3D Facial Modeling
Term Project

EE133: Digital Image Processing
Spring 2017

Department of Electrical and Computer Engineering

Tufts University

Instructor: Prof. Eric Miller

Date submitted: May 10, 2017

Jiacheng Qu

Xiaozheng Guo

Introduction

As an aspect of image processing, face recognition has been rising in importance due to the development of computer vision. Based on the preliminary research done by the team, this topic includes image cropping, filtering, reconstruction feature finding, registration, etc. It consists of lots of pre-existing knowledge which could be acquired both in and out of class. Thus, the team agreed that this might be an adequate topic to work on. The team decided to accomplish a project using a portrait photo as input, detect the face and crop[1] facial proportion then resize it. On the other hand, label the facial features of a referenced depth map by setting control points at critical positions[4]. Map those standard feature points onto the inputted face image. Eventually, generate a facial model that could output profile image from designated angles. Such a process involves dilation-erosion, Gaussian filtering, and denoising.

Objective

The goal of this project is to discover and analysis existing facial modeling methods. Also, other than implementations of the modeling method, the team found that testing and verification are necessary for not only the project completeness but also the functional completeness. However, the testing and verification of the result could be challenging and time-consuming.

Individual Contributions

As the team has two members, we decided to divide the entire term project into two parts: facial recognition and facial mapping. Jiacheng is in charge of the first part while Xiaozheng is responsible for the second one. To ensure the integrity of this term project, the facial recognition must make all the detected face images maintain the same aspect ratio and aligned properly, so it can fit in to the position map of the three feature spots. In other words, the coordination of the eyes and mouth must be the same. Also, the project report has been split into two parts, and each of the team member is assigned to one of them.

Approach

1. Face recognition based on Gaussian skin color detection

To detect the face, the method skin-color modeling is used here for finding the distribution of skin color. For the simplicity of the calculation, the YCbCr color space is used here.

In the YCbCr color space, Y is the intensity value which represents brightness and can be removed in the following of this paper. This is because the Chromatic colors (Cr Cg) are related to skin color[8].

$$r = \frac{R}{(R+G+B)},$$
$$g = \frac{G}{(R+G+B)}.$$

The reason why the intensity could be neglected is that, in the RGB space, two pixels with the same color but different brightness can be expressed as the following:

$$\frac{r_1}{r_2} = \frac{g_1}{g_2} = \frac{b_1}{b_2},$$

Here the team found two primary methods in doing this. The first one is applying thresholds on Chromatic colors Cb Cr. According to the paper written by Shaik et al. [6] skin color Cb Cr are falling in the range of [100, 150] and [150, 200].

The second approach is called Maximum Likelihood Adaptation. Based on the paper written by Hasan et al. [1], skin-colors can be modeled by multivariate Gaussian distribution. or N((**μ**, **Σ**) where expected value: **μ** = (Cr, Cg)

$$\Sigma = \begin{bmatrix} \sigma_{rr} & \sigma_{rg} \\ \sigma_{gr} & \sigma_{gg} \end{bmatrix}.$$

Therefore, the likelihood of the selected pixel value as skin-color, p can be represented using the following formula:

$$p(x;\ \mu, \Sigma) = \frac{1}{(2\mu)^{d/2}|\Sigma|^{1/2}} exp(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu))$$

Although the range varies in different papers, the proportion of image other than skin can be successfully filtered out. In either approach, always set everything above the range to 0.

Following is the result of the thresholding method 1:

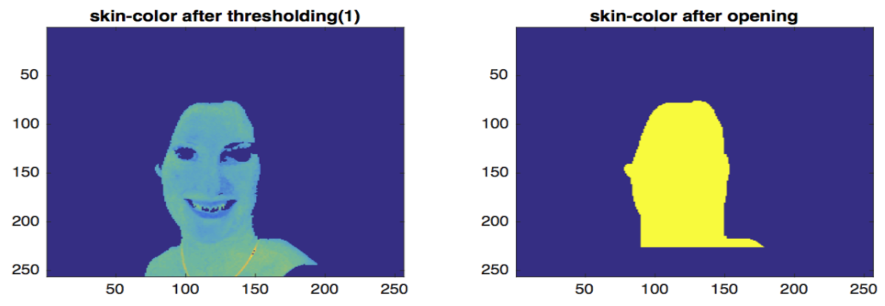
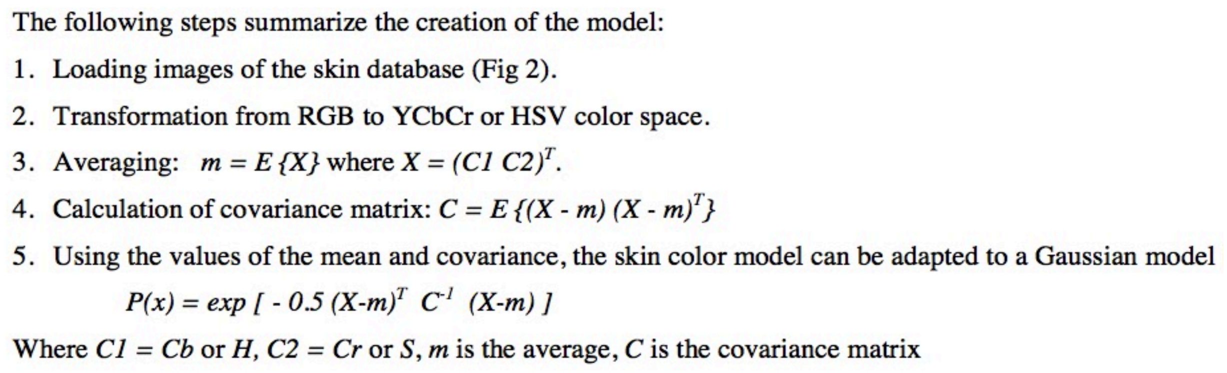


