# FACE SYNTHESIS USING FACIAL TRAIT CODE AND ITS APPLICATION TO CREATING SUSPECT'S PHYSICAL PROFILES

*Ping-Han Lee*

Graduate Institute of Computer Science and Information Engineering
National Taiwan University
d93922027@ntu.edu.tw

*Yi-Ping Hung*

Graduate Institute of Networking and Multimedia
National Taiwan University
hung@csie.ntu.edu.tw

## ABSTRACT

In this work, we propose a novel approach to synthesize human faces. The proposed approach is based on the Facial Trait Code, or FTC for short, which is originally proposed for solving the face recognition problem. The FTC decoding transforms codewords into faces. We find that the FTC code space is typically huge and only a very small subset of FTC codewords render smooth faces. We propose an algorithm which extracts codewords that render smooth faces efficiently, and we dub these codewords as legitimate codewords. The extraction of legitimate codewords is accompanied by a coarse-to-fine face synthesis scheme, which renders faces by mosaicking multiple patches of different facial parts. We implement a GUI to demonstrate that the proposed approach is able to help eyewitnesses to create suspect's physical profiles efficiently in the law enforcement environments, without the aid of face painters.

***Index Terms*—** Face Synthesis, Facial Trait Code

## 1. INTRODUCTION

Computer algorithms which synthesize novel faces have many applications to both surveillance and entertainment industries. There exist some approaches in the literature [1] [2] [3] [4]. Most of these works applied *deformable template* to localize multiple facial feature points in 2-D facial images. All these feature points were aligned to fit a generic 3D face model. Then a texture-mapping technique was applied to synthesize lifelike faces. The face synthesis technique can be applied to generate novel speaking faces [2], or faces with different facial expressions [1]. It can also improve the face recognition accuracy by synthesizing faces with illumination effects removed [3]. [4] proposed a component-based 3D face synthesis scheme, in which 3D faces are segmented into 5 manually defined subregions, and the fusion of these subregions renders novel faces.

In this paper we propose a novel face synthesis scheme which does not rely on 3-D models, and is able to help the eyewitnesses to create suspect's physical profiles efficiently. This task is typically done by face painters. It is sometimes time consuming and usually provides sketches of suspects only. The proposed scheme, on the other hand, provides photorealistic images. The most related work is the Face Matrix [5]. However, Face Matrix is a tool that helps face painters to create suspect's physical profiles, while the proposed scheme can help eyewitnesses to create suspect's physical profiles by themselves, without the aid of face painters.

In our previous work, we proposed the **Facial Trait Code**, or **FTC** for short, for solving the face recognition problem [6]. As pointed out in [6], FTC is also applicable to face synthesis task. In this work, we elaborate the face synthesis procedure using FTC. The contributions of this work is summarized as the following. (1) We elaborate the original FTC decoding scheme to render more visually pleasing faces. (2) We propose a novel measure to evaluate the legitimacy of codewords. According to our definition, a *legitimate codeword* is the codeword that renders a smooth face. This issue is not discussed in [6]. (3) The code space of the FTC is typically enormously huge. We find that the collection of legitimate codewords is only a very small subset in this space, and propose an efficient algorithm to generate legitimate codewords. (4) We implement a GUI and it demonstrates the proposed approach is able to help eyewitnesses to create suspect's physical profiles.

The rest of the paper is organized as the following. Section 2 gives a brief introduction to the Facial Trait Code, and we have implemented a FTC system in section 2.1. We elaborate the face synthesis scheme in section 3. In section 4 we describe the GUI that helps eyewitnesses to create suspect's physical profiles. Finally the conclusion and the future work are given in section 5.

