

Example-based Facial Sketch Generation with Non-parametric Sampling

Hong Chen^{1*}, Ying-Qing Xu¹, Heung-Yeung Shum¹, Song-Chun Zhu² and Nan-Ning Zheng³

¹ Microsoft Research, China

² The Ohio State University, USA

³ Xi'an Jiaotong University, China

Abstract

In this paper, we present an example-based facial sketch system. Our system automatically generates a sketch from an input image, by learning from example sketches drawn with a particular style by an artist. There are two key elements in our system: a non-parametric sampling method and a flexible sketch model. Given an input image pixel and its neighborhood, the conditional distribution of a sketch point is computed by querying the examples and finding all similar neighborhoods. An “expected sketch image” is then drawn from the distribution to reflect the drawing style. Finally, facial sketches are obtained by incorporating the sketch model. Experimental results demonstrate the effectiveness of our techniques.

1. Introduction

It is always fascinating to see how an artist draws a picture of someone’s face. Sketches are perhaps the simplest form of drawings because they consist of only lines. Somehow the artist can quickly distill the identifying characteristics of a face and highlight them with a small number of strokes.

It has been shown that a simple sketch can reveal interesting characteristics of a face. Based on psychological hypotheses, for example, Rhodes and her colleagues [12] investigated how people decode a face’s identity to uncover the secrets of the human face, and reported the experiment using sketches of famous faces to examine when a caricature looks good.

There have been a few attempts to interactively or automatically synthesize facial caricatures. Brennan [1] presented perhaps the first interactive caricature generator. Murakami et al. [13, 7] developed the template-based facial

caricature system PICASSO and Web-PICASSO[14]. Li et al.[8] proposed an automatic facial sketch system that uses a generalized symmetry operator, rectangle filter, and characteristic shapes to detect the locations of facial feature points. Many approaches have also been proposed to generate facial caricature [10, 6]. But these approaches, without observing and learning from the artist’s products, in general produce stiff and inexpressive sketches.

In fact, few systems have been proposed to teach a computer to automatically generate a stylistic facial sketch by observing images drawn by artists. For example, Librande [9] developed an example-based character drawing system called Xspace, which extends the traditional concept of drawing with a learning module, Radial Basis Function[11]. Xspace can generate many dramatic sketches but is restricted in the types of images it can manipulate. Recently, Freeman et al [5] presented an example-based system for translating a sketch into different styles. But it focused on transferring styles instead of generating a stylistic sketch from an image.

In this paper, we present an example-based stylistic facial sketch system. Although it is possible to elicit some rules (e.g., how many lines for each sketch) from a set of sketches drawn by an artist, subtle styles cannot be modeled explicitly or easily. We propose a novel non-parametric sampling scheme for sketch generation to learn these subtle styles. From a set of training images and their associated sketches drawn by an artist with a particular style, we can generate a sketch from an input image automatically.

The rest of this paper is organized as follows. We present our sketch system in Section 2. The detailed algorithms for non-parametric sampling and sketch generation are presented in Section 3. Results are shown in Section 4. In Section 5, we discuss the limitations of our work and possible applications.

* Visiting from Artificial Intelligence and Robotics Lab, Xi’an Jiaotong University, China

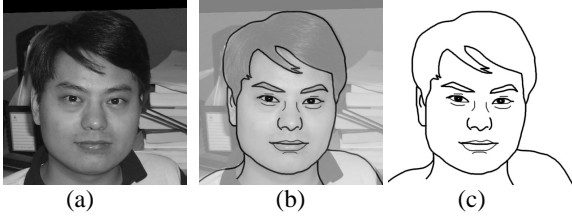


Figure 1. An example from the data set. (a) The original image; (b) the sketch drawn on top of the original by an artist; (c) the final sketch.

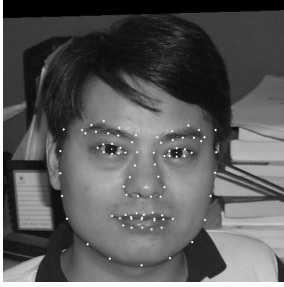


Figure 2. The pre-defined feature points on a face image.

