Automatic On-Line Signature Verification

VISHVJIT S. NALWA

Automatic on-line signature verification is an intriguing intellectual challenge with many practical applications. I review the context of this problem and then describe my own approach to it, which breaks with tradition by relying primarily on the detailed shape of a signature for its automatic verification, rather than relying primarily on the pen dynamics during the production of the signature. I propose a robust, reliable, and elastic local-shape-based model for handwritten on-line curves; this model is generated by first parameterizing each on-line curve over its normalized arc-length and then representing along the length of the curve, in a moving coordinate frame, measures of the curve within a sliding window that are analogous to the position of the center of mass, the torque exerted by a force, and the moments of inertia of a mass distribution about its center of mass. Further, I suggest the weighted and biased harmonic mean as a graceful mechanism of combining errors from multiple models of which at least one model is applicable but not necessarily more than one model is applicable, recommending that each signature be represented by multiple models, these models, perhaps, local and global, shape based and dynamics based. Finally, I outline a signature-verification algorithm that I have implemented and tested successfully both on databases and in live experiments.

I. Introduction

Signature verification is an art. Whereas we may bring objective measures to bear on the problem, in the final analysis, the problem remains subjective. This art is both well studied and well documented as it applies to human verification of signatures whose only records are visual [13], [5], [10]—that is, as it applies to signatures during whose production no measurement is made of the pen trajectory or dynamics. Let us call such signatures, for which we have only a static visual record, *off-line*, and let us call signatures during whose production the pen trajectory or dynamics is captured *on-line*. Whereas attempts to automate the verification of off-line signatures have fallen well short of human performance to this point, I shall demonstrate that automatic on-line signature verification is feasible.

In a break with tradition, I challenge the notion that the success of automatic on-line signature verification hinges on the capture of velocities or forces during signature production. Whereas velocities and forces can assist us in automatic on-line signature verification, I contend that we should not depend on them solely, or even primarily. *If we were indeed unavoidably consistent over the dimensions of time and force when we signed, the use of pen dynamics during signature production—over and above that of signature shape—would be very useful in detecting forgeries, as dynamic information pertinent to a signature is not as readily available to a potential forger as is the shape of the signature, given just the signature's off-line specimens.* However, I have seen no substantive evidence to the effect that our pen dynamics is as consistent as, or more consistent than, our final signature shape when we sign. My own informal experiments indicate that we typically exhibit similar temporal variations over the production of similar handwritten curves: In general, our speed along high-curvature curve segments is low relative to our speed along low-curvature curve segments, with our average overall speed varying greatly from one instance of a pattern to another irrespective of whether we are producing our own pattern or forging someone else's. This observation suggests that at least the requirement of consistency over time during signature production is of limited value beyond that of consistency over shape. At any rate, irrespective of the velocities and forces generated during the production of a signature, for us to declare two signatures to be produced by the same individual, clearly, it is necessary that the shapes of the signatures match closely.

Hence, I have based my signature-verification strategy primarily on the shapes of signatures; although, at this point, I do depend on time, this dependence is weak and could be removed, as I shall explain. Thus although my verification technique does require the capture of pen trajectories during signature production, unlike other reported on-line signature verification techniques, my technique can do without the explicit capture of any temporal, force, or pressure information during signature production.

I propose that each handwritten on-line signature be represented by multiple models: local and global, shape based and time based, including a model that is local and purely shape based. Whereas global models are easier to devise than local models—and, hence, global models are more widely used than local models—for signatures whose various instances are shaped consistently, global models are less discriminating than and less robust than local-shape-based models. My principal contributions to automatic on-line signature verification are twofold. First, I suggest the weighted and biased harmonic mean as a graceful mechanism of combining errors from multiple models of

Manuscript received September 5, 1996; revised November 22, 1996.

The author is with Bell Laboratories, Holmdel, NJ 07733 USA (e-mail: vic@bell-labs.com).

Publisher Item Identifier S 0018-9219(97)01463-1.

which at least one model is applicable but not necessarily more than one model is applicable. Second, I devise a robust, reliable, and elastic local-shape-based model for handwritten on-line curves. This model is generated by first parameterizing each on-line curve over its normalized arc-length and then representing along the length of the curve, in a moving coordinate frame, measures of the curve within a sliding window that are analogous to the position of the center of mass, the torque exerted by a force, and the moments of inertia of a mass distribution about its center of mass. I have implemented and tested my signature-verification algorithm successfully both on databases and in live experiments.

Successful on-line signature verification that is based on comparing the varying local shapes of signatures offers several important advantages over alternative techniques, especially over those that are not shape based.

- Local-shape-based signature verification is more likely than alternative techniques to reject only those genuine signatures that will be accepted by original signers as nonrepresentative of their signatures, because such signers would typically base their judgment of the fidelity of their signatures on an *a posteriori* visual examination of the detailed shapes of their signatures, rather than on the velocities or forces generated during the production of these signatures. Such acceptance by nonfraudulent signers—of the inevitable rejection of some genuine signatures by a signature-verification system—is key to the acceptance of the signature-verification system by consumers in the marketplace.
- Local shape-based comparisons of signatures, in contrast to global comparisons, avoid lumping together differences between signatures irrespective of their causes, which is important to us because we would like to distinguish between errors that are caused by isolated mistakes, such as inadvertent isolated gaps in writing, and errors caused by systematic deviations, such as those due to different writing styles.
- Local-shape-based signature verification can potentially highlight, for human consumption, local "nonobvious" similarities and discrepancies between the shapes of two signatures—perhaps so that a customer or a court of law can *see* why a particular signature was accepted or rejected.
- Shape-based signature verification does not require us to be consistent over the additional dimensions of time and force when we sign, a requirement that would alter the traditional expectation from us that we be consistent over only the shape of our signature when we sign. Alteration of this traditional expectation, it seems, would force many of us to change "the way in which we do business," weakening what is probably the strongest argument in favor of the continued use of handwritten signatures for verification (see Section II).

Of course, if we were unavoidably consistent over the dimensions of time and force when we signed, this last item would be a nonissue.

I would like to point out here, that because of their time independence, most of the tools I have developed for the elastic local comparison of handwritten shapes are immediately applicable both to on-line handwriting verification—which could be used to verify a user's identity by requesting the user to write something specific—and to on-line handwriting recognition. Note here, however, that my use of the pen trajectory to parametrize each on-line curve implies that visually identical curves that are traversed differently will be represented differently. Whereas this aspect of the representation I propose is advantageous to verification, it is disadvantageous to recognition. Further, note that signature verification is both easier and more difficult than handwriting recognition. Verification is easier than recognition because, in verification, we know *a priori* what pattern to expect: All that successful verification entails is the comparison of an input pattern with a stored model. However, verification is more difficult than recognition because, unlike in recognition, where we are justified in assuming a cooperative human, in verification, we must allow for an adversary who is keenly intent on deceiving the system. Hence, whereas the answer in verification might simply be a *yes* or a *no*, successful verification requires the ability to detect subtle differences between patterns, an ability not required by recognition. More specifically, *successful signature verification hinges on the ability to distinguish between inadvertent intrasigner variations on the one hand, and intersigner variations and advertent intrasigner variations on the other hand*. We shall discuss this assertion at length in Section III.

I have organized this paper into ten sections. In Section II, I categorize the various techniques commonly used to verify the identities of individuals. In Section III, I describe what constitutes successful signature verification. In Section IV, I summarize the state of the art of automatic on-line signature verification as recorded in the published literature. In Section V, I highlight the key features of my approach, several of these features differentiating my approach from the prior art. In particular, I flesh out the following fundamental concepts that underlie my approach:

A) harmonic mean;
B) jitter;
C) aspect normalization;
D) parameterization over normalized length;
E) sliding computation window;
F) center of mass;
G) torque;
H) moments of inertia;
I) moving coordinate frame and saturation;
J) weighted cross correlation and warping.

In Section VI, I outline my algorithm, which I have implemented and tested both on databases and in live experiments. In Section VII, I illustrate my algorithm with a detailed example. In Section VIII, I describe the performance of my implementation on three databases created by Bell Laboratories. In Section IX, I describe a particular

signature-verification system that runs my algorithm in real time on a notebook personal computer, with this computer coupled to an electronic writing tablet for capturing on-line signatures. I conclude with Section X, where I list some of the outstanding issues in automatic on-line signature verification.

