State of the Art: Signature Biometrics Verification

Mohamed Soltane
Electronics Department, Faculty of Engineering Science, “Badji Mokhtar” University of Annaba,
Annaba, 23000, Algeria
soltane@lri-annaba.net & xor99@hotmail.com

Noureddine Doghmane
Electronics Department, Faculty of Engineering Science, “Badji Mokhtar” University of Annaba
Annaba, 23000, Algeria
ndoghmane@univ-annaba.org

Nourddine Guersi
Electronics Department, Faculty of Engineering Science, “Badji Mokhtar” University of Annaba,
Annaba, 23000, Algeria
guersi54@yahoo.fr

Abstract

This paper presents a comparative analysis of the performance of three estimation algorithms: Expectation Maximization (EM), Greedy EM Algorithm (GEM) and Figueiredo-Jain Algorithm (FJ) - based on the Gaussian mixture models (GMMs) for signature biometrics verification. The simulation results have shown significant performance achievements. The test performance of EER=5.49 % for "EM", EER=5.04 % for "GEM", and EER=5.00 % for "FJ", shows that the behavioral information scheme of signature biometrics is robust and has a discriminating power, which can be explored for identity authentication.

Keywords: Biometric authentication, behavioral, signature, soft decision and Gaussian Mixture Modal, EM, GEM and FJ.

1. Introduction

BIOMETRIC is a Greek composite word stemming from the synthesis of bio and metric, meaning life measurement. In this context, the science of biometrics is concerned with the accurate measurement of unique biological characteristics of an individual in order to securely identify them to a computer or other electronic system. Biological characteristics measured usually include fingerprints, voice patterns, retinal and iris scans, face patterns, and even the chemical composition of an individual's DNA [1]. Biometrics authentication (BA) (Am I whom I claim I am?) involves confirming or denying a person's claimed identity based on his/her physiological or behavioral characteristics [2]. BA is becoming an important alternative to traditional authentication methods such as keys (“something one has", i.e., by possession) or PIN numbers (“something one knows", i.e., by knowledge) because it is essentially “who one is", i.e., by biometric information. Therefore, it is not susceptible to misplacement or forgetfulness [3]. These biometric systems for personal authentication and identification are based upon physiological or behavioral features which are typically distinctive, although time varying, such as fingerprints, hand geometry, face, voice, lip movement, gait, and iris patterns. An identity verification system has to deal with two kinds of events: either the person claiming a given identity is the one who he claims to be (in which case, he is called a client), or he is not (in which case, he is called an impostor). Moreover, the system may generally take two decisions: either accept the client or reject him and decide he is an impostor.

Some works based on biometric signature identity verification systems has been reported in literature. **A. Perez-Hernandez et al. [13]** Propose a simple adaptive off-line signature recognition method based on the feature analysis of extracted significant strokes for a given signature. Their system correctly decides on the majority of tested patterns, which include both simple and skilled forgeries. Experimental results have showed a good trade-off between response time and reasonable recognition accuracy. **Hugo Gamboa et al. [14]** describe a new behavioral biometric technique based on human computer interaction. They developed a system that captures the user interaction via a pointing device, and uses this behavioral information to verify the identity of an individual. Using statistical pattern recognition techniques, they developed a sequential classifier that processes user interaction, according to which the user identity is considered genuine if a predefined accuracy level is achieved, and the user is classified as an impostor otherwise. Two statistical models for the features were tested, namely Parzen density estimation and a uni-modal distribution. The system was tested with different numbers of users in order to evaluate the scalability of the proposal. Experimental results showed that the normal user interaction with the computer via a pointing device entails behavioral information with discriminating power that can be explored for identity authentication. **Ibrahim S. I. Abuhaiba [4]** presents a simple and effective signature verification method that depends only on the raw binary pixel intensities and avoids using complex sets of features. The method looks at the signature verification problem as a graph matching problem. The method is tested using genuine and forgery signatures produced by five subjects. An equal error rate of 26.7% and 5.6% was achieved for skilled and random forgeries, respectively. A positive property of the algorithm is that the false acceptance rate of random forgeries vanishes at the point of equal false rejection and skilled forgery false acceptance rates.