2. The sketch system

There is no precise rule of grammar in a sketching language; even artists themselves can rarely explain their own styles in words. Therefore, in order to automatically generate a stylistic facial sketch of a given image, we have chosen to take an example-based approach.

2.1. Training data

Examples in our training data include 162 images and their corresponding sketches drawn by an artist. We also make the following important prerequisites.

- Frontal view only (no hats and glasses).
- Plain style, no exaggeration.
- Each pair of image and sketch matches perfectly.

As we shall see later, matched image and sketch pairs make the learning process much simpler. We asked the artist to draw the sketch on top of the original image (e.g., with a different layer in PhotoShop). Figure 1 shows one example of training images and sketches, and more example can be found in Figure 11.

To take different geometrical shapes of faces into account, we create an average shape from all examples. For each face I_i in the training set, we label all the feature points

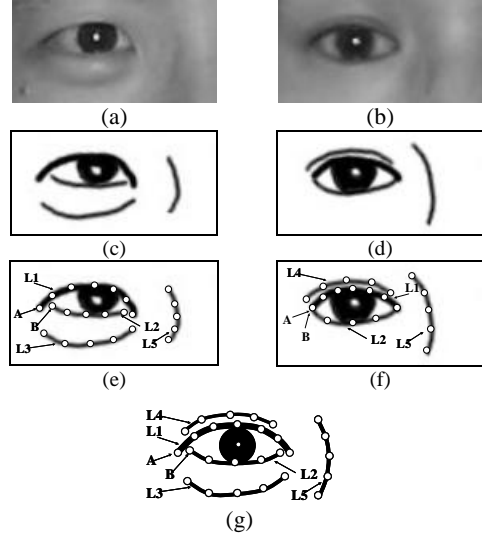


Figure 3. The model S has a flexible definition.

and place their positions into a vector $Shape_i$. Then we can compute the average of all vectors and call it $MeanShape$. We can now define a geometrical transformation G for any given $Shape$, so that $MeanShape = G(Shape)$. Geometrically transformed images are shown at the top of Figure 4. By warping (I, S) to $(I' = G(I), S' = G(S))$, we can discount the geometrical variations of different shapes.

We train an Active Shape Model (ASM [2]) to locate the key features in each face. As shown in Figure 2, we manually label 83 feature points on each face image. Directly applying the ASM model, however, generates some unsatisfactory results at high-curvature feature points such as the canthus and the corners of the mouth. Therefore, we modify the ASM algorithm to search for and match the high-curvature feature points. Specifically, we estimate the Gaussian model of the 2D grey-level structure and search for these feature points in a 2D local region instead of along the 1D normal direction. An example using the modified ASM is shown at the lower left corner in Figure 4.

2.2. The sketch model

Based on all examples from the training set, the model for sketches is defined. A sketch can be represented as a set S that contains a fixed number of lines. An on-off switch, its width, and the position of control points define each line. Note that these control points are different from those in the ASM model above. Specifically, $S = \{L_i : i = 1, 2, \dots, k\}$, where k is the number of lines. For the i -th line $L_i = \{c_i, \omega_i, \theta_i\}$:

- c_i : On-off switch. $c_i = 0$ signifies not to draw this line; and $c_i = 1$ means to draw it.

