AGGRESSIVE COMPRESSION OF THE DYNAMICS OF HANDWRITING AND SIGNATURE SIGNALS

Homayoon S. M. Beigi

Recognition Technologies, Inc.†
300 Hamilton Avenue
White Plains, NY 10601, USA
+1 (914) 288-0029 (Tel.) *+1 (914) 243-4965 (Fax.)*
E-mail: beigi@RecognitionTechnologies.com

ABSTRACT

The fields of biometrics and handwriting recognition can benefit from the aggressive compression techniques presented here. A novel approach to compressing the on-line handwriting signal without losing the dynamic information (i.e., velocity) is presented which provides the means to fit almost any signature sample within the 57-byte limit of one of the free magnetic strips on the back of conventional credit cards. The method starts with segmenting the on-line handwriting signal into portions which may be approximated with a set of features from which the velocity and position may be reconstructed. These features are then compressed further using a standard compression technique. Here, a Huffman coding technique is used to bring the size of the average signature including its dynamic information below 57 bytes which is the magic limit imposed by conventional, magnetic credit card strips. Normally two of these strips are blank and may be used for storing the template of the owner's signature for future verification.

1. INTRODUCTION

In the 1990s, Kodak scientists presented an aggressive image compression technique for coding the photo of a credit card holder in one of the three 57-byte magnetic strips available on the back of conventional credit cards. A more important or perhaps complementary piece of information which should be stored on one of these strips is the signature of the individual. Currently, the signature is written on the back of the credit card and is available to any forger to see at the moment of making purchases. In addition, this is only a static image of the individual's signature and lacks the dynamic information such as velocity, order and pressure at different portions of the signature. This dynamic information can substantially increase automatic signature verification results.

Low quality tablets have recently been installed at many department store cash registers to obtain electronic signature samples from customers. These samples are, however, still being treated as images and their dynamic information is not being used. For on-line verification, part of the problem is the storage of the signature templates which would have to be done at a central location accessible to stores through a network. This would be an expensive system which although is quite warranted, will not be easily attainable in practice since many stores do not have network connectivity. A self-contained signature verification system would be made possible if the credit card of the individual included his or her signature sample electronically. The electronic information may also be used to render the signature on a screen for the store manager to review and compare against the individual's signature at the time of purchase.

A methodology is presented her for compressing all the dynamic information within an average signature to less than 57 bytes so that it may be included on the back of a credit card. In addition, it requires less storage than storing the actual x and y coordinates and it maintains all the dynamic information as well. In addition to biometric applications, since the order and velocity information are conserved, stored on-line handwriting data may be used by on-line recognizers without much degradation in accuracy results.

Next section presents a loss-less coding with perfect reconstruction capabilities. Then, the promised lossy

† http://www.RecognitionTechnologies.com

compression technique is formulated by first segmenting the signature into smaller pieces. Each piece is subsequently modeled by only six or less parameters. These parameters may be further compressed using quantization and tabulation. Finally, a Huffman coding scheme is used to further reduce the size of the signature code.

2. LOSS-LESS CODING

This section describes a loss-less compression technique for applications which require exact reconstruction of the handwriting signal from the coded signal. Consider the writing as a sequence of strokes, themselves defined as sequences of $x(t)$ and $y(t)$ coordinates beginning with a pen-down and ending with a pen-up signal. Almost all commercial tablet provide the $x(t)$ and $y(t)$ coordinates at a fixed sampling rate in the order of $100Hz$. In lieu of recording the $x(t)$ and $y(t)$ coordinates directly, one may compute a difference approximation to the local speeds $\dot{x}(t)$ and $\dot{y}(t)$. Since the speed of writing has an upper limit, the number of bits needed to represent $\dot{x}(t)$ and $\dot{y}(t)$ are quite small. Also, by keeping the coordinates of the initial position of each stroke, the stroke may be re-integrated with almost no loss. In fact to avoid any loss, only the Δx and Δy for each time step are recorded and the computation of the real velocity is not even attempted. Before, presenting the results, a Huffman coding scenario is discussed which may be used to code these differences in a table with variable number of bits representing the data. Less frequent differences are coded with longer number of bits and more frequent ones are coded with less bits. This compression proves to be very practical and provides considerable compression of the on-line data without loss of the sequence and velocity information.

This type of compression requires over 1000 bytes for coding an average on-line signature. This is not nearly enough for our objective of an upper limit of 57 bytes, but it certainly has its own applications.

3. AGGRESSIVE LOSSY CODING

To reduce the code-length substantially to meet the upper limit of 57 bytes, we would have to consider a sequence of aggressive compressions. The first of these compressions is the reduction of the point representation of the handwriting to a model representation where the model parameters would describe the handwriting. This will give us a substantial reduction in the code-length. To achieve this, a special segmentation would have to be performed which will be discussed below. Each segment of handwriting will be represented by a constant number of model parameters.

