Virtual Makeover Using MATLAB

Akash I. Mecwan
akash.mecwan@nirmauni.ac.in

Vijay G. Savani
vijay.savani@nirmauni.ac.in
Electronics and Communication Department,
Institute of Technology, Nirma University

Abstract

Softwares' like Adobe Photoshop are readily available today which enables us to make desired modifications in any image. But this is limited to only those who have an expertise in using that software. For others, it becomes too difficult. So, selecting a specific domain, the one dealing with images of human faces, we present an algorithm using which anyone can modify various facial features of human face like eyes, nose, lips, ears, hairstyle etc. Moreover, in order to make this procedure user friendly, we implement the algorithm using a GUI in Matlab. By making a GUI, we intend make the task of image morphing simpler and easy to implement.

Keywords: *Adobe Photoshop, Matlab, Virtual Makeover, Abdominoplasty.*

1. Introduction

A makeover is a term applied to changing one's appearance, usually through cosmetics. Makeovers can range from something as simple as a new haircut, to the use of cosmetic surgery, to the extreme of the implantation of dental veneers, eye-color-changing contact lenses, and the use of appearance-altering gastric bypass surgeries, providing massive, permanent fat loss in obese persons, and the associated plastic surgeries such as ***Abdominoplasty***. Computer software and online tools can also be used for performing what are known as Virtual Makeovers. Using a photograph of a human face, software can apply cosmetics, hairstyles, and various eyewears such as contact lenses and sunglasses in order to allow users to visualize different looks without physically trying them on. The software presented here is a little different. It performs the task of changing various facial features of human face as well as changing the skin complexion.

2. Problem Statement I

The first task to accomplish the desired goal is to detect the skin regions from the input image and thereby extract the face portion from the image. This is required to be done because whatever modifications are supposed to be made to the face cannot be made until only the face can be stored in the form of an image. For example, if the input image shows complete

photograph of a person, the software will first detect the skin regions from the complete image and then it will simply extract the face region from the detected skin region and store it in an image.

2.1. Practical Approach I

The Canny edge detection operator was developed by John F. Canny in 1986 and uses a multi-stage algorithm to detect a wide range of edges in images. Most importantly, Canny also produced a computational theory of edge detection explaining why the technique works. Canny's aim was to discover the optimal edge detection algorithm. In this situation, an "optimal" edge detector means:

- ***Good Detection:*** The algorithm should mark as many real edges in the image as possible.
- ***Good localization:*** Edges marked should be as close as possible to the edge in the real image.
- ***Minimal response:*** A given edge in the image should only be marked once and where possible, image noise should not create false edges.

To satisfy these requirements Canny used the calculus of variations - a technique which finds the function which optimizes a given functional. The function in Canny's detector is described by the sum of four exponential terms, but can be approximated by the first derivative of a Gaussian. In Matlab, there is a function called EDGE (canny) using which edges of a given image can be directly detected , and the function uses Canny's algorithm to detect the edges. After determining the edges of the image, the next task is to extract the edges of only face portion from the image and then finding the extremities of the face. The next task would be to find the geometric ratio. Edges of the face can be used to find the extreme co-ordinates of the face to a good approximation but it does not produce accurate results. So, leaving aside the above mentioned approach, the algorithm follows the approach which is mentioned below.

2.2 Practical Approach Followed

The algorithm presented here follows a rather practical and relatively simpler approach to accomplish the task. In this approach, skin portion is first isolated by using a threshold range. This threshold range is a value corresponding to various skin complexions. Using a for loop, the algorithm checks the value of each and every pixel, and those pixels for which the RGB value falls within the threshold range are converted to 1 i.e. white while the remaining are converted to 0 i.e. black. Thus, a binary image is obtained in which the white portion displays the skin region.

In order to make the produced results more accurate, the algorithm uses a function 'imdilate' available in Matlab. This is required in order to smoothen the edges of the skin region. After this, using 'imagecrop' function, the face portion of the image is extracted but it may still have some undesirable skin portion then just face due to limitation of manual cropping. In order

to remove this undesirable portion the algorithm uses 'imboundary' function and a for loop can be used to find out the coordinates of the extremities of the face.