II. Alternatives to Signature Verification

Signature verification is only one of several techniques commonly used to verify the identities of individuals. Broadly, the various techniques used for this purpose adopt one or more of five strategies [19].

1) You remember some information, such as a password or a personal identification number (PIN).
2) You are privy to some personal detail, such as your date of birth or your mother's maiden name.
3) You possess some object, such as a magnetic card or a key.
4) You possess some unique physical characteristic, such as a fingerprint or a retinal vascular pattern.
5) You possess the ability to perform some action consistently at will—such as sign your name or speak a phrase—in a fashion that is difficult to duplicate by others.

See [14] for a detailed discussion of the various possibilities. Among the above strategies, Strategies 1)–3) can be inadvertently or intentionally compromised. In contrast, Strategy 4) cannot be compromised, and Strategy 5) is generally difficult to compromise. Strategy 4) is clearly more objective than Strategy 5). However, Strategy 4) has a social stigma associated with it because of this strategy's widespread use in the criminal justice system. Further, Strategy 4) is easier to compromise under coercion than is Strategy 5), not only because the latter entails a voluntary action, but also because this action is likely to deviate from its norm under stress. Among the various possibilities that lie within the realm of Strategy 5), the handwritten human signature is without doubt the most popular, especially in financial transactions. Further, there seems to exist a strong cultural bias toward the continued use of handwritten human signatures for authorization and authentication. Thus, there is money to be made in the robust and reliable automation of signature verification. Such an automation would not only detect attempts at fraud, but also greatly discourage such attempts; we saw several additional advantages of automatic signature verification in the introduction.

III. Evaluating Performance

For a signature-verification system to be useful, the system must commit few errors in practice. The strategy often adopted to obtain an indication of a system's error rates, without actually introducing the system into the marketplace, is to *field test* the system on a limited scale; but even a limited field test can be expensive and time consuming. Hence, it is useful to devise criteria to help us decide whether to field test a system. I have come up with the following *two criteria to evaluate a signature-verification system* that is yet to be field tested; we shall discuss both criteria in detail.

Criterion 1: When you try the system in person, it must *work*.

Criterion 2: When you test the system on large databases, it must exhibit low statistical error rates.

Neither criterion is sufficient, and both are necessary.

Criterion 1 is not sufficient in itself because any evidence of performance gathered from one or a few isolated individuals is anecdotal: Chances are slim that these individuals are representative of the population at large. Criterion 2 is not sufficient in itself as, in the evaluation of a verification system, unlike in the evaluation of a recognition system, it is necessary to consider determined forgers who have access to feedback from the system. The difficulty of a successful forgery, given such feedback, provides a more realistic assessment of the vulnerability of a verification system than does a preexisting database. Bear in mind here that genuine signers would also adapt to the system in the marketplace, learning quickly what it is they have to do to have their signatures accepted by the system in the first try—assuming, of course, that the answer put out by the system is correlated to characteristics of the signature apparent to the signer. This raises two important points. One, as users of the system, both genuine signers and forgers, adapt to the system, the observed performance of the system will change: Both fewer genuine signers and fewer forgers will be rejected by the system. Two, the observed performance of the system for new users of the system will be different than that for accustomed users. Let us now discuss both Criterion 1 and Criterion 2 in turn.

Criterion 1, of course, begs the issue unless we can reach agreement on what *work* means. I can think of at least three conditions that must be met for us to declare a system to *work* when it is tried in person.

- The system must recognize your visually similar scribbles consistently, notwithstanding discrepancies in velocities during the production of these scribbles, and notwithstanding minor inadvertent discrepancies in their shapes.
- You must find it difficult, if not impossible, to forge someone else's signature successfully—even more so, to do so consistently—irrespective of whether you trace the signature, copy the signature, practice the signature first, have complete knowledge of the signature's generation, or know precisely the strategy adopted by the signature-verification system.
- You must not be able to generate a scribble that is visually disparate from your signature and is yet accepted by the signature-verification system as your signature.

The first condition enables genuine transactions. The second condition hinders *second-party fraud*—that is, fraud by an entity other than the genuine signer. The third condition hinders *self fraud*—that is, fraud by the genuine signer.

Omission of the third condition would open up the possibility of a genuine signer authorizing a transaction with the *a priori* intent of later denying this authorization by pointing to the visual discrepancy between the authorizing signature and the expected signature as proof of second-party fraud.

In the context of on-line signature verification, we need to qualify the first condition above aimed at the in-person evaluation of a signature-verification system—namely, the condition that the system must recognize your visually similar scribbles consistently. In on-line signature verification, we require that all genuine instances of a signature be traversed qualitatively similarly as functions of time (i.e., with similar trajectories), irrespective of whether we require that these instances be produced quantitatively similarly as functions of time (i.e., with roughly similar, or even proportional, velocities). Whereas the latter requirement might be an imposition on genuine signers as we discussed, the former requirement is met naturally by genuine signers. *The critical advantage of on-line signatures over off-line signatures in the automation of their verification is precisely the availability of the pen trajectories during the production of on-line signatures*. It is these trajectories that we match in on-line signature verification, rather than their end products, which are just off-line signatures. If we assume that the information in each on-line signature is a superset of the information in its off-line counterpart, it is clear that successful automatic on-line signature verification is a precondition for successful automatic off-line signature verification.

I stress again that Criterion 1 is essential for evaluating a verification system, more so than for evaluating a recognition system, because the quality of a verification system hinges on the inability of determined individuals to defeat the system. Then, to evaluate a verification system thoroughly, we must provide determined forgers complete access to the system—access such as forgers might gain once the system is introduced into the marketplace. Such access provides individuals with the opportunity to experiment with the system, to learn from the system response, and eventually to discover loopholes in the system, if any. It is prudent to assume that a system's loopholes would sooner or later be discovered if the system were ever introduced into the marketplace. Whereas the evaluation of Criterion 1 is subjective, I must mention that I have had little difficulty finding willing and determined forgers: Most individuals seem to relish the gamelike nature of trying to beat the system, and go to the task with a vengeance. However, none of my forgers have been professional forgers.

What factors contribute to a successful forgery? Let us assume that the forger has complete knowledge of the production of the signature to be forged and of the signature-verification strategy used. Then, in my experience with amateur forgers, the *two foremost factors that contribute to a successful forgery* are these:

1) inconsistency across instances of the genuine signature used by the verification system to build a model of the signature;
2) simplicity of the genuine signature, this simplicity characterized by the domination of the signature by a few low-curvature strokes.

The simpler the signature, the greater the consistency needed to thwart forgeries. Another factor that contributes to successful forgery is inherent similarity between the writing style of the forger and the writing style of the original signer. Deftness of the forger at drawing also helps, with such deftness enabling the forger to reproduce pen strokes gracefully and with control. In the circumstances particular to my live experiments, the capacity of the forger to understand the feedback provided by the verification system, perhaps only intuitively, and learn from this feedback was also a factor. Another contributing factor, not to be overlooked, is the determination of the forger to succeed.

Criterion 2 for evaluating a signature-verification system also requires some background—as did Criterion 1. Now, in any verification task, there are two types of errors we can commit: false rejects and false accepts. In the current context, a *false reject* is a signature that we reject even though the signature is not a forgery, and a *false accept* is a signature that we accept even though the signature is a forgery. Clearly, we can trade off one type of error for the other type of error. In particular, if we accept every signature as a genuine, we shall have 0% false rejects and 100% false accepts, and, if we reject every signature as a forgery, we shall have 100% false rejects and 0% false accepts. Thus, in the statistical evaluation of a verification system, whether on a database or otherwise, we must determine the percentage of false accepts as a function of the percentage of false rejects. The ensuing curve—the *error tradeoff curve*—which trades off false accepts for false rejects, is often characterized by its *equal-error rate*, which is the error rate at which the percentage of false accepts is equal to the percentage of false rejects. The equal-error rate, despite its convenience as an indicator of system performance, of course, is no substitute for the actual tradeoff curve, especially if we intend to operate the system in a range outside the immediate vicinity of the equal-error rate.

