Face Verification and Identification using Facial Trait Code

Ping-Han Lee
Graduate Institute of Computer Science and Information Engineering, National Taiwan University
d93922027@ntu.edu.tw

Gee-Sern Hsu
Department of Mechanical Engineering, National Taiwan University of Science and Technology
jison@mail.ntust.edu.tw

Yi-Ping Hung
Graduate Institute of Networking and Multimedia, National Taiwan University
hung@csie.ntu.edu.tw

Abstract

We propose the Facial Trait Code (FTC) to encode human facial images. The proposed FTC is motivated by the discovery of some basic patterns existing in certain local facial features. We call these basic patterns Distinctive Trait Patterns (DTP), which can be extracted from a large number of faces. We have also found that the fusion of these DTP's can accurately capture the appearance of a face. The extraction of DTP involves clustering and boosting for maximizing the discrimination between human faces. The extracted DTP's can be symbolized and used to make up the n-ary facial trait codes. A given face can be encoded at some prescribed facial traits to render an n-ary facial trait code with each symbol in its codeword corresponding to the closest DTP. We applied FTC to a face identification and verification problems with 3575 facial images from 840 people under different illumination conditions, and it yielded satisfactory results.

1. Introduction

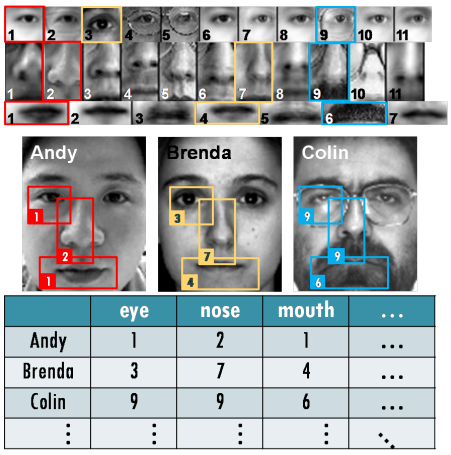
Most 2D face recognition algorithms can be classified into global approach, part-based approach, and some hybrid of both. A comprehensive survey is given in [25]. The global approach considers the whole facial area, mostly from eyebrows to chin and ear to ear; while the part-based extracts some local features, mostly eyes, nose, mouth, and maybe others. Part-based approaches often start with some feature extraction methods and are followed by classification schemes. Liao and Li [14] extracted 17 local features using the Elastic Graph Matching (EGM), and each of these 17 features has its own specific spot on a face, for example, the corners of eyes, the ends of eyebrows, and the centers of lips. Deformable graphs and dynamic programming are used in [1] to determine eyes, nose, mouth, and chin. A two-level hierarchical component classifier is proposed in [10] to locate 14 feature patches in a face, and [9] shows that face recognition using these 14 feature patches outperforms the same recognition method but using the whole face as a global feature. Ivanov et. al. [11] extended this study by experimenting with a few different recognition schemes using this same set of 14 feature patches. Few have different perspective toward the definition of such local features, as the features are perceivable to our nature, and many part-based approaches have yielded promising performance. More part-based approaches can be found in [3] [19].

Some imperceivable local features, however, have been emerging in recent years that have created some additional dimension for the computer analysis of human faces. Using the Adaboost algorithm, Viola and Jones [22] developed a highly effective real-time face detector that utilizes a large number of patches of different sizes, orientations, and locations across a face as the features for discriminating faces from non-faces. Similar features are used again in [12] for face recognition. To study the properties of these patch features, we have carried out some experiments and found the following interesting observations:

1. Some typical patterns seem to exist in many patch features, and the number of patterns in one patch feature can vary from one feature to another.
2. If the patterns in each patch feature are numerated, a face can be represented by a series of numbers, each of which shows the specific pattern of a specific patch feature on that face. More importantly, the series of numbers seems different for a different face.

These observations motivated our development of Facial Trait Code (FTC). From an exhaustive set of the patch features collected from a large number of faces, FTC extracts those with relatively stronger strength in discriminating different faces. The extracted patch features are called *facial traits*, and each facial trait has its own specific location, size, and orientation on a face. The patterns in each facial trait are determined using some clustering approach, and the resulting patterns are then numerated. Given a face, one can

978-1-4244-3991-1/09/$25.00 ©2009 IEEE



	eye	nose	mouth	...
Andy	1	2	1	...
Brenda	3	7	4	...
Colin	9	9	6	...
⋮	⋮	⋮	⋮	⋱

Figure 1. Illustration of Facial Trait Code.

encode it using the numbers of the patterns at all facial traits that best describes the appearance of this face. These numbers constitute the facial trait code for this face. An illustrative example is given in Figure 1, where three facial traits are shown in each of the three faces. According to FTC, the faces can be encoded into [1, 2, 1], [3, 7, 4], and [9, 9, 6], respectively.