Figure 1 result of thresholding method 1

As the formula is shown above, the second one implements a Gaussian detection algorithm on the yCrCb version of the target image. Here, the standard deviation does not matter, and therefore, have been neglected

Following is a more precise expression, starting from the second step [5]:



The following steps summarize the creation of the model:

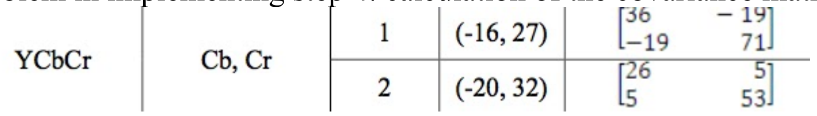
1. Loading images of the skin database (Fig 2).
2. Transformation from RGB to YCbCr or HSV color space.
3. Averaging: $m = E\{X\}$ where $X = (C1\ C2)^T$.
4. Calculation of covariance matrix: $C = E\{(X - m)(X - m)^T\}$
5. Using the values of the mean and covariance, the skin color model can be adapted to a Gaussian model

$$P(x) = \exp[-0.5 (X-m)^T C^{-1} (X-m)]$$

Where $C1 = Cb$ or H, $C2 = Cr$ or S, m is the average, C is the covariance matrix

Similar to the first method, mean value m is chosen from the range of Cb and Cr.

In step 5, p is the probability of the selected pixel value is human skin. However, the team have had some problem in implementing step 4: calculation of the covariance matrix



YCbCr	Cb, Cr	1	(-16, 27)	$\begin{bmatrix} 36 & -19 \\ -19 & 71 \end{bmatrix}$
		2	(-20, 32)	$\begin{bmatrix} 26 & 5 \\ 5 & 53 \end{bmatrix}$

The team tried a couple of covariance sets, but none of them gives the correct result in step 5. From our understanding, the covariance used in the formula above should be a set of the matrix depends on the mean value of Cb and Cr of human skin, which supposed to be in proportional to the actual covariance of skin-color Cb and Cr.

Other papers using Gaussian detection have been analyzed thoroughly, the statistical results (mean and covariance matrix) are similar to those mentioned above and therefore have been used in the following part of this project.

$$m = (156.5599, 117.4361)^T$$

$$C = \begin{bmatrix} 299.4574 & 12.1430 \\ 12.1430 & 160.1301 \end{bmatrix}$$

$$m = \left(\overline{C_r}, \overline{C_b}\right)^T$$

Note, in some paper, the Cb-Cr have been swapped, but the concepts are the same [6].

The threshold has been determined based on the mean and standard deviation of the likelihood distribution P of the skin-color.

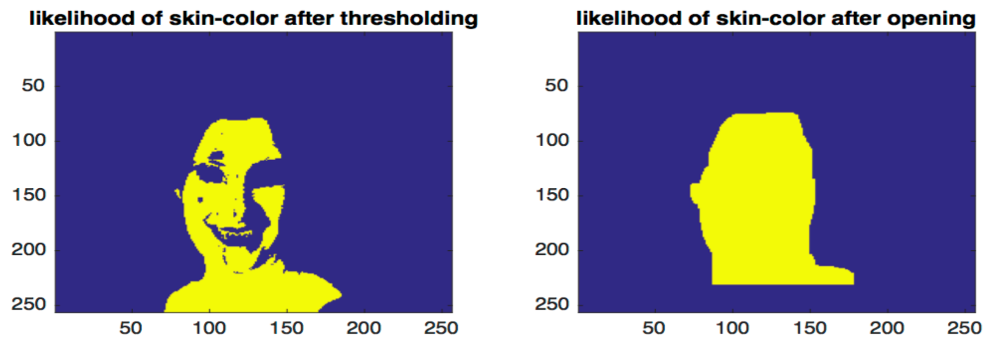


Figure 2 Result of thresholding method 2

Since the output needs to be a clear cropped Passport photo. For this reason, the opening method introduced in the second homework is used here to sew up the filtered face.

