.

### *Light conditions:*

In function of the color of skin of the subject, the light conditions must provide some pictures with a minimum of saturation. For example, the nose area shouldn't be too white.

### *Background:*

The background color must be white to obtain a good contrast between the head and the background.

### *Format:*

The algorithm treats the PNG picture files.

- **Side view**

*Subject position:*

The subject has to stand as perpendicularly as possible to the camera direction looking to the right side. A neutral expression with closed mouth is recommended.

*Distance:*

The distance between the subject and the camera must be the same as to obtain portrait pictures; the required proportion is still 75% of the whole picture.

*Center:*

The center of the picture should be centered as well as possible on the facial symmetrical axe.

*Light conditions:*

In function of the color of skin of the subject, the light conditions must provide some pictures with a minimum of saturation. For example, the white background shouldn't influence on the facial edges too strongly.

*Background:*

The background color must be white to obtain a good contrast between the head and the background.

*Format:*

The algorithm treats the PNG picture files.  
(cf. **ANNEX 5** for more details about PNG file)

## Conclusion

My six months end of studies internship in the Faculty of Electrical Engineering and Information Technology of Bratislava is completed.

The worked presented previously is like the root of the Adaptable Speech Synthesis Project of KTL Department. Thanks to the algorithm of facial features localization developed, the adaptable 3D head modeling did a new step to the final multimedia product. What was expected from my work in this project was realized holding the deadlines. Moreover this training course brought me a lot in different ways.

First of all it provided me the opportunity to develop of my skills in the image processing field more deeply, and learn more about the C++ language. I found out how is difficult to teach an artificial intelligence to solve simple human problem.

Moreover this engineer internship had a particular human side, maybe due to the fact that it occurred in a Telecommunication Department of a University. I will keep contact with the other team project members and the supervisor Doc Ing Gregor ROZINAJ to follow the life of this multimedia project and for the reuse of my program.

Finally another detail and not the smallest enriched me a lot, the contact with a new culture opened my mind in different ways, first of all I realized how high is the life level in France. So due to the many friendships built in and out of my work, I had the opportunity to learn a new language, and improved my English daily. I took this chance that could open some new promising horizons. Indeed the Slovak and Check Republic are young countries but after ten years of preparation they are going to enter the European Community. Maybe one day it will allow me to apply for job more easily in these countries.

Thank you very much to Dr Safwan EL ASSAD for this internship opportunity and to Doc Ing Gregor ROZINAJ for supervising the project.



## References

- [1] MPEG-4 FACIAL ANIMATION The Standard, Implementation and Applications  
Edited by: Igor s. Pandzic Robert Forchheimer
- [2] Le langage C++ Le tout en poche Campus Press Stephane Dupin
- [3] <http://face.ee.nus.edu.sg/NewFace/NetResource/NetResources.htm>
- [4] <http://pics.psych.stir.ac.uk/cgi-bin/PICS/New/pics.cgi>
- [5] <http://xavier.chassagneux.free.fr/tipe/tipe.htm>
- [6] [http://ise0.stanford.edu/class/ee368a\\_proj00/project16/skincolor.html](http://ise0.stanford.edu/class/ee368a_proj00/project16/skincolor.html)
- [7] <http://www.esil.univ-mrs.fr/~morph3d/rapport.html>
- [8] <http://cswww.essex.ac.uk/mv/021/node2.html#SECTION00020000000000000000>
- [9] <http://astronomy.swin.edu.au/~pbourke/analysis/correlate/>
- [10] <http://www.bib.ulb.ac.be/coursmath/normale.htm>
- [11] <http://web.media.mit.edu/~jebara/uthesis/node40.html>
- [12] <http://www1.cs.columbia.edu/~jebara/htmlpapers/UTHEISIS/>
- [13] <http://raphaello.univ-fcomte.fr/IG/TraitementImages/>



# ANNEXES