Putting the facial trait patterns into codes has the following advantages:

1. A face can be effectively denoted by a finite length of codeword in which each symbol gives not just a specific facial trait with a fixed size and location, but also the pattern in this facial trait that best describes the face in that specific facial trait. This implies that the FTC offers an effective descriptor to a given face.

2. Face identification and verification problems can be formulated as code matching problems, and thus some merits from coding perspective can be preserved. Error correcting is one such merit needed in the development of the FTC, and more details will be given subsequently.

3. Coding consists of encoding and decoding. The former transforms a face into a codeword, and the latter, according to the proposed FTC, transforms a codeword to a subject in the gallery for face identification. This is, however, just an option in the FTC decoding. Our on-going research show that FTC decoding can be made to generate faces with different levels of similarity and some caricature faces that preserve certain features of a real face. This implies that the FTC may also be useful for other applications, for example in animation or entertainment areas.

1.1. Related Works

A few works were proposed that put together coding and facial recognition. Kittler et al [13] and Windeatt et at [23] applied an *Error Correcting Output Coding (ECOC)* approach for face verification. This work shows that ECOC can decompose a multi-classification problem into a set of complimentary binary classification problems solvable by, for example, MLP (Multi-Layer Perceptron) binary classifiers. The input to the binary classifiers are the holistic facial features extracted using Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA). The output of the binary classifiers defines the ECOC feature space, where the patterns of different faces are claimed to be well separated. Followed by [13], Xie and Kumar proposed Face Class Code (FCC) [24] to encode each subject's facial class into a binary string. They designed classifiers to discriminate '1' or '0' for each bit in the binary string for the determination of the class label. [24] aimed to fix the computationally expensive one-classifier-per-subject problem when the number of subjects is large. Given an N-subject recognition task that generally requires N binary classifiers, FCC only creates $log_2(N)$ binary classifiers.

The proposed FTC is distinctive from all of the previous works upon coding for facial recognition in following aspects. Firstly, the codes are developed from *unperceivable and local* features. Secondly, each symbol in a codeword denotes some specific pattern existing in a facial trait, i.e., a local rectangle patch of a certain size and location. Thirdly, the FTC is an n-ary codeword instead of the binary ones in all the previous works. This paper begins with the specification of the FTC in Section 2 that elaborates how the facial trait codes are determined and specified from a large collection of faces. When the specification of the FTC is determined, one can encode a given facial image and decode a FTC codeword for face recognition, and the details are given in Section 3. The experimental setup and the databases used for the performance evaluation is given in Section 4. This paper ends at a conclusion in Section 5.

2. Extraction of Facial Traits and Their Patterns

Given a facial image, one can specify a local **patch** by a bounding box $\{x, y, w, h\}$, where x and y are the 2-D pixel coordinates of this bounding box's upper-left corner, and w and h are the width and height of this bounding box, respectively. If this bounding box is moved from left to right and top to bottom in the face with a step size of Δx and Δy pixels in each direction, one can obtain many patches with the same size but different locations. If w and h can further change from some small values to large values, we will end up with some exhaustive set of local patches across the face. Some similar set of such an exhaustive collection of local

Figure 2. Illustration of some of the patches resulting from the local patch selection scheme. The highlight areas in each face indicates the location of the patches.

patches was used in [22] to determine the features good for facial detection.

It can be seen that the number of the patches grows with (1) the size of the facial image, (2) the size range of local patches, and also (3) the reduction in the step size. The above items (2) and (3) are *size-and-step parameters*. For a given facial image, these size-and-step parameters can be appropriately determined based on the spatial-frequency contents of the facial image. Some of our research upon the spatial frequency analysis of facial images are underway, and the outcome will be reported later. The size-and-step parameters in this paper are determined from some experimental observations showing that the characteristics of a face are mostly in the low spatial frequencies. These experiments have determined the following size-and-step parameters for the local patches:

1. The patches with sizes less than 6% of the whole face are ignored because these small patches carry less frequency contents of the face and more high frequency noises.