3. Problem Statement II

To make the task relatively simpler, a template based approach is followed. In this, it is required to make templates of a human face as well as of various facial features like eyes, hairs, nose, lips etc. Further, it is also required to develop an algorithm which would place these facial features at their respective positions in the template. And finally to develop a GUI which would do all these processing in the simplest possible manner.

3.1. Practical Approach I

As a practical solution to the above mentioned problem, first various templates of human face, eyes, nose, lips etc. are made. Some of these templates are shown below:

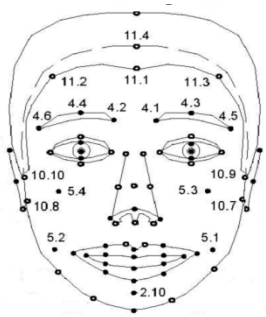

Fig. 1 : Face Template

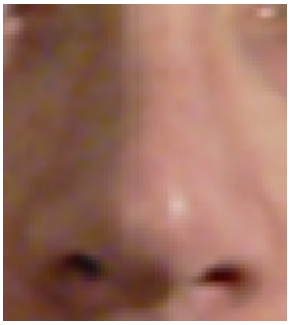

Fig.2 : Nose

In template-based approach, an input face is compared with a predefined standard face template, which can capture the whole face or regions corresponding to the location of facial features, such as face contour, eye, nose, mouth etc. First the coordinates of centre of various facial features like eyes, nose, lips etc. are found from the template shown above. These coordinates are shown below:

Right eye	(266,290)
Left eye	(266,128)
Nose	(330,215)
Lips	(445,215)

Now, in order to determine the centre coordinates of the eye template, it is first divided into three horizontal blocks and two vertical blocks. Then, intersection horizontal line of the third horizontal block and the vertical line is found out and its coordinates are obtained as shown in the figure 3.

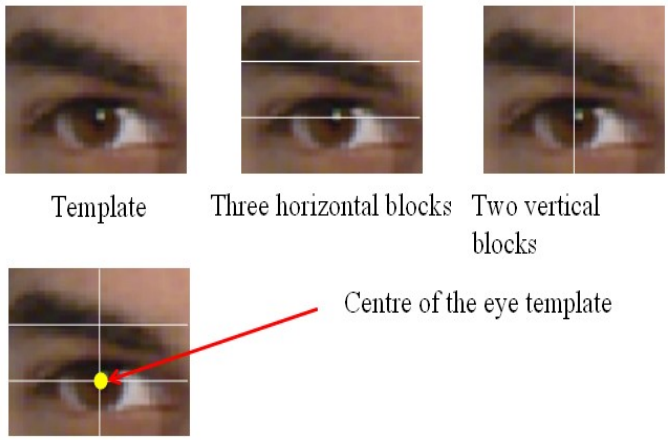

Fig. 3 : Finding the center of the eye template

Now the center of eye as obtained above is placed on the center of eye of the template, thus replacing the eye in the face template by the eye template. In the same way, center of the nose is found by dividing it into two vertical and two horizontal blocks and in a way similar to that of eye, the nose is placed at its relevant position on the face template. Similarly, center of the lips is obtained by dividing it into two horizontal and two vertical blocks and then, it is placed at its respective position on the face template. The final image is shown in figure 4.

Fig. 4 : Facial features placed on the face template

3.2 Adjusting Skin Complexion

Having placed the templates of various facial features at their relevant faces on the face template, the next task of the algorithm is to adjust the skin complexion of the entire face so that the newly obtained face looks like a natural human face. As the templates of various facial features may vary in their skin complexion, it is extremely necessary to set the skin complexion of the entire face.

If one considers a natural human face, one finds that some areas of the face are dark, some bright, some other still brighter and some still darker. This is not because the skin color is different in different areas of the face, but because the amount of light falling on different areas is different. In technical terms, the variance in skin complexion is due the variation in luminance at different areas of the face. Thus, in order to adjust the skin complexion, it is required to adjust the luminance level of all facial features so that it looks like a natural face.

Fig. 5 : Reference Image

Consider a reference image as shown in fig. 5, a skin portion near the eye is selected and its value (RGB) is used to alter the RGB values of other pixels by subtracting this reference value from each and every pixel.

