國立交通大學

多媒體工程研究所

碩 士 論 文

經由照片資料庫實現繪圖與卡通人像真實化之研究

Realizing Cartoon Faces and Portrait Painting through Photograph Collections

研 究 生：李映萱

指導教授：林奕成　教授

中 華 民 國　一 百 零 一　年 九 月

經由照片資料庫實現繪圖與卡通人像真實化之研究

Realizing Cartoon Faces and Portrait Painting through Photograph Collections

研 究 生：李映萱　　Student：Yin-Hsuan Lee

指導教授：林奕成　　Advisor：I-Chen Lin

國立交通大學

多媒體工程研究所

碩士論文

A Thesis

Submitted to Institute of Multimedia Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer Science

September 2012

Hsinchu, Taiwan, Republic of China

中華民國一百零一年九月

經由照片資料庫實現繪圖與卡通人像真實化之研究

學生：李映萱　　指導教授：林奕成

國立交通大學

多媒體工程研究所

摘要

要將不真實的卡通人像或是畫像真實化是一件很困難的事，對於人臉結構方面需要高度的想像力以及理解能力。人眼的高敏銳度更使得人臉真實化更為困難。根據我們的調查，目前對於從卡通人像半自動或是自動化去產生出真實化的樣貌的研究非常的少。如果我們使用目前有的方法直接 warping 或是 matching，會使很多較為精細的資訊找不到且亦可能會使重要的個人特徵遺失。在這篇論文當中，我們提出了一個 example-based 方法來合成真人化的卡通人像或是畫像。我們使用 graph-cut-based optimization 來拼出和輸入圖最相似且合適的 piece-up 臉。下一步，我們更提出了 chromatic gain compensation 和 multi-level blending 的方法來無縫的拼貼 patches。我們的實驗結果更顯示我們提出的 example-based method 是能夠對於輸入的卡通圖產生出真實且新穎的結果。

關鍵字：Example-based method, graph-cut-based method, 卡通人像真實化, chromatic gain compensation

Realizing Cartoon Faces and Portrait Painting through Photograph Collections

Student: Yin-Hsuan Lee Advisor: I-Chen Lin

Institute of Multimedia Engineering

National Chiao Tung University

ABSTRACT

Realizing unrealistic cartoon faces or paintings is a complicate task required high imagination and comprehension of face structures. The high sensitivity of human eyes makes face realizing a difficult problem. According to our survey, there are rare works on fully- or semi-automatic generation of a realistic face from a cartoon face. When directly applying face warping or matching methods, existing methods does not have much detailed information or may lose the personally characteristics. In this thesis, we propose an example-based method to synthesize the reality of cartoon faces and portrait paintings. We use graph-cut-based optimization to find the most similar and adequate piece-up faces according to the input image. Next, we further present chromatic gain compensation and multi-level blending to seamlessly stitch the patches. Our experiments show that the proposed example-based method is able to provide realistic and novel result of input faces.

Keywords: Example-based method, graph-cut-based method, realizing cartoon face, chromatic gain compensation

Acknowledgement

首先，我要感謝我的指導教授林奕成老師，在我遇到困難以及挫折時，都適時地幫助我解答疑惑並正確的指導我研究方向使我能夠順利完成這篇論文。在兩年之間的學習以及研究途中，當遇到了困難和瓶頸，老師總是給予耐心細心的指導、諄諄教誨以及莫大的支持。這兩年來從學校、實驗室和老師所學習到的各方面知識讓我成長了許多，克服了各種困難完成了屬於自己的一份作品也增強了對自我的自信與勇氣，我想這些都是我進入研究所最大的收穫。

感謝實驗室的同學星寒、世祐和蕙芸，還有博士班學長 Parker，不僅在課業研究上相互砥礪，一同學習成長，平時也因為你們的聊天陪伴在失去信心的時候得到勇氣。感謝學長姊在碩一的時候給予課業上的指導，感謝學弟妹在碩二的時候總是給予精神上的陪伴與鼓勵。

最後，最重要的是要特別感謝我的爸爸、媽媽和姊姊長期以來的支持與照顧，讓我能全心全意地專心在我的學業以及研究。每當在學習途中遇到了困難與挫折，家人總是給予我最大的支持以及鼓勵，也許家人無法直接在專業領域上給予協助，但總能從關切支持之中得到持續努力向前的動力。另外，還要感謝所有在學習途中在各個方面幫助過我的人。謹以此論文獻給所有給予我關心與協助的親友們。

Table of Contents

Tables

Figures

Chapter 1

Introduction

1.1 Motivation

For the past few years, multimedia technology has been rapidly developed and widely used in many different fields especially in interactive applications. Digital interactive applications are widely used in each region, such as mobile app, movie, cartoon, games and etc. In 2011, societies are attracted by the famous ancient Chinese painting “Along the River During the Ch’ing-ming Festival” coming alive. The painting captures the daily life of people in the Song dynasty of ancient China. People are interested in how forefathers lived and acted in that drawing in ancient time. Making antique alive is not only interesting but also educational. People enjoy interacting with such antique paints or portraits through digital media.

In addition to making the ancients move, restoring their realistic appearance can be more astonishing. There are few existing artworks on this topic but they required intensive human interactions. So we propose a semi-automatic system to score this goal.

A new rising concept named “Untoon”, is to enhance photorealistic details for artwork of cartoon characters while preserving their cartoon-like characteristics. This new concept comes up on the internet. Thereafter, the trend of untooning has continued to be practiced by other graphic artists on the internet.

These talented “untoon” artists mostly use image editors like “Photoshop” to create those astonishing works. Nevertheless, making one untooned image is time-consuming, tedious and requires considerable expertise. For instance, they have to search for a well fitted human photo first, and then they use various editing functional tools for photo retouching.

The concept of untooning is to realize cartoon characters with extraordinary facial features. There are difficulties to automatically process the untooning idea. For instance, the problem of flat lighting of the input image leads to the flat lighting result is not an issue can be solved in a short time. Although the smooth term in our algorithm can keep the local lighting and similarity but it cannot handle large region lighting problem. We do not have an image relighting model especially for faces. The lighting for the untooned image is natural due to the artist’s attention and concern. Artist spends a lot of time to find the best matching, fitting and reasonable sources and retrieves the patches to synthesize by image editors. It is a difficult problem to solve by automatic tool in present stage. Therefore, although it is not our target but we can seem this idea as a long term future works to accomplish.

In this thesis, we try to develop a new and automatic tool with manually adjustable options to ease the workload for the expertise and can even be used by novices. However, we had to confront the considerable variations of human faces, e.g. identity, pose, expression, hairstyle, lighting, and other factors. Furthermore, people are highly familiar with faces and are especially sensitive to synthetic defects.

Human faces pose a difficult confront as they display considerable variation due to identity, pose, expression, hairstyle, lighting, and other factors. Furthermore mans have massive visual experience of faces and may be especially sensitive to errors.

For the face shape and the proportion of facial features of input images that are not real

enough as human, we provide option for the users to symmetrize the input images first before starting the afterward steps. The symmetrize option works on frontal faces.

The technique we proposed can also be applied to the aspect of entertainment. Such technique would leads towards automatically creating realistic humanoid actors and avatars. It can also create realistic human characters for games and movies.

The main insight of this paper is that we formulate the process as an optimization problem and perform on 2D space with an appropriate amount of photograph collections. We do not have to precisely recognize the details in images. We transform each of our database images to the same viewpoints as user input painting. Then, we progressively choose the best sources for each patch, and finally the optimal parts are stitched to be our result. The system overview in the later section explains how our framework works. In the third and fourth chapter, we elaborate our method and recount the reason for choosing such methods. Finally, we manifest our experiments and discuss the results.

1.2 Overview

We present our system overview and illustrate the function of each component. Our method focuses on realizing and preserving feature structure of human faces on paintings. It is applicable for exaggerated or highly unrealistic painting. Nevertheless, Japanese animation characters with over-emphasized round eyes will not be applicable to our tools. There are three stages in our system: preprocessing stage, graph-cut-based multi-label optimization and seamless patch stitching. With these stages, we extract the most likely patches from the database for our input painting and remove the patch boundaries to form a smooth result.

In the preprocessing stage, we collect a large amount of images to form our database. We

collect our images from internet, such as photo albums and image searching engines. These photos are open and free to use. We collect our database with individual photo that has a clear face with sufficient size. After collections, we find the feature points of facial features on each photo with Stacked Trimmed Active Shape Model (STASM) [12]. It is extended from Active Shape Models (ASM) which developed by Tim Cootes *et al.* [13]. User interaction and correction is available in this step. Next, we transform and warp photos in database to the view point of input painting with feature points by perspective projection warping. Nevertheless, perspective projection warping may not warp the feature points to the exact positions we offer; we further apply local warping to amend it.

The next stage, we find out the best and most similar pieced up face for input image. Mohammed *et al.* [4] proposed a Visio-lization algorithm which locally extracted the images into given size patches. But with fixed size and shape patches, the edges on face may not be continuous. Therefore, we try not to limit the size and fixed shape for our patches on faces. Instead of cutting the faces into fixed size and shape, we formulate the variable-size patch search as a graph-cut-based optimization problem.

For the third step, we have tried to use Poisson image editing [5] to solve gaps between patch boundaries. However, this method is easily influenced by large illumination changes and does not match our expectation. We had to abandon this thought and we choose to adjust the method proposed by Brown *et al.* [6] in use. Gain compensation helps lessen the color differences between adjacent patches.

Even though gain compensation has made the patch boundaries resemble in color, there are still notable seams between patches. Consequently, we use Gaussian-low pass filter to filter our images into two levels: the high level and the low level. To make the boundary seamless and smooth, we use weighted blending along the patch border. After that, we put

back the high level information. Detailed textures (e.g. wrinkles, pores) will still be preserved.

This paper is organized as follows. In chapter 2, we introduce several classical articles and other related writings. The overview of our system is described in addition. The detail of our algorithm is explained in chapter 3 and 4. In chapter 5, we present the result and experiments of our system. The last chapter described our conclusion and the future work. Figure 1.1 shows the overview of our system.

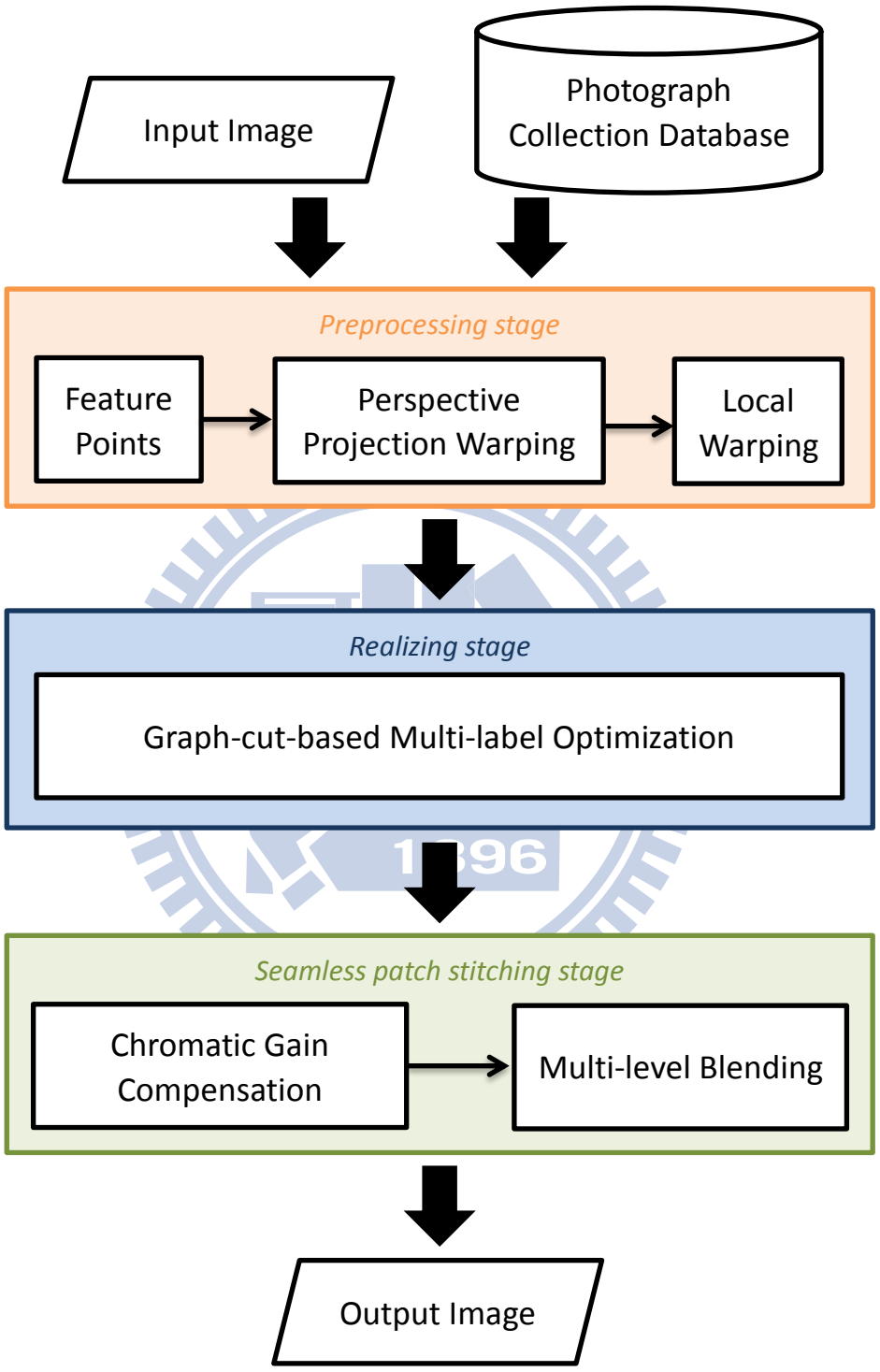


Figure 1.1 The overview of our system

Chapter 2

Related Work

It is not an easy task to analyze and synthesize novel facial images with limited quantity of photographs with example-based methods. Several classical papers targeted this goal. In this section, we introduce several state-of-the-arts of this field and related articles, such as image editing and composition, face analysis and synthesis and synthesis and analysis from sketch or sparse input.

2.1 Image Editing and Composition

Image editing and composition are techniques like image stitching, image blending and image warping. In our method we use projective warping and 2D local warping for transformation. Image stitching is a close concept to ours, which is to pieced image together with well join. Image blending is to remove the color gaps of boundary and make the image seamlessly, which is another technique we have to use. Related articles about image stitching and blending are introduced below.

"Image Stitching Using Structure Deformation" [19] is to achieve seamless stitching images. They prevented the artifact from severe intensity inconsistency and the misalignment of structure. They input images that are roughly aligned and solve by structure deformation and propagation for the overall uniformity in image structure and intensity.

In the last step of our approach, seamless patch combination, we initially intended to use Poisson image editing [5] proposed by Pérez *et al*. Poisson image editing can copy a region from source image into a destination image seamlessly in gradient domain. However, this method performs satisfactory only when the gradients at source and target image borders are similar. It is easily affected by the significant color changes e.g. shadow or make-up near patch boundaries, especially for 3 channel color space. If the patch boundary steps in a region that differs dramatically, that color will progressively propagate into the region which cause the texture tainted.

Another reason for not using Poisson editing is that if we use Poisson method for seamless stitch it smooth out the boundary of the region when we divide our image into smaller and irregular patches. The more patches we had, the more blurred regions we get.

Therefore, we refer to "gain compensation" proposed by Brown *et al* [6]. With the overlap regions between all the patches, it narrows down the color differences between all the nearby patch pairs. This method is originally created for panorama and it makes the later blending easier and better. It aimed at optimizing the sum of gain normalized intensity errors, for all overlapping pixels.

Chromatic gain compensation is the method we adjusted from gain compensation. The original gain compensation is for panorama, which the images are taken in a close and short time, so the color difference in between each patches are only slightly changed. Therefore, they used only one gain variable for each patch. The patches in our cases come from different sources and the colors in between varies a lot, so we use three gain variables for each patch, which the results are more satisfying.

Heeger *et al.* [7] utilized manageable pyramid to obtain various frequency bands and

resolution-aware methods are applied to each sub-band respectively to preserve image details. A multi-scale representation and acquisition technique was presented by Bickel *et al.* [8]. This technique is for animating high resolution facial geometry and wrinkles. Facial expressions are categorized from fine scale (e.g. spots, pores) to coarse scale (e.g. eyelids, cheeks, lips, nose), and for different scale features they used different equipment.

2.2 Face Analysis and Synthesis

To finding landmarks for face and facial features, recent approaches have used structural matching methods such as the Active Shape Model (ASM) [13]. Another method closely related is the Active Appearance Model (AAM) [20]. These two models are similar but ASM is more flexible when new face is added.

Blend shape is one of the data-driven approaches that proposed creating novel facial expressions from a set of example appearances. For instance, Pighin *et al.* [1] combined the geometries and textures of example models on convex vector space. The main idea of blend shape is to represent each example expression as a convex vector. Combining those example expressions by the convex vector can create a novel expression. However the expression editing system they proposed requires user manually specify the coefficients of convex combination.

Zhang *et al.* [2] solved the troublesome problem of parameter tuning by using a geometry-driven expression synthesis system. Although blend shape can synthesize diverse facial expressions, but high-resolution facial details such as wrinkles and pores may be blurred during image blending.

Mohammed *et al.* [4] proposed a Visio-lization algorithm for generating novel realistic

facial images by using a well-aligned model from real photos. This method contains two parts, local and global models. With global parametric model a blurry image is generated by principle component analysis (PCA). For local model, images are firstly extracted into given size patches. A patch is chosen if it is visually consistent with existing patches in its left and above. Finally, the patches are stitched by Poisson Image editing [5].

Visio-lization algorithm performs efficiently. The synthesized faces by Viso-lization look realistic. However, with fix size and rectangular patches is not adjustable to fit face features. For faces, straight borders may cut the continuous textures. Also due to the fixed size and shape patches, wrinkles on face may be cut off and cause discontinuity. Therefore, we propose a flexible structure without limiting its size and shape for the patches. Our method can avoid the wrinkle-cut problem by connecting the edges between neighboring patches.

Liu *et al.* [11] proposed a technique, called the expression ratio image (ERI). It can map facial expression details to another face and seizes the illumination change and facial feature motion.

Suo *et al.* [22] presented an article about face aging, where a compositional and dynamic model is applied. Each faces in different age group is presented by a hierarchical And-Or graph. They synthesized different age faces by the different details they calculated from the model.

“Exploring Photobios” [21] is able to create a face animation from a collection of photographs of the same person. After given the source and target image, it built a facial similarity graph to find the best fitting transition images between two photos.

Bitouk *et al.* [3] proposed “Face Swapping” to replace human face by similar faces. The

similar face is ranked by pose, color, lighting and blending cost. This method is good at replacing the whole face, but not for generating novel faces.

2.3 Synthesis and Analysis from sketch or sparse input

Sketch-based face searching or recognition is another related topic to what we do. Face sketch is a brief yet recognizable form of people. Klare *et al.* [23] performed face recognition based on the Scale-Invariant Feature Transform (SIFT) feature descriptors and multi-scale local binary patterns. Given a forensic sketch, the algorithm recognized and finds the most similar mug shot photo. This approach performs well on face sketch but not on cartoon paintings. Cartoon paintings are more concise and exaggerated than sketches. And do not have the details as face sketches, so it is unfit for our goal.

The CG2Real [9] system used a large collection of photographs to decorate photo-realistic computer-generated images with details. By transferring color, tone and texture from photos of similar scenes to improve realism in a CG scene. Mean-shift cosegmentation algorithm is the key ingredient in the system that matches regions in the CG image with regions in the photographs.

Recently, Shrivastava *et al.* [10] presented a method that can match paintings or sketches to real photographs. The main goal of this paper is to find images that are visually similar. Visually similar means images have to be found even if they are distinct at the raw pixel level.

Chapter 3

Preprocessing Stage

In the preprocessing step, we apply the Stacked Trimmed Active Shape Model (STASM) [12] to find the facial feature points of our database photographs. Users can unrestrictedly revise the positions of these feature points. Based on the more precise future points, we can synthesize better results. Afterward we use projective warping to transform database photographs into the same aspect and angle of input image. By projective warping, the photographs are warped to a closest pose to fit the input face features, but the positions are not precisely aligned. We then apply local warping to adjust these warped point coordinates. After the photographs of database are aligned, our system can then process the task of face synthesis. The later processes are discussed in the next chapter.

3.1 Preprocessing

Collecting Photographs

The very first step of our system is to build a database with photographs of humans. We gathered photographs from internet image searching engines and photo albums. These photographs we gathered are publicly shared. We mainly collect female photographs to form our database but our method can easily be used for male faces either. Gathering individual photo with clear and big face is the foremost norm we pursue.

Feature Points

We use a software package named Stacked Trimmed Active Shape Model (STASM) [12] to find feature points for face alignment. STASM is used to locate landmarks for faces, and it is extended from Active Shape Models (ASM) [13]. Before the landmarks search, STASM have to find the face region in the photograph. Here we use the Viola Jones global face detector [15] [16].