2. The patches with sizes between 6% and 16% of the whole face are obtained with step sizes $1/3$ of the width w or the height h.

3. The patches with sizes larger than 16% of the whole face are obtained with step sizes $1/5$ of the width w or the height h.

The above scheme will result in a couple thousands of patches, some of them are overlapped with each other, for a face with 80x100 pixels in size. Figure 2 illustrates some of the resulting patches.

2.1. Pattern Clustering in Each Patch

Given a stack of K frontal facial images aligned by the centers of both eyes, we can crop a stack of K corresponding patch samples for each patch resulting from the above local patch selection scheme. To cluster the K patch samples in each patch stack, we first extract their features using the Principal Component Analysis (PCA). Considering the case that the K facial images can be from L subjects ($L \leq K$, i.e., one subject can have multiple facial samples), in each patch stack we calculate the L mean facial images for L subjects, and the Linear Discriminant Analysis (LDA) is applied to determine the L most discriminant low dimensional patch features. 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 proposed by Figueiredo and Jain [7] can be applied to identify the MoG patterns in each patch stack.

The method given by Figueiredo and Jain [7], abbreviated as FJ algorithm in the following, can determine the number of clusters in a mixture model for a given data set, and avoid the drawbacks of the standard Expectation Maximization (EM) algorithm such as the sensitivity to initialization and possible convergence to the parameter space boundary. Instead of the often usual approach by choosing one from a set of candidate models with different numbers of clusters, their method combines parameter estimation and model selection in one single algorithm. If there are M patch stacks, then the FJ algorithm can be applied to 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 are considered the patterns existing in the i-th patch stack, and will be called **patch patterns**.

2.2. From Patches to Facial Traits

Basically M, the number of the local patches over a face, is large. However, if the effectiveness of each *patch* in the discrimination of the subjects whose faces were used to determine the overall *patch patterns* can be compared and ranked, this can give a way to reduce M. Those patches with their patterns that are good at discriminating the subjects will be called **Facial Trait**, and the corresponding patch patterns will be called **Distinctive Traits Patterns**.

To extract the facial trait, one must come up with a measure able to assess each patch's effectiveness for face discrimination. This measure can be defined via a matrix, called **Patch Pattern Map** or PPM for short. PPM is defined for each different patch, and it shows which subjects' faces reveal the same pattern at that specific local 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 subjects, and the entry at (p, q), denoted as $PPM_i(p, q)$, is defined as follows:

$$PPM_i(p,q) = \begin{cases} 0 & \text{if the patches on the faces of the } p\text{-th and the } q\text{-th subjects are clustered into the same patch pattern} \\ 1 & \text{otherwise} \end{cases}$$

Given N patches and their associated PPM_i's stacked to form a $L \times L \times N$ dimensional array, there are $\frac{L(L-1)}{2}$ N-dimensional binary vectors along the *depth* of this array because each PPM_i is symmetric matrix and one can 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 subjects in terms of these N local patches. More unities in $v_{p,q}$ indicates more differences between this pair of subjects, and on the contrary, more zeros shows more similarities.

The binary vector $v_{p,q}$ motivates our application of the Error Correcting Output Code (ECOC) [6] to this research. If each subject's face is encoded using the most discriminant patches, or 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 propose an Adaboost algorithm to maximize the d_{min}, the minimum Hamming distance, for the determination of the facial traits from the overall patches. This algorithm is summarized in Algorithm 1.

Algorithm 1 Extraction of Facial Trait Patterns

Require: PPM_i, $i = 1 \sim M$

Ensure: selected N facial traits that yield maximum d_{min}

$F = \{$the set of M patches$\}$; $\hat{F} = \{\emptyset\}$

$C(p,q) = 0$; $\omega(p,q) = 1$, where $p = 1 \sim L$ and $q = 1 \sim L$

for $t = 1$ to N **do**

Normalizing the weight $\omega(p,q) = \frac{\omega(p,q)}{\sum_{p,q}\omega(p,q)}$.