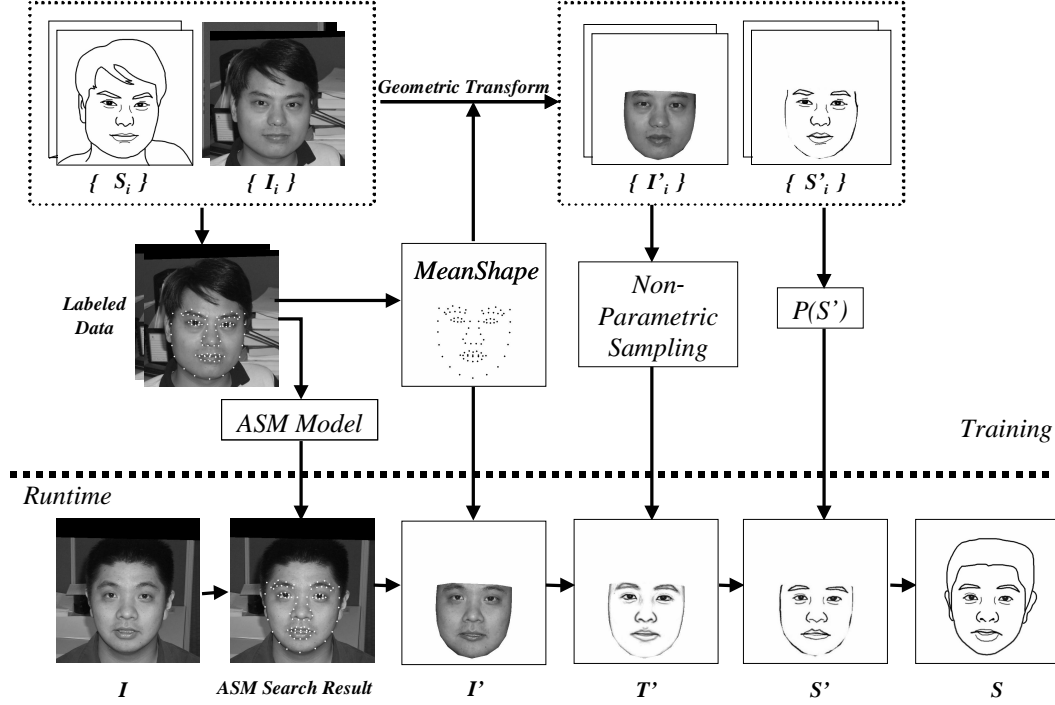


Figure 4. The system framework. Our system consists of a training phase and a runtime phase.

- ω_i : Width of this line.
- $\theta_i = \{(x_j, y_j); j = 1, 2, \dots, n_i\}$: The control points of this line, where n_i is the number of control points.

It is important to have a flexible sketch model so that the style can be captured. For example, the sketch models (Figure 3(c) and 3(d)) for the right eyes (Figure 3(a) and Figure 3(b)) are defined as a combination of five lines and an eyeball as shown in Figure 3(g). There is a line ($L3$) below the eye in Figure 3(e), but not in Figure 3(f). And there is a line ($L4$) above the eye in Figure 3(f), but not in Figure 3(e). The on-off switch is therefore necessary to model this kind of effect. In addition, there is no restriction on the location of lines or their endpoints. For example, point A and point B are separated in Figure 3(e) but they coincide in Figure 3(f).

In addition, we define three types of lines in our system, depending on whether the line is affected by others or not.

- Always appears;
- Probably appears, but independent of other lines.
- Depends on other lines.

We can build a prior model $P(S')$ based on these three types of lines from the set $\{S'_i\}$. Each line is modeled with a Gaussian, much like in ASM [2].

2.3 The system framework

Our system consists of a training phase and a runtime phase. In the training phase, we start with a set of image and sketch pairs, with manually labeled feature points.

- An ASM model is first trained to automatically locate the facial feature points in any input image.
- Based on these feature points, we define the average shape of all input sketches as *MeanShape*.
- A geometric transformation is then defined to warp any input shape to the *MeanShape*, $S' = G(S)$.
- Estimate prior probability $P(S')$ from $\{S'_i\}$.

At runtime, for a given image I , we generate a sketch by the following steps:

- Apply ASM to extract the facial feature points.
- Apply geometric transformation: $I' = G(I)$.
- Employ non-parametric sampling to obtain the “expected sketch image” T' .
- Apply $P(S')$ to obtain sketch S' from T' .
- Compute final sketch S from the inverse geometric transformation: $S = G^{-1}(S')$.