The specification of a tradeoff curve relating false accepts to false rejects assumes knowledge of, and agreement on, a *ground truth*. In reality, it is not always clear what a false reject is. In particular, is it an error to reject a signature that is produced by the original signer but that *looks* substantially different from that signer's specimen signatures? If not, who or what decides whether two signatures *look* different? If yes, are we affording opportunities to individuals to disown their signatures after the fact? Note that, even if we were to reach a consensus on what constitutes a false reject, two systems with similar error tradeoff curves could perform very differently in practice. In particular, the nature of the false rejects—and false accepts—of the two systems could be quite different. Such a difference can be very important from a practical standpoint. For instance, whereas a consumer might be willing to assume responsibility for a false reject that is visually dissimilar to the consumer's

typical signature, the very same consumer is likely to be annoyed if a false reject is visually similar to the consumer's typical signature. Thus, given two systems with similar error tradeoff curves, one system might be accepted by the marketplace, and the other rejected.

It is thus clear that the evaluation of a signature-verification system cannot simply be reduced to a graph, or to a set of numbers. Numbers might help, but they cannot suffice; at best, numbers obtained from a database, or from a field trial, provide a sample of system performance. It should not surprise us that there is no clear-cut objective criterion to evaluate a signature-verification system. Signature verification, after all, is an art, and even though we might bring objective measures to bear on the problem, in the final analysis, the problem remains subjective.

IV. Prior Art

My review of the literature, and of the most comprehensive published survey [9] with its accompanying bibliography [15] (see also [7]), indicates the existence of a widely held belief that the temporal characteristics of the production of an on-line signature are key to the signature's verification. I am not sure what the basis for this belief is—after all, we have for centuries relied on a visual examination of a signature to verify the signature's authenticity. Of the many possible reasons for this belief, two reasons come readily to mind. The first reason is that, in experiments, the temporal characteristics of signature production are seen to provide better system performance than alternate characteristics. The second reason is that the production of a signature is believed to be necessarily a *reflex action*, or a *ballistic action*, rather than a *deliberate action* [9]. Ballistic handwriting is characterized by a spurt of activity, without positional feedback, whereas deliberate handwriting is characterized by a conscious attempt to produce a visual pattern with the aid of positional feedback.

I challenge, on two counts, the belief that signature production is necessarily ballistic and also the more widely held notion that the temporal characteristics of signature production are key to signature verification. The first count is that many signers—including most of my acquaintances—can produce their signatures both ballistically and deliberately, with the exact mechanism of production in a particular instance depending on the urgency and importance of the task. In general, it is fast handwriting that is ballistic [3] rather than signature production *per se*, and many of us have and exercise control over the speed with which we sign. The second count is that even if we were to group together all the instances of the ballistic production of a signature, there is no compelling reason why these instances would exhibit temporal consistency. I suspect that the apparent success of the use of the temporal characteristics of on-line signatures in their verification is, at least partially, an artifact of the testing methodology: It is clearly easy to detect forgers on the basis of time when these forgers, being unaware that time is critical to verification, are making every effort to reproduce the shape of the signature they are trying to forge, with little attention to time. I must emphasize that I am not arguing here that the temporal characteristics of signature production are not potentially useful for automatic signature verification, but rather that these characteristics should not be the primary determinants of our decision.

The various strategies reported in the literature for the automatic verification of on-line signatures rely typically either on comparing specific features of signatures or on comparing specific temporal functions captured during signature production, or, perhaps, on both [9]. Although the signature features that are compared are typically global—such as the total time taken, or the average or rms speed, acceleration, force, or pressure [2], [12]—these features could be local, such as the starting orientation or speed. Typical signature functions that are compared include pressure versus time, and the horizontal and vertical components of position, velocity, acceleration, and force, each versus time [18]–[20]. The more sophisticated among the temporal-function-based approaches allow the horizontal axes of the functions to warp during comparison [18], [20], and approaches that rely on comparing temporal functions reputedly perform better, in general, than approaches that rely solely on comparing features. Barring the straightforward representation of the coordinates, orientation, and curvature of a signature along its length, all as functions of time, few attempts have been made to characterize the local shape of a signature. One exception to this observation is the work of Hastie and his coauthors [6], who match signatures by first segmenting them at places of low speed and then seeking the optimal affine transformation between each segment and its stored prototype. However, segmentation-based approaches are, in general, not robust owing to their rapid deterioration in the presence of segmentation errors that are bound to occur sooner or later.

I have had the opportunity to try out, in person, only one well-known signature-verification system created outside Bell Laboratories, and the various statistical results reported in the literature are difficult to compare because of the very disparate conditions under which these results were produced. In database testing, we can in practice obtain almost any desired statistical tradeoff between false rejects and false accepts if we allow ourselves the luxury of suitably pruning or restricting the database on which we test the system. Such pruning is often easy as the performance of a verification system on a database is typically limited by the database's *goats*, a term used to describe the typically few individuals who account for a large majority of the errors—in our case, by producing signatures that are either inconsistent or degenerate; see, for instance, [2] and [9]. Note, here, also that the false-reject statistics obtained in laboratory settings are likely to be overly optimistic vis-á-vis results that would be obtained in more unregulated settings [9]. Further, I point out that the validity of many of the results reported in the literature is suspect in real-world settings, because, in most experiments, forgers are not provided all the knowledge that they could gain over time if the verification system were ever introduced into the

marketplace. For instance, it is clearly easy to detect forgers who are making every effort to duplicate a shape while all that the verification system is measuring is the total time taken; under such circumstances, a forger would clearly have greater success by ignoring the shape completely and concentrating on duplicating the total time taken. I believe this artifact of testing to be a significant contributor to the widespread emphasis given to the temporal characteristics of on-line signatures in their automatic verification.

There is a plethora of reasons other than those that I have just mentioned why a direct comparison of the various published statistical results is of little value. Some tests allow each user multiple tries to have a signature validated by the verification system, whereas other tests do not permit multiple tries.[1] In some experiments, the users are highly motivated—for instance, by financial reward [19]—whereas in other experiments, the users are largely unmotivated. In some experiments, the false-accept statistics are based on so-called random forgeries that typically have little or no similarity to the genuine signatures they are supposed to represent. A *random forgery*, as its name suggests, is a pattern that by design is not related to the original signature; such a forgery is to be expected when a forger does not have ready access to the original signature, as might happen, for instance, if a credit card were stolen in transit before a genuine signature could be produced on the card.

All the reasons stated above point to the difficulty of comparing the various published statistical results. Hence, as a practical matter, we have no choice but to take recourse to our *common sense* in judging the quality of the various efforts toward automatic on-line signature verification. My own examination of the various published techniques makes me very sceptical of their efficacy in practice as stand-alone techniques. This scepticism is borne of my conviction that the varying local shape of a signature, as we proceed along the length of the signature, is key to the signature's verification, and, in my judgement, the published techniques are by and large conceptually inadequate at capturing the local shape of a signature.

V. Key Concepts

In this section, I describe some of the key ideas that underlie my approach to on-line signature verification. Many of these ideas are novel, and others have not been applied previously to signature verification. The presentation of the ideas might seem disorderly, but I shall bring all of them together in an algorithm in Section VI.

The first idea, that of the harmonic mean, provides a graceful mechanism of combining errors from multiple models of which at least one model is applicable, but not necessarily more than one model is applicable. The second idea, that of jitter, provides a measure of any abrupt adjustments in the pen trajectory during the course of signing, a large value of jitter often indicating a forgery. The third idea, that of aspect normalization, allows individuals to scale their signatures unequally along the horizontal and vertical dimensions, such unequal scaling of signatures often observed in practice.