For every element $f_i \in F$, $\alpha(f_i) = \sum_{p,q} PPM_i(p,q)\omega(p,q)$

Select $\hat{f}_t = \arg\max_i \alpha(f_i)$

Update $C(p,q) = C(p,q) + PPM_i(p,q)$

Calculate $d_{min}(d_{max})$, which is the minimum(maximum) element in $C(p,q)$

$$\omega(p,q) = \begin{cases} \text{L} & \text{if } C(p,q) = d_{min} \\ 0 & \text{if } C(p,q) = d_{max} \\ 1 & \text{otherwise} \end{cases}$$

Update sets $F = F - \hat{f}_t$ and $\hat{F} = \hat{F} \bigcup \hat{f}_t$.

end for

In Algorithm 1, F is the set of the overall patches, initially contains M patches. $\hat{F}$ is the set of selected facial traits, which will finally reach N traits in total. C is a $L \times L$ dimensional array where $C(p,q)$ gives the number of ones in $v_{p,q}$. ω is a weight array with the same dimension as that of C, and $\omega(p,q)$ is the weight for the subject pair p and q. In each run, the patch able to maximize the updated d_{min} is selected as one new facial trait.

The N facial traits with their trait patterns *symbolized* from $1, 2, ..., N$ define the basic structure of the proposed Facial Trait Code. Each codeword in the FTC is of length N and n-ary where n is the largest number of the trait patterns found in one single trait. And the smallest distance between codewords is d_{min}. In summary, given a set of frontal faces, we can define N facial traits, $\sum_{j=1}^{N} k_j$ trait patterns, and $\prod_{j=1}^{N} k_j$ faces (or FTC codewords).

3. FTC Encoding, Decoding, and Application to Face Recognition

The following facial sets need to be defined so that one can apply the proposed FTC for face recognition.

1. **Trait Extraction Set**: A large collection of *neutral* frontal faces which best covers both genders and a wide range of ages, races, and possibly other parameters. Those with variations caused by poses, illuminations, and expressions are excluded. The facial trait patterns and thus the facial trait codes are defined upon such a set. This set specifies (1) the number of facial traits, and thus the codeword length for a given face; (2) the patterns in each facial trait, and thus the range of each symbol in a codeword; (3) the location and the size of each facial trait.

2. **Trait Variation Set**: This is the set that encompasses the images with the variations excluded in the trait extraction set. This set does not alter any specifications given by the trait extraction set, but substantially enriches the spectrum of each predetermined trait pattern by adding in samples with variations.

3. **Gallery Set**: This is the set that those enrolled used to register their faces to a face recognition system. It is allowed in the FTC and other algorithms that images with the aforementioned variations can be included.

4. **Probe Set**: This is a disjoint set from the gallery set, and is used to test the recognition rate and other performance indices. This set includes the images of those enrolled but taken at different time and conditions, and also the images of those not enrolled. It often includes images with a tremendous amount of the aforementioned variations.

With a pre-selected length of the FTC codeword, N, the trait extraction set defines N facial traits of different sizes, orientations, and locations, and also the patterns in each facial trait. Each facial trait pattern is tagged with a number, which will be used as a symbol in the FTC codeword. In

the FTC encoding, a given face is firstly decomposed into N patches according to the specifications given by the N facial traits, and each patch is then classified into a specific facial trait pattern and numbered as the pattern tag. An SVM (Support Vector Machines) classifier is used for the patch classification[1], and it is trained using both the trait extraction set and the trait variation set for encompassing possible variations. The given face is therefore encoded into a n-ary FTC codeword of length N.

In practice the images in the gallery set are firstly encoded into **gallery codes**. Given a probe, an image from the probe set, it is also firstly encoded into a **probe code**. The FTC **decoding** matches this probe code against all gallery codes, and finds the 'closest' one using the Hamming distance as the measure.

Given two codewords $\mathbf{g}_c = [g_1 g_2 ... g_N]$ and $\mathbf{p}_c = [p_1 p_2 ... p_N]$, the Hamming distance can be easily interpreted using the code difference $\mathbf{d}_c = [d_1 d_2 ... d_N]$ where

$$d_i = \begin{cases} 0 & if\ p_i = g_i \\ 1 & otherwise. \end{cases}$$

Then the Hamming distance between $\mathbf{g}_c$ and $\mathbf{p}_c$ is given by the following,

$$D(\mathbf{g}_c, \mathbf{p}_c) = \sum_{i=1}^{N} d_i. \quad (1)$$

4. Performance Evaluation–A Comparative Study

4.1. Face Samples under Different Illumination Conditions

To diversify the face samples in our experimental study, we selected samples from some commonly used databases, namely AR [15], FERET [18], FRGC [17], PIE [20], XM2VTS [16], and an in-house one, called FVI (Face, Voice and Iris) database. Only frontal faces with illumination variations were considered, and the whole selected dataset consists of 840 subjects with 3575 facial images. The details of the samples selected from the 6 databases are given in Table 1.