STASM will automatically return the landmarks it found, as the red dots shown in Figure 3.1 (a). However, it does not include the landmarks in the region of forehead, which is also the important information we need. Accordingly, we choose to do region upward propagation according to the skin color from the eyebrows edge. We adopt the canny edge as our edge detector. The forehead feature points are demonstrated in Figure 3.1 (b). The actual feature points we use are not identical to the landmarks STASM returns. We drop several unreliable feature points. Figure 3.1 (c) displayed the 61 landmarks that are in use.

(a) (b) (c)

Figure 3.1 (a) The landmarks found by STASM (b) The landmarks found by STASM and the forehead feature points we found (c) The feature points we use

Next we further represent the facial feature by continuous contours. On the contour, we can create or sample more corresponding landmarks. We use Catmull-Rom Splines [17] for this purpose.

Given two control points P_0 and P_1, the slopes of the tangents at each point are denoted as P_0' and P_1'. The parametric cubic curve, passing through P_0 and P_1 with the slopes, can be defined by the following polynomial function with coefficients a_0, a_1, a_2 and a_3:

$$P(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \tag{3-1}$$

We can derive from the above function:

$$\begin{aligned} P(0) &= a_0 \\ P(1) &= a_0 + a_1 + a_2 + a_3 \\ P'(0) &= a_1 \\ P'(1) &= a_1 + 2a_2 + 3a_3 \end{aligned} \tag{3-2}$$

Substitute Equation (3-2) into the original polynomial equation and isolate P(0), P(1), P′(0) and P′(1), we get a clear cubic polynomial form matrix as below:

$$P(t) = \begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -3 & 3 & -2 & -1 \\ 2 & -2 & 1 & 1 \end{bmatrix} \begin{bmatrix} P(0) \\ P(1) \\ P'(0) \\ P'(1) \end{bmatrix} \tag{3-3}$$

Moreover, for Catmull-Rom Spline the tangent at each point P_i can be formulated using the previous and the next point on the spline.

$$P(t) = \begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -3 & 3 & -2 & -1 \\ 2 & -2 & 1 & 1 \end{bmatrix} \begin{bmatrix} P_i \\ P_{i+1} \\ \frac{P_{i+1} - P_{i-1}}{2} \\ \frac{P_{i+2} - P_i}{2} \end{bmatrix} \tag{3-4}$$

Figure 3.2 Representing facial features by the Catmull-Rom Spline

After organizing Equation (3-4), we obtain:

$$P(t) = \frac{1}{2}\begin{bmatrix}1 & t & t^2 & t^3\end{bmatrix}\begin{bmatrix}0 & 2 & 0 & 0\\ -1 & 0 & 1 & 0\\ 2 & -5 & 4 & -1\\ -1 & 3 & -3 & 1\end{bmatrix}\begin{bmatrix}P_{i-1}\\ P_i\\ P_{i+1}\\ P_{i+2}\end{bmatrix} \quad (3\text{-}5)$$

With Equation (3-5), we input the 61 landmarks mentioned before as control points for Catmull-Rom Spline, Figure 3.2 shows as our result.

Like all automatic techniques, STASM and our forehead feature point finder algorithm may not always be as accurate as a human landmarker. It is still possible to get incorrect positions by the methods. Therefore, we provide user interaction in this step. Users can adjust the feature points to more accurate positions.

3.2 Transformation

Perspective Projection Warping

Image warping and transformation is a way to rearranging the position of pixels of an

image by performing geometric transformation. This skill is often used in both image processing and computer graphics (e.g. texture mapping, image mosaicing). Warping a source image into a destination image requires a mapping function between source space (u,v) and destination space (x,y).

We define the source image and target image as I and I'. The corresponding points would be the extracted facial feature points by above mentioned methods. A feature point in I defined as $p_i=(u_i,v_i)^T$ and the desired new position in destination image $q_i=(x_i,y_i)^T$, i=1,…,n. The goal of image warping is to find a transform function subject to $f(p_i)= q_i$. For each pixel i, $I'(f(p))= I(p)$. $I(p_i)$ and $I'(q_i)$ is the intensities or colors of the images.

Here we use projective mapping, also known as the perspective or homogeneous transformation. It projects points from one plane onto another plane:

$$\begin{bmatrix} wx \\ wy \\ w \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \tag{3-6}$$

Each point correspondence generates two equations:

$$\begin{aligned} x(A_{31}u + A_{32}v + A_{33}) &= A_{11}u + A_{12}v + A_{13} \\ y(A_{31}u + A_{32}v + A_{33}) &= A_{21}u + A_{22}v + A_{23} \end{aligned} \tag{3-7}$$

A projective mapping has 8 degrees of freedom which need at least 4 correspondences to solve. After rearrange equations above, this can be rewritten as:

$$\begin{bmatrix} u_1 & v_1 & 1 & 0 & 0 & 0 & -x_1u_1 & -x_1v_1 & -x_1 \\ 0 & 0 & 0 & u_1 & v_1 & 1 & -y_1u_1 & -y_1v_1 & -y_1 \\ u_2 & v_2 & 1 & 0 & 0 & 0 & -x_2u_2 & -x_2v_2 & -x_2 \\ 0 & 0 & 0 & u_2 & v_2 & 1 & -y_2u_2 & -y_2v_2 & -y_2 \\ u_3 & v_3 & 1 & 0 & 0 & 0 & -x_3u_3 & -x_3v_3 & -x_3 \\ 0 & 0 & 0 & u_3 & v_3 & 1 & -y_3u_3 & -y_3v_3 & -y_3 \\ u_4 & v_4 & 1 & 0 & 0 & 0 & -x_4u_4 & -x_4v_4 & -x_4 \\ 0 & 0 & 0 & u_4 & v_4 & 1 & -y_4u_4 & -y_4v_4 & -y_4 \end{bmatrix} \begin{bmatrix} A_{11} \\ A_{12} \\ A_{13} \\ A_{21} \\ A_{22} \\ A_{23} \\ A_{31} \\ A_{32} \\ A_{33} \end{bmatrix} = 0 \qquad (3\text{-}8)$$

We can regard this question as a least-square problem. We solve this problem by calculating the Singular Value Decomposition (SVD) of the matrix of Equation (3-8) to get the smallest singular value.

The reason why we choose to use projective warping instead of the others is described below. In our case, we need to warp one face to the other, so it will confront the perspective distortion problems due to different viewpoints. Other warping method may only be applicable to view angel difference up to 15 degrees.

After derived the mapping function from the source image to the destination image, we need to warp each point on source image to destination image. If we use forward mapping, each point in source image after transformation has a real-valued coordinates. It needs to round off to integer coordinates, but it may have error and misalignment.

Therefore, we find the inverse of mapping matrix which maps from destination image to source image. Then, instead of using forward mapping, we use backward mapping or named inverse mapping. Every point in destination image is filled with the intensity from source image. Backward mapping prevent from round-off error and the misalignment problem.

Local Warping

Even though perspective warping solved the problem of view point change, it does not guarantee the perfectly-aligned correspondence. Therefore, we try to solve this problem by applying local warping in the following step.

Each pixel q is transform to pixel p by the projective mapping function above mentioned, which is $p=T(q)$. For more accurate alignment, we find the new position p' by a local warping function f. The new adjusted position by local warping can be calculated by:

$$p' = f(p) \tag{3-9}$$

There are m pairs of corresponding feature points. Feature point s_j from the source image is corresponding to feature point d_j from the destination image to form one pair of corresponding feature points. If the projective mapping function is accurate, $d_j=T(s_j)$, point s_j is supposed to transform to point d_j. However, with the projective mapping function, $d_j'=T(s_j)$ $d_j'\neq d_j$, point s_j is warped to point d_j' instead of point d_j.

The last term in Equation (3-10) is the translation between d_j and d_j'. The c in Equation (3-11) is a small constant to avoid zero and η_j is the weight for translation. Equation (3-12) is for normalization.

$$f(p) = p + \sum_{j=1}^{m} \eta_j (d_j - d_j') \tag{3-10}$$

$$\hat{\eta}_j = \frac{1}{|s_j - q|^2 + c} \tag{3-11}$$

$$\eta_j = \frac{\hat{\eta}_j}{\sum_{h=1}^{m} \hat{\eta}_h} \tag{3-12}$$

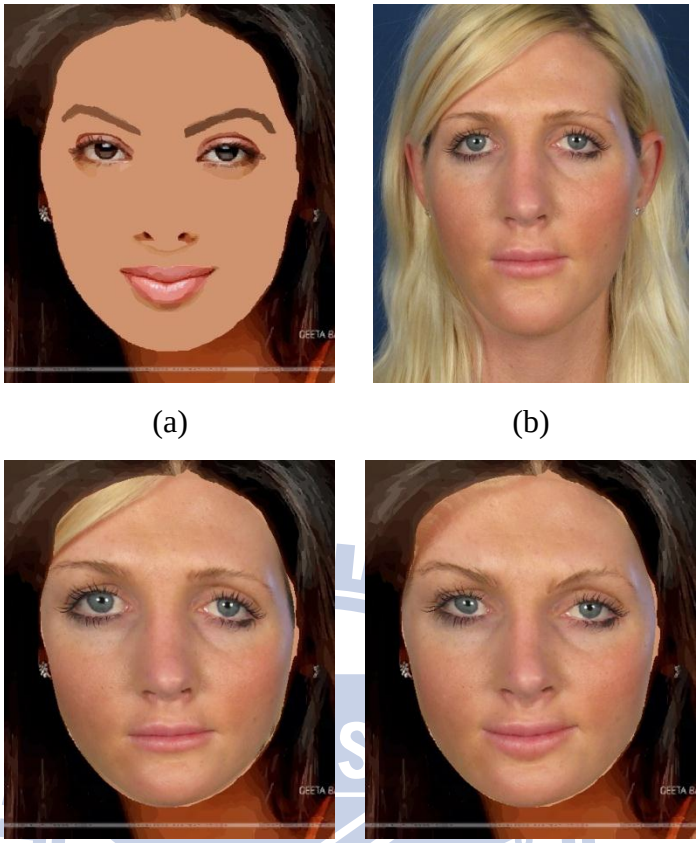
The coordinate for pixel p' may not be exactly on the grid, so directly sampling the color of nearby pixel is not accurate enough. Therefore, we use the adjacent left, right, up and down pixels t_h to interpolate mapped color. We derived the color of p' by the RGB values of adjacent four pixels. The weight ϕ_h is calculated with the distance between each nearby pixels. Equation (3-15) is for normalization.

$$C(p') = \sum_{h=1}^{4} \phi_h C(t_h) \tag{3-13}$$

$$\hat{\phi}_h = \frac{1}{\sqrt{(t_h - p')^2}} \tag{3-14}$$

$$\phi_h = \frac{\hat{\phi}_h}{\sum_{g=1}^{4} \hat{\phi}_g} \tag{3-15}$$

In Figure 3.3, image (a) is the destination image and image (b) is the source image, we warp image (b) onto image (a) with the facial feature points. Figure 3.3 (c) showed that the result of perspective projection warping is visual satisfactory. However the corresponding positions are not exactly aligned. It can be seen from the shape and position of eyebrows, eyes, and mouth. Local warping adjusted image (c) slightly and the correspondence are now in precisely aligned.



(a) (b)

(c) (d)

Figure 3.3 (a) The target image (b) The source image
(c) The result of perspective projection warping (d) The result after local warping

Chapter 4

Realizing Cartoon and Painting Faces

Aligning the database photographs to the input face, we are now able to use graph-cut-based multi-label optimization [14] to find the best matched patches in arbitrary shapes. The best match goal in our method is to find the highest similarity in colors and edge, and keep locally spatial affinity. Once the most similar face is pieced up from database, we present two procedures to remove visible discontinuities between these patches on face. Gain compensation, tuning the intensity of each patch, is used to lessen the color differences between adjacent patches. At last, we separate the patch boundaries into multi-level and blend each channel separately to generate realistic and novel faces.

4.1 Graph-cut-based Multi-label Optimization

Graph-cut is a method that finds an optimal node partition. It is equivalent to max-flow min-cut. In computer vision, graph cut methods are often used to tackle optimization of discrete elements, like pixels. For instance, graph-cut can segment the image into the background and foreground regions.

Graph-cut regards node partition as a labeling problem. For example, in image segmentation, it labels the foreground pixels as one and the background as zero. That is a two-label problem. To find the best partition for an input image, graph cut solve the problem by maximizing the flow according to edge capacities.

When we assign a label to each pixel, labels should have spatial affinities while preserving possible sharp discontinuities. These tasks are stated as energy minimization. Solving global minimization of energy function is NP-hard problem. Therefore, Boykov *et al.* [14] proposed alpha-expansion, which is an efficient approximation algorithm.

Later, more papers about applying multi-label optimization were presented. However, most of them were applied for segmenting different regions on an image. The multi-labels in our work represent the region indices of image pixels. The patch can therefore be in arbitrary shapes.

We have various requirements about portrait realization. We wish each pixel on input image comes from the corresponding position of the most similar photograph in database as source. Besides, we also require the synthesized face should be similar to input image and also realistic. Therefore, multi-label optimization matches our needs. The labels in our situation refer to the photograph ID in database.

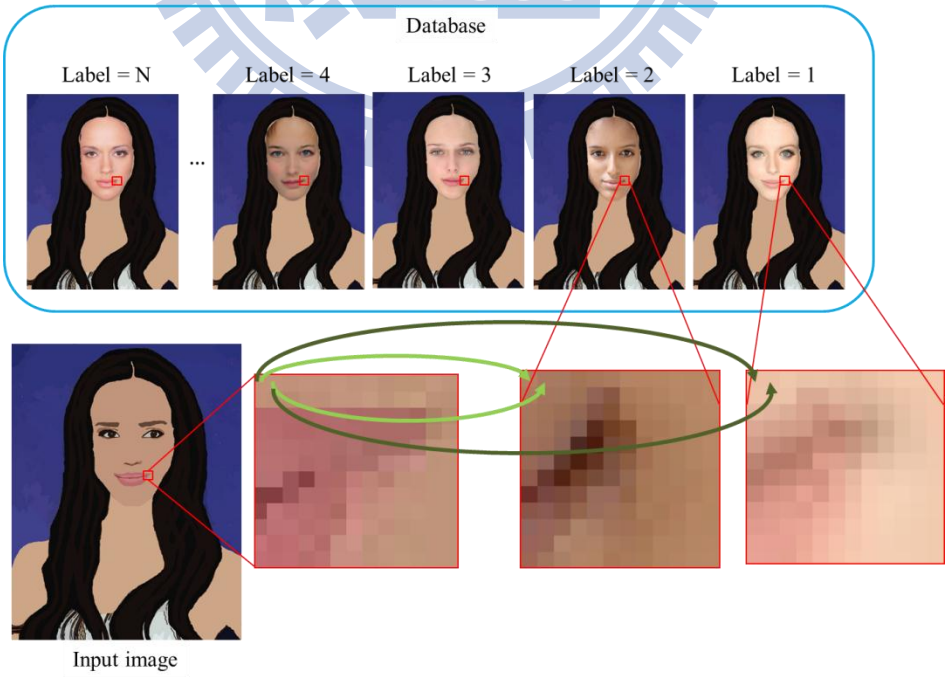


Figure 4.1 Matching multi-labels to our problem

As Figure 4.1 shown, the labels in our cases are presented as each image index in database. For each pixel in the input cartoon or painting image, it can match the pixels at the same aligned position of all photos in database. Corresponding pixels refers to the pixel that has the same coordinate as the input image pixel after aligning with feature points. Each input image pixel will pick one label (photo ID). Which label the pixel picked means which source of image index it want to come from. Below, we are going to discuss about how the pixels pick the most adequate labels.

It is obvious that we should compare the corresponding pixel colors to determine which label to choose. The greedy method is to find the pixel labels with minimum color difference. However, if we only compare the pixel color differences, the result will be almost the same as original input image. It lost the realism of faces and creates the discontinuity in faces, so that is not fit for our problem.

In order to preserve the realism of face, we expect the extracted labels should carry local and reality details from real faces. Therefore, the individual pixel labels would gather into patches. We consult the labels and color differences of the adjacent pixels. If we only determine the label by the color difference of its own, we can see an example from Figure 4.2. Node *p* would choose the most similar label 3 as its label, and the neighbors choose label 2. However, label 2 is also node *p*'s secondary similar label. So in this case we want node *p* to choose label 2 to follow its neighbors, and become a group with the identical label, called patch. This approach preserves the realism and smoothness of face. Therefore, for our energy function, we would consult the terms of color difference by each pixel and also the neighbor affinity of labels according to color differences between neighbors.

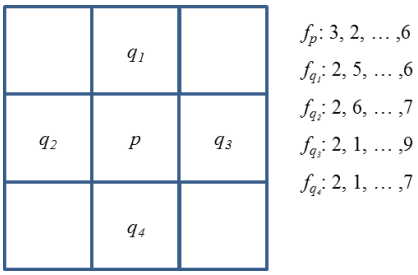


Figure 4.2 Node *p* and 4-neighbors

TABLE I

GRAPH-CUT-BASED MULTI-LABEL OPTIMIZATION SCHEME

1. Perform Canny edge detection for input image and each database images.
2. Perform Gaussian filter to propagate the range of edge.
3. Set up the neighborhoods for smooth term and symmetry term.
4. Iteratively solve our energy optimization function by alpha expansion with step 5.
5. Compute data term, smooth term, symmetry term and edge term penalty.

We formulate our task as a labeling problem of Markov Random Fields (MRF). The input image I is represented by a weighted graph $\mathbf{G} = (P, N)$. Each pixel stands for one node $p \in P$ in the graph **G**. The pairwise adjacent pixels are linked by edge $\langle p,q \rangle \in N$. 4-neighbors are used here. Next, the MRF can be formulated as the following energy function:

$$E(f) = \sum_{p \in P} D_p(f_p) + \sum_{p,q \in N} S_{p,q}(f_p, f_q) \tag{4-1}$$

We use alpha-expansion method to solve our energy optimization function. The graph-cut-based multi-label optimization method is briefly described in Table I.

Data term

The first term in Equation (4-1) is known as the data term, as described in Figure 4.3. As mentioned above, each pixel of the input image choose a belonging label by comparing the color difference with the corresponding pixels. For location *p*, we calculate the difference between the intensity of input image and database photograph with label *f*. As described before, the label in our cases represents the index of database photograph. Data term ensures that pixel p in label f photograph is similar to pixel p in input image. It gives a large penalty when pixel *p* in label f photograph is too different to the observed pixel p. The data term D_p in Equation (4-1) can be defined as:

$$D_p(f_p) = \frac{\left\| I_{input}(p) - I_{f_p}(p) \right\|^2}{w}, \quad p \in P \tag{4-2}$$

$I_{input}(p)$ and $I_{f_p}(p)$ denote the intensities of location p in input image and label f photograph. We calculate the color difference by Euclidean distance. The variable w normalizes the color difference range to zero to one.

Data term penalty prevents each pixel from matching with a very dissimilar pixel from photograph in database. The more dissimilar pixel is labeled the more penalties it will get. Therefore, it can keep the similarity of the result and input image. However, keeping intensity makes the image fragment into discontinuous and separate crumbs from multitudinous sources. It lowers the reality of result faces.

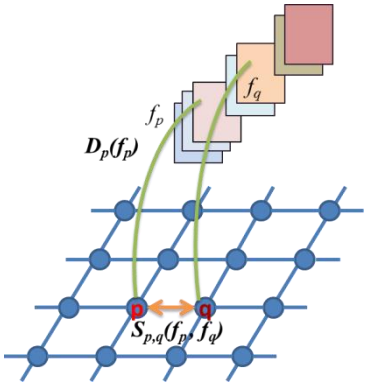


Figure 4.3 The illustration for data term and smooth term

Smooth term

The smoothness in Equation (4-1) is the second penalty term that can retain neighbor affinity. As showed in Figure 4.2. It is based on 4-connected neighborhood system. The purpose of the smooth term purpose is to keep the local labeling smooth, so it penalizes two neighbor pixels *p* and *q* if their colors are similar in input image but is assigned with different labels. f_p and f_q represent the assigned labels to pixel *p* and *q*.

The smooth penalty is defined as follows:

$$\begin{cases} S_{p,q}(f_p, f_q) = 0 \Leftrightarrow f_p = f_q \quad \text{or} \quad S_{p,q}(f_p, f_q) \neq 0 \Leftrightarrow f_p \neq f_q \\ S_{p,q}(f_p, f_q) = S_{p,q}(f_q, f_p) \geq 0 \\ S_{p,q}(f_p, f_q) \leq S_{p,g}(f_p, f_g) + S_{g,q}(f_g, f_q) \end{cases} \tag{4-3}$$

Assume that *p*, *q* are two adjacent pixels. The first rule tells that the penalty of edge *p*, *q* should be non-zero if the label $f_p \neq f_q$. If the energy equals to zero, then the labels must be the same. The second rule defines the energy between adjacent nodes is mono-combination. The last rule form to the triangular inequality, where the energy for a shortcut is always lesser than taking the indirect path.

