Biometric Hash based on Statistical Features of Online Signatures

Claus Vielhauer[1,2], Ralf Steinmetz[1], Astrid Mayerhöfer[3]

[1] – Technical University Darmstadt – Institute for Industrial Process- and System Communications
Merckstrasse 25, D-64283 Darmstadt, Germany
[2] – Platanista GmbH
Pankratiusstrasse 7, D-64289 Darmstadt, Germany
[3] – Fraunhofer IPSI
Dolivostrasse 14, D-64287 Darmstadt, Germany

{Claus.Vielhauer, Ralf Steinmetz}@KOM.tu-darmstadt.de, Astrid.Mayerhoefer@ipsi.fraunhofer.de

Abstract

This paper presents a new approach to generate biometric hash values based on statistical features in online signature signals. Whilst the output of typical online signature verification systems are threshold-based true-false decisions, based on a comparison between test sample signals and sets of reference signals, our system responds to a signature input with a biometric hash vector, which is calculated based on an individual interval matrix.

Especially for applications, which require key management strategies (e.g. e-Commerce, smart cards), hash values are of great interest, as keys can be derived directly from the hash value, whereas a verification decision can only grant or refuse access to a stored key. Further, our new approach does not require storage of templates for reference signatures, thus increases the security of the system.

In our prototype implementation, the generated biometric hash values are calculated on a pen-based PDA and used for key generation for a future secure data communication between a PDA and a server by encryption. First tests show that the system is actually able to generate stable biometric hash values of the users and although the system was exposed to skilled forgeries, no test person was able to reproduce another subject's hash vector. During tests, we were able to tune the system to a FAR of 0% at a FRR level of 7.05%.

1. Introduction

Analysis of online handwriting analysis is a research subject since many years and besides the handwriting recognition, writer verification has been identified as a major application area. Verification systems require two parameters (actual test sample and a reference data set) and respond with a binary result (verified or not). On the problem of how to decide upon the similarity between the test sample and the reference, many approaches have been published. Typically, they are either based on statistical function or parameter comparison [1] or neural networks [2]. In order to identify a writer with these technologies, the system will need to compare an actual test signal against a database of all registered users, thus leading to a large number of verification processes in large environments. Our system presents an approach to generate a biometric hash function, which returns a vector of 24 elements, representing a hash value of the online signature, upon entering the actual test signature signals and an interval matrix (IM), which is individually obtained during enrollment.

In the following chapters, we first give an overview of the system. Then the 24 feature parameters, that are extracted from the signal input, are presented and the Interval Matrix is defined in chapter 4. The following chapter deals with the problem of how to determine tolerance factors, while in chapter 6, we describe the calculation of the biometric hash value. Chapter 7 presents our test results while chapter 8 concludes this paper and shows areas of future work.

2. System Overview

Data acquisition of our system is performed on a digitizer tablet or a pen-sensitive computer display, like the ones included in pen-based Personal Digital Assistants (PDA). Writing position signals x(t) and y(t) are sampled as well as the binary pen-up/pen-down signal $p_{0|1}(t)$. Although the discrete writing pressure signal p(t) has been identified as a relevant feature for writer verification [3], our system limits to the binary representation, as most PDAs do not provide discrete pressure information. During Feature Extraction, 24 parameters are being calculated from the input sample, which are then computed in the 24 component hash vector by interval mapping, see figure 1 System overview.

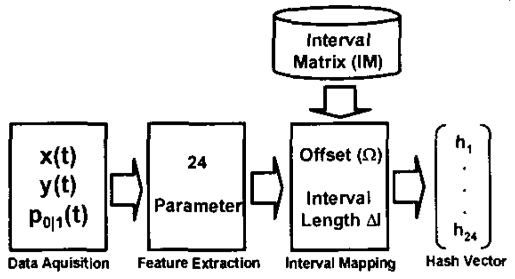

Figure 1. System overview

The system is implemented on a Palm Vx PDA based on a Motorola 68328 CPU clocked at 20 MHz and 8 MBytes of RAM.