All facial images were aligned to the centers of the eyes, and normalized to 80x100 pixels in size. About 34%[2] of the 3575 samples were considered with significant variations caused by illumination. Figure 3 shows a few samples with variations caused by illumination. To alleviate the impacts made by illumination variations, all samples were processed to have mean 128 and variance 10.

[1] We used libsvm [5] and applied the C-SVC SVM with a RBF kernel. The associated parameters, C and γ were determined using the *grid search* strategy for each trait.

[2] 34% gives 1216 facial samples.

Database	#subjects	#faces	#face/subject
AR	126	1008	8
FERET	200	400	2
FRGC	191	876	$1 \sim 18$
FVI	34	283	$1 \sim 18$
PIE	43	344	8
XM2VTS	246	664	$1 \sim 4$
total	840	3575	4.26

Table 1. Face samples selected from 6 benchmark databases.

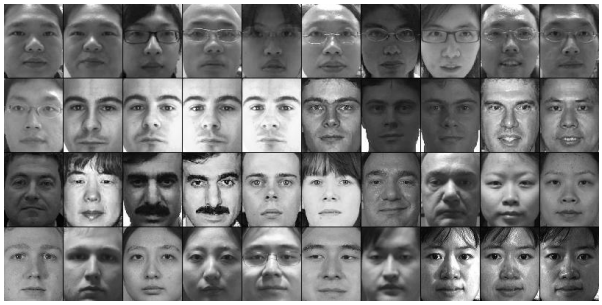


Figure 3. Facial samples used in our experimental study.

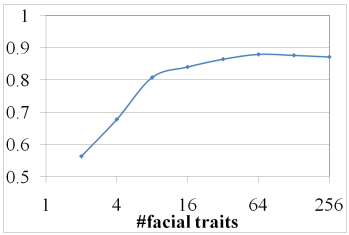
4.2. Test Scenarios and Performance Comparison

To evaluate the FTC's performance, two typical face recognition tasks were carried out: identification and verification. In identification, each probe image had one unique match to identify in the gallery set. In verification, each probe image with a claimed subject were both presented to the verification algorithm, which would either accept or reject the claim. A claim would be rejected when the probe failed to match the claimed subject, no matter whether the subject of the probe was in the gallery set or not.

To compare the FTC's performance with others, we implemented two methods using local facial features, *Heisele03* [9] and *Ahonen06* [2], one algorithm using ECOC, *Kittler01* [13], and two baseline methods, *Eigenface* [21] and *Fisherface* [4]. The implementation of the ECOC method [13] applied [127, 15, 27] binary BCH code to generate the codewords. This 127-bit code was able to correct up to 27 error bits. The FTC codeword was also made to have 127 symbols from the 127 facial traits.

The experiments were carried out with the following four setups with different partitions of the overall dataset.

1. *Overall Identification*: The whole set of 3575 facial images was divided into two disjoint sets, $\mathbf{S}_g$ and $\mathbf{S}_p$, separated by the time when the images were taken. $\mathbf{S}_g$ contained 1895 images from the overall 840 subjects and $\mathbf{S}_p$ contained 1680 images from 777 subjects. The subjects in $\mathbf{S}_p$ were all in $\mathbf{S}_g$, and those with only one image available were all in $\mathbf{S}_g$. $\mathbf{S}_g$ was used as the gallery set and $\mathbf{S}_p$ as the probe set.



(a)

Figure 4. FTC identification performance with different numbers of facial traits.

2. *Subset Identification*: To study the identification performance with different sizes of datasets, three subsets were randomly selected from $\mathbf{S}_g$. $\mathbf{S}_{g1}$ had 100 subjects, $\mathbf{S}_{g2}$ had 200 subjects, and $\mathbf{S}_{g3}$ had 400 subjects. The three corresponding probe sets, $\mathbf{S}_{p1}$, $\mathbf{S}_{p2}$, and $\mathbf{S}_{p3}$ were also selected from $\mathbf{S}_p$. This test was repeated for more than 20 times to obtain some average statistics.