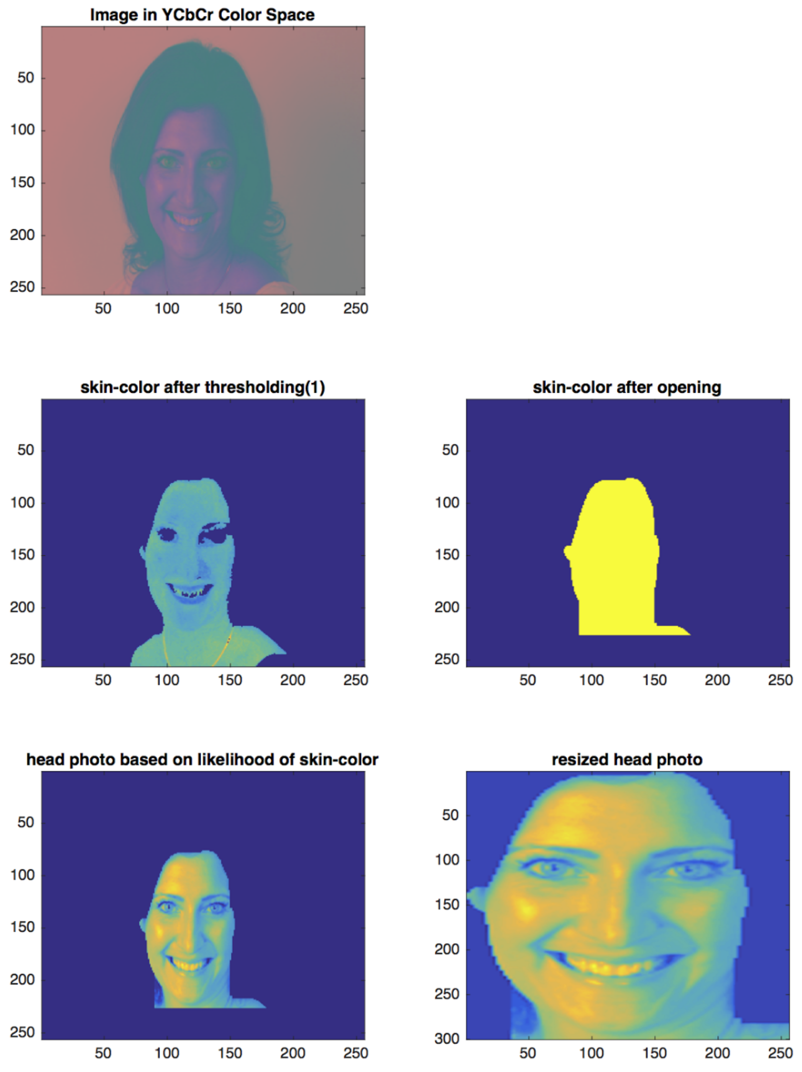


Figure 3 Result of thresholding method 1

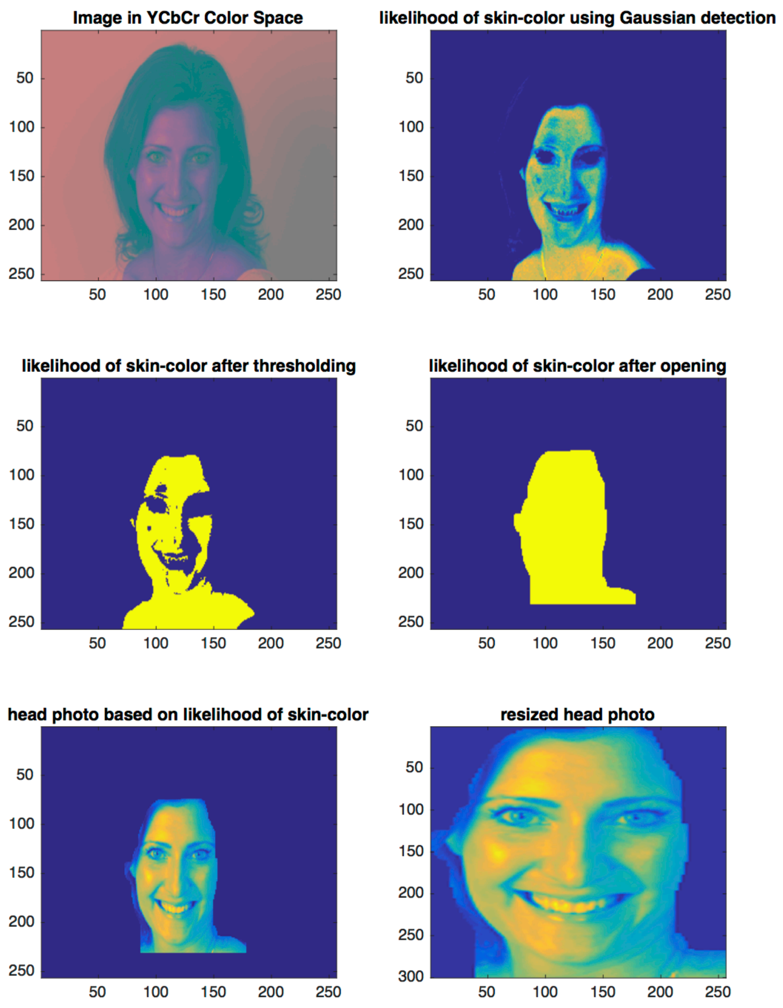


Figure 4 Result of thresholding method 2

2. Facial feature locating and depth mapping

As we mentioned above, after doing some researches on existing 3D face molding techniques, the team chose to reach their goal in a more feasible way. Rather than bother with unknown albedo map image[7] and an unknown depth map of the same image[2, 3], the team decided to use an approach similar to Flexible appearance models(FAM)[4]. However, since time is limited and the team members have heavy workloads at the end of this semester, after further consideration, the team decided only to implement a semi-automatic algorithm.