The smooth term of our energy function is defined below:

$$S_{p,q}(f_p, f_q) = \begin{cases} 0 & , \text{if } f_p = f_q \\ \lambda \exp\left(-\dfrac{\left\| I_{f_p}(p) - I_{f_q}(q) \right\|^2}{w}\right), & \langle p, q \rangle \in N \quad , \text{other case} \end{cases} \tag{4-4}$$

λ is a trade-off weight, which decide how smooth the system is. When λ value is big, the results will mostly have the same source, which makes the result do not look like the input cartoon image. The constant *w* is for range normalization.

Figure 4.3 illustrate the smooth term. If node *p* and node *q* have the same label, which mean they will get intensity data from the same source, the smoothness between them is conceivable continuous. So we give no penalty for this pair of nodes. When node *p* chooses different label as its neighbor *q*, we give punishment for discontinuity by Equation (4-4). If the color difference of pixel $I_{f_p}(p)$ and pixel $I_{f_q}(q)$ is large then the penalty for node *p* and node *q* is large for the extensive discontinuity.

Symmetry term

With the data term and smooth term, we can get satisfactory results in most cases. However, human eyes are sensitive to faces, especially to symmetry of human appearance. In certain cases, the results might not be satisfying for users due to asymmetric eyes or others. Therefore, we want to adjust the problem of symmetry through the third kind of penalty, called symmetry term.

Huang *et al.* [18] proposed an idea in the paper “RepSnapping”. They add a new term to the graph-cut energy function, other than data term and smooth term. A new term, repetition term is added to the energy function. This term link more neighbors in the RGB color space, while these linked nodes might be far away in spatial relationship. With this term,

"RepSnapping" is able to segment similar objects in an image even if the objects are spatially far from each other.

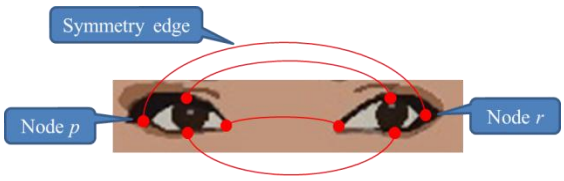


Figure 4.4 Symmetry Neighborhoods

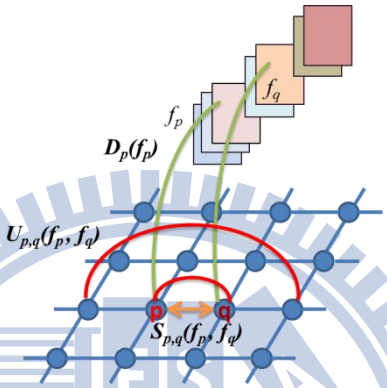


Figure 4.5 The illustration for symmetry tem

On the other hand, we propose a symmetry term to our energy function. It is used to connect symmetry facial features even if they are separated. For instance, we can make new links between left eye and right eye, and also link left eyebrow and right eyebrow nodes to become special neighborhoods. As showed in Figure 4.4, each node *p* is linked to node *r* based on facial symmetry to form a symmetry edge and symmetry Neighborhood *M*. Our energy function becomes:

$$E(f) = \sum_{p \in P} D_p(f_p) + \sum_{p,q \in N} S_{p,q}(f_p, f_q) + \sum_{p,r \in \mathrm{M}} U_{p,r}(f_p, f_r) \tag{4-5}$$

The symmetry term of our energy function is defined below:

$$U_{p,r}(f_p, f_r) = \begin{cases} 0 & ,\text{if } f_p = f_r \\ \mu \exp\left(-\dfrac{\left\| I_{f_p}(p) - I_{f_r}(r) \right\|^2}{w}\right), & \langle p, r \rangle \in \mathrm{M} \quad ,\text{other case} \end{cases} \tag{4-5}$$

μ is a trade-off weight. The constant *w* is for range normalization.

Figure 4.5 shows the link for symmetry term on the graph. For pairs of symmetry node, node *p* and node *r*, since it comes from the same person, we believe if they have the same label that the eyes is symmetrical. So we give no penalty in this case. If they have different label, we give some penalty for choosing from different source that may cause asymmetry. If the pair of nodes chooses from different source but they have similar color then the penalty is less. However, if the pair of nodes chooses different sources and has large color difference, the penalty is large for causing a bad result.

Edge Penalty

Moreover, another new term, edge penalty, is also added to our energy function. We merge this edge penalty into the data term. The originally data term is comparing only with color differences, shown in Equation (4-2).

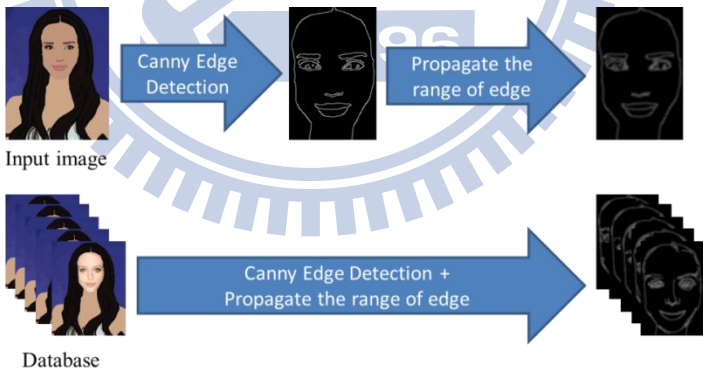


Figure 4.6 The overview for preprocessing the edge information

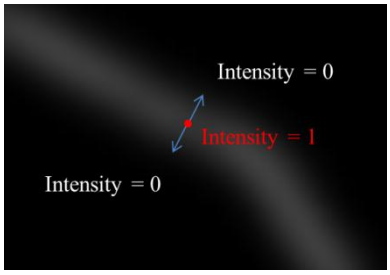


Figure 4.7 Gaussian filter to propagate the range of edge

The purpose of this edge penalty is to deal with the discontinuous edges on synthesized face from multiple sources. As we mentioned before, Visio-lization algorithm [4] use fixed size and shape patches, the edges on face may not be continuous across patches. We solve this problem by using arbitrary size and shape of our patches on faces. The arbitrary patches solve most of the discontinuous edge problem. However, using the edge penalty can perform even better.

We use Canny edge detector to find the edges in the input cartoon image and all the database photographs, shown in Figure 4.6. Pixels with edge crossing over will have values equals to one, other pixels will have zero. Next, we use Gaussian Filter to propagate the range of edge. After filtering, the value of pixels near edge will be closer to one and farther approach to zero. This can be seen in Figure 4.7. The middle of the edge has the highest value and progressively down to zero. Propagating the range of edge is for increasing the edge region because the database image edges may not be exactly on the edge of input image. Besides, the gradual change of edge intensities can help the optimization process progressively move the parameters during multiple iterations. The new data term after adding edge penalty becomes:

$$D_p(f_p) = \frac{\left\| I_{input}(p) - I_{f_p}(p) \right\|^2}{w} + \sigma \left| E_{input}(p) - E_{f_p}(p) \right|, \quad p \in P \tag{4-6}$$

σ is a weight, which decides the proportion of edge penalty in energy function. With variable size and shape patches we can prevent the face from discontinuity in most of the cases.

We use Figure 4.8 to describe several different cases of edge penalty, where the three pixels are all assigned to the same label *K*. Location *p* of the three cases are denoted as p_1 to p_3 in the Figure. The white pixel has the highest value one which is the strongest edge and

black pixel has the lowest value zero which is far from the edge. The gray pixels have the blending value, such as 0.49 and 0.85. For the first case, p_1 in input image is on the edge center which has edge value one and it chooses label *K*. In the image *K*, p_1 is also on the edge center which returns no penalty. It is a good match for matching edge center to edge center. Location p_2 is in the center of the edge matched to the close surrounding of edge. Although it does not match exactly to the center of the edge but it has small probability that it will cause discontinuous for the result, so it gets a low penalty. Case 3 describes an edge pixel matching to a pixel with no edge crossing and far away from the edge, which had a very big chance of causing discontinuity. Therefore, case 3 gets a large penalty.

Figure 4.9 (a) is the result without edge term. The lower lip did not match the edge across patches well, so the lip line is not smooth enough. After we add the edge term to our energy function, it preserves lip line well and smooth such as Figure 4.9 (b). Figure 4.9 (c) shows under different variable setting, even if here are different patches connecting it can still get a satisfying results.

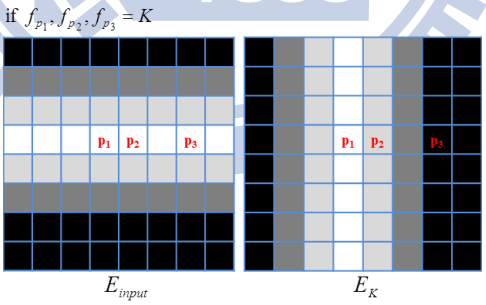
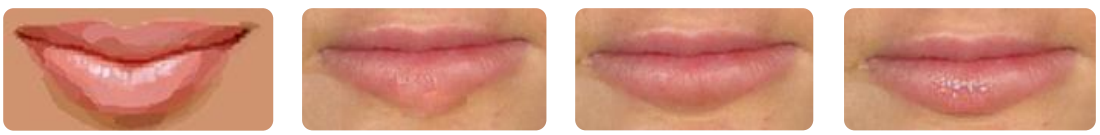


Figure 4.8 Cases of edge term penalty



(a) (b) (c) (d)

Figure 4.9 (a) The cartoon mouth (b)(c) Compared results between without edge term and with edge term (d) Other result with edge term

4.2 Seamless Patch Stitching Stage

In seamless patch stitching stage, we want to solve the problem of the intensity gaps between adjacent patches from different sources. Every photograph has different color tones, so the patches we retrieve have intensity gaps between them. Therefore, we use gain compensation and multi-level blending to seamlessly stitch the patches.

Chromatic Gain Compensation

Gain compensation is proposed by Brown *et al.* [6]. This concept is originally presented for stitching panorama. With the overlap regions between all the patches, gain compensation narrows down the color differences between all the adjacent patches. In our work, we adjust for arbitrary shape patch stitching.

Panorama use images taken in a close and short time, so the color difference between each images are only slightly different. Therefore, the originally gain compensation used only one gain variable for each stitching patch. However, the stitching patches in our cases come from different images and for different sources the colors differ much. A single gain variable is not sufficient in our cases to narrow down the color differences between adjacent patches. Therefore, we use three gain variables for each patch and each gain is for each color channel.

The method of chromatic gain compensation is briefly described in Table II. First, for each patch, we expand the patch region to form overlap regions with every adjacent patches. Here we define *i* and *j* to be the indices of patches. The pairwise adjacent patches are defined as $\langle i,j \rangle \in$ H. H is a set including all the pairwise patches with overlap region. Each patch *i* comes from database image index f^i. $I^i_{f^i} \cap I^j_{f^j}$ is the overlap region of patch *i* and patch *j*. If the overlap regions are too small, the result may be easily affected by the outliers like noise.

TABLE II

SEAMLESS PATCH STITCHING SCHEME (PART I)

1. Apply chromatic gain compensation
   a. Expand each patch region to form overlap regions with every adjacent patches.
   b. Find the dominate RGB cluster center for each patch on both input image and piece-up image.
   c. Compute the initial gains for each patch by scaling dominate input patch color by dominate piece-up patch color.
   d. Use the initial gains find in step c and randomly chosen multiple guesses of gain in the range of (0, 2] to start the iteration of step e.
   e. Iteratively finds the minimum sum of error function as Equation (4-10) by optimizing the gain value by L-M method.
   f. After finding the minimum sum of error function of step d and step e, the variable gain is apply to each patch to adjust the color of each patch.
2. Apply multi-level blending

Next, we find the dominate RGB cluster center for each patch by clustering the RGB color of each pixel in each patch. We do this process on both input image and piece-up image. Piece-up image for now is the pieced up face by multi-label optimization in previous step. Figure 4.10 is an example for description. Looking at only one patch at a time, the initial gain tries to make the out patch similar to input patch in color. Therefore, we compute the initial gains for each patch by scaling dominate input patch color by dominate piece-up patch color. The error function is the sum of gain normalized intensity errors for all overlapping pixels:

$$E = \frac{\sum_{i=1}^{n} \sum_{\langle i,j \rangle \in H \cap i<j} \sum_{p \in I^{i}_{f^{i}} \cap I^{j}_{f^{j}}} (g_i I_{f^{i}}(p) - g_j I_{f^{j}}(p))^2}{m} \tag{4-7}$$

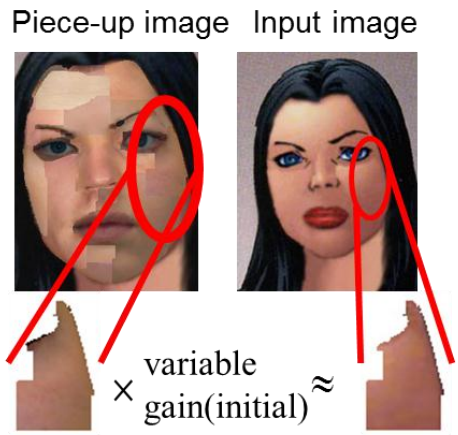


Figure 4.10 Simple case for dominate RGB color and initial guess of variable gain

The variable n stands for total n patches and *j* is the adjacent patch index of patch *i*. The parameter *p* is the pixel in the overlapping region. The variable g_i is the intensity gain for patch *i*. We put multiple guesses for our gain value. One of the multiple guesses remains for the initial gain we find in previous step, others are randomly chosen in the range of (0, 2]. We use these multiple guesses to start the iteration. The variable *m* is the number of pixels in the entire overlapping region, used for normalizing.

We optimize the gain value by Levenberg-Marquardt method (L-M method). It updates the parameter through first deriving the partial differentiation of each gain and Jacobian matrix of *E*, which is from Equation (4-7).

$$J_E = \left[\frac{\partial E}{\partial g_i} \quad \cdots \quad \frac{\partial E}{\partial g_n} \right] \tag{4-8}$$

Equation (4-9) is the parameter's delta movement of L-M method. *I* is the identity matrix and μ is a small constant to prevent the diagonal of matrix from zero. The unknown variable δx is the shift vector of the variable gain. After the new gain value derived, we can go back to Equation (4-7) to iteratively find the optimization of error function.

$$(J_E^T J_E + \mu I)\delta x = -J_E^T E \tag{4-9}$$

We add another constraint to the error function in Equation (4-7). Equation (4-10) is the complete form of our error function. *n* patches have *n* different gain variables. For each gain *i*, compare it to the initial value of gain *i*. We do not want our gain to be too different to the initial state, so we set up this penalty term. The variable *r* is the number of color channel and *n* is the number of patches. θ is the weight of each error term.

$$E = (1-\theta)\frac{\sum_{i=1}^{n}\sum_{\langle i,j\rangle\in H\cap i<j}\sum_{p\in I^i_{f^i}\cap I^j_{f^j}}(g_i I_{f^i}(p) - g_j I_{f^j}(p))^2}{m} + \theta\frac{\sum_{i=1}^{n}(g_i - \tilde{g}_i)^2}{r\cdot n} \tag{4-10}$$

After gain compensation finds the minimum sum of error function, the variable gain is apply to each patch to adjust the color of each patch and lessen the gaps of patch boundaries. The above mentioned energy function is written in a form for a single gain variable for each patch due to easy explanation. In our method, we use three gains for each patch. Figure 4.11 shows the best gain value we retrieved after optimization and apply to each patches to adjust the intensity for each patch.

Figure 4.13 shows the results for gain compensation. Image (a) is without gain compensation. Image (b) presents the result of gain compensation. It is clearly that gain compensation narrow down the gaps between patches.

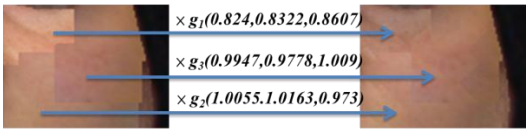


Figure 4.11 An example of apply the best gain value retrieved after optimization to each patches to adjust the intensity for each patch.

TABLE III

SEAMLESS PATCH STITCHING SCHEME (PART II)

1. Apply chromatic gain compensation
2. Apply multi-level blending
    a. Find the center point for each patch by the mean value of each pixel's coordinate.
    b. Use Gaussian low-pass filter to divide the database images into multi-level.
    c. For each pixel in overlap region, blend the low-pass information by Equation (4-11).
    d. Put back the high-pass information.

Multi-level Blending

Even though we apply the gain compensation method to draw closer the color of adjacent patches, the boundaries between them may still not be seamless. So we further use multi-level blending to smooth the patch boundaries. The method of multi-level blending is briefly described in Table III.

For the overlapping regions defined in chromatic gain compensation, we blend the color in the overlap region of each pixel *t*. For the first step, we find the center point for each patch by the mean value of each pixel's coordinate. We then use Gaussian low-pass filter to divide the images into multi-level. To the low-pass information we blend it by Equation (4-11).

$$I'_{t=(i,j)} = \sum_{q \in G_t} w_{f^q}(t) I'_{f^q}(t) \tag{4-11}$$

$$\hat{w}_q(t) = \frac{1}{\left\| h_q - t \right\|^2 + c} \tag{4-12}$$

$$w_q(t) = \frac{\hat{w}_q(t)}{\sum_{v \in G_t} \hat{w}_v(t)} \tag{4-13}$$

The set G_t include the patch index that overlaps at coordinate t which is (i, j). The variable q is the patch indices that overlap location t. Equation (4-13) is for normalizing the weight. For each pixel t in the image, we blend its new color by all the overlapping patches q at pixel t. The new color are derived by the color of coordinate (i,j) in patch q with the weight $w_q(t)$. The weight is calculated by the inverse proportion to the distance of the center coordinates of patch q and the coordinate of pixel t. The variable h_q in Equation (4-12) is the center coordinates for patch q.

Figure 4.12 shows an example of multi-level blending. We blend the low-pass image patches to smooth the colors of gaps and keep the high resolution details in the high-pass parts. We can retrieve the facial details by put back the high-pass image at the last step. Two-level blending is sufficient in our cases. Figure 4.14 (a) and (b) shows the image of before and after Multi-level blending.

Figure 4.12 An example of multi-level blending

(a) (b)

Figure 4.13 (a) Before chromatic gain compensation (b) After chromatic gain compensation

(a) (b)

Figure 4.14 (a) Before multi-level blending (b) After multi-level blending

Chapter 5

Experiment and Results

In this chapter, we present the experiment and results of our method, and other related setting, environment and experiences. With the results, we further compare our result with those generated by other existing methods. In the end of this chapter, we discuss the advantages and limitation of these works.

5.1 Experiment Setup

The image database and input images we use are collected from internet. We gathered photographs that are publicly shared from internet photo albums and image searching engines. Female photographs are our main targets but male faces can also be easily used with our method. For face and neck database, there are 50 photographs. Hair database has 94 photographs due to the variety of hair style and color, so we gathered more for hair database.

The 61 feature points we use for face warping is shown in Figure 3.1 (c). We use other 6 points to frame the region of the neck and 1 chin points from the 61 feature points to adjust the aspect.

There are few libraries used in our method. STASM [12] library is used for feature point extraction. For solving multi-label optimization, we use the gco-v3.0 library provided by author [14]. Our algorithm is written in C and OpenCV library. We run our algorithm on a

computer with Intel Core i5-2400 CPU (3.10GHz) and 3.49 GB RAM. In our experiment, we take 13 minutes to computing a 400x600 resolution image.

5.2 Results

In this section, we present the results of our algorithm. To demonstrate the effects of each component, we also show some other results without the functional terms we add to our algorithm. Discussion about the results is also presented in this section.

Our Results

The results for our algorithm are in Figure 5.1-1~Figure 5.1-7. We apply our methods on cartoon images. Parts of the characters are from famous comics and some cartoon images are retrieved on internet. These cartoon images have various painting styles. Another type of painting we tried is the portrait painting of ancient art work. The results are shown in Figure 5.2-1~Figure 5.2-5. We also tried our method on sketch paintings, in Figure 5.3-1~Figure 5.3-5.

The outcomes in Figure 5.1-1~5.3-5 show satisfying photorealistic results, especially for cartoon image and sketch paintings. The results for portrait paintings such as images in Figure 5.2-1~Figure 5.2-5 is not as surprising as the other types of painting because the portrait painting is realistic in the first place. However, comparing to the original portrait paintings, our results shows more details of human skin and facial texture. For the sketch paintings, we can even use gray level intensity for our data term and use 3 channel color intensity for the smooth term. In this way, the similarity and smoothness can be kept and the results can be colored instead of monochrome.