3. *Subset Verification*: The three probe sets $\mathbf{S}_{p1}$, $\mathbf{S}_{p2}$, and $\mathbf{S}_{p3}$ were used along with three disjoint but equal-sized probe sets, $\mathbf{S}_{q1}$, $\mathbf{S}_{q2}$, and $\mathbf{S}_{q3}$, respectively, to match against $\mathbf{S}_{g1}$, $\mathbf{S}_{g2}$, and $\mathbf{S}_{g3}$. $\mathbf{S}_{q1}$, $\mathbf{S}_{q2}$, and $\mathbf{S}_{q3}$ were used as *imposters* who were trying to break in using false identities. This test was also repeated for more than 20 times.

4. *Generalization Test*: This test was designed to evaluate the generalization performance of the FTC. The 840 subjects were randomly selected to form three disjoint sets, $\mathbf{V}_1$ had 440 subjects, $\mathbf{V}_2$ had 200 subjects, and $\mathbf{V}_3$ had the rest 200 subjects. $\mathbf{V}_2$ was further partitioned into two disjoint subsets $\mathbf{V}_{2a}$ and $\mathbf{V}_{2b}$, which had the same subjects but with images taken at different time. $\mathbf{V}_1$ was used as the trait extraction set and trait variation set in the FTC, and also as the basis extraction set for the approaches as PCA and LDA. When the basis components were defined by $\mathbf{V}_1$, the faces in $\mathbf{V}_{2a}$, $\mathbf{V}_{2b}$ and $\mathbf{V}_3$ would be decomposed accordingly. $\mathbf{V}_{2a}$ was then used as the gallery set, $\mathbf{V}_{2b}$ as the probe set, and $\mathbf{V}_3$ as another probe set formed by imposters only. $\mathbf{V}_{2a}$ and $\mathbf{V}_{2b}$ were used in identification test, and with $\mathbf{V}_3$ added in for verification test. This test was repeated for at least 20 times for 4 different gallery sizes, 20, 40, 80, and 160.

With the first setup *Overall Identification*, figure 4 shows the the FTC's performance with different numbers of facial traits, and thus with different length of codeword. It can be seen that the identification rate increases with the number of the facial traits, and it reaches some upper bound between 64 and 128 traits. If 127 traits were selected, we would end up with a $[127, 840, 41]$ FTC[3], which could correct up to 41 error digits in the 127 symbol positions. The BCH code proposed in *Kittler01* in our experiment could only correct 27 error bits. We also found that the number of trait patterns in each different trait was typically from 15 to 30. A few major facial traits with their patterns are shown in figure 5.

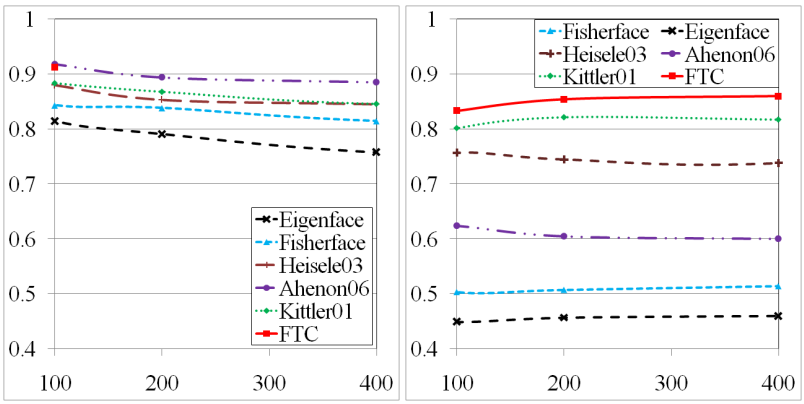
Figure 5. An example of the FTC codebook.

Figure 6 (a) and (b) gives results for the *Subset Identification* and *Subset Verification*, respectively. The verification performance was evaluated as the *Hit Rate* when the *False Acceptance Rate* was 0.001. The most difficult test scenario was given by the *Generalization Test*, and the identification and verification rates are shown in Figure 7 (a) and (b), respectively. In both Figure 6 and 7, we use 127 facial traits for the proposed FTC method and a 127-bit BCH code for the algorithm *Kittler01*.

Our experiments have shown that the ECOC algorithm *Kittler01* performs well in most setups except the most challenging *Generalization Test*. This is because its binarization at each bit is determined solely by the BCH coding method, resulting in dichotomies irrelevant to the appearance of a human face. It sure fails in the generalization test. In all our experiments, FTC outperformed other methods, including *Ahonen06*, which drew some attention in the face recognition community recently. FTC performed better than *Ahonen06* in the subset(generalization) test for $23.9\%(15.8\%)$ higher in the verification rate.