Figure 4 shows various steps in both the training phase and the runtime phase. Detailed algorithms are discussed in next two sections. Note that the hair is not part of learned

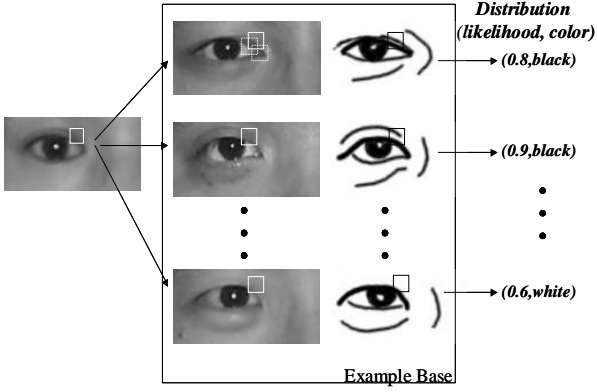


Figure 5. Construction of the sketch distribution at a pixel. Each pixel is compared with the sample images to find the best matches. For each matched pixel in a sample image, its corresponding sketch point is kept for the distribution.

sketch but added in a post-processing step by finding out the hair contours.

3. The algorithm

The subtle styles of facial sketches are embedded in the complex statistical relationship between I' and S' . We introduce a non-parametric method that generates sketches from images. Non-parametric sampling has been recently applied successfully for texture synthesis by Efros and Leung [3].

3.1. Non-parametric sampling

Similar to Markov networks [4], in this work, the probability distribution of a sketch point, for a given pixel in the input image and its neighbors, is assumed to be independent of the rest of the image. For each pixel in the input image, sample images are queried and corresponding pixels are found. Only those pixels with a small enough difference (the k-nearest-neighbors) are used to determine how likely it would be drawn as a point in the sketch, or the distribution of a sketch. Cross-correlation is used to compare pixels with square neighborhoods. Local search is employed to find the best match between two images to deal with slight geometrical mis-alignments that may exist even after warping the images to the *MeanShape*.

Figure 5 illustrates the process of constructing such a distribution.

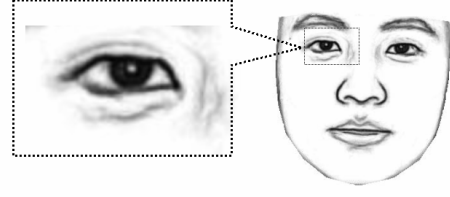


Figure 6. An expected sketch image T' and the blown-up right eye.

	Example 1	Example 2	Example 3
I'			
S'			
$K=1$			
$K=4$			
$K=20$			
$K=80$			

Figure 7. Expected sketch images generated with K nearest neighbors; Search window is set at 5×5 and neighborhood is 7×7 .

3.2. Extracting sketches

From the computed distribution of possible sketch points, we can sample possible sketches. One way to do so is to integrate the distribution and obtain an “expected sketch image” (T') or *ESI* as shown in Figure 6. Then the final sketch can be extracted from the “expected sketch image”.

How many samples should we integrate to generate such an *ESI*? Or how many k-nearest-neighbors should be considered? Figure 7 shows the effect of different numbers of integration samples. If we use all of them, we will lose significant details. For example, the top line and the right line of Example 1 become almost invisible when K is 80. On the other hand, if we use too few, the results will be noisy, as evident from Figure 7. In practice, we have found that 25% of the samples to be a good choice for this particular style.

The neighborhood size also affect the generated sketches. Figure 7 shows the comparison with different neighborhood sizes when the search window size is fixed at 5×5 and the nearest neighbor number K is set to 20.
















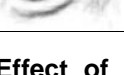
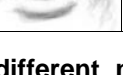

	Example 1	Example 2	Example 3
I'			
S'			
$N=3*3$			
$N=7*7$			
$N=11*11$			
$N=15*15$			

Figure 8. Effect of different neighborhood sizes. Search window is set at $5 * 5$ and K is 20.

When the neighborhood size is too large, the details will disappear. On the other hand, if the neighborhood size is too small, the sketches will be noisy. In practice, we have found that $7 * 7$ is a good size for the neighborhood.

We have also studied the effect of search window size. In our experiments, especially under the current geometrical alignment, search window size does not affect significantly the sketch results and we select $5 * 5$ in our system.