## 2. FACIAL TRAIT CODE

This section gives a brief introduction to our previous work, the Facial Trait Code (FTC) [6]. We begin with specifying a local **patch** on a face by a rectangle bounding box $\{x, y, w, h\}$, where $x$ and $y$ are the 2-D pixel coordinates of the bounding box's upper-left corner, and $w$ and $h$ are the width and height of this bounding box, respectively. If the bounding box is moved from left to right and top to bottom in the face with various sizes of steps, denoted by $\Delta x$ and $\Delta y$ pixels in each direction, and if $w$ and $h$ can change from some small values to large values, we will end up with an exhaustive set of local patches across the face. With an extensive experimental study on the size range and step, [6] selected slightly more than a couple thousands of patches for a face with 80x100 pixels in size. In the following, we assume $M$ patches in total obtained from a face.

Assuming $K$ faces available for FTC construction, and all faces aligned by the centers of both eyes, the above running box scheme will give a stack of $K$ patch samples in each *patch*. To cluster the $K$ patch samples in each patch stack, the Principal Component Analysis (PCA) is firstly applied to extract the features. Considering the case that the $K$ facial images can be from $L$ individuals ($L \leq K$, i.e., one individual may have multiple facial samples), for each patch stack the Linear Discriminant Analysis (LDA) is then applied to determine the $L$ most discriminant low dimensional patch features for the $L$ individuals. It is assumed that the $L$ low dimensional patch features in each patch stack can be modeled by a Mixture of Gaussian (MoG), then the unsupervised clustering algorithm can be applied to identify the MoG patterns in each patch stack. Assuming $M$ patch stacks are available, this algorithm can cluster the $L$ low dimensional patch features into $k_i$ clusters in the $i$-th patch stack, where $i = 1, 2, ..., M$. The $k_i$ clusters in the $i$-th patch stack were considered the patterns existing in this patch stack, and they are called the **patch patterns**.

A scheme is proposed in [6] that selects some combination of the patches with their patch patterns able to best discriminate the training individuals by their faces. This scheme first define a matrix, called **Patch Pattern Map** ($PPM$), for each patch. $PPM$ shows which individuals' faces reveal the same pattern at that specific patch. Let $PPM_i$ denote the $PPM$ for the $i$-th patch, $i = 1, 2, ..., M$. $PPM_i$ will be $L \times L$ in dimension in the case with $L$ individuals. Its entry at $(p, q)$, denoted as $PPM_i(p, q)$, equals to 0 if the patches on the faces of the $p$-th and the $q$-th individuals are clustered into the same patch pattern, and equals to 1 otherwise.

Given $N$ patches and their associated $PPM_i$'s stacked to form a $L \times L \times N$ dimensional array, there are $L(L-1)/2$ $N$-dimensional binary vectors along the *depth* of this array because each $PPM_i$ is symmetric matrix and one can only consider the lower triangular part of it. Let $v_{p,q}$ $(1 \leq q < p \leq L)$ denote one of the $N$-dimensional binary vectors, then $v_{p,q}$ reveals the local similarity between the $p$-th and the $q$-th individuals in terms of these $N$ local patches. More unities in $v_{p,q}$ indicates more differences between this pair of individuals, and on the contrary, more zeros shows more similarities in between.

If each individual's face is encoded using the most discriminant patches, defined as the **facial traits**, then the induced set of $[v_{p,q}]_{1 \leq q < p \leq L}$ can be used to define the minimum and maximum Hamming distance among all encoded faces in the corresponding code space. The $v_{p,q}$ with the least (most) of unities gives the minimum (maximum) Hamming distance. To maximize the robustness against possible recognition errors in the decoding phase, we proposed an Adaboost algorithm to maximize the $d_{min}$, the minimum Hamming distance, for the determination of the facial traits from the overall patches [6].

Assuming $N$ facial traits selected from the the overall $M$ patches, and each with trait patterns *symbolized* by $1, 2, ..., k_j$, $i = 1, 2, ..., N$, one can now define the codewords in FTC. Each codeword is of length $N$ and $n$-ary where $n$ is the largest number of the trait patterns found in one single trait. In summary, given a large collection of training faces, one can define $N$ facial traits, $\sum_{j=1}^{N} k_j$ trait patterns, and $\prod_{j=1}^{N} k_j$ faces (or FTC codewords).