2. Biometric Signature Verification

Handwritten signature is one of the first accepted civilian and forensic biometric identification technique in our society [4]. Human verification is normally very accurate in identifying genuine signatures. A signature verification system must be able to detect forgeries and at the same time reduce rejection of genuine signatures. The signature verification problem can be classified into categories: offline and online. Offline signature verification does not use dynamic information that is used extensively in online signature verification systems. This paper investigates the problem of offline signature verification. The problem of offline signature verification has been faced by taking into account three different types of forgeries: random forgeries, produced without knowing either the name of the signer or the shape of his signature; simple forgeries, produced knowing the name of the signer but without having an example of his signature; and skilled forgeries, produced by people who, looking at an original instance of the signature, attempt to imitate it as closely as possible.

Figure 1. Wacom Graphire3 digitizing TabletPC

A. Feature Extraction

The coordinate trajectories (x_n, y_n) and pressure signal $\mathbf{p_n}$ are the components of the unprocessed feature vectors $u_n = [x_n, y_n, p_n]^T$ extracted from the signature signal [5], where $n = 1,...,N_s$ and N_s is the duration of the signature in time samples. Signature trajectories are then pre-processed by subtracting the centre of mass followed by rotation alignment based on the average path tangent angle. An extended set of discrete-time functions are derived from the pre-processed trajectories consisting of sample estimations of various dynamic properties. As s result, the parameterised signature O consists in the sequence of feature vectors $o_n = [x_n, y_n, p_n \theta_n, v_n, \dot{x}_n, \dot{y}_n]$, $n = 1,...,N_s$, where the upper dot notation represents an approximation to the first order time derivative and θ and v stand respectively for path tangent angle, path velocity magnitude.

$$v_i = \sqrt{\dot{x}_i^{\,2} + \dot{y}_i^{\,2}} \text{ and } \theta_i = \arctan(\dot{y}_i, \dot{x}_i) \text{ and } \dot{x}_i = x_i - x_{i-1} \text{ and } \dot{y}_i = y_i - y_{i-1}$$

A whitening linear transformation is finally applied to each discrete-time function so as to obtain zero mean and unit standard deviation function values. Seven dimensional feature vectors are used for GMM processing described in the following section. Figure 3 shows x-, y-, p- and velocity signals of an example signature.

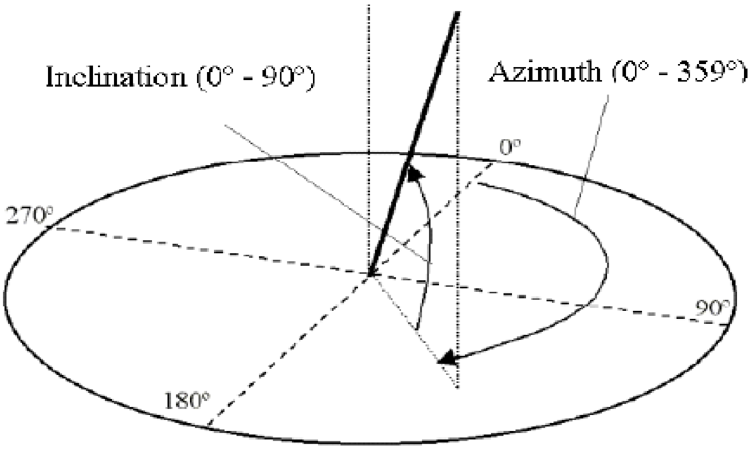

Figure 2. Azimuth and inclination angles of the pen respect to the plane of the graphic card GD-0405U from Wacom Graphire3 digitizing TabletPC