Our approach identifies each face by setting a group of control points[4] at critical positions, and we assume the depth[3] of two same points on different faces are equal. Therefore, we can set control points on a reference depth face image first, store depth value for each point. After that, we can set control points for each input image. After we get the position and depth of each control points, we can use these points to do a rough 3D face reconstruction. These are described in more detail below.

First, we need to set control points for a referenced depth image and use these points as our referenced control points. In our approach, we decided to set 90 points at every critical position, as shown in Figure 5. After that, we store a gray value of depth map image for each control point. These values are the referenced depth values of corresponding control points.

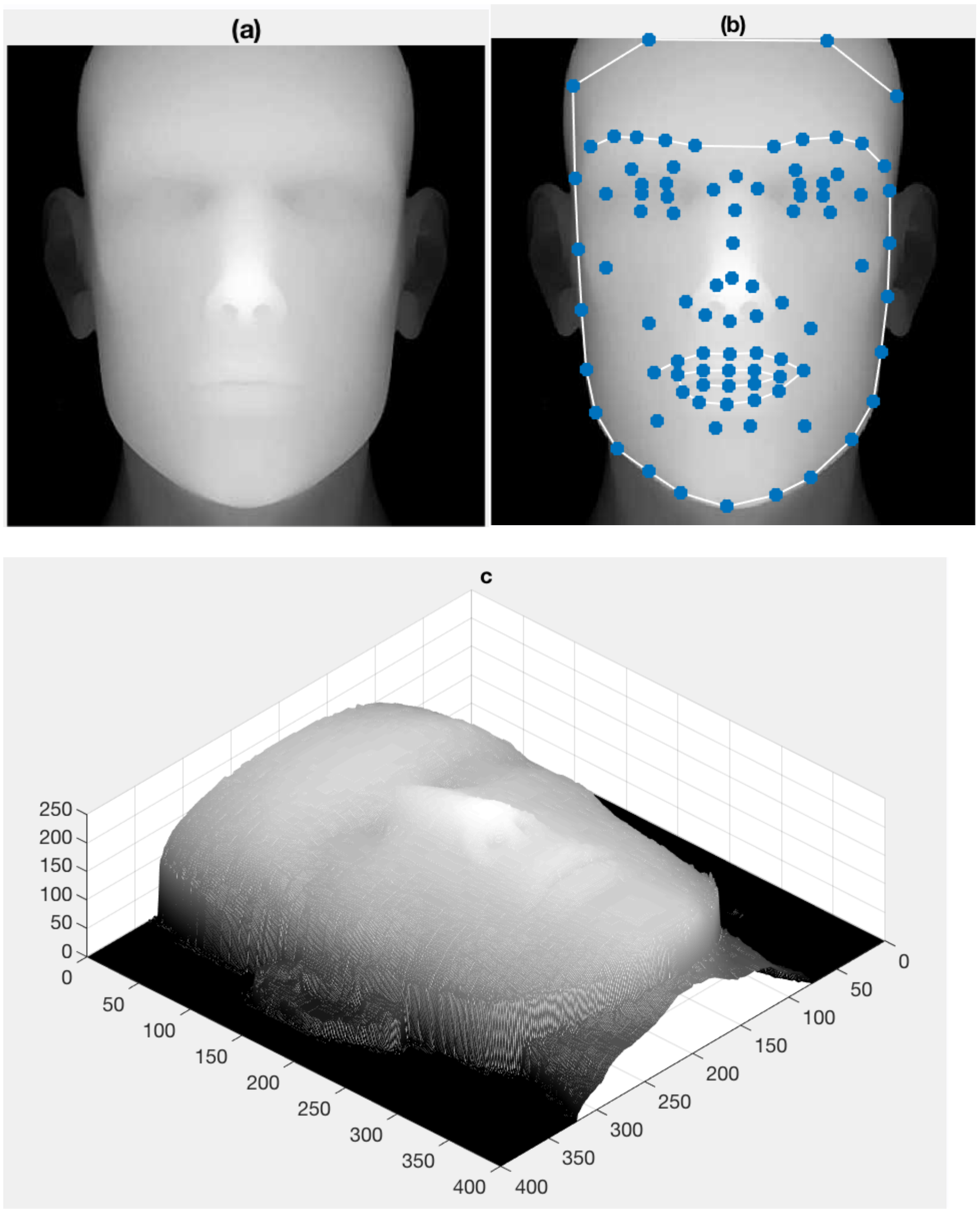


Figure 5(a) is a depth map image, we found it on Google. The value of each pixel is the depth of this point of the face. (b) is depth map image after we set up 90 control points, this is also the default control points set. (c) is the 3D plot of depth map image

After getting referenced depth data and control points, we can map it to any input image. We have an automatically cropped facial area of the input image using Gaussian detection which is described above. Because input image might have a different size from referenced image, to map referenced control points to the input image, we need to select a set of points on critical positions in the input image, calculate a linear transform matrix with corresponding points in referenced control points set. In our approach, we chose to select 3 points on eyes and mouth. In our implementation, we used Matlab ginput() function to select these points. After we get the linear transform matrix, we use this matrix to calculate the coordinates of every control points, as shown in figure 6.