(a) (b) (c)

Figure 5.4 (a) The input portrait painting (b) The halfway result using Poisson image editing (c) The results using Poisson image editing

Poisson Image Editing

We have tried to use Poisson image editing [5] to solve gaps between patch boundaries. However, this method is easily influenced by large illumination changes and does not match our expectation of the result. Figure 5.4 shows the progressive results for using Poisson image cloning. Image (a) is the original input painting. After we seamlessly cloned a few patches, the results seams fine but not very clear, such as in image (b). For image (c), it adds all the patches, but the result is blurred due to the inherent problem of Poisson Editing. We believe the reason is that we had many small patches and they are irregular shaped. Each time one patch is stitched with Poisson Editing, the boundary of the patch is smoothed for one time. After merging more patches, the image becomes seriously blurred. Therefore, instead of using Poisson image editing, we propose using gain compensation to lessen the color differences between adjacent patches.

Edge Term

Figure 5.5 shows the results for the penalty term. Image (c) is the results with edge

penalty term and image (d) show the results without edge penalty term. From image (a) the original input sketch paint, we can see that the lower lip has shining reflections on it. On the edge maps also show the edge of the reflections. Therefore our result finds shining patches for the lower lip based on the edge. If the edge term is turned off, with only the intensity similarity, the shining reflection is not shown in the results.

(a) (b) (c) (d)

Figure 5.5 (a) The input sketch painting (b) The edge from Canny edge detector (c) Result with edge penalty term (d) Without edge penalty term

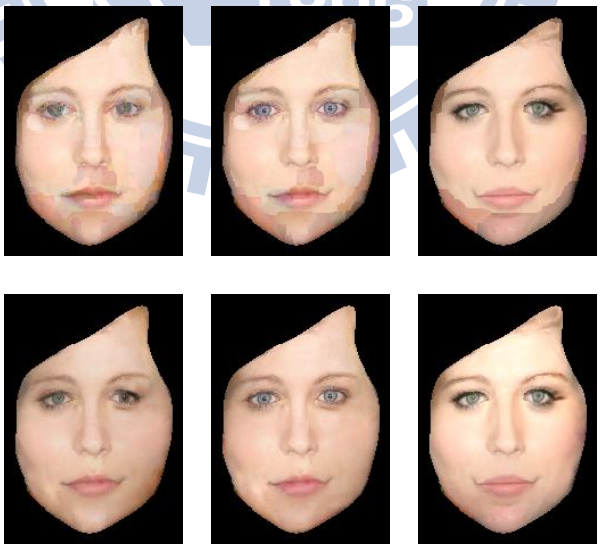
(a) (b)

Figure 5.6 (a) With edge term (b) Without edge term

Figure 4.9 (a) in the previous chapter, shows the result of the discontinuity of the lip if edge term is not added. Figure 5.6 also shows the discontinuity on the lip and eyes. The lip for Figure 5.6 (b) does not seem smooth and continuous as in image (a). The eye for Figure 5.6 (b) appears a segment of outer corner of the eye, which is should appears.

Smooth term

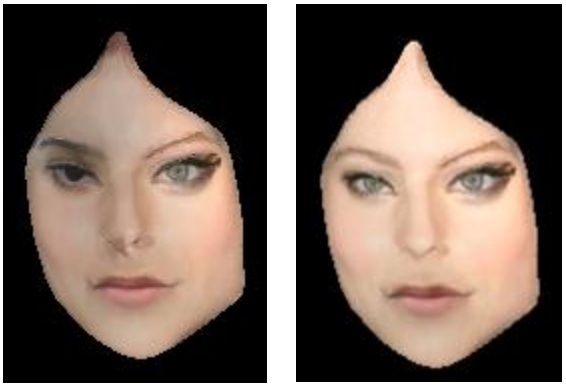
The variable λ determines the size of the patches. Here we show some results of different λ. Figure 5.7 shows when λ is small the patches are smaller with the face is similar to the input. However, if the patches are too small it sometimes may affect the result for outlier pixels. When λ is larger, the patch size is bigger and it is more smooth. However, it lost the similarity to the input. So how to decide the value of λ is an interesting question. It depends on user's requirement. In our cases, most of the image use 0.1~0.3 as the smoothness term weight. In most of the cases, 0.2 return satisfying results, though sometimes other values return better result.



λ=0.1 λ=0.2 λ=0.3

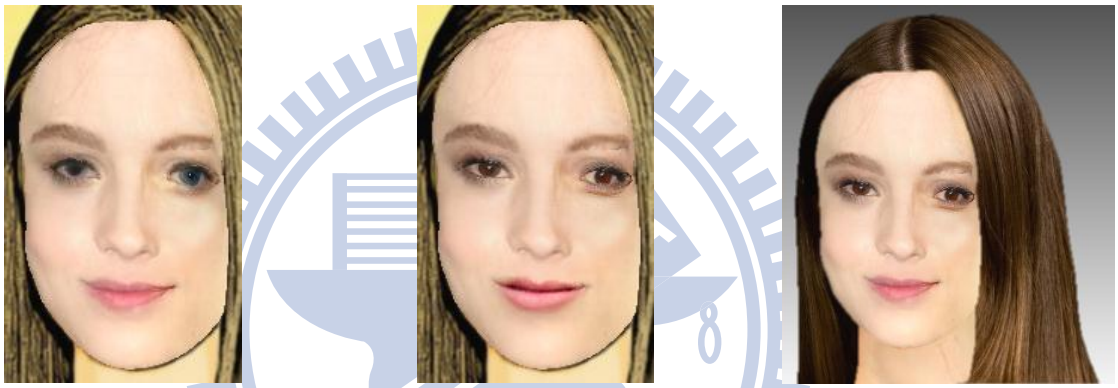
Figure 5.7 The effects of different λ(*[3])

(*[3])The upper row is piece-up image without seamless stitching. The lower row is the result after seamless stitching from upper row.



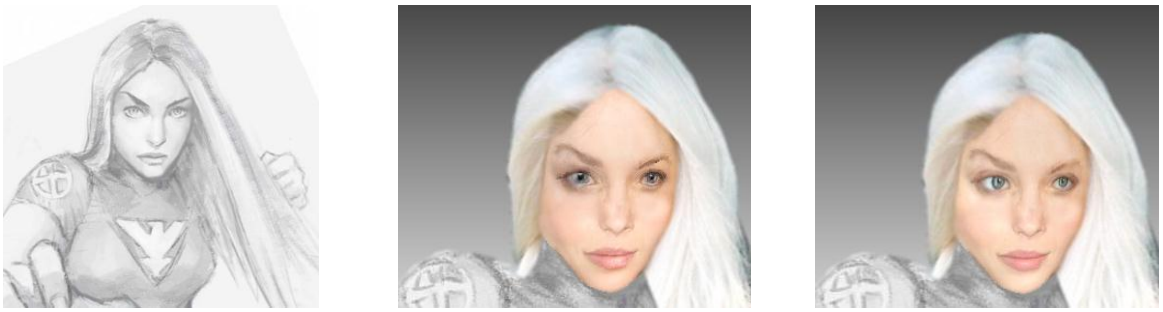
(a) (b)

Figure 5.8 (a) Result without symmetry term (b) Result with symmetry term



(a) (b) (c)

Figure 5.9 (a) Without option eyes and mouth completeness (b) Without option eyes and mouth completeness (c) The result with complete eyes and blended mouth



(a) (b) (c)

Figure 5.10 (a) The input image for (b) and (c)

(b) Without completeness (c) With completeness

Symmetry term

In Figure 5.8, image (a) is the results without the symmetry term and image (b) is with the symmetry term. If there is no limitation, there is a high probability that the eyes and eyebrows may choose from different sources. But human eyes are sensitive to symmetry of faces, so symmetry term is added to our energy function.

Identical source of eyes or mouth

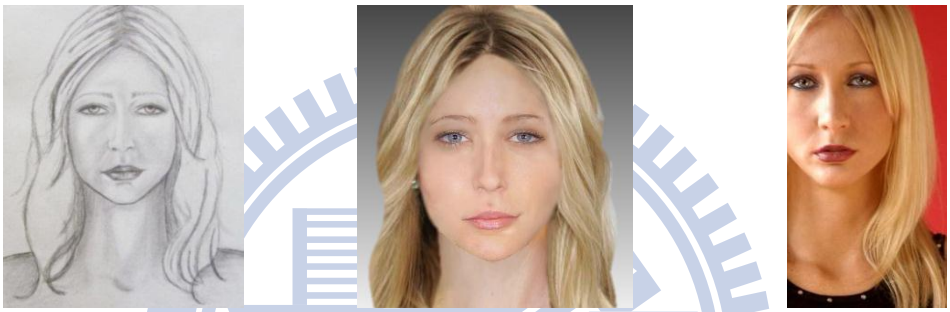
There is a large rage of color in eyes, so occasionally blending methods makes the eyes unreal. We provide an option for the user to choose whether they want the whole eyes or mouth is extracted from the same source. We call it the completeness constraint. The completeness is for the eye and mouth.

For Figure 5.9, image (a) is the results without eyes and mouth completeness and image (b) is the results with completeness for both eyes and mouth. It is obvious that the eyes in image (a) after blending looks more similar to the input image but not realistic and image (b) has more realistic eyes. So the user can choose the option for eyes completeness for better result. For the mouth in image (b), even though it is completely from the same source but it is not as good as the result for mouth in image (a). The mouth in image (a) is more realistic and natural. So in this case, the user can choose not to limit the completeness for the mouth. Figure 5.10 (b) and (c) give another example for this case. The eyes without completeness are more similar to the input image, and the result is also satisfying. However, it lacks the reality of human eyes.

Ground truth

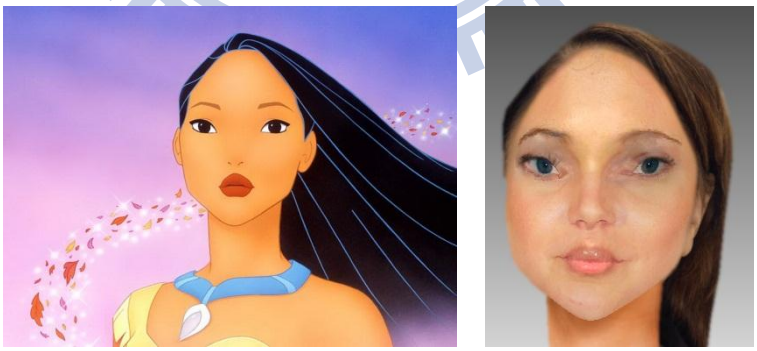
Image (b) is the result of our algorithm as the sketch painting of image (a) as the input

image. Figure 5.11 (a) is the sketch painting that the artist draw from photograph (c), even though the drawing is not exactly the same as the photograph, we can still seem photograph (c) as our ground truth to compare with. Our result and input sketch painting is quite similar. However, the similarity between our result and ground truth is adequate. The reason for this is because the sketch painting and ground truth is not exactly the same. If the similarity between sketch painting and ground truth is higher, our result can be more similar to the ground truth photograph.



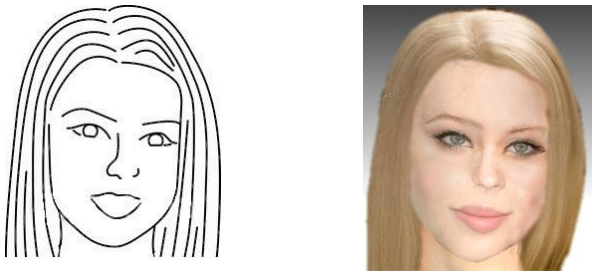
(a) (b) (c)

Figure 5.11 (a) Input image (sketch painting) (b) Our Result (c) The photograph (ground truth)



(a) (b)

Figure 5.12 (a) Pocahontas (b) Our Result



(a) (b)

Figure 5.13 (a) Simple line drawing (b) Our Result

Special Case

Figure 5.12 (a) is the input cartoon image and (b) is the result we generated. This case is special because though the result looks satisfying but it is not really similar to Pocahontas. The reason is we do not have Indian photographs in our database.

If we use an input image that has only very simple and easy line drawing, we can get the result as in Figure 5.13 (b). When very simple line drawing are used, it means that we have very little information even lesser than the cartoon images. In this case, our result still gets satisfactory image appearance quality. However, the geometry of our result is limited to the original input image's geometry. The geometry of our result is similar to human or not is depending on the input geometry.

5.3 Comparison

In this section, we show the output image of different methods, **best-matched face**, **best-matched-regions** and **regular-patch-based** method. Best-matched face method is to choose the most similar warped image from database by comparing color and edge of the whole face. Best-matched-regions method is to select the most similar skin by color and edge, and use Poisson image editing to synthesis the facial features from each similar sources. Here

we limit the symmetry, which the two eyes are from the same source and so is the eyebrows. Regular-patch-based method is the patch finding and stitching method proposed in Visio-lization [4]. The results for these comparative methods are also shown in Figure 5.1-1~5.3-5. Below, we only pick some images from our results to show the comparing.

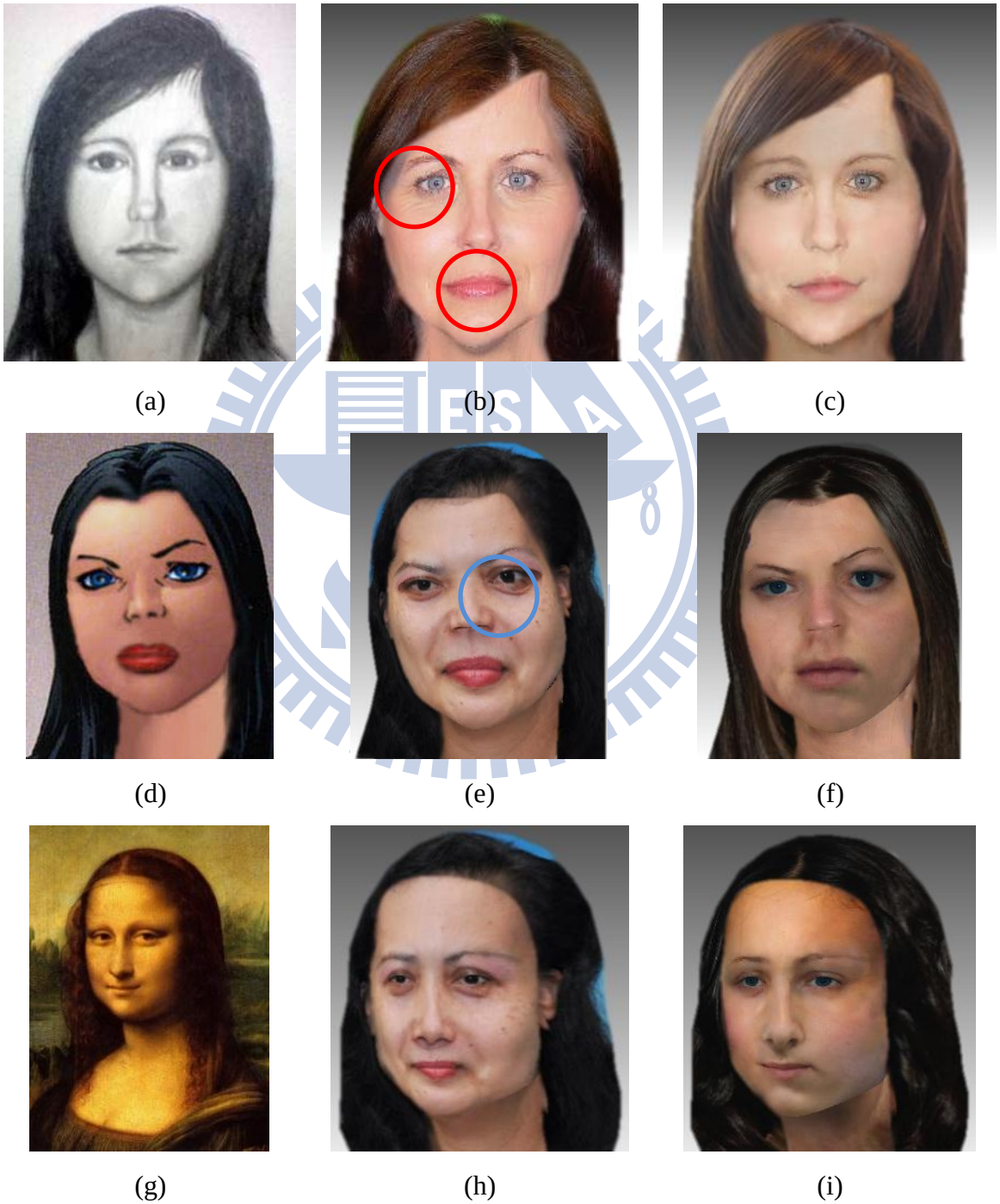
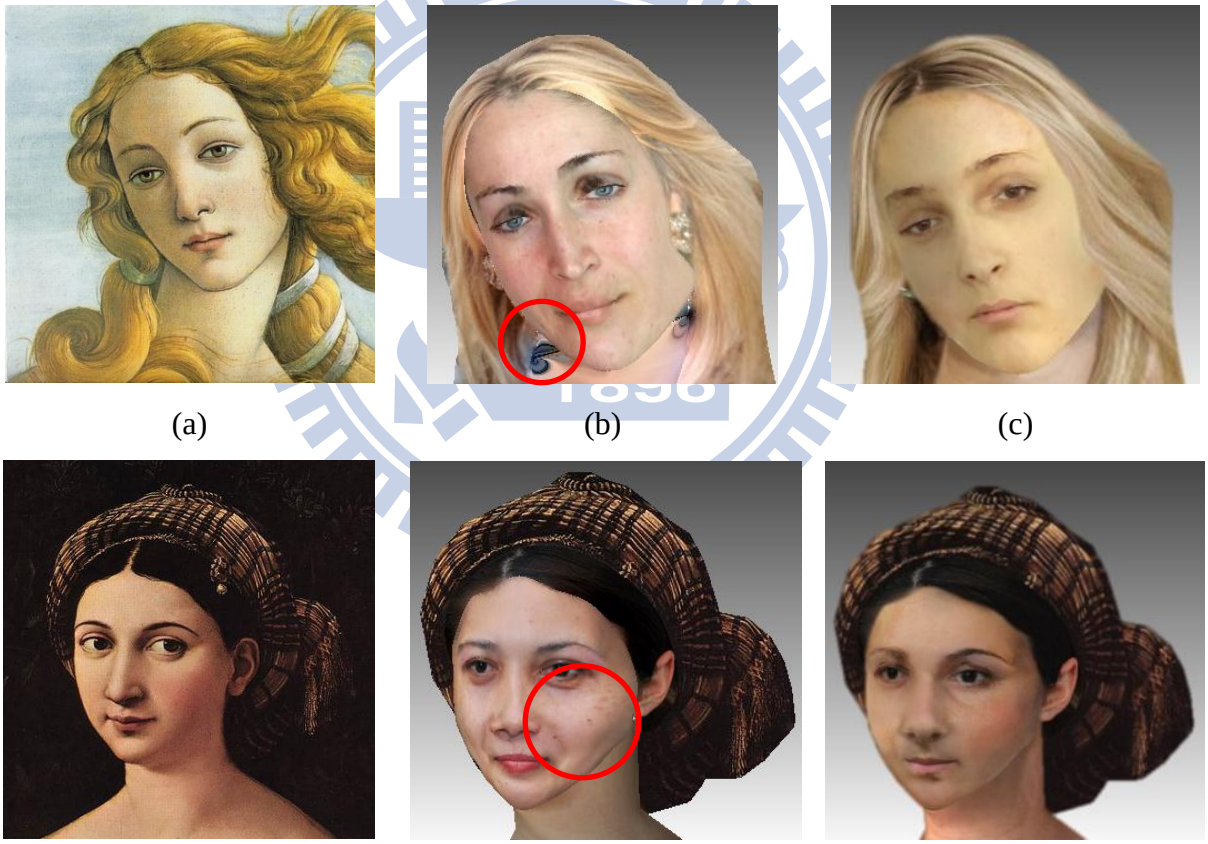


Figure 5.14 (a)(d)(g) Input image (b)(e)(h) Best-matched face (c)(f)(i) Our Result

Best-matched face

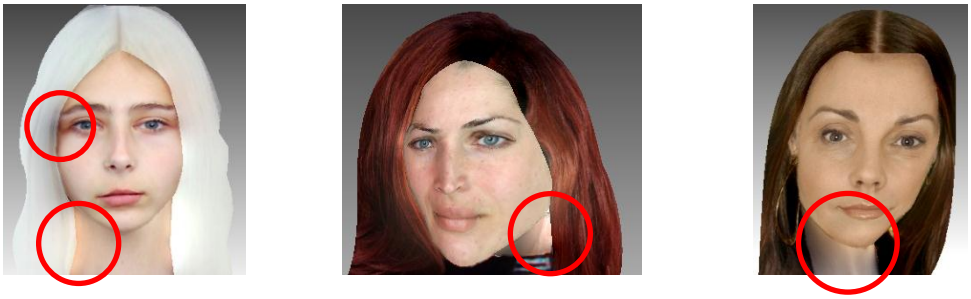
Figure 5.14 (b) is the result for the most similar directly warped image. Although it seems similar, but it can be seen from the red circle that the original input image does not appear some many wrinkles around eyes which this is not fit to the input image. The next is the shape of the mouth does not fit to the input image mouth. For the blue circle in image (e), from the original image there is a strong edge shows the wrinkle but best-matched face lost this important features. Sometime, as in Figure 5.14 (h), the result may be significantly different from the input image.