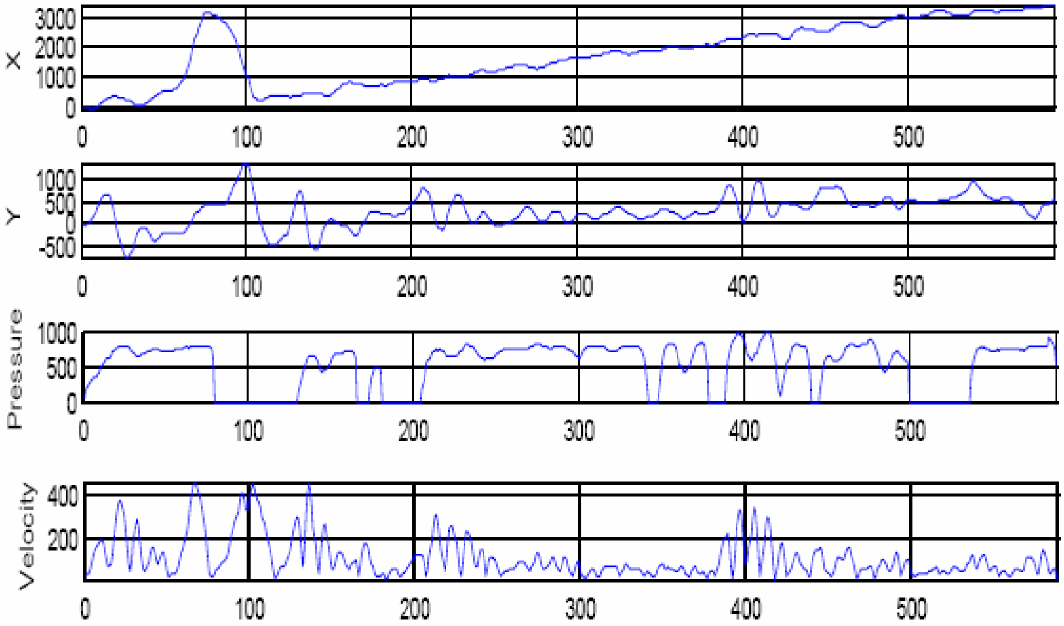

Figure 3. Signals (x-, y- position, pen pressure and velocity) of one signature fragment.

B. Maximum Likelihood Parameter Estimation

Given a set of observation data in a matrix X and a set of observation parameters θ the *ML* parameter estimation aims at maximizing the likelihood $L(\theta)$ or log likelihood of the observation data $X = \{X_1, ..., X_n\}$

$$\hat{\theta} = \arg\max_{\theta} L(\theta). \quad (1)$$

Assuming that it has independent, identically distributed data, it can write the above equations as:

$$L(\theta) = p(X \mid \theta) = p(X_1, ..., X_n \mid \theta) = \prod_{i=1}^{n} p(X_i \mid \theta). \quad (2)$$

The maximum for this function can be found by taking the derivative and set it equal to zero, assuming an analytical function.

$$\frac{\partial}{\partial \theta} L(\theta) = 0. \quad (3)$$

The incomplete-data log-likelihood of the data for the mixture model is given by:

$$L(\theta) = \log(X \mid \theta) = \sum_{i=1}^{N} \log(x_i \mid \theta) \ (4)$$

which is difficult to optimize because it contains the log of the sum. If it considers X as incomplete, however, and posits the existence of unobserved data items $Y = \{y_i\}_{i=1}^{N}$ whose values inform us which component density generated each data item, the likelihood expression is significantly simplified. That is, it assumes that $y_i \in \{1, ..., K\}$ for each i, and $y_i = k$ if the i-th sample was generated by the k-$^{\text{th}}$ mixture component. If it knows the values of Y, it obtains the complete-data log-likelihood, given by:

$$L(\theta \mid Y) = \log p(X, Y \mid \theta) \ (5)$$

$$= \sum_{i=1}^{N} \log p(x_i, y_i \mid \theta) \quad (6)$$

$$= \sum_{i=1}^{N} \log(p(x_i \mid \theta) p(x_i \mid y_i, \theta)) \ (7)$$

$$= \sum_{i=1}^{N} (\log p_{y_i} + \log g(x_i \mid \mu_{y_i}, \sum y_i)) \ (8)$$

which, given a particular form of the component densities, can be optimized using a variety of techniques [6].

C. EM algorithm

The expectation-maximization (EM) algorithm [7][8] [9][10] is a procedure for maximum-likelihood (ML) estimation in the cases where a closed form expression for the optimal parameters is hard to obtain. This iterative algorithm guarantees the monotonic increase in the likelihood L when the algorithm is run on the same training database.