The next step is to fit a sketch model from the expected sketch image. It may still be hard to solve a general feature location problem. Fortunately, for our problem, after the coarse feature localization by ASM, the possible solution of each line segment is constrained to a very small region. In addition, the dimension of parametric space of each line is often low. To search for the parameters of each line, it is very efficient to directly sample from $P(S')$, and followed by a local search.

4. Sketch results

Finally, we combine all the lines to generate a sketch for the given image of a frontal face. There are three types of lines as we defined in the sketch model: definitely appear, probably appear without any constraints, and appear with constraints. Figure 9 shows two examples synthesized by our approach. And more detailed images of the eyes are shown in Figure 10 to illustrate that some lines can appear or disappear in different sketches. Figure 10 also illustrates that different styles of sketches can be generated for the nose, as either a continuous line or two separated lines. These various styles are all learnt from the example base. More examples are illustrated in Figure 12 to demon-

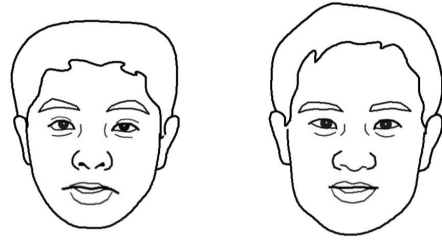


Figure 9. Two examples of generated sketches.



Figure 10. Brown-up views of right eyes and noses from the images in Figure 9.

strate the richness of generated sketches. Note that ears and hairs are generated simply with tracked contours by ASM and image segmentation.

It takes 10 minutes on a Pentium III 500 Mhz PC to generate a sketch of size $256*256$. We have recently accelerated the process to generate a sketch in 15 seconds. The most time-consuming segment is the sampling from example bases. We should point out that some details are lost around the mouth because lines above and below the mouth are not modeled in the current sketch model.

5. Discussion and conclusions

We have presented an approach to automatically generating good-quality sketches from input images. These sketches not only preserve facial features of the original image but also simulate the artist's style. A novel non-parametric sampling scheme is employed to learn complex statistical characteristics between an image and its sketch. Combining a flexible sketch model with the non-parametric sampling method is the key to our system.

There are a number of future directions we can take. For example, our system cannot generate sketches for non-frontal views. Speeding up the non-parametric sampling is another important problem. A severe limitation of current work is that no exaggeration can be produced, yet many cartoons and caricatures are interesting because the facial characteristics are highlighted and exaggerated. Nevertheless, the sketch system we have presented in this paper have a wide range of applications from Internet chatting to video compression.

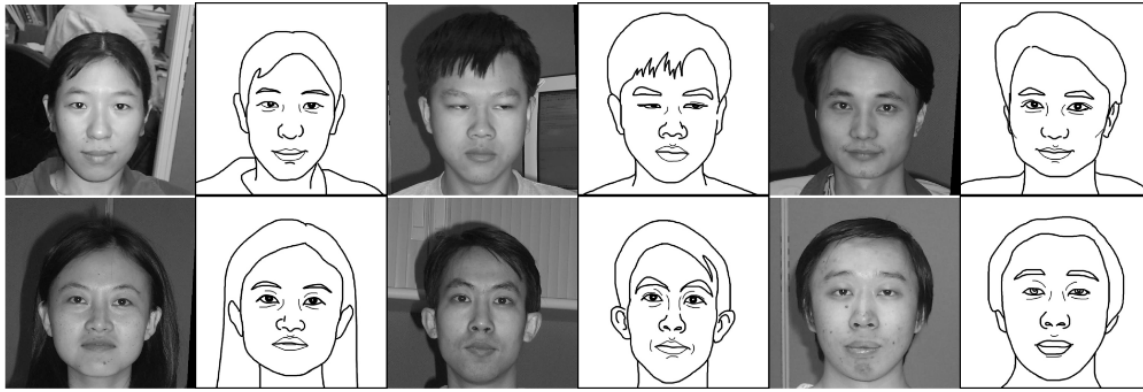


Figure 11. A set of examples of our training set. Note the difference of eyes and noises for different subjects.