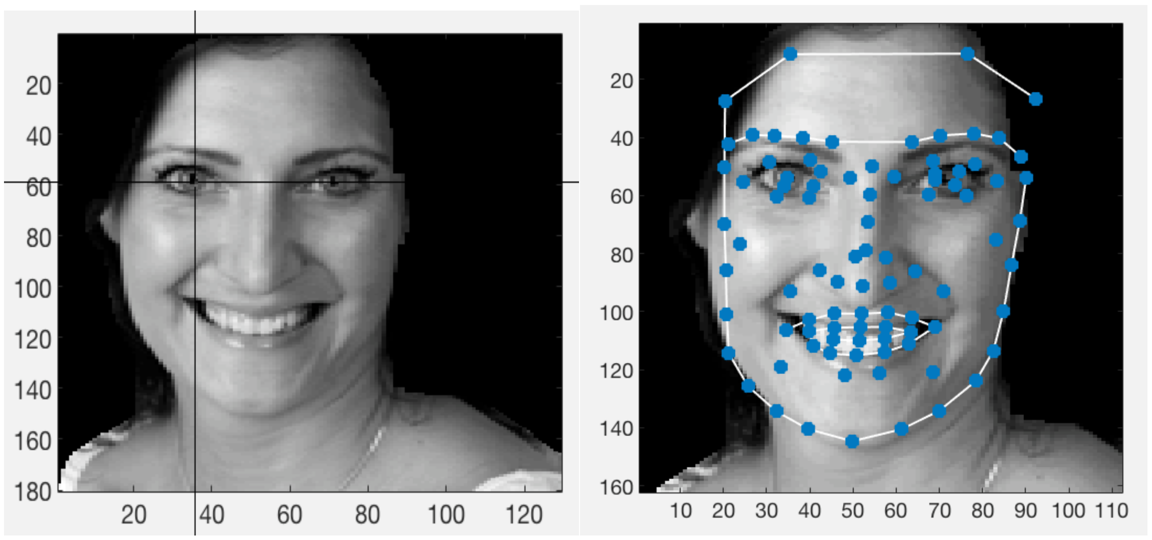


Figure 6 Left is the first step that we need to select eyes and mouth using our finger. The right image is the automatically map of control points.

We can see that the positions of these automatically generated control points are very rough. Therefore we need to adjust them by hand. In our approach, we used Matlab ginput function to adjust the coordinate of each point. When we want to change the position of a point, we click that point first, after that, the program will calculate the nearest control point from the position we clicked, then we can re-click to set a new coordinate of that control point.