(a) (b) (c)

(d) (e) (f)

Figure 5.15 (a)(d) Input image (b)(e) Best-matched-regions method (c)(f) Our Result



(a) (b) (c)

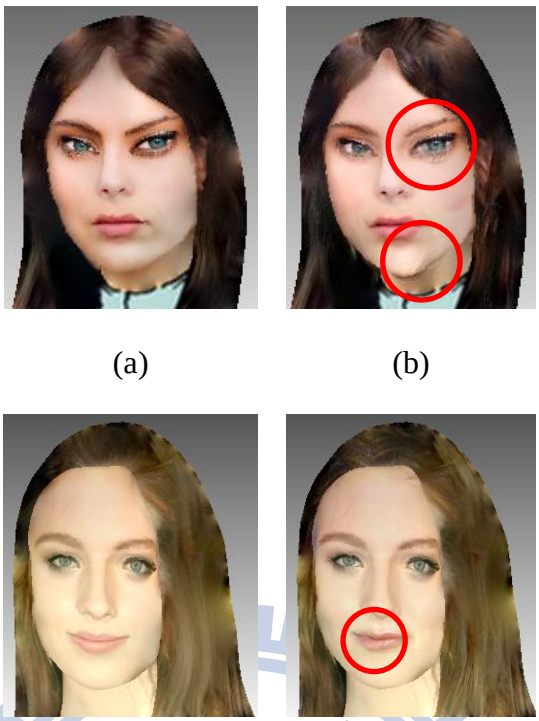
Figure 5.16 Some examples of artifacts caused by Poisson image editing

Best-matched-regions

The red circle in Figure 5.15 (b) presents a problem occurs when we define skin as a region. Occasionally, when aspect changes differently, the transformation become difficult and may transform error. In this case, region-based method cannot avoid from retrieving this error patches. However, with our method these erring pixels can be evaded. Image (d) shows that the blusher is obvious on the original face, yet it is not presented on the cheek of image (e). The Poisson image editing may cause artifacts like in Figure 5.16.

Regular-patch-based method

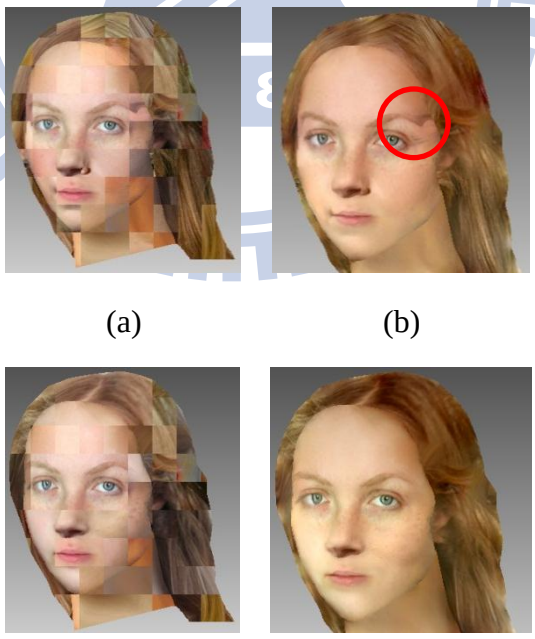
The regular-patch-based method in Visio-lization [4] uses feature vectors of RGB. We here use RGB+E (Edge) to be our feature vectors, which also slightly adjust this method to return a better result. The results in our comparison are all implement with RGBE. We show the difference between the results when we add edge to be our feature vector in Figure 5.17. It showed some discontinuity in the results at where the red circle marked. Figure 5.18 shows how the feature vector E helped from retrieving a bad patch.



(a) (b)

(c) (d)

Figure 5.17 (a)(c) The results with feature vectors of RGBE (b)(d) The results with feature vectors of RGB



(a) (b)

(c) (d)

Figure 5.18 (a)(c) The regular patches (b) The result of RGB after Poisson image editing (d) The result of RGBE after Poisson image editing

The method of regular-patch-based method can synthesis realistic and novel faces, but important features may be lost. We can see from the experiment in Figure 5.19 that images (a)~(c) is the original experiment set and images (d)~(f) is the experiment set we add a small but important feature. For image (d), we add a small but very important feature which is a mole on the face. We can see that with our arbitrary shape and size patches, the important feature is preserved in our result but not in regular-patch-based method.

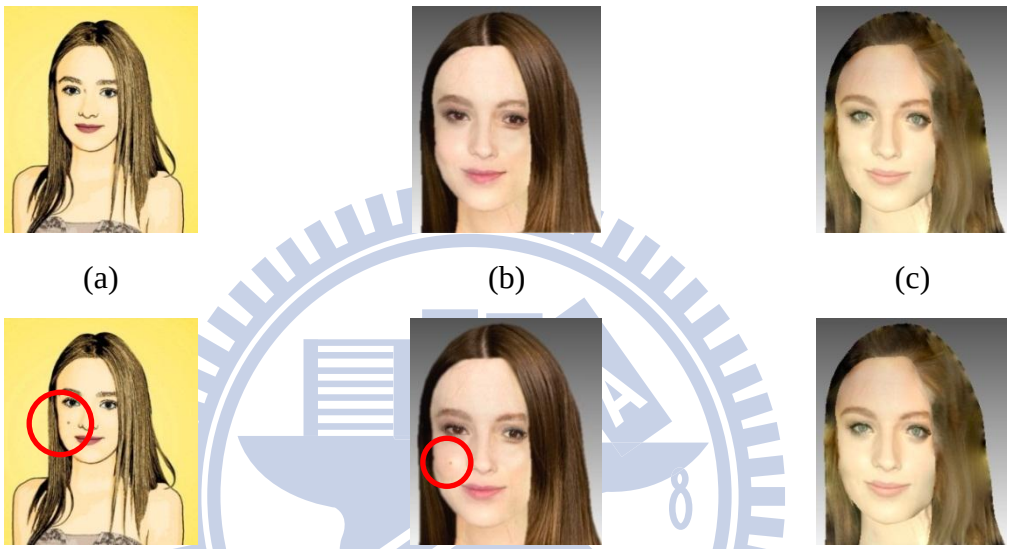


Figure 5.19 (a) The input image (d) We draw a mole (important but small feature) on the input face (b)(e) Our results (c)(f) Regular-patch-based method results

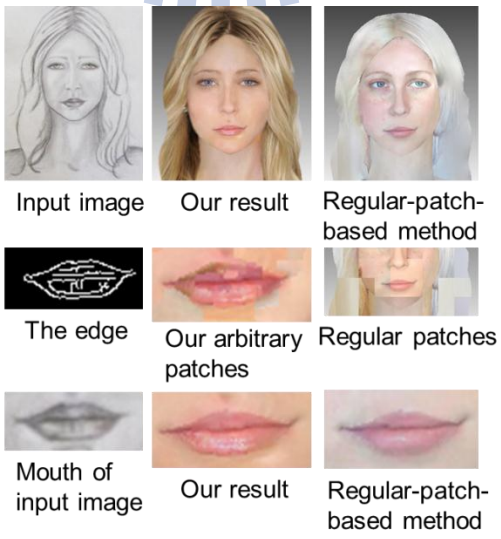
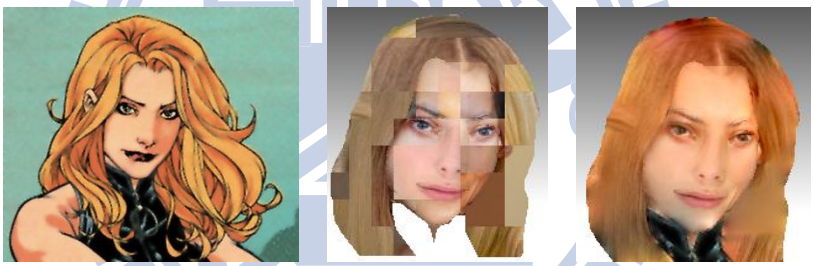


Figure 5.20 Another example for preserving the important feature

Figure 5.20 shows another example of our algorithm can preserve important features well than the method of regular-patch-based. We can see that with our arbitrary patches and edge penalty we can synthesize the luster of the lips. However, regular-patch-based method cannot synthesize this effect due to the patch size and shape.

The results for regular-patch-based method can also affected by Poisson image editing. Like in Figure 5.21, the boundary constraint comes from the input cartoon image and the color crossing boundaries differs a lot which makes the cartoon hair color propagate all the way into the face region. Figure 5.22 shows a different kind of artifact caused by the regular size and shape patches. Figure 5.23 (c) shows the discontinuity caused by regular-patch-based method. This problem also can be prevented by our edge penalty term.

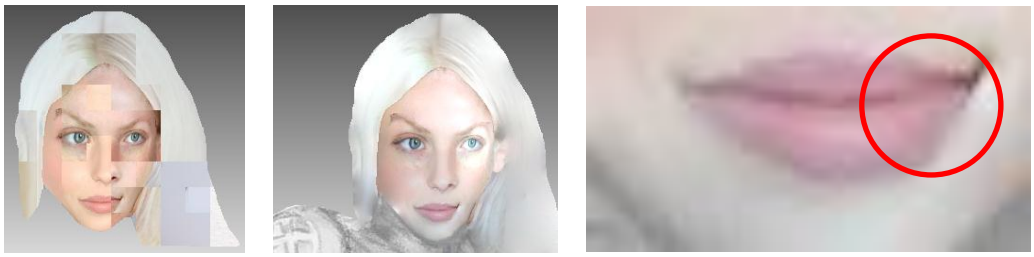


(a) (b) (c)

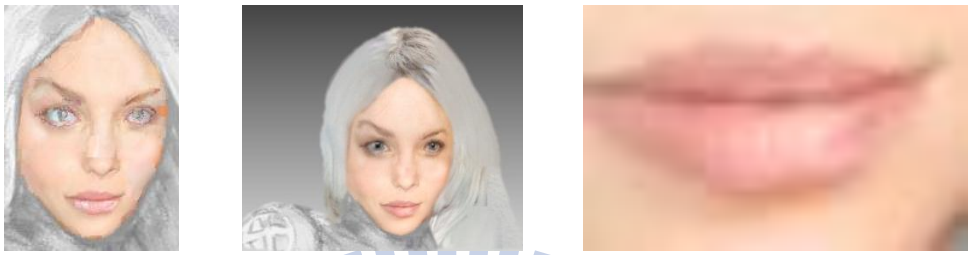
Figure 5.21 Visual artifacts of regular-patch-based method

(a) (b) (c) (e)

Figure 5.22 Visual artifacts (a) Regular patches (b) The close-up for image (a) (c) The result of regular-patch-based method (e) Our result



(a) (b) (c)



(d) (e) (f)

Figure 5.23 (a) Regular patches (d) The piece-up image of our algorithm (b) The result of regular-patch-based method (c) Close-up of image (b) (discontinuity) (e) Our result (f) Close-up of image (e)

Figure 5.24 The face mostly comes from the same source after limiting the symmetry for eyes and eyebrows (Regular-patch-based-method)

Figure 5.25 Unsatisfactory results for sketch painting (Regular-patch-based-method)

We also found out some other problems for regular-patch-based method. As in Figure 5.24, the algorithm in Visio-lization [4] limits the symmetry for the faces. Here we limit the eyes and eyebrows symmetry. We can see that the patches of face will return mostly from the same source, which lower the novelty. Another problem occur for monochrome paintings, because regular-patch-based method used Poisson image editing to seamless synthesize the results the boundary constraint comes from the input image which cause the result to be a weird skin color. It can be seen from the example in Figure 5.25.

5.4 Discussion

The most simple and easy method is to choose the most similar faces by the difference of color and edge information from warped faces. The result might be broadly similar. But the important features may be very different. And sometimes the results may look significantly different from the input image.

Best-matched-region method is to separately find the similar facial feature and skin. Once the skin is retrieved from a single source, the chosen facial features are synthesized by Poisson image editing. In this way, the facial features may be similar to the input image but the important feature on the skin is disappeared. The boundary of skin and hair might cause an artifact by Poisson image when the hair and skin color is very different.

Regular-patch-based image synthesis is part of the method in Visio-lization [4]. Their algorithm using regular-patch-based image synthesis performs efficiently. The synthesized faces look realistic and novel when a considerable large photo database is used. However, a patch is similar to another patch does not means the important feature including is also similar. If the size of the patch is not well determined, the source of adjacent patches may be quite

different and it result in visual artifacts. If the patch size is too small the discontinuity may occur.

Others like Suo *et al.* [22] proposed a statistic-based image synthesizing method. The result of this method is impressive well and the wrinkles and other features synthesis look realistic. However it needs more accurate warping and segmentation. And it can only process with the frontal faces. Similar to regular-patch method, it needs a large database to comprise high varieties within each region.

Bitouk *et al.* [3] proposed "Face Swapping" to replace human face by similar faces. The similar face is ranked by pose, color, lighting and blending cost. This method is used as replacing the whole face to another so it cannot preserve the important features and cannot generate a novel face. While replacing the whole face to cartoon image, the Poisson Editing may cause artifact due to the simple color of the cartoon character skin.

Our method synthesizes satisfying results. The results are realistic and novel. Figure 5.26 shows the output image of multi-label optimization is stitched by 38 different sources. Image (c) is the all the source used in this case. Image (b) shows the different color patch represent different sources.

Our method can synthesize most of the important feature in the input image. Under the premise, this feature is shown in the collection of our database and the edge of this feature can be found by Canny edge detector. With our edge term penalty and the edge map of input image, we can prevent the discontinuity in most of the cases.

Another advantage of our algorithm is that we can apply on sketch painting like monochromes. Our method generates satisfying colored images for these sketch paintings. If

Poisson image editing is used, the results are interference by the input image gray level colors. The results are shown in Figure 5.3-1~Figure 5.3-5. Even though the patches are cloned seamlessly by Poisson image editing, it lost the original color. Instead, our method preserves the image color of source in a reasonable condition.

Due to our flexibility of patch size and shape, we can prevent from finding bad warping patches. Sometimes when the aspect changes significantly, the warping become difficult and bad warping may occur. With our arbitrary-shape patches, the method can alleviate these bad warping regions.

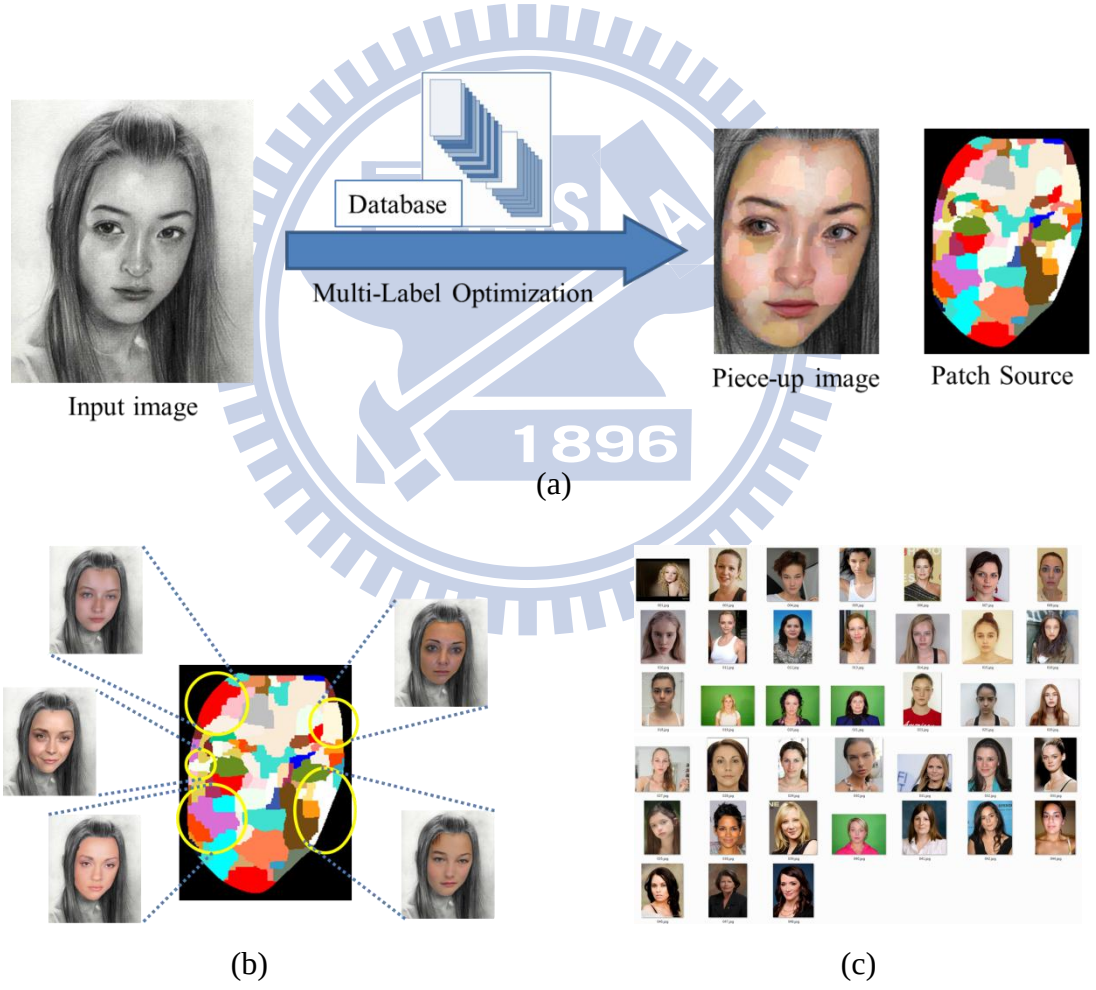


Figure 5.26 (a) The color patches are the different source to form our output (b) Different colors represent different sources (c) The output of image (a) used 38 sources from database

	Best-matched face	Best-matched-regions	Regular-patch based(*4)	Statistical-based	Face Swapping	Our method
Amount of database			Large	Large	Large	Small
Realistic for cartoon image	Yes	Yes	Yes	Yes	No	Yes
Important Features	No	No	Yes(*1)	Yes	No	Yes(*2)
Monochrome to colored image	Yes	No	No	No	No	Yes
Novel Face	No	No	Yes	Yes	No	Yes
Computing time	Fast	Fast	Fast	Fast	Fast	Relatively slow(*3)
Continuity	Yes	Yes	No	Yes	Yes	Yes
Previous Training	No	No	No	Yes	No	No

Table 5.1 The comparative table with our method and other methods

*1 When the patch size is right.

*2 If the feature is able to be found by Canny edge detector.

*3 The computing time can be accelerate and shorten by parallel computation.

*4 Visio-lization [4]

◆ Talia al Ghul (DC Comics)

Figure 5.1-1 Comparison for realizing cartoon images

A	B	C
D	E	
F	G	

A is the original input image. B is the result for **best-matched face**. C is the result for **best-matched-region**. D and E are our results with different weight λ for the hair. F is the patches retrieved by **regular-patch-based** method using feature vector of RGBE. G is the result after Poisson image editing for image F.