The other ideas relate to creating and comparing robust, reliable, and elastic local-shape-based models of handwritten on-line curves. The local-shape-based model of a handwritten on-line signature I propose is based on first parameterizing the signature over its normalized arc-length, and then representing, as functions of arc-length, entities analogous to the position of the center of mass, the torque exerted by a force, and the moments of inertia of a mass distribution about its center of mass, each measurement made over a window that is sliding along the length of the signature in unison with the motion of a coordinate frame with respect to which the measurement is made. Let us call these functions of arc-length—functions that we are using to represent the signature—the *characteristic functions* of the signature. The proposed characteristic functions provide robust descriptions of local shape that depend on the position, orientation, and curvature of the curve along its length. The straightforward approach to measuring the orientation or curvature of a curve would involve estimating the curve's derivatives, an operation that is well known to emphasize noise.[2] Given the proposed representation, we compare a curve to its prototype by computing the cross correlation of each of the curve's characteristic functions with the function's prototype. In the computation of these cross correlations, we weight more heavily those portions of each characteristic function over which the original signer is relatively consistent, and less heavily those portions over which the original signer is relatively inconsistent. Further, in the computation of these cross correlations, we allow all the characteristic functions of a curve to warp simultaneously along their lengths so as to maximize an overall measure of the cross correlation between the characteristic functions and their prototypes.

A. Harmonic Mean

The most popular method of combining two errors is to compute their root weighted-mean square. In particular, if we represent two errors by ξ_1 and ξ_2, then the *root weighted-mean square* of the two errors is $\xi = [\alpha_1\xi_1^2 + \alpha_2\xi_2^2]^{1/2}$, whose isocontours are ellipses—that is, each of whose loci in the ξ_1–ξ_2 plane for a particular ξ is an ellipse. An immediate generalization of this error combination is

[1] If we assume that each attempt by an individual to have a signature accepted by a verification system is independently identically distributed (i.i.d.), and that the probability of a false reject in a single try is p_r, and that the probability of a false accept in a single try is $p_a \ll 1$, then the probability of a false reject in each of n tries is p_r^n, and the probability of a false accept in at least one of n tries is approximately np_a.

[2] The effect of differentiation on signal noise is appreciated easily by examining the effect of differentiation's discrete implementation, differencing, on a discrete signal. Consider a discrete one-dimensional (1-D) signal z_i corrupted by additive, i.i.d., zero-mean noise η_i with variance σ_η^2. Say, $s_i = z_i + \eta_i$. Then, the noise $(\eta_{i+1} - \eta_i)$ in the first difference $(s_{i+1} - s_i)$ of the signal is easily seen to have zero mean and variance $2\sigma_\eta^2$ [11]. That is, the variance of the noise in the first difference of a signal is twice the variance of the noise in the original signal. It follows that the variance of the noise in the nth difference of a signal is 2^n times the variance of the noise in the original signal. Finally, note that the straightforward discrete computation of a curve's orientation and curvature entails taking its first and second differences, respectively.

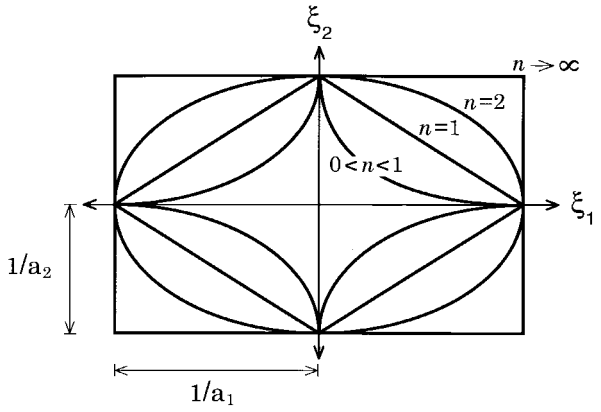

Fig. 1. Superellipses.

$\xi^n = a_1^n|\xi_1|^n + a_2^n|\xi_2|^n$, where a_1, a_2, and n are all positive. Fig. 1 illustrates the isocontours of this expression for various n when $\xi = 1$; curves such as those illustrated, which are generalizations of ellipses, are called *superellipses* [11]. A possible drawback of this ubiquitous family of error combinations is that this family takes into account each of the two errors, *irrespective of the other error*. In a sense, this mechanism of combining errors AND's the errors—assuming here that a low error corresponds to a Boolean 1, and a high error corresponds to a Boolean 0. But, what if we wish to OR the errors? We might want to do this, for instance, if ξ_1 and ξ_2 are derived from two different models of which at least one model is applicable, but not both models are necessarily applicable.

One mechanism of ORing two errors, if you will, is to replace the superelliptical isocontours above by superhyperbolic isocontours. We can accomplish this goal easily by constraining n in the superelliptic error expression above to be negative instead of positive. Say, $m = -n$. Then we get $1/\xi^m = 1/(a_1^m|\xi_1|^m) + 1/(a_2^m|\xi_2|^m)$, where a_1, a_2, and m are all positive. Fig. 2 illustrates the isocontours of this expression for various m when $\xi = 1$; curves such as those illustrated, which are generalizations of hyperbolas, are called *superhyperbolas*. Now, if we put $m = 1$, and assume that both ξ_1 and ξ_2 are positive, then ξ will become the *weighted harmonic mean* of ξ_1 and ξ_2. That is, we will have

$$\frac{1}{\xi} = \frac{1}{a_1\xi_1} + \frac{1}{a_2\xi_2}$$

or equivalently

$$\xi = \frac{a_1a_2\xi_1\xi_2}{a_1\xi_1 + a_2\xi_2}.$$

The isocontours of the weighted harmonic mean are simply hyperbolas with asymptotes $\xi_1 = \xi/a_1$ and $\xi_2 = \xi/a_2$, as illustrated in Fig. 3 for $a_1 = 2$ and $a_2 = 1$. Now, if we put $a_1 = a_2 = 2$ in the expression for the weighted harmonic mean, ξ will simply become the unweighted harmonic mean of ξ_1 and ξ_2. Whereas the unweighted arithmetic mean of a collection of numbers is their average—the sum of the numbers divided by their count—the reciprocal of the unweighted harmonic mean of a collection of numbers is

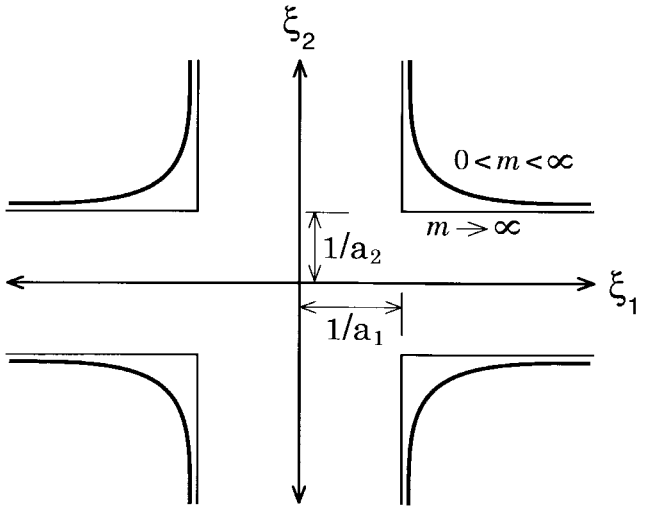

Fig. 2. Superhyperbolas.

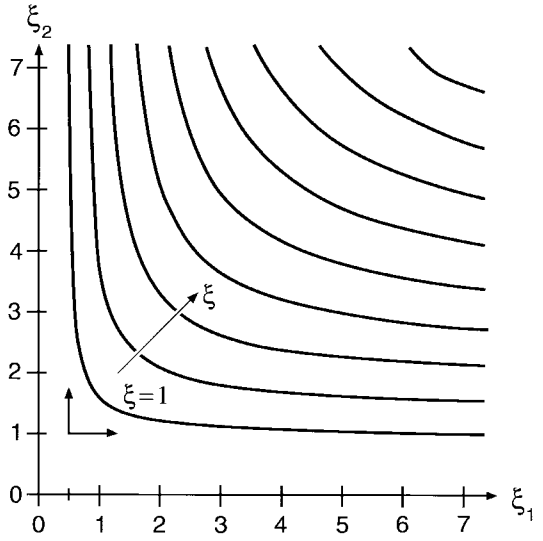

Fig. 3. Isocontours of weighted harmonic mean.

the average of the reciprocals of the individual numbers, as reflected in the above expression for the harmonic mean [1]. Before we move on, note that, if we had put $m = 2$ instead of $m = 1$ while deriving the above expression for the weighted harmonic mean, then ξ would have become the root *weighted-harmonic-mean square* of ξ_1 and ξ_2, and we would have

$$\xi = \left[\frac{\alpha_1\alpha_2\xi_1^2\xi_2^2}{\alpha_1\xi_1^2 + \alpha_2\xi_2^2}\right]^{1/2}$$

where we have made the substitutions $\alpha_1 = a_1^2$ and $\alpha_2 = a_2^2$.