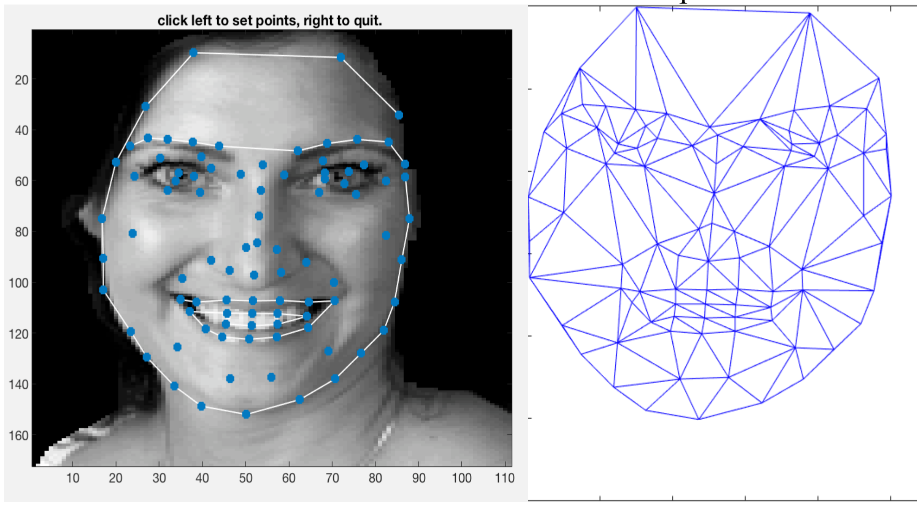
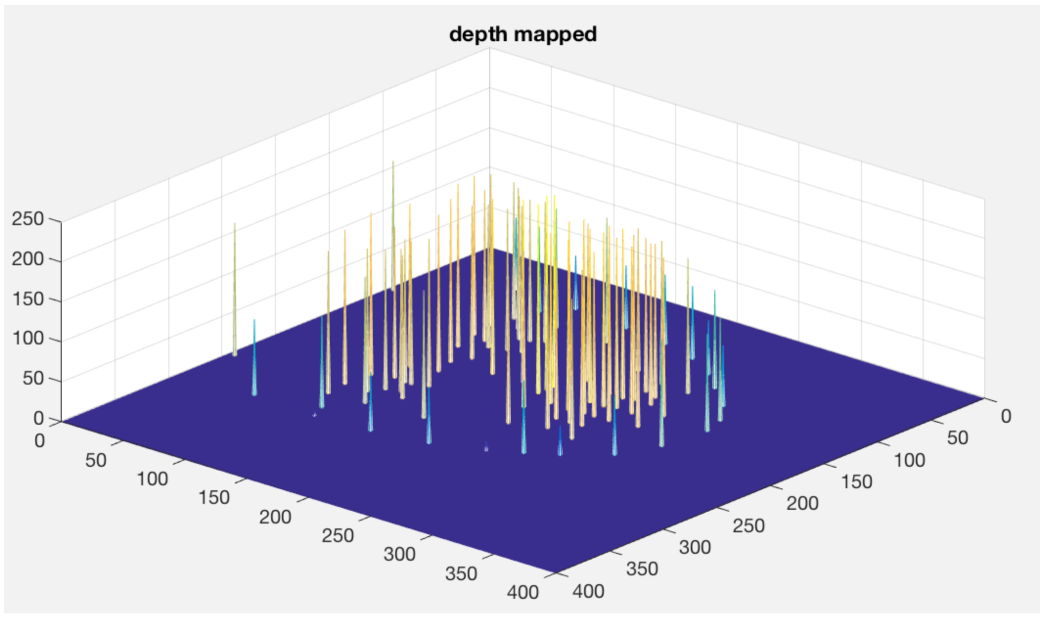


Figure 7left: control points set properly. Right: triplot of control points using Matlab.

After we set each control point properly, as shown in Figure 8, we are now ready for map referenced depth data to these adjusted control points.



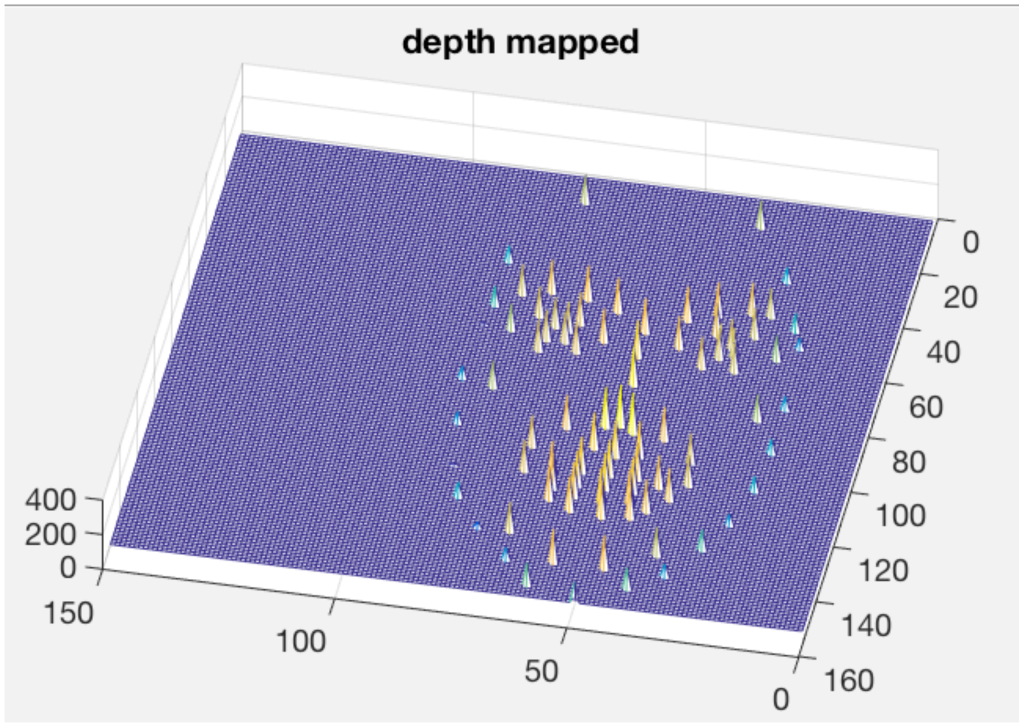


Figure 8 Control points with mapped depth.

Finally, we can do 3D reconstruction using these control points and mapped depth. First, we create a triangulation representation of control points using Matlab built-in function Delaunay() as shown in Figure 7. Then, we can do a roughly 3D reconstruct using Matlab function trisurf().