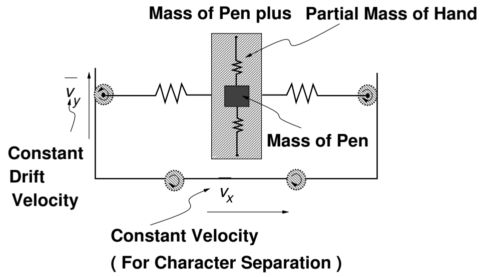

Figure 1: Handwriting Generation Model

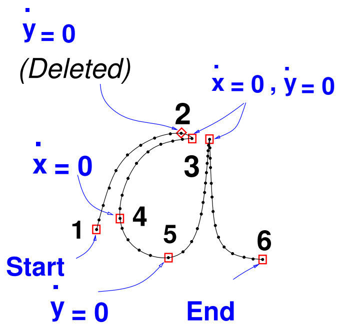

Figure 2: Segmentation of character *a*

3.1. Dynamic Modeling

The author has presented the grounds for the approximations of this paper in [1]. Considering these arguments, let us assume that the differential equations for the handwriting generation process may be approximated by a two dimensional second order differential equation with linear time-invariant coefficients along a piece of writing between any two consecutive extreme positions in each coordinate (x and y), given by figure 2. Under these assumptions, the solution of the approximate differential equation, in the nominal region, would be in the usual sine and cosine form. Therefore, the velocity in each coordinate will also have the same form. For the sake of modeling the handwriting it is better to consider the velocity rather than the position for the apparent reasons of robustness to noise and pre-emphasis. Thus, the velocities in x and y directions under these very crude assumptions would be of the form given in equation 1.

$$\dot{x}(t - t_0) = A_x sin(\omega_x(t - t_0) + \phi_x) + \bar{v}_x$$

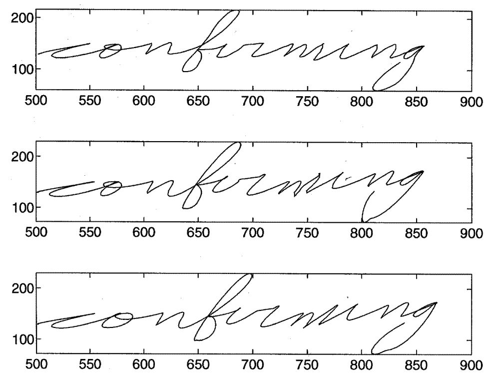

Figure 3: test caption

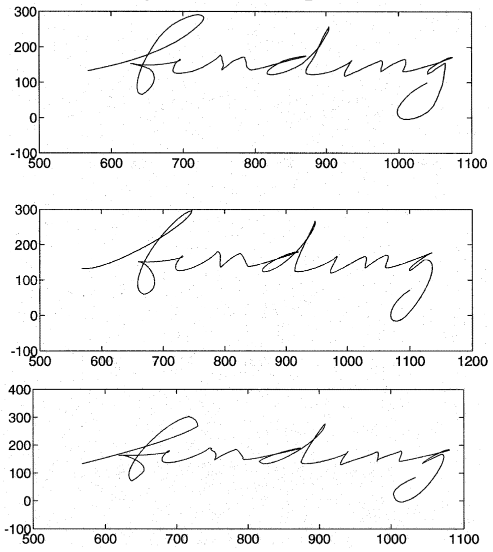

Figure 4: test caption

$$\dot{y}(t - t_0) = A_y sin(\omega_y (t - t_0) + \phi_y) + \bar{v}_y \quad (1)$$

where A_x, ω_x, ϕ_x and $\bar{v}_x$ are the amplitude, frequency, phase and mean velocity for the x direction. Also, A_y, ω_y, ϕ_y and $\bar{v}_y$ are the counterparts in the y direction. Therefore, the first level of compression is achieved by estimating these six parameters which may be used to reconstruct the segment through an integration process starting from the beginning of the stroke at recorded initial values of x and y. See [1] for further information on pre-processing, filtering and the parameter estimation technique. The estimation is formulated as a quadratic optimization problem in [1] and solved by minimizing the error function given by equation 2.

$$E_\eta = (\dot{\vec{\eta}}_d - A_\eta \vec{s})^T (\dot{\vec{\eta}}_d - A_\eta \vec{s}) + \alpha(\omega_\eta - \tilde{\omega}_\eta)^2 + \beta(A_\eta - \tilde{A}_\eta)^2 \quad (2)$$

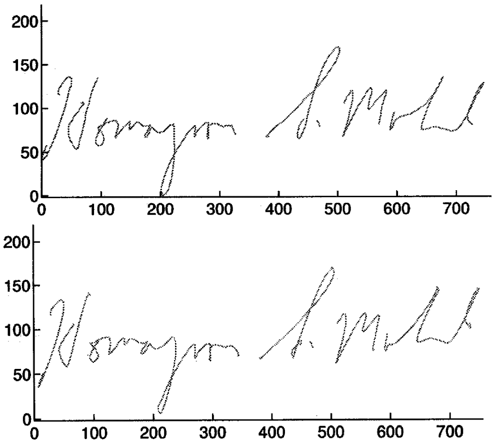

Figure 5: test caption

where, $\vec{s} : s_k = sin(\omega_\eta t_k + \phi_\eta) \quad 1 \leq k \leq n$ and η is the generalized coordinate which may be replaced by x or y. α and β are penalty factors which penalize the the amplitude and frequency from deviating too much from their initial estimates represented by $\tilde{\omega}_\eta$ and $\tilde{A}_\eta$ respectively. Note that, the frequency tends to converge to large numbers to reduce the error E_η without the presence of such penalties. We have a good initial estimate of ω_η based on zero-crossings. We also have a good initial estimate of the amplitude A_η from the original signal. Figures 3 and 4 show the original long strokes of the words, **confirming** and **finding** on top and their reconstruction in the middle of the figures.