The probability density of the Gaussian mixture of k components in $\mathcal{R}^{\text{d}}$ can be described as follows:

$$\phi(x) = \sum_{i=1}^{N} \pi_i \varnothing(\mathrm{x} \mid \theta_\mathrm{i}) \quad \forall x \in \mathcal{R}^{\mathrm{d}}, \qquad (9)$$

where $\varnothing(x|\theta_i)$ is a Gaussian probability density with the parameters $\theta_i = (m_i, \Sigma_i)$, m_i is the mean vector and Σ_i is the covariance matrix which is assumed positive definite given by:

$$\varnothing(\mathrm{x} \mid \theta_\mathrm{i}) = \varnothing(\mathrm{x} \mid \mathrm{m_i}, \Sigma_i) = \frac{1}{(2\pi)^{\frac{n}{2}} \, |\Sigma_i|^{\frac{1}{2}}} e^{-\frac{1}{2}(x-m_i)^T \Sigma_1^{-1}(x-m_i)} \quad (10)$$

and $\pi_i \in [0,1]\,(i = 1,2,...,k)$ are the mixing proportions under the constraint $\sum_{i=1}^{k} \pi_i = 1$. If it encapsulates all the parameters into one vector: $\theta_k = (\pi_1, \pi_2, ..., \pi_k, \theta_1, \theta_2, ..., \theta_k)$, then , according to (8), the density of Gaussian mixture can be rewritten as:

$$\phi(x \mid \Theta_k) = \sum_{i=1}^{k} \pi_i \varnothing(\mathrm{x} \mid \theta_\mathrm{i}) = \sum_{i=1}^{k} \pi_i \varnothing(\mathrm{x} \mid \mathrm{m_i}, \textstyle\sum_i). \ (11)$$

For the Gaussian mixture modeling, there are many learning algorithms. But the EM algorithm may be the most well-known one. By alternatively implementing the E-step to estimate the probability distribution of the unobservable random variable and the M-step to increase the log-likelihood function, the EM algorithm can finally lead to a local maximum of the log-likelihood function of the model. For the Gaussian mixture model, given a sample data set $S=\{x_1, x_2, \ldots, x_N\}$ as a special incomplete data set, the log-likelihood function can be expressed as follows:

$$\log p(S \mid \Theta_k) = \log \prod_{t=1}^{N} \varnothing(\mathrm{x_t} \mid \Theta_k) = \sum_{t=1}^{N} \log \sum_{i=1}^{k} \pi_i \varnothing(\mathrm{x_t} \mid \theta_i) \ (12)$$

which can be optimized iteratively via the EM algorithm as follows:

$$P(j \mid x_t) = \frac{\pi_j \varnothing(\mathrm{x_t} \mid \theta_j)}{\sum_{i=1}^{k} \pi_j \varnothing(\mathrm{x_t} \mid \theta_j)}, \ (13)$$

$$\pi_j^+ = \frac{1}{N} \sum_{t=1}^{N} P(j \mid x_t), \ (14)$$

$$\mu_j^+ = \frac{1}{\sum_{t=1}^{N} P(j \mid x_t)} \sum_{t=1}^{N} P(j \mid x_t) x_t \ , (15)$$

$$\Sigma_j^+ = \frac{1}{\sum_{t=1}^{N} P(j \mid x_t)} \sum_{t=1}^{N} P(j \mid x_t)(x_t - \mu_j^+)(x_t - \mu_j^+)^T \quad (16)$$

Although the EM algorithm can have some good convergence properties in certain situations, it certainly has no ability to determine the proper number of the components for a sample data set because it is based on the maximization of the likelihood.

D. Greedy EM Algorithm

The greedy algorithm (GEM) [7][8][10][11] starts with a single component and then adds components into the mixture one by one. The optimal starting component for a Gaussian mixture is trivially computed, optimal meaning the highest training data likelihood. The algorithm repeats two steps: insert a component into the mixture, and run EM until convergence. Inserting a component that increases the likelihood the most is thought to be an easier problem than initializing a whole near-optimal distribution. Component insertion involves searching for the parameters for only one component at a time. Recall that EM finds a local optimum for the distribution parameters, not necessarily the global optimum which makes it initialization dependent method. Given p_c a C-component Gaussian mixture with parameters θ_c. The general greedy algorithm for Gaussian mixture is as follows:

1. *Compute the optimal (in the ML sense) one-component mixture* p_1 *and set* $C \leftarrow 1$.
2. *Find a new component* $\mathcal{N}(x;\mu',\Sigma')$ *and corresponding mixing weight* α' *that increase the likelihood the most:*

   $$\{\mu',\Sigma',\alpha'\} = \arg\max_{\{\mu,\Sigma,\alpha\}} \sum_{i=1}^{N} \ln[(1-\alpha)p_C(x_i)+\alpha\mathcal{N}(x_i;\mu,\Sigma)] \quad (17)$$

   while keeping p_C *fixed.*
3. Set $p_{C+1}(x) \leftarrow (1-\alpha')p_c(x)+\alpha'\mathcal{N}(x;\mu',\Sigma')$ and then $C \leftarrow C+1$.
4. Update p_C using **EM** (or more other method) until convergence.
5. *Evaluate some stopping criterion; go to step 2 or quit.*