Fig. 6 Image Showing the Deviation

In this image, there is still variation in the skin region of the face. To make skin region uniform, we again subtract the pixel values from a reference value so that the resultant image obtained has uniform intensity in skin region as shown in figure 6. Now the skin complexion of the reference image is set to desired tone. First of all, the RGB image in Figure 5 is converted to HSV image where the three planes indicate Hue (H), Saturation (S) and Intensity (V). The HSV color space image obtained is shown in figure 6.

Fig. 7: HSV Pixel Value

Secondly, the algorithm takes the same reference pixel value and subtracts each and every pixel value of Figure 7 from this reference value thus finding the deviation of each pixel in terms of hue, saturation and intensity with reference to the desired reference pixel value. Still the image doesn't have uniform skin complexion, so again it is converted to HSV as shown in fig 8.

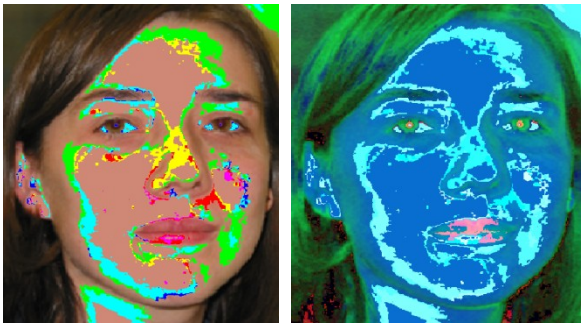

Fig. 8: Re-conversion for Uniform Skin complexion

Further the desired image with desired skin complexion is constructed by comparing the figure 8 with the image in figure 6 and the desired image is shown in figure 9.

Fig. 9 Image with Complexion Variation

The algorithms developed above are merged into a single algorithm and the skin complexion is set.

Fig. 10 (a) Reference image

Fig. 10(b) Image after changing the complexion

Consider figure 10(a) as the reference image. A reference pixel value is selected, and this value is subtracted from all the other pixels of figure 4. Also, this same value is subtracted from all the pixel values of figure 10(a). Now, the deviation value of each and every pixel from the reference value is added to the selected complexion value which is specified by the user. Thus, by applying the algorithm on Figure 4, we obtain the final output image with desired skin complexion as shown in figure 10(b).

The algorithm presented here provides an innovative and simpler approach to virtual makeover by simply using logical manipulation of input image.

5. Developing a GUI

In order to make the application user friendly and simpler, a Graphics User Interface (GUI) would be developed which would enable the user to select various templates of facial features like eyes, nose, lips etc. just by a click. Thus, the user will have to do only selection of desired facial features while the other processing will be done in the background by the software. To develop a Graphics User Interface (GUI), the Guide feature available in Matlab is used. By using objects such as push buttons, axis, text box, dialogue box etc., a user friendly GUI is created which would guide the user to use the software properly as well as give a message each time the user does some mistake.

6. Limitations

As no standard algorithm is available which would enable us to extract only the face portion from the detected skin region, we don't have any other option but to do run time cropping i.e. cropping the face portion manually using the Imcrop function. As the various templates like eyes, lips, nose etc. are of different resolution (size), it is too difficult to maintain a uniformity in the image size of all templates. Thus, we have to make a little compromise, which leads to negligible error in the position of that template in the face template. Our algorithm cannot be applied when the image is taken at any angle other than front. It is difficult to find the centre of the face as the size of faces varies from person to person.

7. Future Plans

Our future prospects include interfacing of a webcam with the software so that it can fetch images in real time. Moreover, we would like to implement the geometric method in the image. Finally we would merge the two approaches discussed above, i.e. we will try and make a software which would fetch image in real time, extract face from it and modify the facial features of that face.

References

[1] Standardization of Facial Biometrics by Farzin Deravi, University of Kent.
[2] Template-based Eye and Mouth Detection for 3D Video Conferencing by Jurgen Rurainsky and Peter Eisert, Fraunhofer Institute for Telecommunications, Germany.
[3] Tutorial on edge detection
[4] Tutorial on Canny Edge Detection Technique
[5] Investigation of New Techniques for face detection by Abdallah S. Abdallah, Virginia Polytechnic Institute & State University.
[6] Major Project Report on Face Recognition Techniques by Chetas, Milind and Shalin, Institute of Technology, Nirma University, Ahmedabad.