3. Feature Parameters

From the physical input functions x(t), y(t) and $p_{0|1}(t)$, the following 24 statistical parameters n_i (i=1..24) are calculated during feature extraction:

Table 1. Feature parameters extracted from the online signature

No.	Parameter Name	Description
1	$N_{Strokes}$	Number of continuous pen-down sequences (strokes)
2	T_{Total}	Duration of the complete writing process in ms
3	$N_{Samples}$	Number of samples for the complete writing process
4	$N_{Extrema}$	Sum of all local Minima and Maxima in writing signals x(t) and y(t)
5	$L_{AspectRatio}$	Aspect ratio of x/y bitmap * 100
6	$L_{TPenDown/TPenUp}$	Ratio of pen-down and pen-up duration for complete writing process * 100
7	A_x	Integrated Area covered by the x-writing signal for the complete writing process
8	A_y	Integrated Area covered by the y-writing signal for the complete writing process
9	v_x	Average of the absolute writing velocity in x-direction
10	v_y	Average of the absolute writing velocity in y-direction
11	a_x	Average of the absolute writing acceleration in x-direction
12	a_y	Average of the absolute writing acceleration in y-direction
13	$(x_{max}-x_{min})$ / T_{Total}	Effective average writing velocity in x-direction
14	$(y_{max}-y_{min})$ / T_{Total}	Effective average writing velocity in y-direction
15-19	$A_{x,Segment\ 1}$ - $A_{x,Segment\ 5}$	$A_{x,\ Segment\ i}$ for the I-th segment of the equidistantly partitioned x-writing signal, $i \in [1..5]$
20-24	$A_{y,Segment\ 1}$ – $A_{y,Segment\ 5}$	$A_{y,\ Segment\ i}$ for the i-th segment of the equidistantly partitioned y-writing signal, $i \in [1..5]$

As the process of interval mapping is implemented in integer arithmetic, all ratios are computed in 2-digit precision and then transposed by 2 magnitudes.

All feature parameters except $A_{x,\ Segment\ i}$ and $A_{x,\ Segment\ i}$ are calculated on a global basis, taking the sampled features over the entire writing process into account. Segmentation of online signatures into local features, e.g. on a stroke level, can lead to significant improvement in verification results [4], therefore we decided to include local features in the parameter set. For the segmentation of those two local feature classes, we divide the complete input signals x(t) and y(t) in 5 equal-length part signals and apply integration over the pen position signals x(t) and y(t), thus leading to feature parameters number 15-19 and 20-24 respectively.

Although a wider number of statistical parameters can be found [5], we limit this approach at this stage to the values listed, as our goal the application on PDA computers with less sophisticated sampling devices and limited computation performance.

4. Determination of the Interval Matrix

For a given number $N_{Parameters}$ of different parameters which are used to built the biometric hash, the Interval Matrix (IM) is a matrix of dimension ($N_{Parameters}$ x 2), which is individually calculated for each user during the enrollment process. The enrollment is based on a transitivity based enrollment strategy presented in [6], with a resulting reference set size of 4 samples. Considering the 24 feature parameters that are calculated in our system, IM can be written as follows:

$$IM = \begin{pmatrix} \Delta I_1\ ,\ \Omega_1 \\ \Delta I_2\ ,\ \Omega_2 \\ \cdots\ \cdots\ \cdots \\ \Delta I_{24}\ ,\ \Omega_{24} \end{pmatrix} \qquad (4.1)$$

where ΔI_i is the iterval length of an interval $[I_{Low}..I_{High}]$ and Ω_i, the interval offset, which are calculated for each parameter value $n_{i,j}$ (i being the reference number 1..24 and j being the sample number 1..4) of the reference sample j as follows. The extended interval length ΔI_i is computed based on the following intervals:

Initial interval:

$$[I_{InitLow} .. I_{InitHigh}] = [MIN(n_{i,j}) .. MAX(n_{i,j})] \qquad (4.2)$$

Extended interval:

$$[I_{Low} .. I_{High}] = [I_{InitLow}*(1-t_i) .. I_{InitHigh}*(1+t_i)] \qquad (4.3)$$

with t_i being the tolerance factor to extend the interval length. This factor has been determined by statistical testing of authentic writing samples against the above intervals and averaging standard deviations, which will be explained in the following chapter. Considering the fact that all parameters are of integer type and test values will be rounded accordingly, the effective interval length ΔI_i can be written to

$$\Delta I_i = I_{High} + 0.5 - (I_{Low} - 0.5) = I_{High} - I_{Low} + 1 \qquad (4.4)$$