The stopping criterion in Step 5 can be for example any kind of model selection criterion, wanted number of components, or the minimum message length criterion. The crucial point is of course Step 2. Finding the optimal new component requires a global search, which is performed by creating CN_{cand} candidate components. The number of candidates will increase linearly with the number of components *C*, having N_{cand} candidates per each existing component. The candidate resulting in the highest likelihood when inserted into the (previous) mixture is selected. The parameters and weight of the best candidate are then used in Step 3 instead of the truly optimal values.

The candidates for executing Step 2 are initialized as follows: the training data set X is partitioned into *C* disjoints data sets $\{A_c\}$, c=1…*C* according to the posterior probabilities of individual components; the data set is Bayesian classified by the mixture components. From each A_c number of N_{cand} candidates are initialized by picking uniformly randomly two data points x_l and x_r in A_c. The set A_c is then partitioned into two using the smallest distance selection with respect to x_l and x_r. The mean and covariance of these two new subsets are the parameters for two new candidates. The candidate weights are set to half of the weight of the component that produced the set A_c. Then new x_l and x_r are drawn until N_{cand} candidates are initialized with A_c. The partial **EM** algorithm is then used on each of the candidates. The partial **EM** differs from the **EM** and **CEM** algorithms by optimizing (updating) only one component of a mixture; it does not change any other components. In order to reduce the time complexity of the algorithm a lower bound on the log-likelihood is used instead of the true log-likelihood. The lower-bound log-likelihood is calculated with only the points in the respective set A_c. The partial **EM** update equations are as follows:

$$w_{i,C+1} = \frac{\alpha\mathcal{N}(x_i,\mu,\Sigma)}{(1-\alpha)p_C(x)+\alpha\mathcal{N}(x_i,\mu,\Sigma)}, \quad (18)$$

$$\alpha = \frac{1}{\mathcal{N}(A_C)} \sum\nolimits_{i \in A_C} w_{i,C+1} \quad (19)$$

$$\mu = \frac{\sum\nolimits_{i \in A_C} w_{i,C+1} x_i}{\sum\nolimits_{i \in A_C} w_{i,C+1}} \quad (20)$$

$$\Sigma = \frac{\sum\nolimits_{i \in A_C} w_{i,C+1} (x_i - \mu)(x_i - \mu)^T}{\sum\nolimits_{i \in A_C} w_{i,C+1}} \quad (21)$$

where $N(A_c)$ is the number of training samples in the set A_c. These equations are much like the basic **EM** update equations in Eqs. (6) - (8). The partial **EM** iterations are stopped when the relative change in log-likelihood of the resulting $C + 1$ –component mixture drops below threshold or maximum number of iterations is reached. When the partial **EM** has converged the candidate is ready to be evaluated.

E. Figueiredo-Jain Algorithm

The Figueiredo-Jain (FJ) [7][8][10][11] algorithm tries to overcome three major weaknesses of the basic EM algorithm. The EM algorithm presented previous section requires the user to set the number of components and the number will be fixed during the estimation process. The FJ algorithm adjusts the number of components during estimation by annihilating components that are not supported by the data. This leads to the other EM failure point, the boundary of the parameter space. FJ avoids the boundary when it annihilates components that are becoming singular. FJ also allows starting with an arbitrarily large number of components, which tackles the initialization issue with the EM algorithm. The initial guesses for component means can be distributed into the whole space occupied by training samples, even setting one component for every single training sample.