3.2. Segmentation

As stated above, the segmentation is done at the points where $\dot{x} - \bar{v}_x$ or $\dot{y} - \bar{v}_y$ are $zero$. Figure 2 shows these segmentation points on the letter **a**.

3.3. Window Count Reduction

Looking at the reconstruction in the middle of figure 3, the **m** and the lower curvature of **g** display, certain anomalies which create sharp changes in the curvature. These are due to short segments being generated in the segmentation process. Figure 2 shows an example of such segment which has been marked **deleted**. Deleting such short segments has two positive outcomes. The first positive result is that the false sharp direction changes are eliminated and a better reconstruction is made possible (see the bottom of figure 3. The second great side-effect is that there are less segments to be represented. Each segment requires a constant number of parameters for its representation regardless of its length. One may at the first glance imagine that the phase does not play an important role, however, if the word is to be segmented in such a way as to use **either** $\dot{x} = 0$ **or** $\dot{y} = 0$, then the phase plays a very important role of synchronization. In cases when a segment boundary is **deleted**, the phase is essential (figure 2).

3.4. Parameter Reduction

The above parametrization of the signal shows reductions in the order of 75% from the loss-less method described earlier in the paper. This leaves the size of an average signature at about 250 bytes which is still not acceptable. One way to do a further reduction in size is to reduce the number of parameters necessary for representing each segment. After looking at 450 signatures, the author has seen a correlation of between 75 and 90 percent between the frequencies ω_x and ω_y. In fact, using ω_x for both dimensions produces quite acceptable reconstructions while reducing the code-length substantially. The bottom of 4 shows the reconstruction of the word **finding** where ω_x was used for both dimensions. Note that the reconstruction even looks better than the one with different frequencies. This is consistently true to the extent that we throw away ω_y reducing the number of parameters to 5. Also, one may only store the phase difference $\phi_y - \phi_x$, further reducing the required parameters from 5 to 4.

3.5. Parameter Quantization and Tabulation

To further reduce the code-length, we can study the dynamic range of the above parameters. By looking at the 450 signatures in our corpus, we have noticed that can get away by using 4 bits to represent the amplitudes. Also, 4 bits is more than enough for keeping the phase difference. Further, the maximum value of 10 rad/s was set as the upper limit of the frequency. Also, a nonlinear table was generated based on data for each parameter to be able to get a better resolution in smaller variations. The frequency was code with 5 bits and was forced to be positive. In this fashion, only 17 bits were needed to represent each segment. Figure 5 shows an original signature on top and a reconstruction based on the 17-bit representation of the segments at the bottom. This brought the number of bytes necessary for an average signature to about 140 bytes. The average number of windows was 61. The maximum number of frames was 133 within the 450 signatures in our database.

3.6. Huffman Coding

Finally, we have many features which portray substantial repetition – a great candidate for Huffman Coding [2]. A 254 element table was generated from all 450 signatures. In practice, this means that a Huffman Table should be computed from a large corpus of signatures and kept for coding and decoding purposes. Finally, after doing the Huffman coding, the average signature came out to about 52 bytes with the maximum at 84 bytes. Huffman coding is loss-less, so the reconstruction results do not differ from those in the previous section.

4. CONCLUSION

We have devised a loss-less compression technique as well as a very aggressive lossy technique. The loss-less method results in signature sized in the order of 1000 bytes. Our lossy method codes keeps most signatures in less than 57 bytes which is the limit on a magnetic strip on the back of conventional credit cards. As opposed to the image compression, the signature compression may still be used if it overflows the 57-byte limit. We may chose to save as much of the signature as we can possibly fit on the magnetic strip. This would be enough for doing a verification since apparently most signatures are not that long. Future work on signature compression will include the pressure information which is readily available by many tablets today. Recognition Technologies is currently developing a signature verification system using this compression technique at its front end. Results of this research will be presented in the future.

5. REFERENCES

[1] Homayoon S.M. Beigi, "Pre-Processing the Dynamics of On-Line Handwriting Data, Feature Extraction and Recognition," *Progress in Handwriting Recognition*, A.C. Downton and S. Impedovo (eds.), World Scientific Publishers, New Jersey, 1997, pp. 191-198.

[2] David A. Huffman, "A Method for the Construction of Minimum-Redundancy Codes," *Proceedings of the IRE*, September 1952, pp. 1098-1101.