Whereas we define the interval offset value Ω_i as

$$\Omega_i = I_{Low} \; MOD \; \Delta I_i \qquad (4.5)$$

5. Determination of the tolerance value table

The tolerance value for each feature was determined during a test, where for each of the 11 test subjects, 5 different writing semantics (signature, numeric code, passphrase, arbitrary word and symbol) were tested against authentic enrollments of the same kind. For all determined feature parameters not being within the interval boundaries, the absolute distance to the interval borders was averaged to μ_i. The maximum deviation for each individual parameter was limited to 20%, values exceeding this threshold were considered as statistically exceptional deviations and not included in μ_i. For the determination of IM as described in the previous chapter, we then set μ_i=t_i for each of the i features, thus leading to the following tolerance value table. Each column lists the tolerance values for each of the 11 test subjects in percent and each row represents one of the 24 feature parameter. The rightmost column shows the tolerance average over all subjects, where values "A" identify tolerances greater than 20%, which are excluded from averaging.

6. Calculation of the biometric hash vector

The biometric hash values are calculated by interval mapping of each single feature parameter of a given writing sample against an Interval Matrix, resulting in a 24 component hash vector.

Table 2. Tolerance value table

Feature	AL	AM	EF	EH	FR	JD	LF	MH	MM	ST	YS	μ=t
1	0	21	0	0	0	0	A	0	0	0	0	2.1
2	7	0	0	5	9	8	0	0	A	3	0	3.2
3	10	0	0	0	8	6	A	0	0	0	0	2.8
4	0	0	0	0	7	0	9	3	0	5	13	3.4
5	0	8	0	A	A	7	0	0	A	12	9	4.5
6	12	26	0	0	10	A	17	0	0	26	0	9.1
7	12	0	A	9	0	5	2	0	3	8	3	4.2
8	0	6	8	1	13	10	0	0	8	0	A	4.6
9	9	0	A	4	51	7	0	0	0	0	0	7.1
10	0	0	A	0	0	0	0	0	0	0	A	0.0
11	0	0	17	0	0	A	12	0	A	0	6	3.9
12	21	0	0	0	0	0	0	0	0	0	A	2.1
13	0	0	0	A	A	A	0	0	A	A	0	0.0
14	0	5	0	2	0	17	0	0	A	8	2	3.4
15	12	0	A	A	0	3	A	8	17	0	0	5.0
16	0	3	7	A	0	5	8	13	15	0	0	5.1
17	5	0	12	A	0	0	11	21	0	0	4	5.3
18	10	11	9	6	0	0	2	8	0	0	0	4.2
19	A	0	A	0	7	A	0	0	0	A	A	2.4
20	0	12	5	A	8	12	0	0	5	1	34	7.5
21	A	0	15	2	11	A	0	0	0	1	0	3.2
22	0	0	0	5	13	0	0	A	A	2	A	2.5
23	0	A	15	0	13	0	21	17	9	3	26	10.4
24	10	1	3	2	A	A	0	0	A	A	A	2.7

Due to the nature of the determination of the interval matrix, all possible values v_1 and v_2 within the extended interval $[I_{Low}..I_{High}]$ as defined in the previous chapter fulfill the following condition:

$$For\ all\ v_1,\ v_2 \in [I_{Low} .. I_{High}]: \left\lfloor \frac{(v_1 - \Omega_i)}{\Delta I_i} \right\rfloor = \left\lfloor \frac{(v_2 - \Omega_i)}{\Delta I_i} \right\rfloor \qquad (6.1)$$

and

$$For\ all\ v_1, v_2 \notin [I_{Low} .. I_{High}]: \left\lfloor \frac{(v_1 - \Omega_i)}{\Delta I_i} \right\rfloor \neq \left\lfloor \frac{(v_2 - \Omega_i)}{\Delta I_i} \right\rfloor \qquad (6.2)$$

That is, all given v_1 and v_2 within the extended interval lead to identical integer quotients, whereas values below or above the interval border lead to different integer values. Thus, we write the hash function for each feature parameter f_i for the i=1..24 feature parameter values to:

$$h_i(f_i, \Delta I_i, \Omega_i) = \left\lfloor \frac{(f_i - \Omega_i)}{\Delta I_i} \right\rfloor \qquad (6.3)$$

In recent tests, it has been observed, that the equations 6.1 through 6.3 require adaptation in cases where I_{Low} is of negative and I_{High} is of positive value. Thus the original algorithm as proposed in the work referred to in the acknowledgement chapter needs to be adapted.