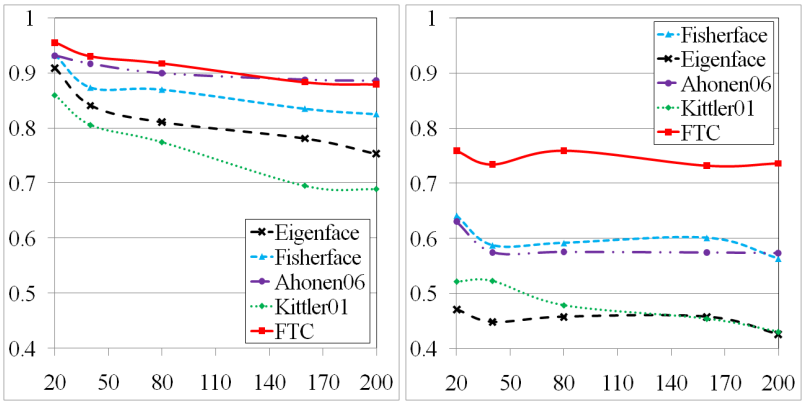
Table 2 summarizes the performance of all algorithms evaluated in our experiments in terms of average identification/verification rates. *Heisele03* were not feasible for the study of the generalization performance, and thus excluded in this table.

[3]We denote a $[N, M, \frac{d_{min}-1}{2}]$ FTC as a code with M valid n-ary codewords of length N. And the smallest distance between these M codewords is d_{min}.



(a) Subset Identification (b) Subset Verification

Figure 6. Experimental results. The x-axis is the number of subjects. The y-axis is the identification rate for (a), and verification rate for (b).



(a) Generalization for Identification (b) Generalization for Verification

Figure 7. Experimental results. The x-axis is the number of subjects. The y-axis is the identification rate for (a), and verification rate for (b).

	Subset		Generalization	
algorithm	ident.	verif.	ident.	verif.
Eigenface [21]	78.8%	45.5%	81.9%	45.3%
Fisherface [4]	83.2%	50.8%	86.7%	59.7%
Kittler01 [13]	86.5%	81.3%	76.5%	48.2%
Heisele03 [9]	85.9%	75.7%	n/a	n/a
Ahonen06 [2]	**89.9%**	61.0%	90.4%	58.6%
FTC	**89.9%**	**84.9%**	**91.3%**	**74.4%**

Table 2. Performance summary of all tested algorithms.

5. Conclusion and Future Work

We propose a novel human face coding scheme called Facial Trait Code in this paper. The proposed code captures the major patterns among local appearance of human faces, and encodes faces into n-ary codewords accordingly. The proposed code can be seen as a type of Error Correcting Code, and its construction follows the principle of maximizing the error correcting capability. In the code space of the proposed code, codewords of different humans are made far away from each other. The resulting discriminative codewords can be regarded as a new descriptor for human faces, and it yield promising results for face verification and identification problems.

As mentioned in introduction, the decoding scheme described in this paper is just an option for the FTC decoding. For the face synthesis purpose, based on the fact that each number in a given codeword refers to a trait pattern, we can renders a facial image by mosaicing these trait patterns from different locations and sizes. The mosaicing involves averaging if a pixel location is overlapped by multiple patches, and smoothing at the edges of the patches to produce a seamless face. We are current working on the application of FTC to face synthesis.

Our future works also include handling faces under

larger out-of-plane rotations. One better solution is to use 3-D facial data. It is worthy of mention that the proposed algorithm can be extended to incorporate 3-D facial data straightforwardly. The whole algorithm remains the same, except that the raw features become range data instead of image intensities.

Acknowledgement

This work is supported in part under the grant 97-EC-17-A-02-S1-032.

References

[1] J. Ahlberg. Facial feature extraction using deformable graphs and statistical pattern matching. In *in Swedish Symposium on Image Analysis, SSAB*, 1999.

[2] T. Ahonen, A. Hadid, and M. Pietikainen. Face description with local binary patterns: Application to face recognition. In *PAMI*, pages 2037–2041, 2006.