### 2.1. Construction of the $FTC_{AR}$

This section gives an example of FTC construction. We apply the facial images in AR face database [7] to construct a FTC. There are 126 different people (70 men and 56 women) in the AR database. Each person participated in two sessions, separated by two weeks (14 days). We included two neutral faces from the two sessions the synthesis task (i.e. faces without variations in illuminations and expressions, and are not partially occluded.). These faces are aligned with the centers of two eyes, converted to gray scale images, and resized to 80-by-100 pixels.

We have total 252 faces from 126 different people, each has 2 images. We randomly divided these images into 2 sets. The first set, denoted as $S_1$, includes 100 people and 200 faces. Based on $S_1$, we constructed a FTC. The second set, denoted as $S_2$, includes the rest 26 people with 52 faces. We 'pretended' these images exist only in the mind of eyewitnesses, and utilize the GUI introduced in section 4 to generate similar ones. Following [6], we construct a FTC with 20 traits, which will be referred to as $FTC_{AR}$ in the rest of the paper. The selected traits with their patterns are illustrated in

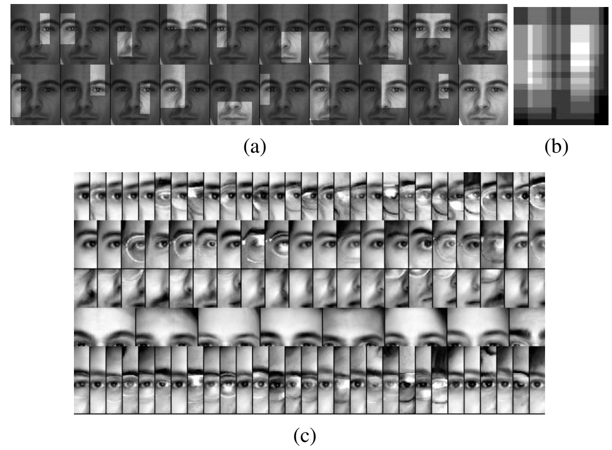


**Fig. 1**. Illustration of the 20 traits for face synthesis application. (a) Their locations on faces. The highlighted regions give the locations and sizes of traits. (b) The union of the covered regions of the 20 traits. Brighter regions are covered by more traits and vise versa. (c) Illustrations of patterns of the some traits in (a). Each row shows patterns belong to the same trait.

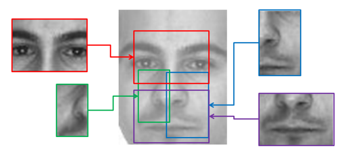

**Fig. 2**. Illustration of the face synthesis using FTC.

figure 1. Each pattern in this figure is the average of all the patch samples belong to this pattern, and its histogram is matched to the *mean face*[1] in order to render smooth faces later. There are total 697 patterns for 20 traits, and each trait has 34.85 patterns on average. The cardinality of the $FTC_{AR}$ code space is roughly $5.936 * 10^{30}$, which is an enormously large number[2]. Please refer to [6] for more details of the code construction. Note that we manually specify the 20-th trait as the whole face itself, to guarantee the fusion of patterns of all 20 traits renders complete faces.

## 3. FACE SYNTHESIS USING FTC

According to [6], the FTC decoding transforms a codeword into a face. In the FTC decoding stage, each number in a given codeword refers to a pattern, and the fusion of the total $N$, which is the number of traits, patterns from different locations and sizes renders a facial image. Figure 2 illustrates this process. Recall that the cardinality of the $FTC_{AR}$ code space is roughly $5.936 * 10^{30}$, and this number is greatly larger than the current total number of people in the world, which is roughly $6 * 10^9$. An interesting problem is: *are all the codewords in the FTC code space able to render lifelike faces*? To answer this question, we begin by noticing that there are some regions on the synthesized faces are overlapped by multiple patterns, as illustrated in figure 2. We define the **appearance consistency**, of

[1]The averaged face over 200 training faces.