Experimental Results

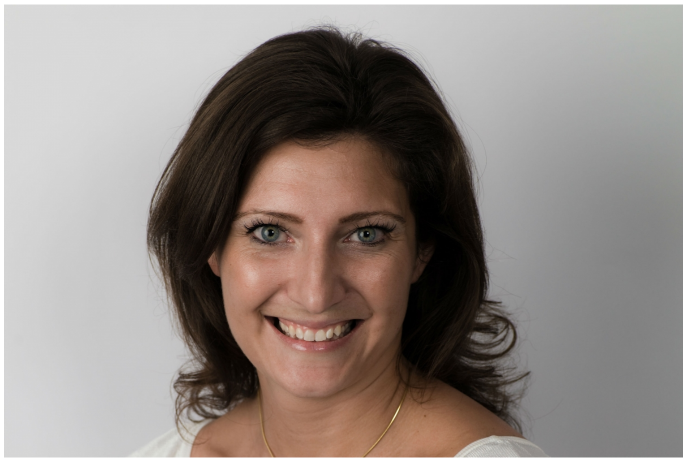


Figure 9 Input image

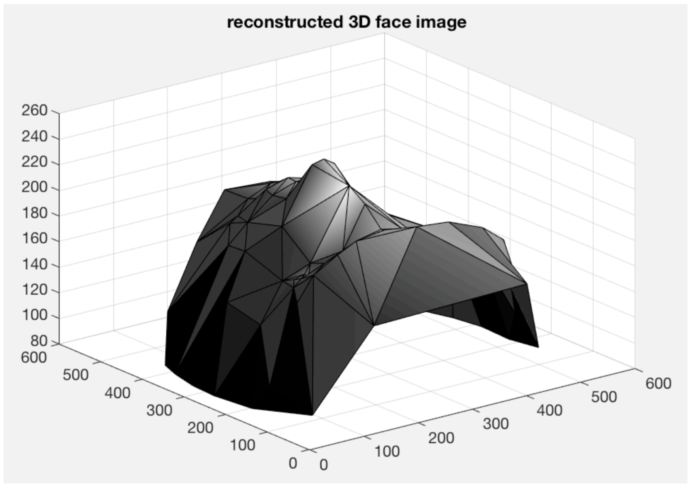


Figure 10 3D model of the target image

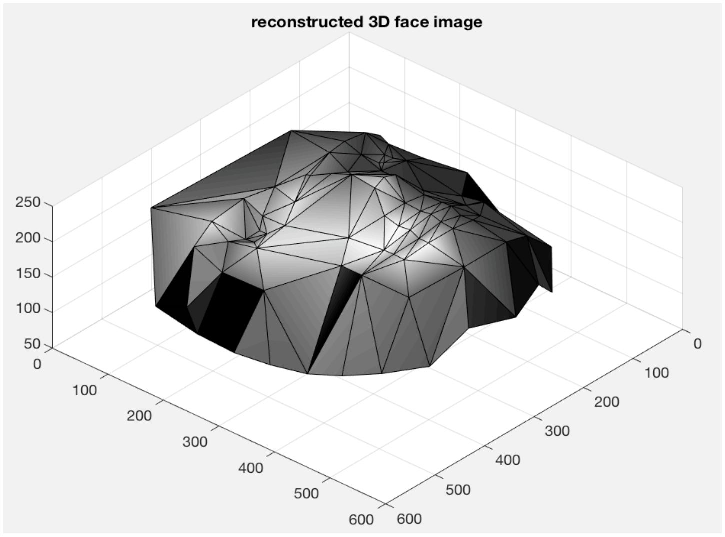


Figure 11 3D model of the target image from a different viewing angle

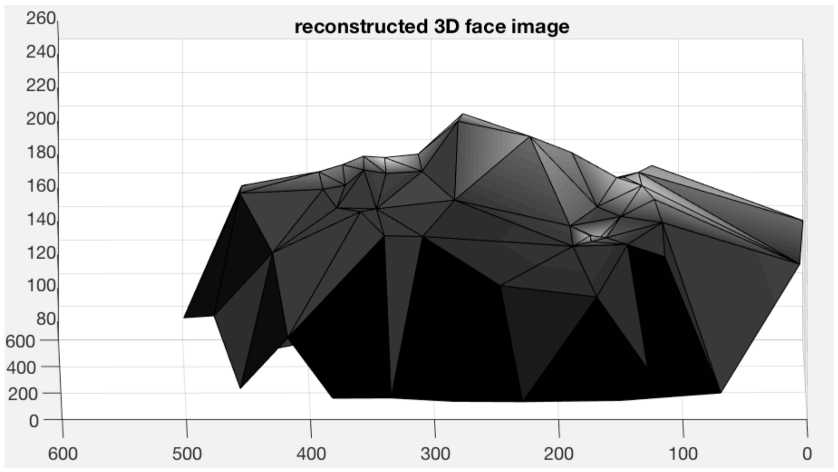


Figure 12 3D model of the target image from the third viewing angle

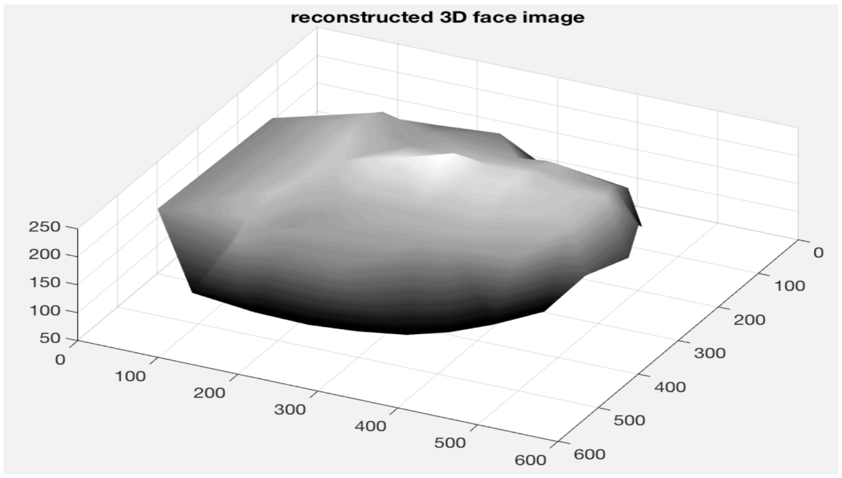


Figure 13 3D model of the target image without lighting condition

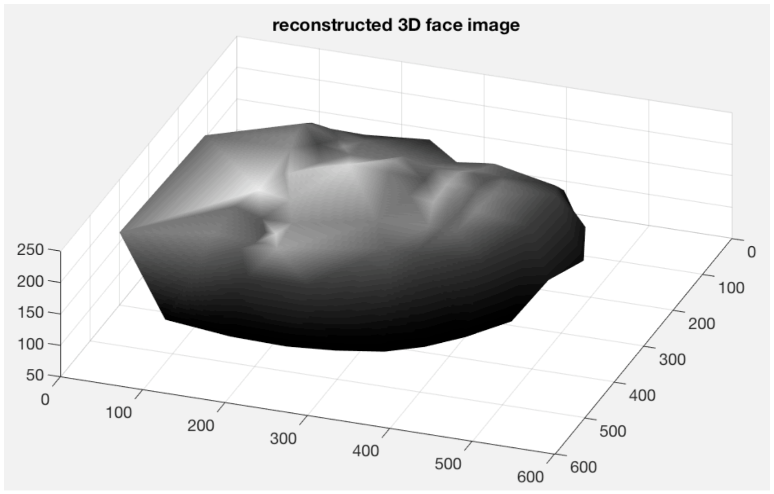


Figure 14 3D model of the target image with lighting condition

Conclusion

In conclusion, by using our approach, we can create a rough 3D reconstructed version of the input image. After optimizing our Gaussian detect algorithm, we do not need to set eyes and mouth positions every time. However, we still need to adjust each control points after the program generating them automatically. From the result, we know that our output 3D image is still very rough. To optimize our result, we need to set more control points. Furthermore, it would be meaningful for us to implement an algorithm that can set these control points automatically; however, given the limited time and knowledge, the team has had some trouble with the notion of completeness. If we were to pursue this topic, we would work more on this topic to make the algorithm can automatically detect feature area on faces and set control points properly.