Finally, let us further generalize the notion of the weighted harmonic mean of two numbers by allowing each of the two numbers to be biased. In particular, let us define the *weighted and biased harmonic mean* ξ of two numbers ξ_1 and ξ_2 to be

$$\xi = \frac{a_1a_2(\xi_1 + b_1)(\xi_2 + b_2)}{a_1(\xi_1 + b_1) + a_2(\xi_2 + b_2)}.$$

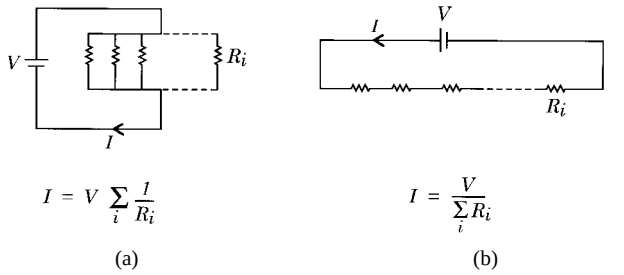

Fig. 4. Currents through resistors in parallel and series.

Here, ξ_1 is said to be biased by b_1, and ξ_2 by b_2. What such biasing does to the hyperbolic isocontours of the weighted harmonic mean—that is, to isocontours such as those illustrated in Fig. 3—is translate them by $-b_1$ along the ξ_1-axis and by $-b_2$ along the ξ_2-axis.

Our discussion of various combinations of individual errors here applies, in all its generality, equally well to errors from multiple models as it applies to errors from just two models. In particular, whereas $\xi^n = \Sigma_i\, a_i^n|\xi_i + b_i|^n$, where a_i and n are positive, provides a convenient mechanism of ANDing ξ_i in a manner of speaking—assuming that a small ξ_i corresponds to a Boolean 1, and a large ξ_i to a Boolean 0—$1/\xi^m = \Sigma_i 1/(a_i^m|\xi_i + b_i|^m)$, where a_i and m are positive, provides a convenient mechanism of ORing ξ_i.

For those of you familiar with elementary circuit theory, an electrical interpretation of the harmonic mean is instructive here. Given a set of resistors arranged in parallel, as in Fig. 4(a), the overall current through the set of resistors will be inversely proportional to the harmonic mean of the individual resistances in the set; in such a circuit configuration, if *any* of the resistors arranged in parallel has a *low* resistance, the overall circuit current will be *large*. In contrast, if the resistors were arranged in series, as in Fig. 4(b), the overall current would be inversely proportional to the arithmetic mean of the individual resistances; in such a circuit configuration, if *any* of the resistors in series has a *high* resistance, the overall circuit current will be *small*.

B. Jitter

When individuals attempt to copy or trace a preexisting curve closely, as often happens in forgery, they produce a "jitter" owing to the act of constantly correcting the pen trajectory to conform to the *a priori* curve. Such jitter is illustrated in Fig. 5. This jitter often exceeds the quantization errors that result from the use of a discrete spatial sampling grid to capture on-line signatures—these quantization errors, of course, depending on the rate of pen motion vis-á-vis the temporal sampling rate. A measure of *jitter* that I have found useful is

$$\text{Jitter} = 1 - \frac{\text{length of polygonal (or other) smoothing approximation to data}}{\text{total sum of intersample distances}}.$$

Note that $0 \leq \text{Jitter} \leq 1$.

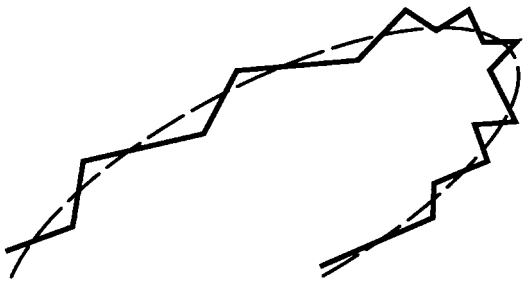

Fig. 5. Jitter.

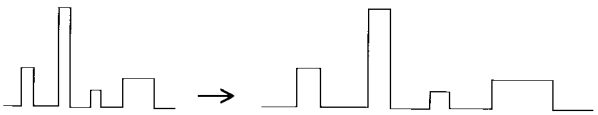

Fig. 6. Aspect variation.

C. Aspect Normalization

Individuals do not scale their signatures equally along the horizontal and vertical dimensions when they sign [4]. You might, for instance, make your signature fatter without making it any taller, as illustrated in Fig. 6. Hence, before we verify the shape of a signature, we must standardize the signature's ratio of height to width—this ratio called the signature's aspect. A measure of *aspect* that I have found useful is

$$\text{Aspect} = \frac{\text{total sum of vertical displacements}}{\text{total sum of horizontal displacements}}.$$

The displacements in this expression are the unsigned vertical and horizontal components of the arc-lengths of curves fitted to the data.

D. Parameterization over Normalized Length

The *parameterization* of a curve is the creation of a one-to-one mapping from a subset of the real line onto the curve. The real line here, which is said to *parameterize* the curve, provides an index or *parameter* by which we can conveniently locate any point on the curve. Once we have parameterized a curve, we can describe various properties of the curve, such as the orientation of the curve, as functions of the curve's parameter.

One possible parameter of an on-line curve is the time instant(s), relative to an arbitrary fixed time, at which the pen is located at a position along the curve. This particular choice of parameter seems to have been adopted universally for on-line signatures in the past, in part, perhaps, because of the ready availability of the pen trajectory as a function of time, and in part because of the widespread belief discussed in Section IV that the temporal characteristics of signature production are key to on-line signature verification. The choice of time as a parameter for on-line signatures seems also to have been influenced by the use of time as a parameter in the relatively well-developed disciplines of speech recognition and speech verification [16]. However, note two important distinctions between speech and handwritten on-line signatures: 1) unlike speech, handwritten on-line

signatures have no inherently unavoidable reason to distort in appearance under distortion of time, and 2) when we sign, we implicitly aim to generate spatially consistent patterns, and not temporally consistent, or even spatiotemporally consistent, patterns. Recall that the primary argument in favor of using handwritten signatures for authorization and authentication is that their continued use would not require us to change "the way in which we do business." Hence, it is probably unacceptable to request individuals to be temporally consistent when they sign; further, it is not even clear that most individuals could meet such a request even if they tried.

I contend that the parameterization of any handwritten on-line curve, including on-line signatures, should be over a spatial metric, rather than over a temporal metric. I suggest that we parametrize each handwritten on-line signature over its normalized arc-length—that is, over the distance traveled by the pen while the pen is in contact with the writing surface, this distance measured as a fraction of the total distance traveled by the pen while the pen is in contact with the writing surface. Let us denote the normalized arc-length of a signature by l. Parameterization of a curve over its arc-length is standard practice in differential geometry [8].

E. Sliding Computation Window

Once we have parameterized a signature over its normalized arc-length, what characteristics of the signature do we represent as functions of the signature's normalized arc-length? The characteristics of the signature we shall represent are derived from the center of mass, the torque, and the moments of inertia of the signature computed over a window that is sliding along the length of the signature in unison with the motion of a coordinate frame. Before we discuss each of these characteristics in sequence next, let us spend some time on the sliding window over which we shall compute these signature characteristics. Let us call this sliding window the *computation window* to distinguish it from another sliding window that we shall discuss in the context of the moving coordinate frame. We shall discuss the moving coordinate frame in Section V-I.