◆ Jean Grey Phoenix X-men (Marvel Comics)

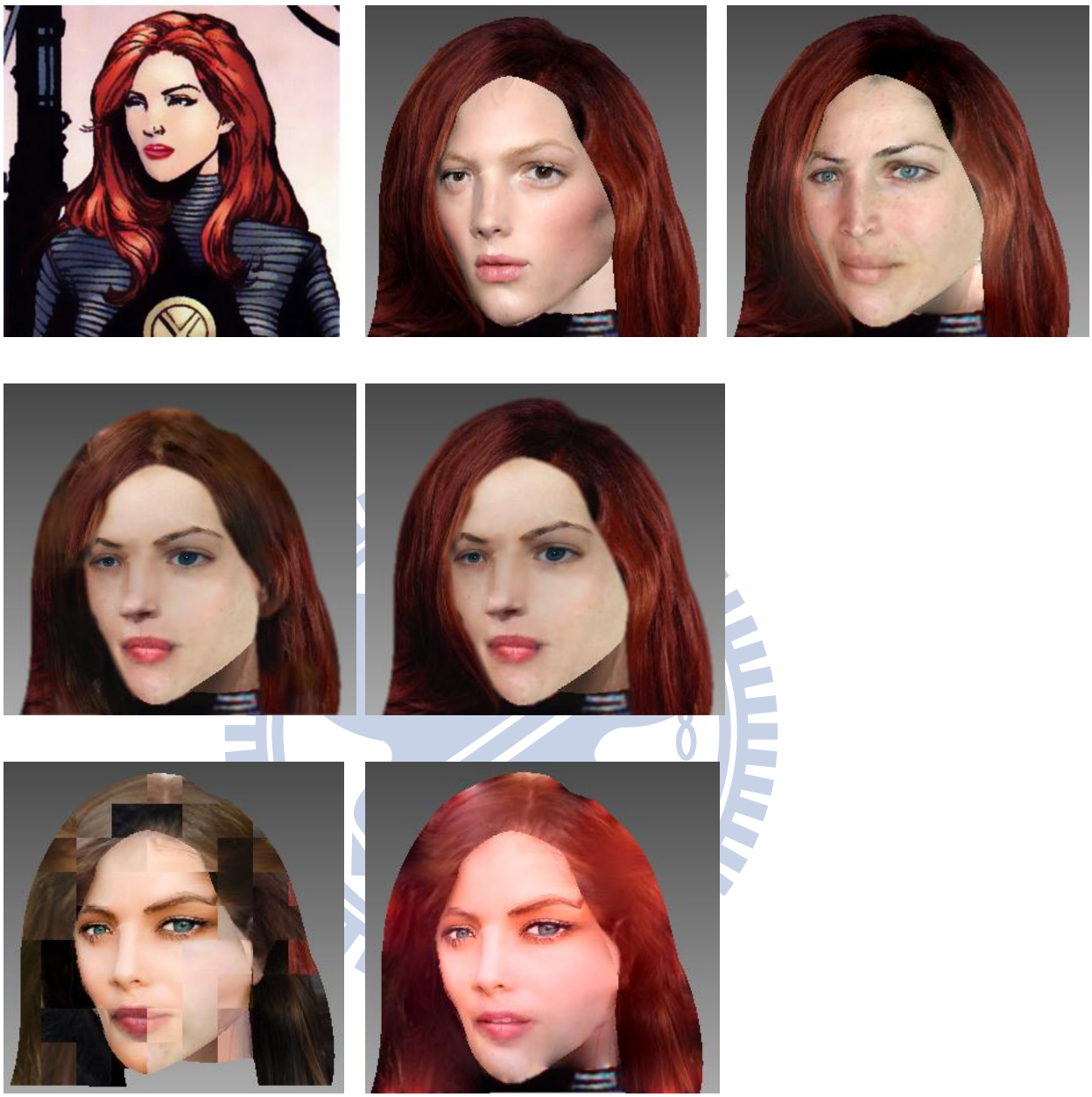


Figure 5.1-2 Comparison for realizing cartoon images

A	B	C
D	E	
F	G	

A is the original input image. B is the result for **best-matched face**. C is the result for **best-matched-region**. D and E are our results with different weight λ for the hair. F is the patches retrieved by **regular-patch-based** method using feature vector of RGBE. G is the result after Poisson image editing for image F.

◆ Black Canary (DC Comics)

Figure 5.1-3 Comparison for realizing cartoon images

A	B	C
D	E	
F	G	

A is the original input image. B is the result for **best-matched face**. C is the result for **best-matched-region**. D and E are our results with different weight λ for the hair. F is the patches retrieved by **regular-patch-based** method using feature vector of RGBE. G is the result after Poisson image editing for image F.

◆ Sara Pezzini (Image Comics/Top Cow)

Figure 5.1-4 Comparison for realizing cartoon images

A	B	C
D	E	
F	G	

A is the original input image. B is the result for **best-matched face**. C is the result for **best-matched-region**. D and E are our results with different weight λ for the hair. F is the patches retrieved by **regular-patch-based** method using feature vector of RGBE. G is the result after Poisson image editing for image F.

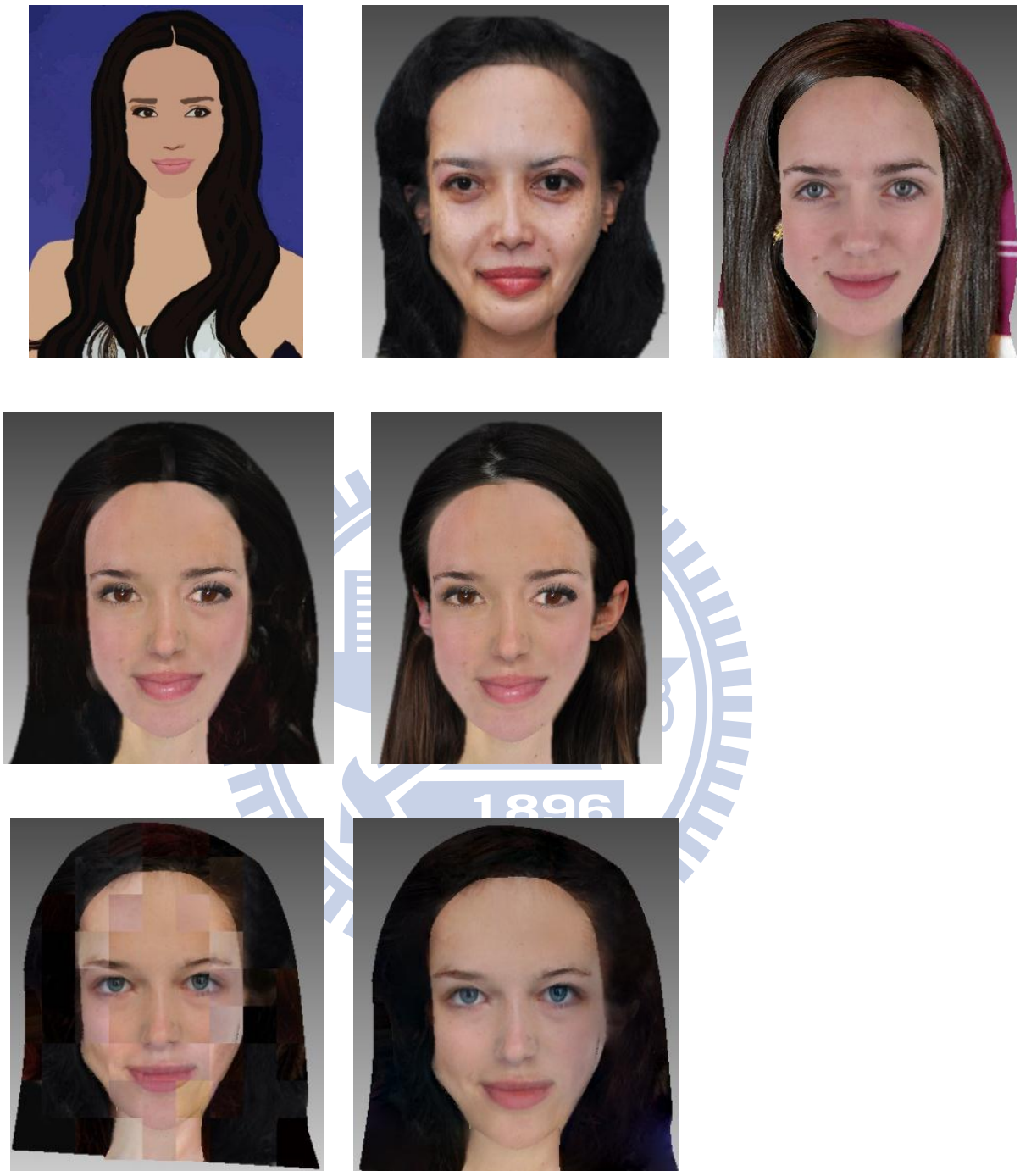


Figure 5.1-5 Comparison for realizing cartoon images

A	B	C
D	E	
F	G	

A is the original input image. B is the result for **best-matched face**. C is the result for **best-matched-region**. D and E are our results with different weight λ for the hair. F is the patches retrieved by **regular-patch-based** method using feature vector of RGBE. G is the result after Poisson image editing for image F.

◆ Dakota Fanning Cartoon

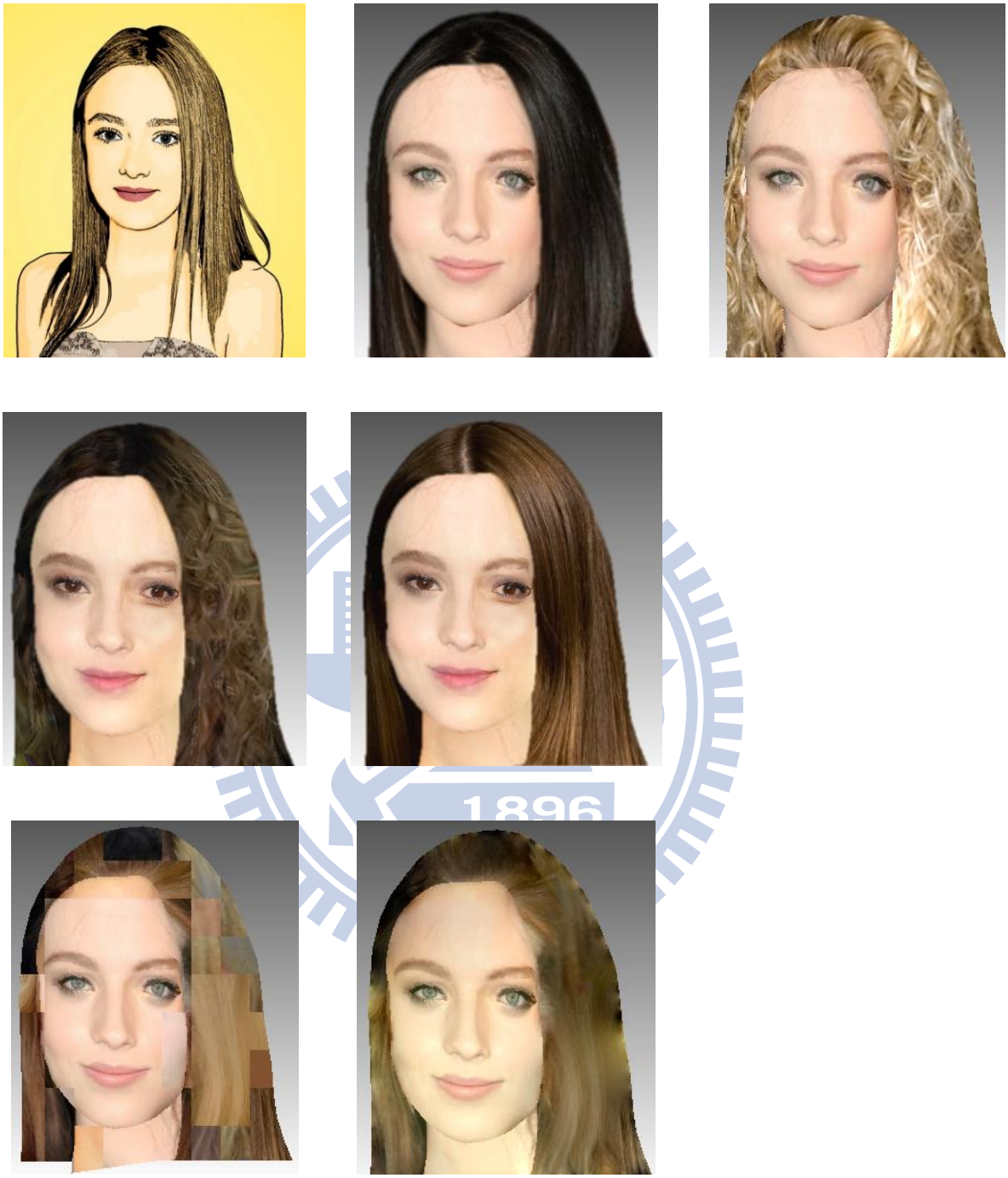


Figure 5.1-6 Comparison for realizing cartoon images

A	B	C
D	E	
F	G	

A is the original input image. B is the result for **best-matched face**. C is the result for **best-matched-region**. D and E are our results with different weight λ for the hair. F is the patches retrieved by **regular-patch-based** method using feature vector of RGBE. G is the result after Poisson image editing for image F.

◆ Emma Watson Cartoon

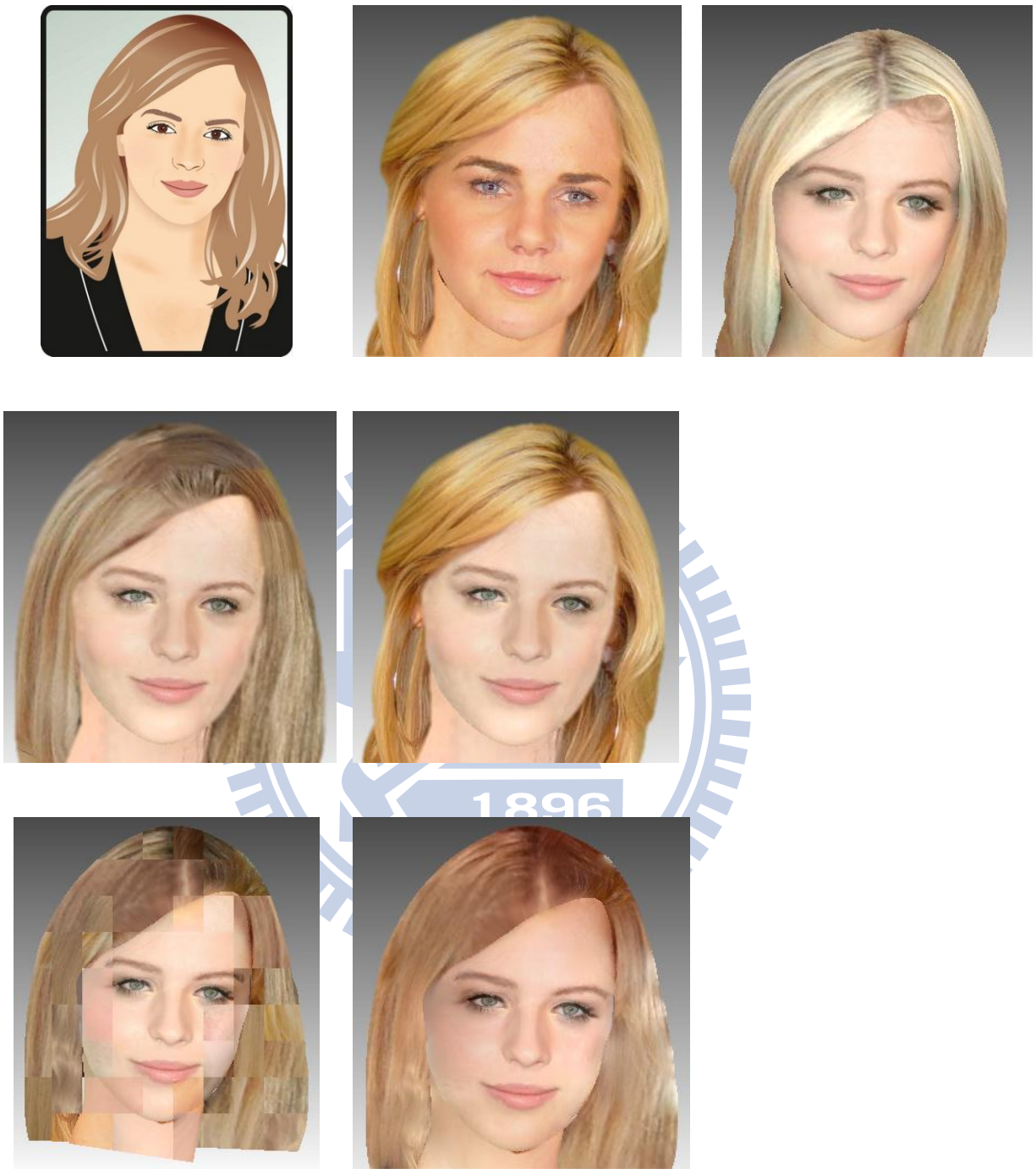


Figure 5.1-7 Comparison for realizing cartoon images

A	B	C
D	E	
F	G	

A is the original input image. B is the result for **best-matched face**. C is the result for **best-matched-region**. D and E are our results with different weight λ for the hair. F is the patches retrieved by **regular-patch-based** method using feature vector of RGBE. G is the result after Poisson image editing for image F.

◆ The Birth of Venus (Botticelli)

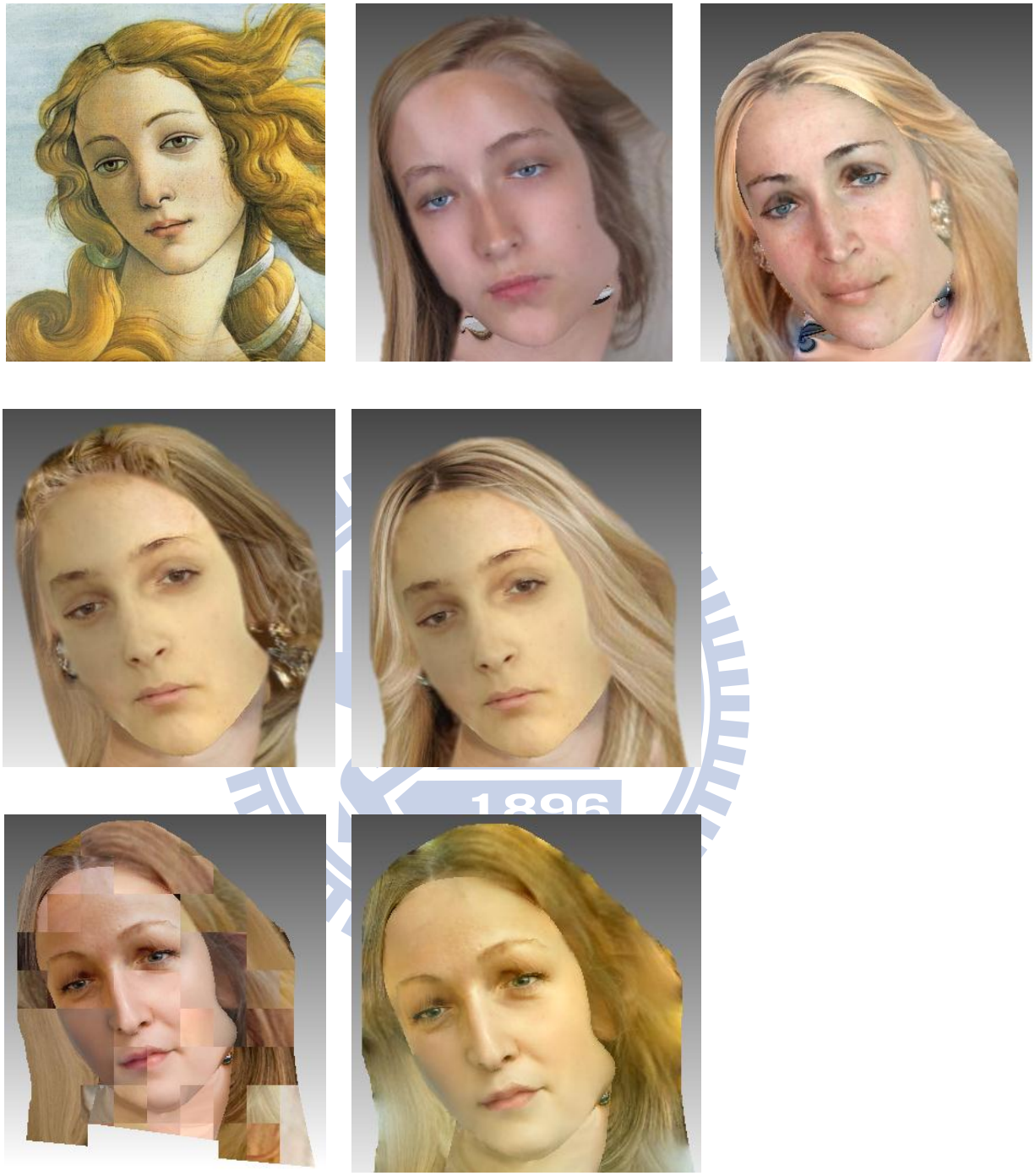


Figure 5.2-1 Comparison for realizing portrait paintings

A	B	C
D	E	
F	G	

A is the original input image. B is the result for **best-matched face**. C is the result for **best-matched-region**. D and E are our results with different weight λ for the hair. F is the patches retrieved by **regular-patch-based** method using feature vector of RGBE. G is the result after Poisson image editing for image F.