Furthermore, we will use more control points such that we can get a more precise result. Also, we will do texture mapping on the output 3D image, such that the result will look more realistic. The mixed results of this project suggest that facial detection and 3D reconstruction are very promising research domain, they are useful and powerful.

Bibliography

1. Hasan, M. M., & Mishra, P. K. (2012). Superior Skin Color Model using Multiple of Gaussian Mixture Model. *British Journal of Science*, *6*(1), 1-14.
2. Kemelmacher, I., & Basri, R. (2006, May). Molding face shapes by example. In *European Conference on Computer Vision* (pp. 277-288). Springer Berlin Heidelberg.
3. Kemelmacher-Shlizerman, I., & Basri, R. (2011). 3D face reconstruction from a single image using a single reference face shape. *IEEE transactions on pattern analysis and machine intelligence*, *33*(2), 394-405.
4. Lanitis, A., Taylor, C. J., & Cootes, T. F. (1995). Automatic face identification system using flexible appearance models. *Image and vision computing*, *13*(5), 393-401.
5. Mohammed, K., Ennehar, B. C., & Yamina, T. (2012). Skin detection using gaussian mixture models in YCbCr and HSV color space. *Global Journal on Technology*, *1*.
6. Shaik, K. B., Ganesan, P., Kalist, V., Sathish, B. S., & Jenitha, J. M. M. (2015). Comparative study of skin color detection and segmentation in HSV and YCbCr color space. *Procedia Computer Science*, *57*, 41-48.
7. Weyrich, T., Matusik, W., Pfister, H., Lee, J., Ngan, A., Jensen, H. W., & Gross, M. (2005). A measurement-based skin reflectance model for face rendering and editing. *None TR*, *71*(4).
8. Yang, J., Lu, W., & Waibel, A. (1998, January). Skin-color modeling and adaptation. In *Asian Conference on Computer Vision* (pp. 687-694). Springer Berlin Heidelberg.