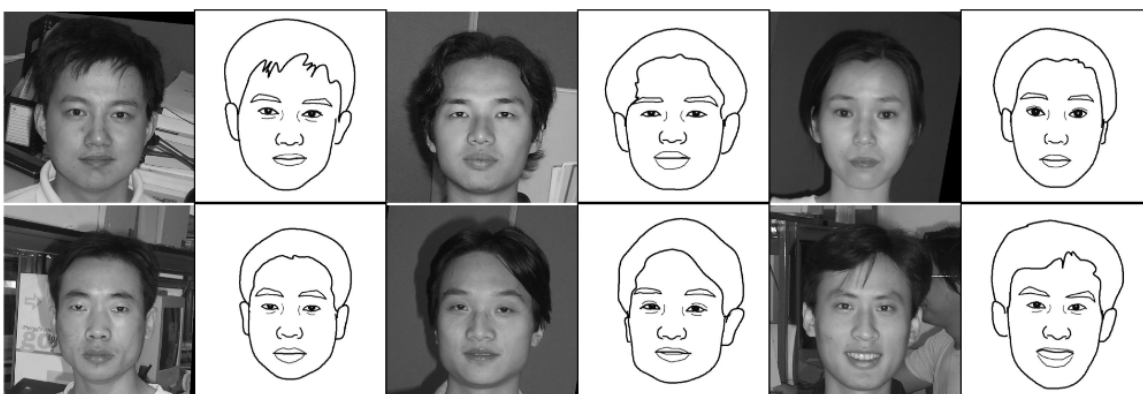


Figure 12. A number of generated sketches using our approach. Our system generates good quality sketches with different expressions, slight rotated angles, and other variations.

References

- [1] S. Brennan. Caricature generator. Master's thesis, Cambridge, MIT, 1982.
- [2] T. F. Cootes and C. J. Taylor. Statistical models of appearance for computer vision. Technical report, University of Manchester, Manchester M13 9PT, U.K., 2000.
- [3] A. A. Efros and T. K. Leung. Texture synthesis by non-parametric sampling. In *the Seventh International Conference on Computer Vision*, pages 20–27, 1999.
- [4] W. T. Freeman and E. Pasztor. Learning low-level vision. In *the Seventh International Conference on Computer Vision*, 1999.
- [5] W. T. Freeman, J. B. Tenenbaum, and E. Pasztor. An example-based approach to style translation for line drawings. Technical Report 11, MERL Technical Report, Cambridge, MA, Feb 1999.
- [6] S. Iwashita, Y. Takeda, and T. Onisawa. Expressive facial caricature drawing. In *IEEE International conference on fuzzy systems*, volume 3, pages 1597–1602, 1999.
- [7] H. Koshimizu, M. Tominaga, T. Fujiwara, and K. Murakami. On kansei facial processing for computerized facial caricaturing system picasso. In *IEEE International Conference on Systems, Man, and Cybernetics*, volume 6, pages 294–299, 1999.
- [8] Y. Li and H. Kobatake. Extraction of facial sketch based on morphological processing. In *IEEE international conference on image processing*, volume 3, pages 316–319, 1997.
- [9] S. E. Librande. Example-based character drawing. Master's thesis, Cambridge, MA. MIT, 1992.
- [10] J. Nishino, T. Kamyama, H. Shira, T. Odaka, and H. Ogura. Linguistic knowledge acquisition system on facial caricature drawing system. In *IEEE international conference on fuzzy systems*, volume 3, pages 1591–1596, 1999.
- [11] T. Poggio and G. Girosi. Networks for approximation and learning. *Proceedings of IEEE*, 78(9):1481–1497, 1990.
- [12] G. Rhodes. Secrets of the face. *New Zealand journal of psychology*, 23(1):3–17, 1994.
- [13] M. Tominaga, S. Fukuoka, K. Murakami, and H. Koshimizu. Facial caricaturing with motion caricaturing in picasso system. In *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, page 30, 1997.
- [14] www.koshi-lab.sccs.chukyo-u.ac.jp/fuji/pica2.