◆ Young Woman with Unicorn by Raphael

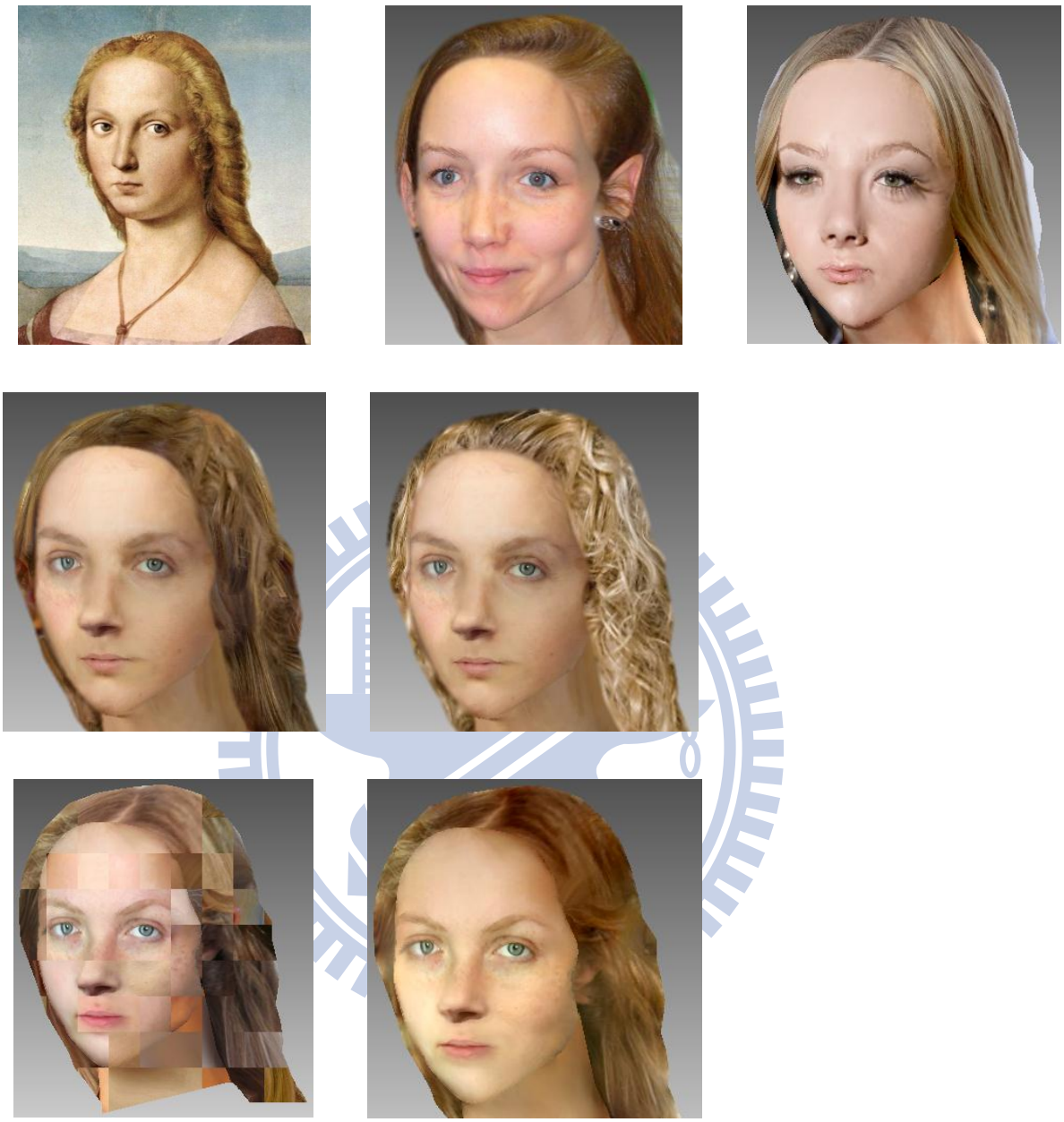


Figure 5.2-2 Comparison for realizing portrait paintings

A	B	C
D	E	
F	G	

A is the original input image. B is the result for **best-matched face**. C is the result for **best-matched-region**. D and E are our results with different weight λ for the hair. F is the patches retrieved by **regular-patch-based** method using feature vector of RGBE. G is the result after Poisson image editing for image F.

◆ La Fornarina by Raphael

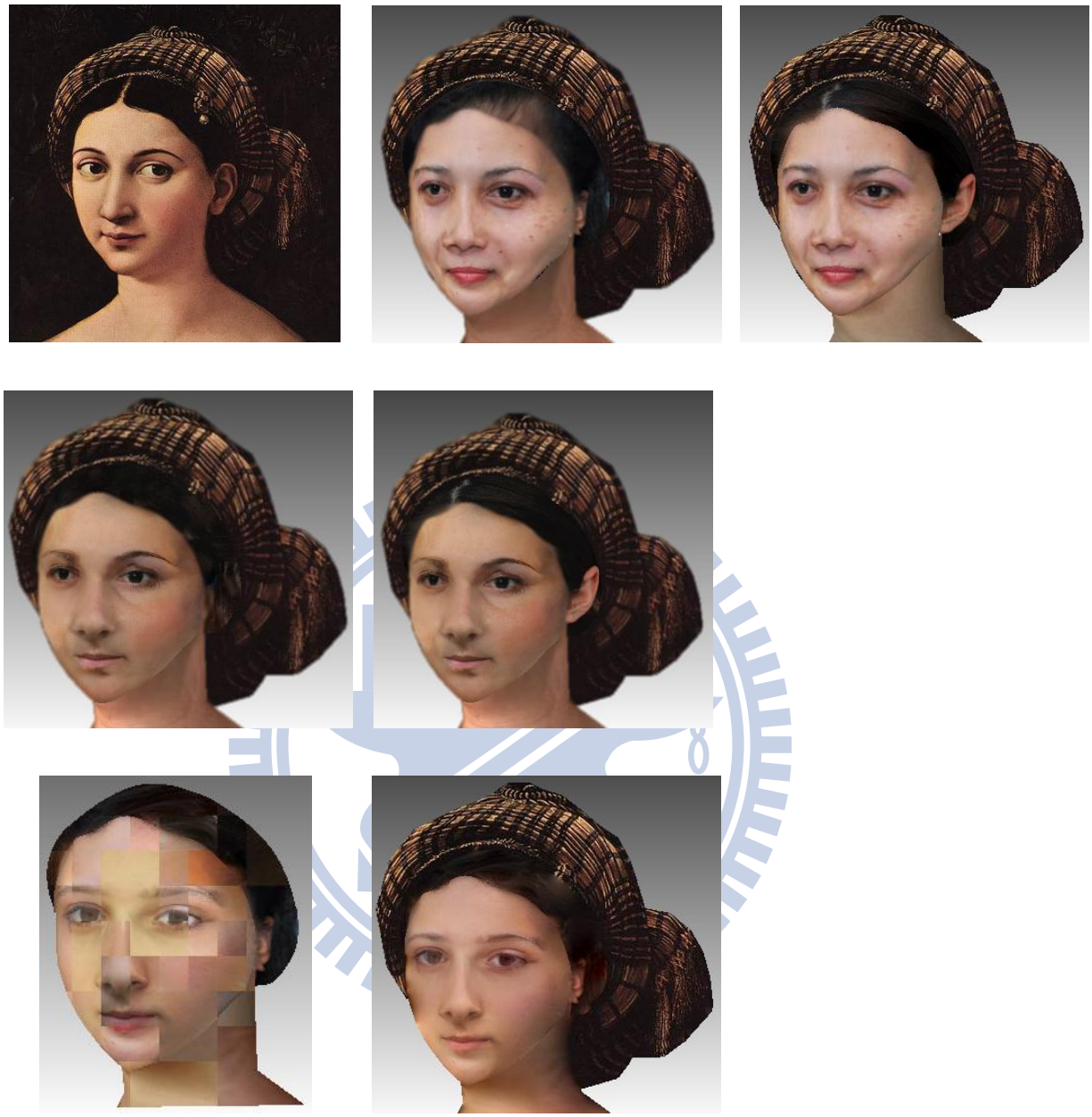


Figure 5.2-3 Comparison for realizing portrait paintings

A	B	C
D	E	
F	G	

A is the original input image. B is the result for **best-matched face**. C is the result for **best-matched-region**. D and E are our results with different weight λ for the hair. F is the patches retrieved by **regular-patch-based** method using feature vector of RGBE. G is the result after Poisson image editing for image F.

◆ Anne Boleyn

Figure 5.2-4 Comparison for realizing portrait paintings

A	B	C
D	E	
F	G	

A is the original input image. B is the result for **best-matched face**. C is the result for **best-matched-region**. D and E are our results with different weight λ for the hair. F is the patches retrieved by **regular-patch-based** method using feature vector of RGBE. G is the result after Poisson image editing for image F.

◆ Mona Lisa or La Gioconda by Leonardo da Vinci

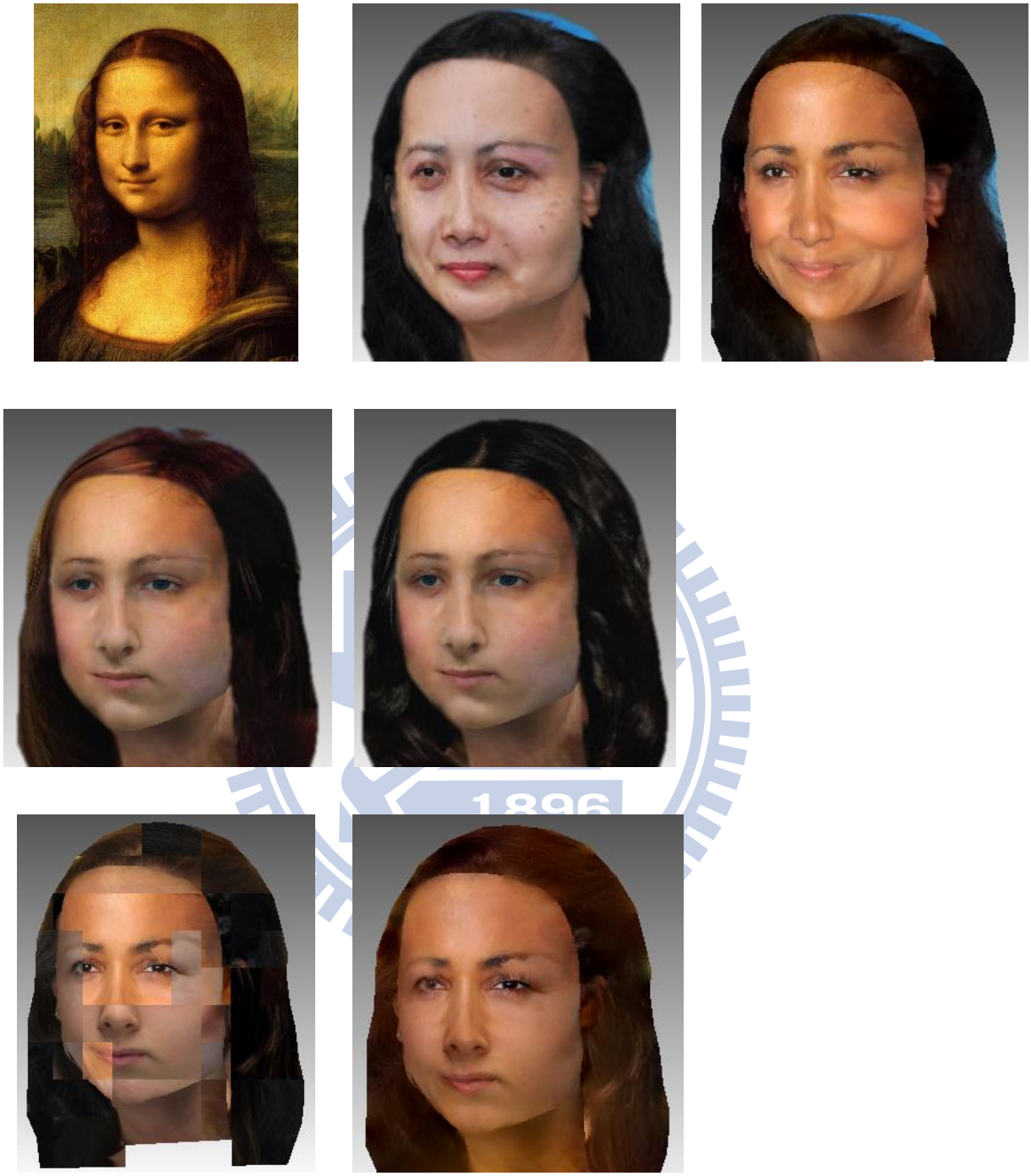


Figure 5.2-5 Comparison for realizing portrait paintings

A	B	C
D	E	
F	G	

A is the original input image. B is the result for **best-matched face**. C is the result for **best-matched-region**. D and E are our results with different weight λ for the hair. F is the patches retrieved by **regular-patch-based** method using feature vector of RGBE. G is the result after Poisson image editing for image F.

◆ From: http://alinn.my-net.cc/post/8/

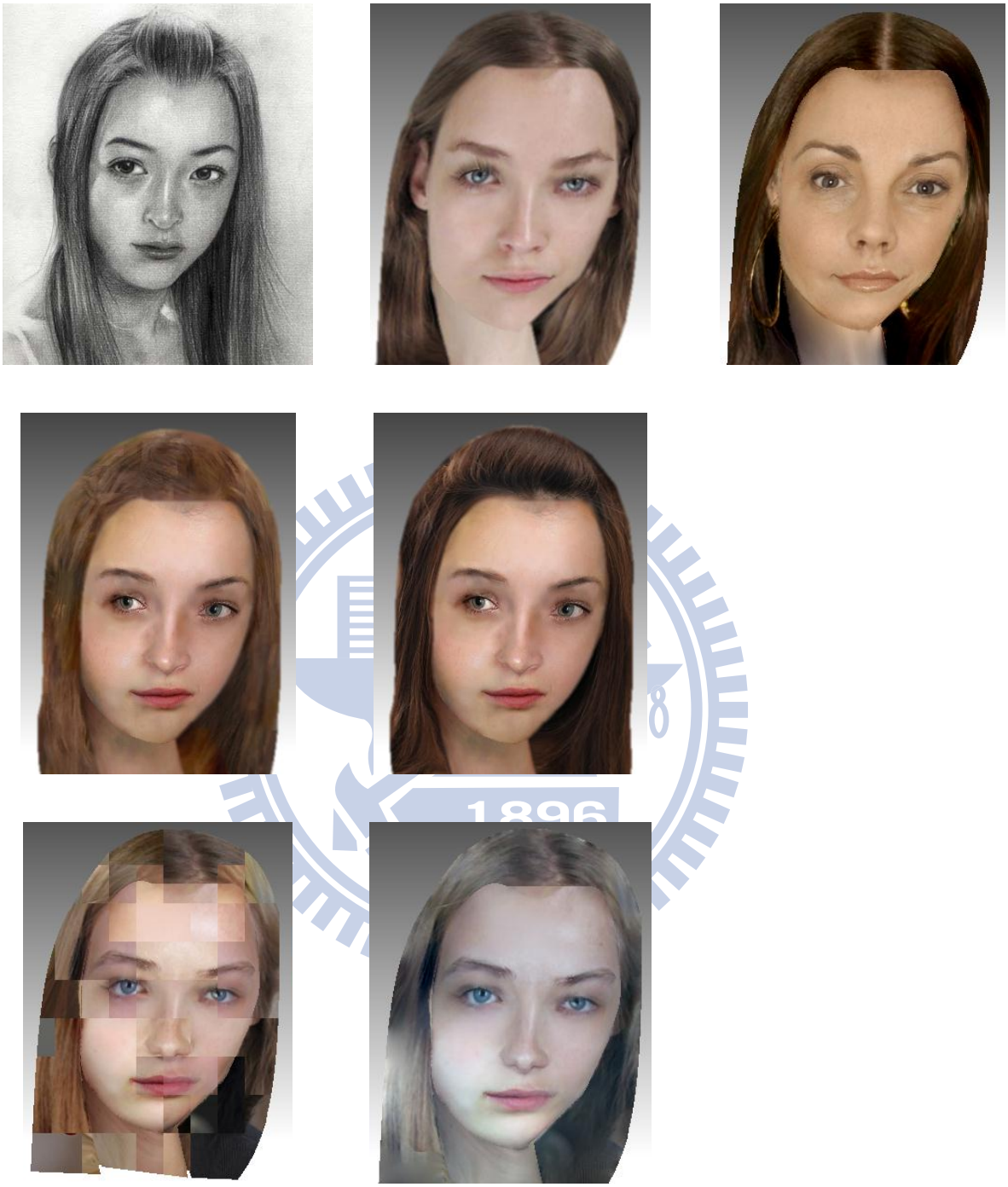


Figure 5.3-1 Comparison for realizing sketch paintings

A	B	C
D	E	
F	G	

A is the original input image. B is the result for **best-matched face**. C is the result for **best-matched-region**. D and E are our results with different weight λ for the hair. F is the patches retrieved by **regular-patch-based** method using feature vector of RGBE. G is the result after Poisson image editing for image F.

◆ From: http://canyouidentifyme.blogspot.tw/2012/03/princess-doe-this-story-is-no-fairy.html

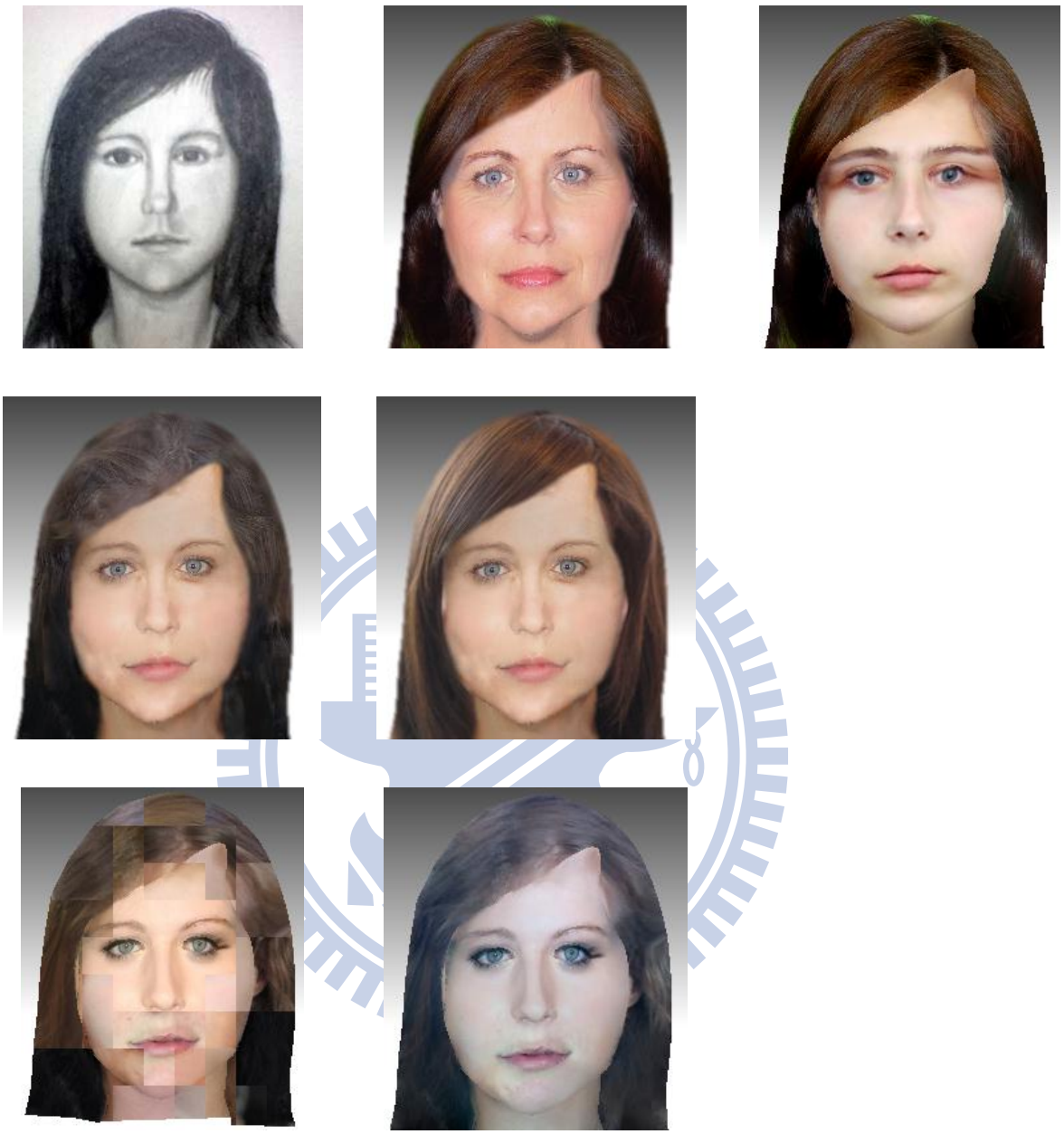


Figure 5.3-2 Comparison for realizing sketch paintings

A B C
D E
F G

A is the original input image. B is the result for **best-matched face**. C is the result for **best-matched-region**. D and E are our results with different weight λ for the hair. F is the patches retrieved by **regular-patch-based** method using feature vector of RGBE. G is the result after Poisson image editing for image F.