Two questions that arise immediately in the context of a sliding window are: what is the window's width, and what is the weighting along the length of the window? With regard to the window's width, the broader we make the sliding window, the more we shall average the signal noise, thus increasingly suppressing the net effect of noise on our computations of signature characteristics.[3] However,

[3] To see the effect of averaging on signal noise, consider once again, as we did in Footnote 2, a discrete 1-D signal z_i corrupted by additive, i.i.d., zero-mean noise η_i with variance σ_η^2. Say, $s_i = z_i + \eta_i$. Then, the noise $\Sigma_i\ w_i \eta_i$ in the weighted average $\Sigma_i\ w_i s_i$ of the signal is easily seen to have zero mean and variance $\sigma_\eta^2\ \Sigma_i\ w_i^2$, where w_i are the weights (all positive), $\Sigma_i\ w_i = 1$, and all summations are implicitly between $i = 1$ and $i = n$ [11]. That is, the variance of the noise in the weighted average of a signal is the sum-of-squares of the individual weights times the variance of the noise in the original signal. As all the weights are positive, with unit sum, this sum-of-squares is at most one. In particular, if all the weights are equal and the average is taken over n samples, then this sum-of-squares is $1/n$.

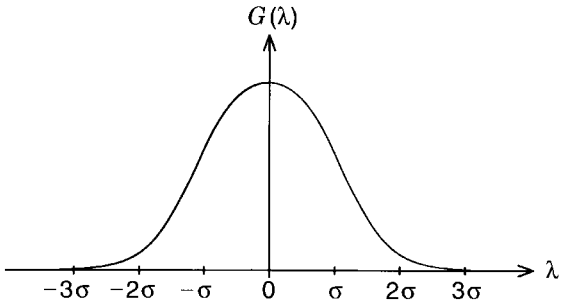

Fig. 7. Gaussian.

a broader window does more than just increasingly smooth out noise: It also increasingly smooths out actual signature variations, making it harder to detect discrepancies between forgeries and genuine signatures. Hence, in choosing the width of our window, we must balance the prospect of undersmoothing the noise against the prospect of oversmoothing the signature. Typically, a reasonable choice for the width of the sliding window is a fraction of the length of an individual "character." We shall discuss this choice at greater length in Section V-I.

Now, coming to the question of the weighting along the length of the window, several choices are possible. We could, of course, weight the window uniformly along its length, but such a weighting would not best serve the desirable goal of gradually phasing in and phasing out our center of attention along the length of the signature as we slide the window along this length. A uniform weighting could lead to relatively abrupt changes in our computed values of signature characteristics when we slide our window along the length of the signature, changes that a discrete implementation of our computations might fail to capture adequately. A straightforward weighting function that would allow us to phase in and phase out our center of attention along the signature gradually is a Gaussian centered at the center of the sliding window, this Gaussian narrow enough for it to taper off to near zero at either end of the window. Recall that the Gaussian is a bell-shaped function—of the type illustrated in Fig. 7—whose equation in 1-D, ignoring a scale factor, is $G(\lambda) = \exp(-\lambda^2/(2\sigma^2))$, where σ controls the width of the Gaussian. Let us adopt a 1-D Gaussian weighting function centered at the center of the sliding window here, the σ of this Gaussian satisfying $L \approx 2\sigma$, where L is half the width of the sliding window. Then, normalizing the Gaussian to have a unit integral over the width of the sliding window, we arrive at the *window function*

$$g(\lambda) = \frac{\exp(-\lambda^2/(2\sigma^2))}{\displaystyle\int_{-L}^{+L} \exp(-\gamma^2/(2\sigma^2))\, d\gamma}, \qquad -L \le \lambda \le +L$$

where $\sigma \approx L/2$, and $g(\lambda) = 0$ outside the range $\pm L$.

F. Center of Mass

Assume the following: The signature is parameterized over its normalized arc-length l, as we discussed in

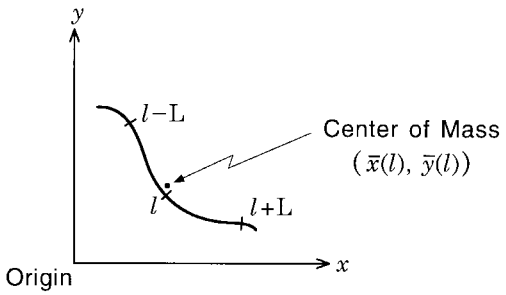

Fig. 8. Center of mass.

Section V-D, the signature has a weighted window $g(\lambda)$ of span $\pm L$ sliding over its length, as we discussed in Section V-E, and the signature has unit mass per unit length. Then, the coordinates of the *center of mass* of the signature within the sliding window are

$$\overline{x}(l) = \int_{-L}^{+L} g(\lambda)x(l+\lambda)\,d\lambda \tag{1}$$

$$\overline{y}(l) = \int_{-L}^{+L} g(\lambda)y(l+\lambda)\,d\lambda \tag{2}$$

where $(x(l), y(l))$ are the point coordinates along the length of the signature. Fig. 8 illustrates the center of mass of a curve segment. The varying coordinates of the center of mass, $\overline{x}(l)$ and $\overline{y}(l)$, computed over a window that is sliding along the length of a signature, together provide us with a robust position-dependent description of the shape of the signature.

G. Torque

The torque $\boldsymbol{T}$ exerted by a vector $\boldsymbol{v}$, which is located at position $\boldsymbol{p}$ with respect to the point about which we measure the torque, is $\boldsymbol{T} = \boldsymbol{v} \times \boldsymbol{p}$. Before we apply this notion of torque to an on-line signature, note that the torque exerted by a vector about a point depends on both the position and the orientation of the vector; further, we can interpret the magnitude of the torque about a point to be twice the area swept by the vector with respect to that point.

Assume the following: the signature is parameterized over its normalized arc-length l (see Section V-D), the signature has a weighted window $g(\lambda)$ of span $\pm L$ sliding over its length (see Section V-E), and the signature is decomposed into a series of infinitesimal vectors, each vector with magnitude equal to its length and with direction pointing in the direction of pen motion. Then we can define the *torque* exerted about the origin by the signature within the sliding window to be

$$\boldsymbol{T}(l) = \int_{-L}^{+L} g(\lambda)[dx(l+\lambda) \quad dy(l+\lambda)] \times [x(l+\lambda) \quad y(l+\lambda)]$$

where $(x(l), y(l))$ are the point coordinates along the length of the signature. Here $dx(l+\lambda)$ and $dy(l+\lambda)$ are the differential changes in x and y at the location $(l+\lambda)$ under a $d\lambda$ change in λ. Given that each of our on-line signatures resides in the x-y plane, $\boldsymbol{T}$ here can point only

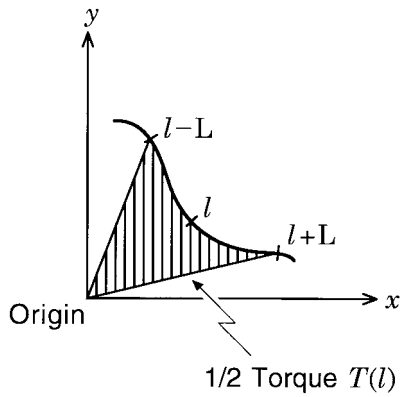

Fig. 9. Torque.

in a direction orthogonal to the x-y signature plane: $\boldsymbol{T}$ will point orthogonally out of the x-y plane if the net torque is counterclockwise, and $\boldsymbol{T}$ will point orthogonally into the x-y plane if the net torque is clockwise. As a result, it suffices for us to consider only the following scalar T, which we obtain by expanding the above vector $\boldsymbol{T}$

$$T(l) = \int_{-L}^{+L} g(\lambda)(y(l+\lambda)\,dx(l+\lambda) - x(l+\lambda)\,dy(l+\lambda)). \tag{3}$$

If we ignore the window function $g(\lambda)$, we can interpret the torque $T(l)$ here to be twice the signed area swept with respect to the origin by the portion of the signature within the sliding window centered at position l, a positive value of $T(l)$ indicating a net counterclockwise sweep and a negative value indicating a net clockwise sweep. Fig. 9 illustrates this physical interpretation of the torque exerted by a planar on-line curve segment about the origin. The varying torque, $T(l)$, computed over a window that is sliding along the length of a signature, can provide us with a robust position- and orientation-dependent description of the shape of the signature, as we shall see next.