7. Test Results

The test environment consists of 2 test blocks (test A and test B), each involving 10 subjects. In test A, the subjects performed an enrollment (calculation of IM) based on 4 writing samples and an additional verification sample. Further, each person was asked to forge the signatures of

the remaining persons. In order to ensure a good quality of the forgeries, a hardcopy of the original signature was made available and the maximum training time was limited to 15 minutes.
In Test B, each person again had been asked to provide a verification sample, plus a forgery for each of the other users. Test B was performed at least a week after Test A. Our system does not generate a true-false response like signature verification systems in general therefore, in order to allow comparison of our new approach with other systems, we define error rates as follows:

- **False Rejection (FR)** error is given, if an authentic individual fails to reproduce the complete biometric hash vector, that is represented by the user's individual IM and is measured in terms of False Rejection Rate (FRR)
- **False Acceptance (FA)** error occurs, if a forger manages to reproduce a biometric hash vector belonging to another person and is measured in terms of False Acceptance Rate (FAR)

Although false acceptances did occur in neither of the tests, the average FRR increased from 3,7% in test A to 10,4% in test B.

Table 3. FAR/FRR during test A

ID	AL	AM	MM	FR	EH	YS	JD	EF	ST	LF
FAR	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
FRR	8%	0%	21%	0%	0%	0%	0%	4%	4%	0%

Table 4. FAR/FRR during test B

ID	AL	AM	MM	FR	EH	YS	JD	EF	ST	LF
FAR	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
FRR	8%	0%	21%	8%	12%	21%	12%	17%	12%	4%

Table 5. Average FAR/FRR

ID	AL	AM	MM	FR	EH	YS	JD	EF	ST	LF
FAR	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
FRR	8%	0%	21%	4%	6%	10%	6%	10%	8%	2%

The average error rates are resulting to FAR=0%, FRR=7.05%.

8. Conclusions and future work

Our new approach of generating hash values from online signatures generates individual patterns for users, based on an actual writing sample and reference data, represented by an interval matrix. This functionality is attractive especially for applications where key management strategies are required, as keys can directly be generated from the system response, the biometric hash vector. Another advantage is the fact, that no reference samples are being stored during the enrollment process, which can avoid replay attacks of individuals being in possession of the reference data, thus increases the security for the overall biometric system. First test results have proven that the parameters selected represent individual handwriting, especially since skilled forgeries were used to determine the FAR. However, the number of subjects was limited and will need to be increased in future test scenarios to allow statistical safe conclusions.
A biometric hash, which can be generated without individual references like the IM, would have a wide number of applications and could offer a great level of security. We will perform further research to validate the possibility of finding appropriate features and global values for the Interval Matrix in order to achieve such functionality based on the approach presented.

9. Acknowledgements

This paper was supported by the work of Michael Haisch during his 2001 master thesis at T.U. Darmstadt: "Design and development of techniques to secure online handwriting reference data for user authentication on mobile devices"

11. References

[1] R. Plamandon, G. Lorette, *Automatic Signature Verifictaion and Writer Identification – the State of the Art*, Pergamon Press plc., Pattern Recognition, 22, 2:107-131, 1989

[2] R. Plamandon, F. Leclerc, *Automatic Verifictaion and Writer Identification: The State of the Art 1989-1993*, International Journal of Pattern Recognition and Artificial Intelligence, 8:643-660, 1994

[3] Y. Sato, K. Kogure, *Online Signature Verification based on Shape, Motion and Handwriting Pressure*, International Conference on Pattern Recognition (ICPR), 2:823-826, Munich, 1992

[4] B. Wirtz, *Stroke-Based Time Warping for Signature Verification*, International Conference on Document Analysis and Recognition (ICDAR), 1, 179-182, 1995

[5] S. Hangai, S. Yamanaka, T. Hamanoto, *On-line signature verification based on altitude and direction of pen movement*, International Conference on Multimedia (ICME), 1:489-492, 2000

[6] C. Vielhauer, R. Steinmetz, *Transitivity Based Enrollment Strategy for Signature Verification*, International Conference on Document Analysis and Recognition (ICDAR), 1:1263-1266, 2001