[3] E. Bart, E. Byvatov, and S. Ullman. View-invariant recognition using corresponding object fragments. In *Proceedings of the Seventh European Conference on Computer Vision*, pages Vol II: 152–165, 2004.

[4] P. Belhumeur, J. Hespanha, and D. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *PAMI*, 19(7):711–720, 1997.

[5] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[6] T. G. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263–286, 1995.

[7] M. Figueiredo and A. Jain. unsupervised learning of finite mixture models. *PAMI*, 24:381–396, 2002.

[8] V. Guruswami and A. Sahai. Multiclass learning, boosting, and error-correcting codes. In *Proceedings of the twelfth annual conference on Computational learning theory*, pages 145 – 155, 1999.

[9] B. Heisele, P. Ho, J. Wu, and T. Poggio. Face recognition: component-based versus global approaches. *CVIU*, 91(1):6–12, 2003.

[10] B. Heisele, S. Thomas, P. Sam, and T. Poggio. Hierarchical classification and feature reduction for fast face detection with support vector machines. *Pattern Recognition*, 36(9):2007–2017, 2003.

[11] Y. Ivanov, B. Heisele, and T. Serre. Using component features for face recognition. In *FGR'04*, page 421, 2004.

[12] M. J. Jones and P. Viola. Face recognition using boosted local features. Technical report, 2003.

[13] J. Kittler, R. Ghaderi, T. Windeatt, and J. Matas. Face verification using error correcting output codes. In *CVPR*, volume 1, pages I–755– I–760, 01.

[14] R. Liao and S. Z. Li. Face recognition based on multiple facial features. In *In Proc. of the 4th IEEE Int. Conf. on Automatic Face and Gesture Recognition*, pages 239–244. Dekker Inc, 2000.

[15] A. Martinez and R. Benavente. The ar face database. Technical Report 24, CVC, 1998.

[16] K. Messer, J. Matas, J. Kittler, J. Luettin, and G. Maitre. Xm2vtsdb: The extended m2vts database. In *AVBPA*, 1999.

[17] P. Phillips, P. Flynn, T. Scruggs, K. Bowyer, J. Chang, K. Hoffman, J. Marques, J. Min, and W. Worek. Overview of the face recognition grand challenge. In *CVPR*, pages 947–954, 2005.

[18] P. J. Phillips, H. Moon, S. A. Rizvi, and P. J. Rauss. The FERET evaluation methodology for face-recognition algorithms. *PAMI*, 22(10):1090–1034, 2000.

[19] M. Savvides, R. Abiantun, J. Heo, S. Park, C. Xie, and B. Vijayakumar. Partial and holistic face recognition on frgc-ii data using support vector machine. In *Proceedings of the 2006 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW06)*, pages 48–48, 2006.

[20] T. Sim, S. Baker, and M. Bsat. The CMU pose, illumination, and expression (PIE) database of human faces. Technical Report CMU-RI-TR-01-02, Carnegie Mellon University, 2001.

[21] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

[22] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *CVPR*, volume 1, pages 511–518, 2001.

[23] T. Windeatt and G. Ardeshir. Boosted ecoc ensembles for face recognition. In *VIE*, pages 165 –168, 2003.

[24] C. Xie and B. V. Kumar. Face class code based feature extraction for face recognition. In *Fourth IEEE Workshop on Publication Automatic Identification Advanced Technologies*, pages 257–262, 2005.

[25] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld. Face recognition: A literature survey. *ACM Comput. Surv.*, 35(4):399–458, 2003.

Appendix: Error-Correcting Property of the ECOC

FTC inherits the error-correcting capability from the ECOC. An (N, L, d_{min}) ECOC $\mathcal{C}$ is a set of L binary vectors of dimension N, called **codewords**, such that the Hamming distance between every pair of distinct codewords is at least d_{min}. This code can correct at least $(\frac{d_{min}-1}{2})$ error bits. Since every codeword has distance at least d_{min} from every other codeword, the closed Hamming balls of radius $(\frac{d_{min}-1}{2})$ around each codeword are disjoint. Hence if a binary vector y differs from some codeword $x \in \mathcal{C}$ in at most $(\frac{d_{min}-1}{2})$ bit positions, then x is still the unique closest codeword in $\mathcal{C}$ to y [8]. Consequently, an ECOC with larger d_{min} is able to correct more error bits. It also means that the L codewords in $\mathcal{C}$ are well separated in the Hamming space of $\mathcal{C}$.