As we noted in the introduction, the straightforward computation of the orientation of a curve based on the curve's derivatives is not robust. It is this lack of robustness that prompted us to devise a new descriptor of shape, the torque, which depends on the orientation of the curve—and also on its position—but does not involve estimating the curve's derivatives explicitly. Is the torque, as we have defined it, a robust descriptor of shape? The answer is *yes*, provided that the point about which we compute the torque is not in the immediate vicinity of the curve segment whose torque we compute. This assertion is established easily by invoking the physical interpretation we provided the torque earlier—that of twice the signed area swept by a curve segment with respect to the point about which we compute the torque. Given this interpretation, it is clear from Fig. 10 that our torque descriptor of shape is robust in the following sense: a slight alteration to the shape of a curve segment would not drastically alter the signed area swept by this curve segment with respect to a point as long as we ensured that the point was not in the immediate vicinity of the

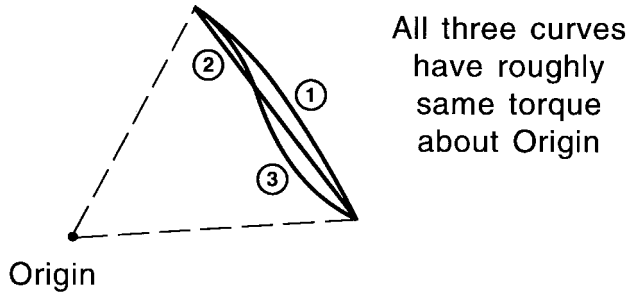

Fig. 10. Stability of torque.

curve segment. It is, of course, possible to devise other robust descriptors of shape that depend on the orientation of a curve—we shall devise two such descriptors in the next subsection—but I must mention that I have found the torque to be particularly useful.

H. Moments of Inertia

Assume the following: The signature is parameterized over its normalized arc-length l, as we discussed in Section V-D, the signature has a weighted window $g(\lambda)$ of span $\pm L$ sliding over its length, as we discussed in Section V-E, and the signature has unit mass per unit length. Then the *moments of inertia* about the y-axis and the x-axis, respectively, of the signature within the sliding window are

$$\overline{x^2}(l) = \int_{-L}^{+L} g(\lambda)x^2(l+\lambda)\,d\lambda$$
$$\overline{y^2}(l) = \int_{-L}^{+L} g(\lambda)y^2(l+\lambda)\,d\lambda$$

where $(x(l), y(l))$ are the point coordinates along the length of the signature. For future reference, let us also define the second-order cross moment here

$$\overline{xy}(l) = \int_{-L}^{+L} g(\lambda)x(l+\lambda)y(l+\lambda)\,d\lambda.$$

The varying second-order moments, $\overline{x^2}(l)$, $\overline{y^2}(l)$, and $\overline{xy}(l)$, computed over a window that is sliding along the length of a signature, when expressed in forms $s_1(l)$ and $s_2(l)$ that we shall derive, together provide us with a robust orientation- and curvature-dependent description of the shape of the signature.

Before we proceed to derive robust combinations of $\overline{x^2}(l)$, $\overline{y^2}(l)$, and $\overline{xy}(l)$, let us examine, in order, the dependence of the moment of inertia of a curve segment about a line on the position and the orientation of the line. Without any loss of generality, consider first the moment of inertia of a curve segment about the line $x = \alpha$, which is parallel to the y-axis. This moment of inertia is readily evaluated to be the following by substituting $[x(l) - \alpha]$ for $x(l)$ in our earlier expression for $\overline{x^2}(l)$

$$\begin{aligned}\int_{-L}^{+L} & g(\lambda)[x(l+\lambda)-\alpha]^2\,d\lambda \\ &= \int_{-L}^{+L} g(\lambda)[x^2(l+\lambda) - 2\alpha x(l+\lambda) + \alpha^2]\,d\lambda \\ &= \overline{x^2}(l) - 2\alpha\overline{x}(l) + \alpha^2\end{aligned}$$

where $\overline{x^2}$ is as defined above, and $\overline{x}$ is the x-coordinate of the center of mass of the curve segment as defined in Section V-F. Thus we see that, given the moment of inertia of a curve segment about a line, and, in addition, the position of the center of mass of the curve segment, we can compute the moment of inertia of the curve segment about any arbitrary line parallel to the original line. Further, if the line about which we compute the moment of inertia of a curve segment is distant from the center of mass of the curve segment—in the expression above, if α is large—the value of the moment of inertia will be dominated by the distance between the line and the center of mass of the curve segment. As such a domination would deemphasize the shape of the curve segment in our measurement—emphasizing instead the position of the curve segment available through $(\overline{x}(l), \overline{y}(l))$—*let us measure the moments of inertia of each curve segment only about lines through the center of mass of the curve segment*. Further, for convenience of analysis, let us position the origin of our coordinate frame for all such measurements at the center of mass of the curve segment.

Let us now turn our attention to the dependence of the moment of inertia of a curve segment about a line on the orientation of the line. Without any loss of generality, consider the moment of inertia of a curve segment about the line through the origin that is at a counterclockwise angle θ with respect to the y-axis. This moment of inertia is readily evaluated to be the following by substituting $[x(l)\cos\theta + y(l)\sin\theta]$ for $x(l)$ in our earlier expression for $\overline{x^2}(l)$:

$$\begin{aligned}I(l,\theta) &= \int_{-L}^{+L} g(\lambda)[x(l+\lambda)\cos\theta + y(l+\lambda)\sin\theta]^2\,d\lambda \\ &= \int_{-L}^{+L} g(\lambda)[x^2(l+\lambda)\cos^2\theta \\ &\quad + 2x(l+\lambda)y(l+\lambda)\cos\theta\sin\theta \\ &\quad + y^2(l+\lambda)\sin^2\theta]\,d\lambda \\ &= \overline{x^2}(l)\cos^2\theta + 2\overline{xy}(l)\cos\theta\sin\theta + \overline{y^2}(l)\sin^2\theta\end{aligned}$$

where $\overline{x^2}(l)$, $\overline{y^2}(l)$, and $\overline{xy}(l)$ are as previously defined. Thus we see that, given $\overline{x^2}$, $\overline{y^2}$, and $\overline{xy}$ for a curve segment in any coordinate frame, we can compute the moment of inertia of the curve segment about any arbitrarily oriented line through the origin of the coordinate frame. This result is independent of the position of the coordinate frame's origin, which we have chosen to be at the center of mass of the curve segment.

Continuing our search for robust combinations of $\overline{x^2}(l)$, $\overline{y^2}(l)$, and $\overline{xy}(l)$, let us visualize the variation over θ, under

constant l, of the moment of inertia $I(l,\theta)$ above. Toward this end, let us first put $I(l,\theta) = 1/r^2(l,\theta)$ in the expression above, and then examine the polar plot of r as a function of θ. Making this substitution, we get

$$1/r^2(\theta) = \overline{x^2}\cos^2\theta + 2\overline{xy}\cos\theta\sin\theta + \overline{y^2}\sin^2\theta$$

where we have not shown the dependence of $\overline{x^2}$, $\overline{y^2}$, $\overline{xy}$, and $r(\theta)$ on l explicitly, which we have fixed here. Converting the polar coordinates (r,θ) in this equation to the Cartesian coordinates (r_h, r_v) via the substitutions $r_h(\theta) = r(\theta)\cos\theta$ and $r_v(\theta) = r(\theta)\sin\theta$, we get the following quadratic equation in r_h and r_v:

$$\overline{x^2}r_h^2(\theta) + 2\overline{xy}r_h(\theta)r_v(\theta) + \overline{y^2}r_v^2(\theta) = 1.$$

Thus we see that the polar plot of r, as a function of θ, is a conic section. Let us now establish that this conic section is an ellipse centered at the origin. Toward this goal, consider the polar plot of $I(l,\theta)$ against θ, for a fixed l. Then, as $I(l,\theta)$ is the moment of inertia of a fixed curve segment, we can deduce that $I(l,\theta)$ is symmetric about the origin, and further, that $I(l,\theta)$ can be zero at most about a single straight line through the origin—a line to which the curve segment must be confined. Hence, $r(\theta)$, which is the inverse square root of $I(l,\theta)$ for a fixed l, must be symmetric about the origin and can be infinite only along a single straight line through the origin. It follows that the polar plot of $r(\theta)$ is an ellipse centered at the origin, where we are including as an ellipse the degenerate case of a pair of infinite parallel straight lines positioned symmetrically about the origin, such a degeneracy occurring whenever a curve segment lies along a single straight line through the origin.