[2]This number is given by $\prod_{i=1}^{20} k_i$, where $k_i$ is the number of patterns for the $i$-th trait.

patterns of two traits as

$$C(t_{i,j}, t_{k,l}) = \frac{1}{|\Omega|} \sum_{\Omega} |t_{i,j}(x,y) - t_{k,l}(x,y)|, \qquad (1)$$

where $t_{i,j}$ is the $j$-th pattern of the $i$-th trait and $t_{k,l}$ is the $l$-th pattern of the $k$-th trait, $i \neq k$. $\Omega$ is the collection of pixel locations where the $i$-th and $k$-th traits are overlapped with each other, and $|\Omega|$ is the area of the overlapped region. To render smooth faces, we need to enforce a constrain: if two patterns $t_{i,j}$ and $t_{k,l}$ are overlapped with each other, the associated $C(t_{i,j}, t_{k,l})$ should be small. Given this constrain, we can verify whether a codeword $c = \{c_1, c_2, ..., c_N\}$, where $c_i$ is the pattern number for the $i$-th trait, can render a smooth face by whether or not the inequality

$$\sum_{i,k,i \neq k} C(t_{i,c_i}, t_{k,c_k}) < \epsilon \qquad (2)$$

is satisfied. We call the codeword that satisfies the inequality (2) a **legitimate codeword**[3].

It is not feasible to exam all these codewords using (2), since it requires examining all the $5.936 * 10^{30}$ codewords in the $FTC_{AR}$ code space. Empirically we found that legitimate codewords contribute a very small portion in the whole code space. If we randomly generate a codeword, most likely it will not be a legitimate one. In the following we propose an algorithm to generate legitimate codewords efficiently. This algorithm also renders a corresponding human face.

### 3.1. An Efficient Novel Face Synthesis Algorithm

Assume we have a FTC with its traits sorted by size in the descending order, and $t_i$ denotes the $i$-th trait. Recall that in section **??**, each trait $t_i$ is specified by a rectangle $\{x_i, y_i, w_i, h_i\}$. We further denote $\Omega_{t_i}$ as the collection of pixel locations within this rectangle. As defined earlier, $t_{i,j}$ is the $j$-th pattern of the $i$-th trait, and it is a $w_i$-by-$h_i$ image patch. A **weight mask** that specifies the weight of each pixel in $t_{i,j}$ is denoted as $\omega_i$, and

$$\omega_i(x,y) = (w_i \cdot h_i)^{-q} \left(\sqrt{(x-x_c)^2 + (y-y_c)^2}\right)^{-p}, \qquad (3)$$

where $(x_c, y_c)$ is the pixel location of the center of trait $t_i$. We enforce $\omega_i(x_c, y_c) = (w_i \cdot h_i)^{-q} \left(\sqrt{2}\right)^{-p}$. According to (3), pixels in the central region of a trait have larger weights than those in the peripheral region, and with $p > 1$ such difference in weights is further enlarged. In this way we can effectively eliminate the *blocking effect*[4] when patterns are mosaicked. Furthermore, $q$ is the coefficient that gives smaller patterns larger weights when $q > 1$. Given the above notations, the proposed random legitimate codeword extraction algorithm is given in Algorithm 1.

In Algorithm 1, $I(x,y)$ is the *face template* which gives the rendering result in the end, and $\omega_I(x,y)$ is the weight mask for $I(x,y)$. $C(t_{i,j}, I)$ is defined in equation (1), note that this time we treat $I$ itself as a 'big' pattern. Patterns of different traits are added to $I$ in turns, and the bigger ones go first. In each turn only the patterns that are similar to the overlapped region of the current $I$ in appearance are considered as a candidate in the random selection process.

[3] The value of the $\epsilon$ is empirically determined so that legitimate codewords render visually pleasing smooth faces.

[4] The mosaic of patterns creates artificial edges along the pattern boundaries.