◆ From: http://theduendestudio.com/blog/?m=201003

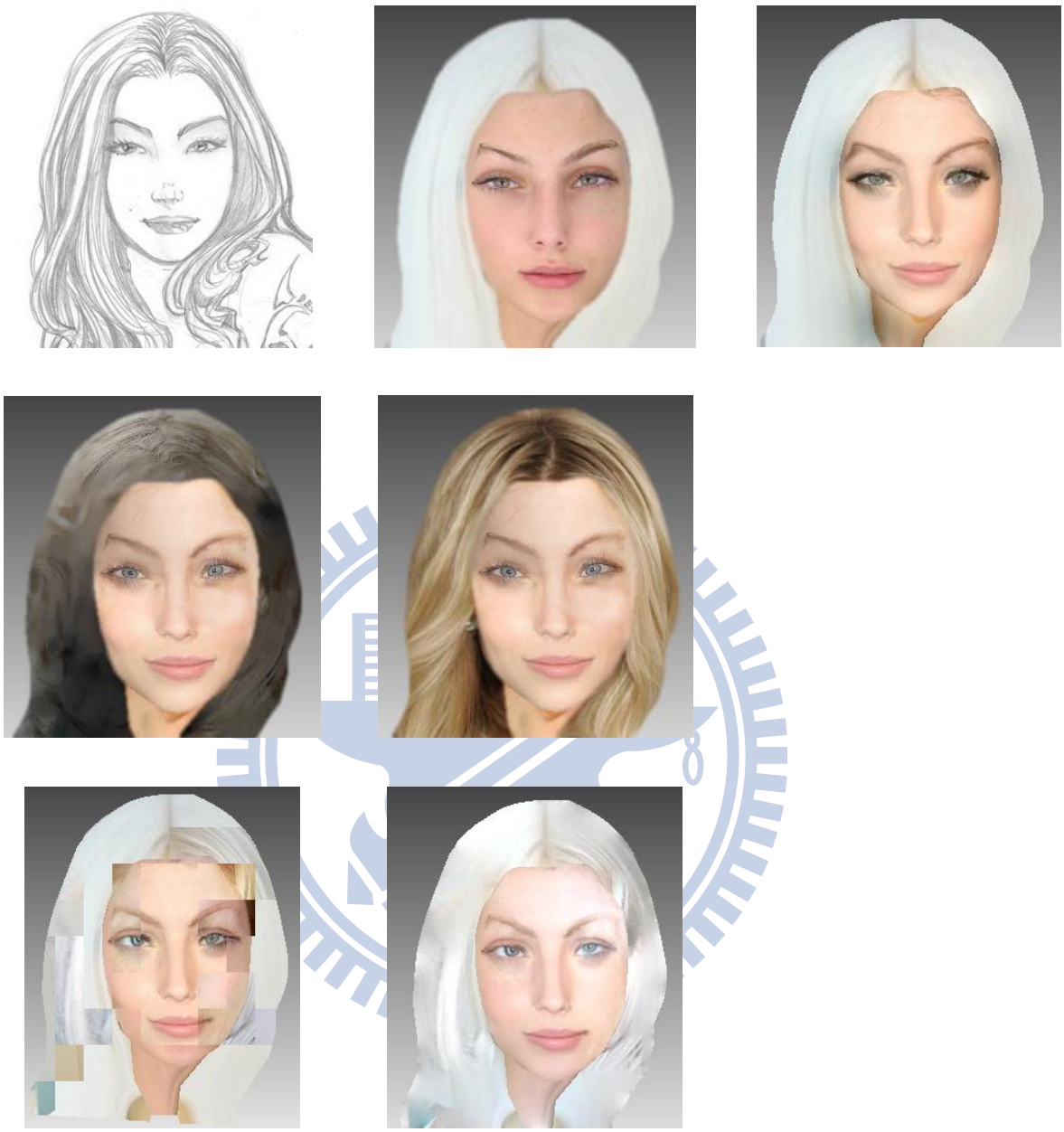


Figure 5.3-3 Comparison for realizing sketch paintings

A	B	C
D	E	
F	G	

A is the original input image. B is the result for **best-matched face**. C is the result for **best-matched-region**. D and E are our results with different weight λ for the hair. F is the patches retrieved by **regular-patch-based** method using feature vector of RGBE. G is the result after Poisson image editing for image F.

◆ From: http://www.myspace.com/theartofvenus/photos/38325268

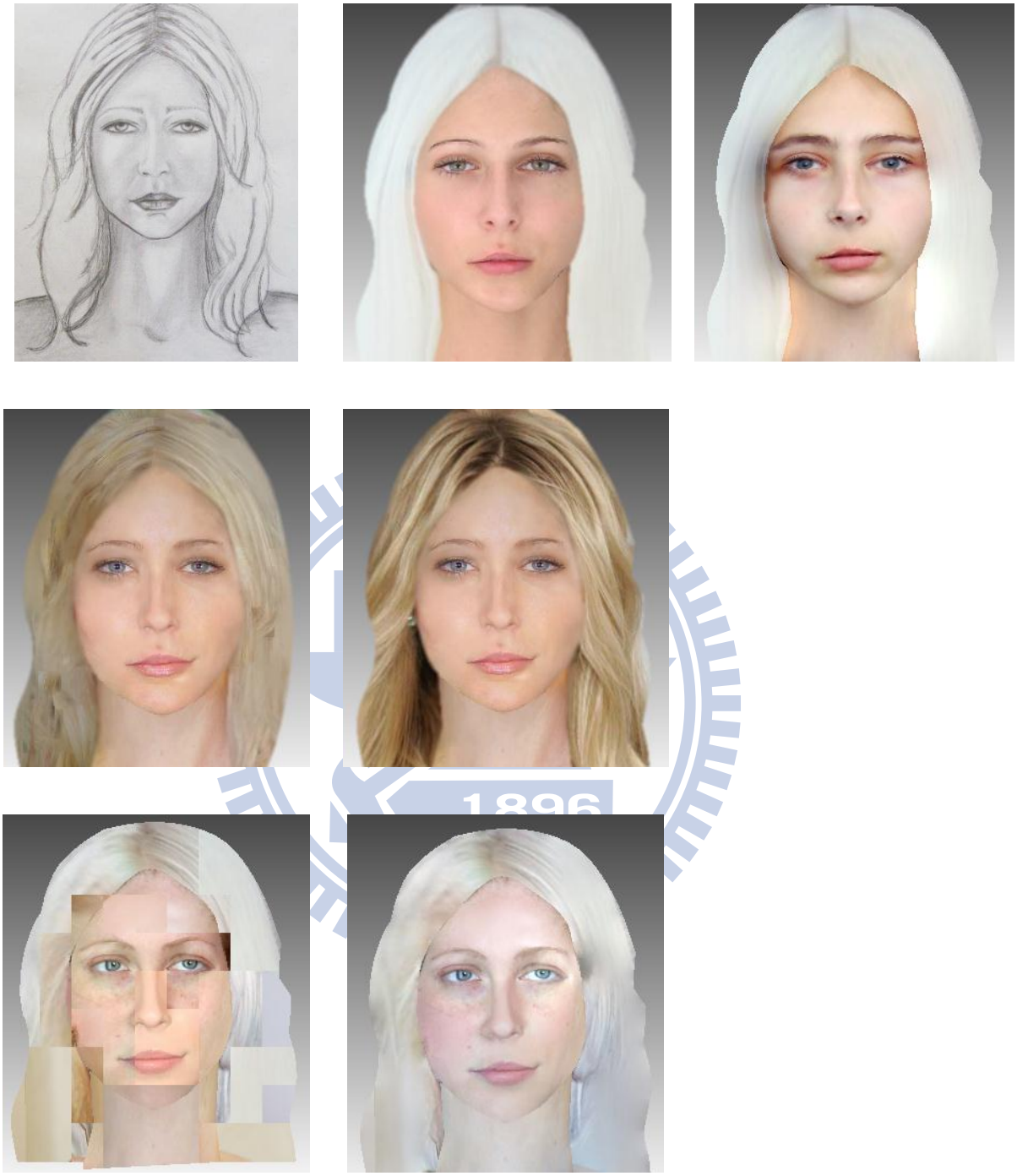


Figure 5.3-4 Comparison for realizing sketch paintings

A	B	C
D	E	
F	G	

A is the original input image. B is the result for **best-matched face**. C is the result for **best-matched-region**. D and E are our results with different weight λ for the hair. F is the patches retrieved by **regular-patch-based** method using feature vector of RGBE. G is the result after Poisson image editing for image F.

◆ Jean Grey Phoenix X-men(http://idrawgirls.com/tutorials/2011/10/27/how-to-draw-jean-grey-phoenix-x-men/)

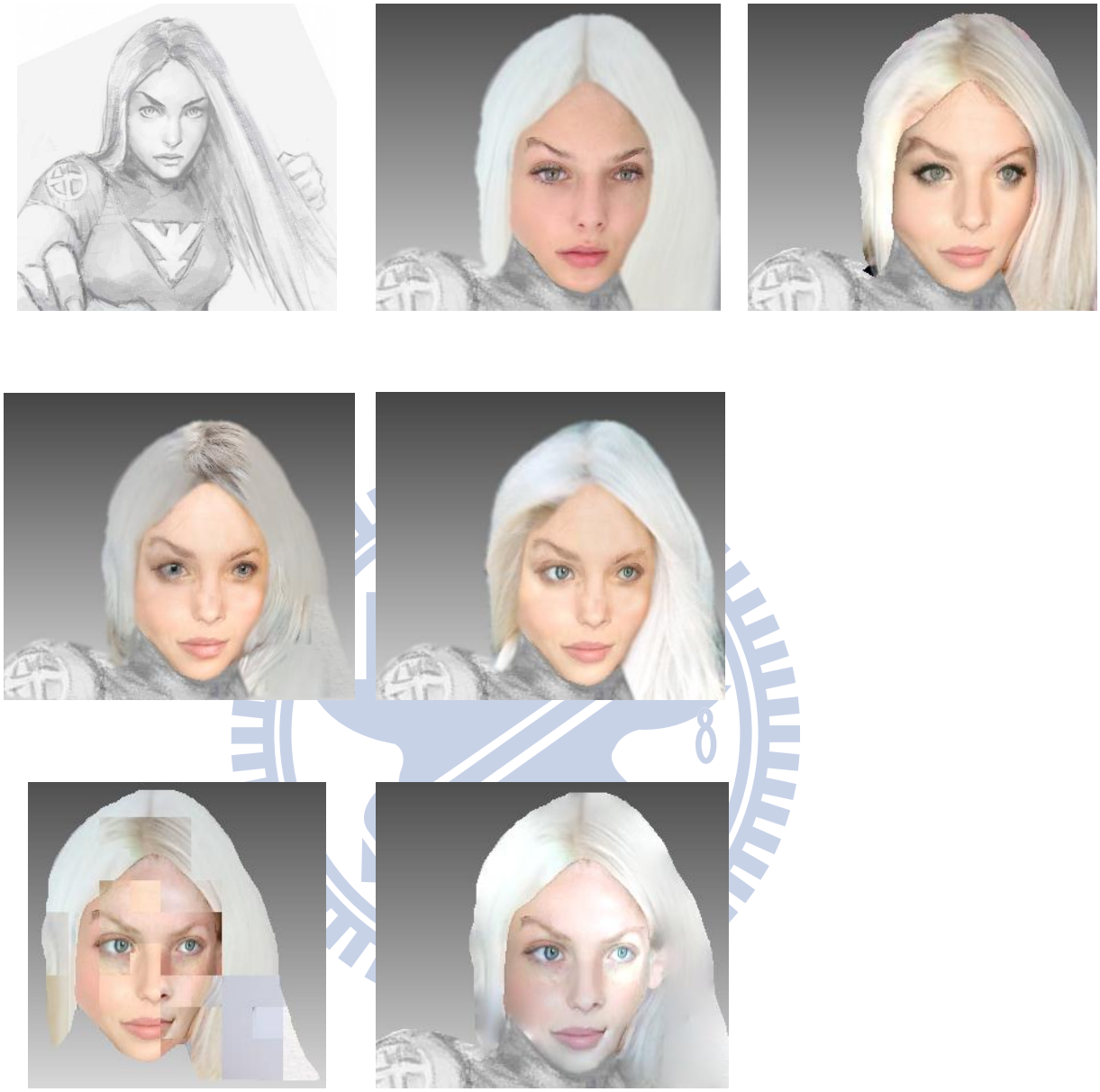


Figure 5.3-5 Comparison for realizing sketch paintings

A	B	C
D	E	
F	G	

A is the original input image. B is the result for **best-matched face**. C is the result for **best-matched-region**. D and E are our results with different weight λ for the hair. F is the patches retrieved by **regular-patch-based** method using feature vector of RGBE. G is the result after Poisson image editing for image F.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

Realizing “unreal” image is a novel and difficult task. That is because cartoon images do not have much detailed information as other paintings and human eyes are very sensitive to faces. In this thesis, we focus on human face realizing. Our method is applicable for exaggerated or highly unrealistic painting. With our method, we extract the most adequate patches from the database and preserve the characteristics of input. Beside it can remove the patch boundaries to form a smooth result.

We propose an example-based method to synthesize the reality of cartoon faces, portrait paintings and sketch paintings. We use multi-label optimization to find the most similar and adequate piece-up faces and seamlessly stitch the patches by chromatic gain compensation and multi-level blending. Our experiments show that the proposed example-based method is able to provide realistic, novel and satisfying results for the input images.

6.2 Future Work

We can improve our system in two aspects: one is the efficiency of our system and the other is the warping stage. For now, our algorithm used two optimization methods and require about dozens minutes for computation. We can accelerate and shorten the computing time by

parallel computation. Furthermore, it is interesting to design specific warping method for faces with exaggerated styles like Japanese cartoon characters, where their facial features are out of the proportion.

List of main figure sources

- Talia al Ghul (DC Comics) {Fig. 5.8, Fig. 5.17, Fig. 5.24, Fig. 5.1-1}
- Jean Grey Phoenix X-men (Marvel Comics) {Fig. 5.1-2}
- Black Canary (DC Comics) {Fig. 5.21, Fig. 5.1-3}
- Sara Pezzini (Image Comics/Top Cow) {Fig. 5.22, Fig. 5.1-4}
- Dakota Fanning Cartoon (http://cartoonized.net/dispceleb.php?img=237) {Fig. 5.9, Fig. 5.17, Fig. 5.19, Fig. 5.1-6}
- Zatanna Zatara (DC Comics/Vertigo) {Fig. 4.10, Fig. 4.11, Fig. 5.6, Fig. 5.14}
- Pocahontas (Walt Disney Pictures) {Fig. 5.12}
- Emma Watson Cartoon (http://www.cartoonstylist.com/celebrities/images/large/emma_watson.jpg) {Fig. 5.1-7}
- Serena Marques (http://models.com/newfaces/modeloftheweek/14945) {Fig. 3.1, Fig. 3.2}
- http://www.drhilinski.com/revision-rhinoplasty/san-diego-revision-rhinoplasty-specialist-discusses-secondary-rhinoplasty-surgery/ {Fig. 3.3}
- http://www.fotolia.com/Content/Comp/742231 (comp image) {Fig. 5.13}
- The Birth of Venus (Botticelli) {Fig. 5.15, Fig. 5.2-1}
- Young Woman with Unicorn by Raphael {Fig. 5.18, Fig. 5.24, Fig. 5.2-2}
- La Fornarina by Raphael {Fig. 5.4, Fig. 5.15, Fig. 5.2-3}
- Anne Boleyn {Fig. 5.2-4}
- Mona Lisa or La Gioconda by Leonardo da Vinci {Fig. 5.14, Fig. 5.2-5}
- Venus (http://www.myspace.com/theartofvenus/photos/38325268) {Fig. 5.5, Fig. 5.11, Fig. 5.20, Fig. 5.3-4}
- http://alinn.my-net.cc/post/8/ {Fig. 4.12, Fig. 4.13, Fig. 4.14, Fig. 5.26, Fig. 5.3-1}
- http://canyouidentifyme.blogspot.tw/2012/03/princess-doe-this-story-is-no-fairy.html {Fig. 5.7, Fig. 5.14, Fig. 5.24, Fig. 5.25, Fig. 5.3-2}
- http://theduendestudio.com/blog/?m=201003 {Fig. 5.3-3}
- Jean Grey Phoenix X-men (http://idrawgirls.com/tutorials/2011/10/27/how-to-draw-jean-grey-phoenix-x-men/) {Fig. 5.10, Fig. 5.23, Fig. 5.3-5}

Special Acknowledgment

We are really thankful and appreciate for the sources of all the images and photographs we surfed on internet. All the images, paintings and photographs helped us a lot. The images, paintings and photographs are not for commercial usage. It is for academic usage only. Last and the most important of all is that we want to show our highest appreciation to all.

References

[1] F. Pighin, J. Hecker, D. Lischinski, R. Szeliski, and D. H. Salesin, "Synthesizing realistic facial expressions from photograph," in *Proceedings of ACM SIGGRAPH*, 1998, pp. 75-84.

[2] Q. Zhang, Z. Liu, B. Guo, D. Terzopoulos, and H. Y. Shum, "Geometry-driven photorealistic facial expression synthesis," *IEEE Transactions on Visualization and Computer Graphics*, Vol.12, 2006, pp. 48-60.

[3] D. Bitouk, N. Kumar, S. Dhillon, P. Belhumeur, S. K. Nayar, "Face Swapping: Automatically Replacing Faces in Photographs," *Proceedings of ACM SIGGRAPH 2008*, vol. 27, no. 3 , August 2008, Article No. 39.

[4] U. Mohammed, S. J. D. Prince, J. Kautz, "Visio-lization: Generating Novel Facial Images," *ACM Transactions on Graphics (Proceedings SIGGRAPH 2009)*, vol. 28, no. 3, August 2009, pp. 57:1-57:8.

[5] P. Pérez, M. Gangnet, A. Blake, "Poisson image editing," *ACM Transactions on Graphics (Proceedings SIGGRAPH 2003)*, vol. 22, no. 3, 2003, pp. 313-318.

[6] M. Brown, D. G. Lowe, "Automatic Panoramic Image Stitching using Invariant Features," *International Journal of Computer Vision*, vol. 74, no. 3, 2007, pp. 364-375.

[7] D. J. Heeger, J. R. Bergeny, "Pyramid-based texture analysis /synthesis," in *Proceedings*

of ACM SIGGRAPH, 1995, pp. 229-238.

[8] B. Bickel, M. Botsch, R. Angst, W. Matusik, M. Otaduy, H. Pfister, M. Gross, "Multi-scale capture of facial geometry and motion," *ACM Transactions on Graphics*, Vol. 26, 2007, Article 33.

[9] M. K. Johnson, K. Dale, S. Avidan, H. Pfister, W. T. Freeman, W. Matusik, "CG2Real: Improving the Realism of Computer Generated Images using a Large Collection of Photographs," *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 9, 2011, pp.1273-1285.

[10] A. Shrivastava, T. Malisiewicz, A. Gupta, A. A. Efros, "Data-driven Visual Similarity for Cross-domain Image Matching," *Proceedings of ACM SIGGRAPH Asia*, vol. 30, no. 6, 2011, Article No. 154.

[11] Z. Liu, Y. Shan, Z. Zhang, "Expressive Expression Mapping with Ratio Images," *Proceedings of ACM SIGGRAPH 2001*, August 2001, pp. 271-276.

[12] S. Milborrow, "Locating Facial Features with Active Shape Models," Master's thesis, University of Cape Town (Department of Image Processing), 2007.

[13] T. F. Cootes and C. J. Taylor, "Statistical models of appearance for medical image analysis and computer vision," *Proc. SPIE Medical Imaging*, Sept, 2001, Online technical report available from http://www.isbe.man.ac.uk/~bim/refs.html.

[14] Y. Boykov, O. Beksler, R. Zabih, "Fast Approximate Energy Minimization via Graph

Cuts," *IEEE transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 11, 2001, pp. 1222-1239.

[15] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, 2001 , pp. 511-518.

[16] R. Lienhart and J. Maydt, "An Extended Set of Haar-like Features for Rapid Object Detection," *IEEE The International Conference on Image Processing*, vol. 1, 2002, pp. 900-903.

[17] E. Catmull. R. Rom, "A class of local interpolating splines," *Computer Aided Geometric Design*, edited by Robert Barnhill, Richard Reisenfeld, Academic Press, 1974, pp. 317-326.

[18] H. Huang, L. Zhang, H. C. Zhang, "RepSnapping: Efficient Image Cutout for Repeated Scene Elements," *Proceeding of Pacific Graphics*, vol. 30, no. 7, September 2011, pp. 2059-2066.

[19] J. Jia, C. K. Tang, "Image Stitching Using Structure Deformation," *IEEE Transactions of Pattern Analysis and Machine Intelligence*, vol. 30, no. 4, April 2008, pp. 617-631.

[20] T. F. Cootes, G. J. Edwards, C. J. Taylor, "Active Shape Models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.23, no. 6, 2001, pp. 681-685.

[21] I. Kemelmacher-Shlizerman, E. Shechtman, R. Garg, S. M. Seitz, "Exploring Photobios," *Proceedings of ACM SIGGRAPH 2011*, vol. 30, no. 4, July 2011, Article No. 61.

[22] J. Suo, S. C. Zhu, S. Shan, X. Chen, “A Compositional and Dynamic Model for Face Aging,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 3, March 2010, pp. 385-401.

[23] B. F. Klare, Z. Li, A. K. Jain, “Matching Forensic Sketches to Mug Shot Photos,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 3, March 2011, pp. 639-646.