The classical way to select the number of mixture components is to adopt the "model-class/model" hierarchy, where some candidate models (mixture pdf's) are computed for each model-class (number of components), and then select the "best" model. The idea behind the FJ algorithm is to abandon such hierarchy and to find the "best" overall model directly. Using the minimum message length criterion and applying it to mixture models leads to the objective function:

$$\Lambda(\theta, X) = \frac{V}{2} \sum\nolimits_{c:\alpha_c > 0} \ln\left(\frac{N\alpha_c}{12}\right) + \frac{C_{nz}}{2} \ln\frac{N}{12} + \frac{C_{nz}(V+1)}{2} - \ln \mathcal{L}(X, \theta) \quad (22)$$

Where N is the number of training points, V is the number of free parameters specifying a component, and C_{nz} is the number of components with nonzero weight in the mixture ($\alpha_c > 0$). The last term $\ln \mathcal{L}(X, \theta)$ is the log-likelihood of the training data given the distribution parameters (Eq. 8). The EM algorithm can be used to minimize Eq. 22 with a fixed C_{nz} it leads to the M-step with component weight updating formula:

$$\alpha_c^{i+1} = \frac{\max\left\{0, \left(\sum\nolimits_{n=1}^{N} w_{n,c}\right) - \frac{V}{2}\right\}}{\sum\nolimits_{j=1}^{C} \max\left\{0, \left(\sum\nolimits_{n=1}^{N} w_{n,c}\right) - \frac{V}{2}\right\}}. \quad (23)$$

This formula contains an explicit rule of annihilating components by setting their weights to zero. The above M-steps are not suitable for the basic EM algorithm though. When initial C is high, it can happen that all weights become zero because none of the components have enough support from the data. Therefore a component-wise EM algorithm (CEM) is adopted. CEM updates the components one by one, computing the E-step (updating W) after each component update, where the basic EM updates all components "simultaneously". When a component is annihilated its probability mass is immediately redistributed strengthening the remaining components. When CEM converges, it is not guaranteed that the minimum of $\Lambda(\theta, X) > 0$ is found, because the annihilation rule (Eq. 23) does not take into account the decrease caused by decreasing C_{nz}. After convergence the component with the smallest weight is removed and the CEM is run again, repeating until C_{nz}=1. Then the estimate with the smallest $\Lambda(\theta, X)$ is chosen. The implementation of the FJ algorithm uses a modified cost function instead of $\Lambda(\theta, X)$.

$$\Lambda'(\theta, X) = \frac{V}{2}\sum\nolimits_{c\,:\,\alpha_c>0} \ln \alpha_c + \frac{C_{nz}(V+1)}{2}\ln N - \ln \mathcal{L}(X,\theta)\,. \quad (24)$$

3. Experiments and Results

The experiments were performed using signatures database obtained from eNTERFACE 2005 [12]. Thirty subjects were used for the experiments in which twenty-six are males and four are females. For each subject, 30 signatures (with dat header) are used. Each line of a (.dat files) consists of four comma separated integer values for the sampled *x*- and *y*- position of the pen tip, the pen pressure and the timestamp (in ms); the lines with values of -1 for *x, y* and pressure represent a pen-up/pen-down event; The device used for recording the handwriting data was a Wacom Graphire3 digitizing tablet. Size of sensing surface is 127.6mm x 92.8mm. With spatial resolution of 2032 lpi (lines per inch), able to measure 512 degrees of pressure. The signature data is acquired with a non-fixed sampling rate of about 100Hz. For the experts, twenty-four signatures from a subject were randomly selected for training, and the other six samples were used for the subsequent validation and testing. Three sessions of the signature database were used separately. Session one was used for training the signature experts. Each expert used ten mixture client models. To find the performance, Sessions two and three were used for obtaining expert opinions of known impostor and true claims.

Performance Criteria:

The basic error measure of a verification system is false rejection rate (*FRR*) and false acceptance rate (*FAR*) as defined in the following equations:

False Rejection Rate (*FRR_i*): is an average of number of falsely rejected transactions. If *n* is a transaction and x(n) is the verification result where 1 is falsely rejected and 0 is accepted and *N* is the total number of transactions then the personal False Rejection Rate for user *i* is

$$FRR_i = \frac{1}{N}\sum_{n=1}^{N} x(n) \quad (25)$$

False Acceptance rate (*FAR_i*) is an average of number of falsely accepted transactions. If *n* is a transaction and x(n) is the verification result where 1 is a falsely accepted transaction and 0 is genuinely accepted transaction and N is the total number of transactions then the personal False Acceptance Rate for user *i* is

$$FAR_i = \frac{1}{N}\sum_{n=1}^{N} x(n) \quad (26)$$

Both FRR_i and FAR_i are usually calculated as averages over an entire population in a test. If P is the size of populations then these averages are

$$FRR = \frac{1}{P}\sum_{i}^{P} FRR_i \quad (27)$$

$$FAR = \frac{1}{P}\sum_{i}^{P} FAR_i \quad (28)$$

Equal Error Rate (*EER*), is an intersection where FAR and FRR are equal at an optimal threshold value. This threshold value shows where the system performs at its best (see Figure 4).