---

**Algorithm 1** Random legitimate code extraction and face synthesis

```
Require: T = {t_1, t_2, ..., t_N}; t_{i,j}, i = 1 ~ N, j = 1 ~ k_i;
    W = {ω_1, ω_2, ..., ω_N}; ε, a predefined threshold.
Ensure: a legitimate codeword c and a human face I.
 1: I = 0; ω_I = 0; c = {c_1 = 0, c_2 = 0, ..., c_N = 0}.
 2: Randomly select a pattern of trait t_1, assume it is t_{1,j}.
 3: c_1 = j.
 4: I(x,y) = I(x,y) + t_{1,j}(x,y)|_{Ω_{t_1}}.
 5: ω_I(x,y) = ω_I(x,y) + ω_1(x,y)|_{Ω_{t_1}}.
 6: for i = 2 to N do
 7:   P = {∅}.
 8:   for j = 1 to k_i do
 9:     if C(t_{i,j}, I) < ε then
10:       P = P ⋂ t_{i,j}.
11:     end if
12:   end for
13:   if P = {∅} then
14:     Abort the current trial. Goto step 1.
15:   else
16:     Randomly select a pattern from set P, assume it is t_{i,j}.
17:     c_i = j.
18:     I(x,y) = I(x,y) + t_{i,j}(x,y)|_{Ω_{t_i}}.
19:     ω_I(x,y) = ω_I(x,y) + ω_i(x,y)|_{Ω_{t_i}}.
20:   end if
21: end for
22: I(x,y) = I(x,y)/ω_I(x,y)|_{ω_I(x,y)≠0}.
23: Perform Histogram Equalization on I.
```

With $q > 1$ in (3), Algorithm 1 presents a *coarse-to-fine* face synthesis scheme. The later added smaller patterns can be regarded as the *refinement* to the current face template. Figure 3 illustrates 40 legitimate novel faces. Note there are artifacts in lower-right corner of some faces, owing to the fact this region is covered by very few traits (see figure 1 (b)). This is the direct result of the FTC trait selection, owing to the poor discriminability of this facial region [6]. Also note the diversity in appearance among these faces.

## 4. AN APPLICATION: CREATING SUSPECT'S PHYSICAL PROFILES

In the previous section we demonstrated the capability of FTC for generating novel human faces based on legitimate codewords. The degree of the similarities among these novel faces are actually related to their *codeword distances*. The distance of two codewords is simply the number of digit locations that the two codewords are different [6]. Face pair with smaller codeword distance are more similar, since they share more patterns in common, and vise versa. Based on this property, we proposed a GUI prototype which help the eyewitnesses to create suspect's physical profiles efficiently, without the aid of face painters. This application is most useful in the law enforcement environment. The GUI is implemented using Matlab. Figure 4 gives the appearance of this GUI, and its features are listed below.

1. There is an array of 5-by-5 *units* in the GUI. Each unit displays a facial image. The Unit in the center location is called the *central unit*; 8 units surround the central unit are call the *inner units*; the rest 18 units are called the *outer units*.
2. Initially the *mean face* is displayed in the central unit.
3. The 8 inner units display faces whose codewords are identical

**Fig. 3**. Examples of randomly generated faces.

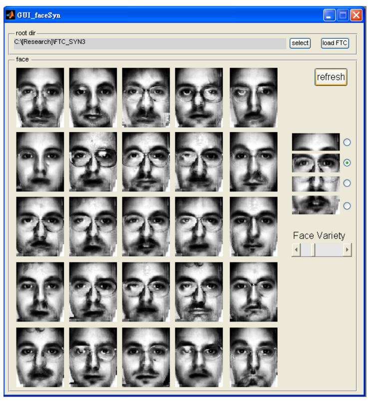


**Fig. 4**. The GUI for face profiling of suspects.

to that of the central face for the first $D$ leading digits. Basically these faces are similar to the central face in the coarse view, but are different in some details.[5] The number $D$ is determined by the "**Face Variety**" slider to the right of the GUI.