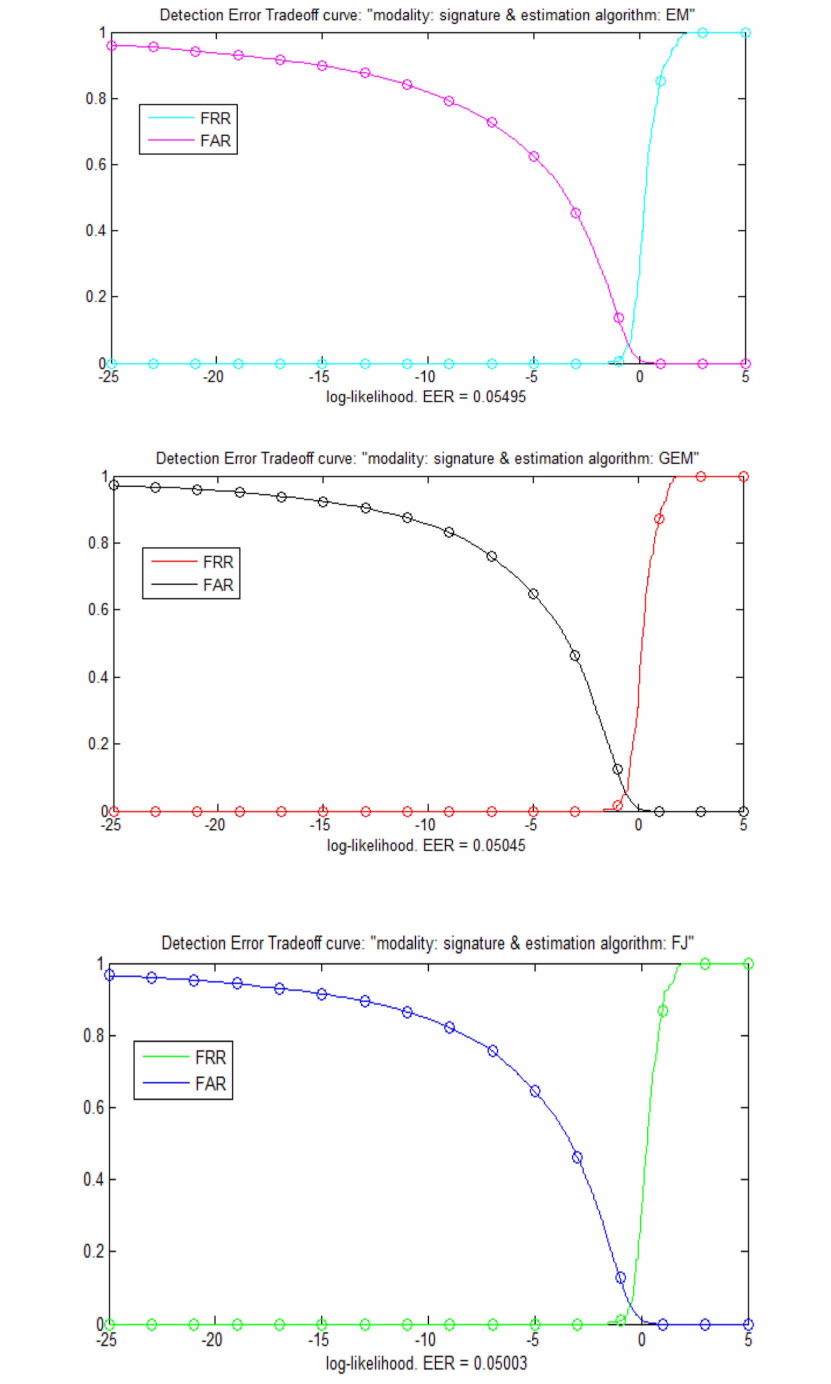

Figure 4. Detection error tradeoff curves

As a common starting point, classifier parameters were selected to obtain performance as close as possible to *EER* on clean test data (following the standard practice in the biometric verification area of using *EER* as a measure of expected performance). A good decision is to choose the decision threshold such as the false accept equal to the false reject rate. In this paper it uses the Detection Error Tradeoff (DET) curve to visualize and compare the performance of the system.