4. The 18 outer units displays faces whose codewords are identical to the codeword of the central face for the first $\frac{D}{2}$ leading digits. Basically these faces are less similar to the central face than those in the inner units.
5. By clicking any face unit, except the central one, this face will be placed in the central unit. Meanwhile, the rest of faces will be updated accordingly.
6. By clicking the "**refresh**" button, faces in the inner and outer units will be randomly generated again, without changing the central face.
7. To the right of the panel also shows the four overlapped horizontal segments of the current central face. By checking the radio button to the right of each segment, the patterns that overlapped with the corresponding facial segment will have reduced probabilities to be altered in the later random face generating process.

Figure 5 shows some suspect's physical profiles created using the proposed GUI. We 'pretend' faces in the odd columns, which comes from $S_2$, exist only in the mind of eyewitnesses, and utilize the GUI

[5]According to the construction in section 3.1, the leading digits correspond to larger traits.

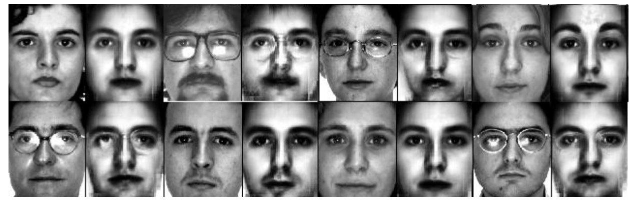

**Fig. 5**. Face profiling examples. The odd columns are given faces. The even columns are faces profiled using our GUI.

to try to generate similar ones in the even columns. Basically the resulting faces are reasonably similar to the given faces. This application also demonstrates the capability of the FTC in extrapolating human faces.

## 5. CONCLUSION AND FUTURE WORK

This work demonstrates the applicability of FTC to face synthesis task. The proposed coarse-to-fine synthesis scheme allows us to change facial appearance in a coarse view or in details. The similarities between the synthesized faces can be measured in a more systematic way using the associating codeword distances, and it is what most existing face synthesis approaches lack. With the above features, we have implemented a GUI prototype that can help eyewitnesses to create suspect's physical profiles effectively.

In our future work, more computer graphics techniques, such as image blending, will be incorporated to further improve the synthesis quality. Meanwhile, we will treat eyeglasses and hair styles as accessories and exclude them from trait patterns extraction. Furthermore, the concept of the codeword legitimacy proposed in this work can potentially benefit the face recognition approach using FTC. We may be able to correct some error digits during the FTC encoding process using the appearance consistency measure between encoded patterns.

## 6. ACKNOWLEDGEMENT

This work was supported in part by the Excellent Research Projects of National Taiwan University, under grant 97R0062-04.

## 7. REFERENCES

[1] Volker Blanz and Thomas Vetter, "A morphable model for the synthesis of 3d faces," in *SIGGRAPH*, 1999, pp. 187–194.

[2] Shiguang Shan, Wen Gao, Jie Yan, Hongming Zhang, and Xilin Chen, "Individual 3d face synthesis based on orthogonal photos and speech-driven facial animation," *ICIP*, vol. 3, pp. 238–241 vol.3, 2000.

[3] Lei Zhang, Sen Wang, and Dimitris Samaras, "Face synthesis and recognition from a single image under arbitrary unknown lighting using a spherical harmonic basis morphable model," in *CVPR*, 2005, pp. 209–216.

[4] Yu Zhang and Shuhong Xu, "Data-driven feature-based 3d face synthesis," *Sixth International Conference on 3-D Digital Imaging and Modeling*, pp. 39–46, 2007.

[5] Yasar Guneri Sahin, Samsun Mustafa Basarici, and Tuncay Ercan, "Face matrix: A quick search and indexing method for suspect recognition in police departments," *Information Technology Journal*, vol. 6, no. 4, pp. 607–612, 2007.

[6] Ping-Han Lee, Gee-Sern Hsu, Tsuhan Chen, and Yi-Ping Hung, "Facial trait code and its application to face recognition," in *4th International Symposium on Visual Computing*, 2008, vol. 5359, pp. 317–328.

[7] A.M. Martinez and R. Benavente, "The ar face database," Tech. Rep. 24, CVC, 1998.