4. Conclusion

The paper has presented a human authentication method of behavioral biometrics signature information. Simulation results show that state-of-the art finite mixture modal (GMM) is quite effective in modeling the genuine and impostor score densities. The (EM), (GEM) and (FJ) estimation algorithms achieve a significant performance rates, EER=5.49 % for "EM", EER=5.04 % for "GEM" and EER=5.00 % for "FJ". Hence, the behavioral information scheme based on signature biometrics is robust and has a discriminating power, which can be explored for identity authentication.

5. References

[1] Gleni, S. & Petratos, P. (2004). DNA Smart Card for Financial Transactions. *The ACM Student Magazine 2004*. Retrieved from http://www.acm.org.

[2] Chetty, G. & Wagner, M. (2006). Audio-Visual Multimodal Fusion for Biometric Person Authentication and Liveness Verification. Australian Computer Society, Inc. *In NICTA-HCSNet Multimodal UserInteraction Workshop (MMUI2005)*, Sydney, Australia.

[3] Poh, N. & Bengio, S. (2004). Database, Protocol and Tools for Evaluating Score-Level Fusion Algorithms in Biometric Authentication, *IDIAP RR 04-44*. Martigny, Switzerland.

[4] Abuhaiba, I. S. I. (2007). Offline Signature Verification Using Graph Matching, *Turk J Elec Engin*, 15(1)

[5] Richiardi, J., Fierrez-Aguilar, J., Ortiga-Garcia, J. & Drygajlo, A. (2004). On-line Signature Verification Resilience to Packet Loss in IP Networks. In: *Second COST 275 Workshop Biometrics on the Internet: Fundamentals, Advances and Applications*. University of Vigo, Vigo-Spain 25-26 March 2004.

[6] Kittler, J., Hatef, M., Duin, R. P. W. and Matas, J. (1998). On combining classifiers. *IEEETransactions on Pattern Analysis and Machine Intelligence*, 20(3), 226–239.

[7] Paalanen, P. (2004). Bayesian classification using gaussian mixcute model and EM estimation: implementation and comparisons. In: Information Technology Project, 2004, Lappeenranta, June 23. Retrieved from http://www.it.lut.fi/project/gmmbayes/

[8] Paalanen, P., Kamarainen, J.-K., Ilonen, J., Kälviäinen, H. (2005). Feature Representation and Discrimination Based on Gaussian Mixture Model Probability Densities: Practices and Algorithms. Department of Information Technology, Lappeenranta University of Technology, Lappeenranta, Finland.

[9] Li, L. & Ma, J. (2008) A BYY Split-and-Merge EM Algorithm for Gaussian Mixture Learning. In F. Sun et al. (Eds.): ISNN 2008, Part I. LNCS 5263 (pp. 600–609), Berlin, Heidelberg: Springer-Verlag.

[10] Verbeek, J.J., Vlassis, N., Krose, B. (2003). Efficient Greedy Learning of Gaussian Mixture Models. *Neural Computation*, 15(2), 469-485.

[11] Nunnink, J. R. J., Verbeek, J. J., Vlassis, N., Accelerated Greedy Mixture Learning, (Research report) Informatics Institute, Faculty of Science, University of Amsterdam Kruislaan, Amsterdam, The Netherlands.

[12] Stylianou, Y., Pantazis, Y., Calderero, F., Larroy, P., Severin, F., Schimke, S., Bonal, R., Matta, F., & Valsamakis, A. (2005). GMM-Based Multimodal Biometric Verification. In *eNTERFACE 2005. The summer Workshop on Multimodal Interfaces*, July 18th – August 12th, Facultè Polytechnique de Mons, Belgium.

[13] Perez-Hernandez, A., Sanchez, A. & Velez, J. F. (2004). Simplified Stroke-based Approach for Off-line Signature Recognition. 2nd COST 275 Workshop – Biometrics on the Internet, Vigo, 25-26 March 2004, Spain.

[14] Gamboaa, H., Fredb, A. A Behavioural Biometric System Based on Human Computer Interaction, (Research Report at Escola Superior de Tecnologia de Setubal, Campo do IPS, Estefanilha, 2914-508 Setubal, Portugal).