Thus we have reduced the problem of describing all possible moments of inertia of a curve segment about lines through the center of mass of the curve segment to the problem of describing an ellipse that is centered at the origin of a coordinate frame. Let us call this ellipse—which is a polar plot of the inverse square root of the moment of inertia of a curve segment about a straight line through the center of mass of the curve segment, as a function of the orientation of the straight line—the *curvature ellipse* of the curve segment.[4] Our motivation for this name will soon become apparent. Fig. 11 illustrates the curvature ellipse of a curve segment.

The curvature ellipse of a curve segment is specified completely by the ellipse's major axis, its minor axis, and its orientation. The *major* axis of the curvature ellipse has twice the magnitude of the inverse square root of the *minimum* moment of inertia of the curve segment about lines through the center of mass of the curve segment, and the *minor* axis of the curvature ellipse has twice the magnitude of the inverse square root of the *maximum* moment of inertia of the curve segment about lines through the center of mass of the curve segment. If the major and minor axes of the curvature ellipse of a curve segment are unequal, then the orientation of the major axis of the curvature ellipse is the orientation through the center of mass of the curve segment about which the moment of inertia of the curve segment is minimum; for a straight-line segment, this orientation is along the straight-line segment. *Let us now superimpose our r_h-r_v and x-y coordinate frames—the former, the frame in which we describe the curvature ellipse of a curve segment, and the latter, the frame in which we measure the moments of inertia of that curve segment.* Further, let us denote by $\phi(l)$ the counterclockwise angle with respect to the y-axis of the major axis of the curvature ellipse of the curve segment spanned by the sliding window centered at position l. Then, as ϕ is simply the θ that minimizes our earlier expression for $I(l,\theta)$, we can determine ϕ by solving $\partial I(l,\theta)/\partial\theta = 0$ for θ. In doing so, on simplification, we get

$$2\phi(l) = \tan^{-1}\left[\frac{2\overline{xy}(l)}{\overline{x^2}(l) - \overline{y^2}(l)}\right].$$

Note that this expression for ϕ provides us with the counterclockwise angles, with respect to the y-axis, of both the major and minor axes of the curvature ellipse; these axes are, of course, 90° apart. Substitution for ϕ into the expression for $I(l,\theta)$ quickly determines which is which, also providing the magnitudes of the semi-major and semi-minor axes of the curvature ellipse.

Our discussion of the curvature ellipse to this point would suggest that, to compare the shapes of two curve segments, we should measure and compare the axes and orientations of the curvature ellipses of the two curve segments. However, it turns out that such direct measurements of curvature ellipses are not robust, this lack of robustness defeating our purpose in measuring the moments of inertia—namely, to

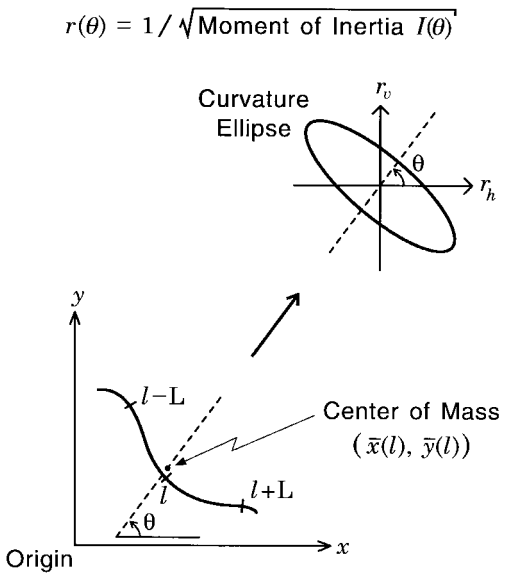

Fig. 11. Curvature ellipse.

[4]Those of you who are familiar with differential geometry [8] will notice that the curvature ellipse of a curve segment is analogous to Dupin's indicatrix of a surface at an elliptic surface point. Dupin's indicatrix of a surface at any point is a polar plot of the inverse square root of the absolute normal curvature of the surface at that point, as a function of the direction (tangent to the surface at the point) in which we compute the curvature.

devise robust descriptors of shape. Hence, we need to adopt a different tack.

Now, it is clear that if we rotate, about its origin, the x-y coordinate frame in which we measure the moments of inertia of a curve segment, the curvature ellipse of the curve segment will simply rotate in the r_h-r_v coordinate frame without changing its shape. Then, from the well-known properties of conics under rotation, we can conclude that the following combinations of the coefficients of the curvature ellipse are invariant under rotation of the x-y axes: $[\overline{x^2}(l) + \overline{y^2}(l)]$ and $[\overline{x^2}(l)\overline{y^2}(l) - \overline{xy}^2(l)]$. (For a description of these properties of a conic, see [17, art. 157]; you can also verify the rotational invariance of the two terms by directly substituting into each of the two terms the expansions for $\overline{x^2}$, $\overline{y^2}$, and $\overline{xy}$ under rotation of the x-y axes.) Let us now combine the two rotationally invariant terms as follows:

$$s(l) = \frac{\overline{x^2}(l)\overline{y^2}(l) - \overline{xy}^2(l)}{[\overline{x^2}(l) + \overline{y^2}(l)]^2}.$$

It is easy to see that $s = 0$ for every straight-line segment, and that $s = 1/4$ for every circle. We can further establish quickly that these two values of s actually bound s. First, it is well known that the combination $[\overline{x^2}(l)\overline{y^2}(l) - \overline{xy}^2(l)]$ of the coefficients of an ellipse is nonnegative; hence, $s \geq 0$. Next, it is clear that

$$1/4 - s(l) = \frac{[\overline{x^2}(l) - \overline{y^2}(l)]^2 + 4\overline{xy}^2(l)}{4[\overline{x^2}(l) + \overline{y^2}(l)]^2} \geq 0.$$

Hence $s \leq 1/4$. Thus we have established that $0 \leq s \leq 1/4$.

Now, we know from our earlier discussion that, barring the degenerate case of a circle, the orientation ϕ of a curvature ellipse satisfies

$$\sin 2\phi(l) = \frac{2\overline{xy}(l)}{\sqrt{[\overline{x^2}(l) - \overline{y^2}(l)]^2 + 4\overline{xy}^2(l)}}.$$

Then let us define the following two measures of a curve segment's shape derived from the curve segment's curvature ellipse:

$$\begin{aligned} s_1(l) &= as(l) + b\sin 2\phi(l) \\ &= a\frac{\overline{x^2}(l)\overline{y^2}(l) - \overline{xy}^2(l)}{[\overline{x^2}(l) + \overline{y^2}(l)]^2} \\ &\quad + b\frac{2\overline{xy}(l)}{\sqrt{[\overline{x^2}(l) - \overline{y^2}(l)]^2 + 4\overline{xy}^2(l)}} \qquad (4) \\ s_2(l) &= as(l) - b\sin 2\phi(l) \\ &= a\frac{\overline{x^2}(l)\overline{y^2}(l) - \overline{xy}^2(l)}{[\overline{x^2}(l) + \overline{y^2}(l)]^2} \\ &\quad - b\frac{2\overline{xy}(l)}{\sqrt{[\overline{x^2}(l) - \overline{y^2}(l)]^2 + 4\overline{xy}^2(l)}} \qquad (5) \end{aligned}$$

where a and b are positive weights. Together, $s_1(l)$ and $s_2(l)$ provide us with measures of the curvature and orientation of a curve segment that are independent of scale—that is, that are invariant under uniform magnification or reduction of the curve segment. Let us, for brevity, call s_1 and s_2 *curvature-ellipse measures*. We measure $as(l) \pm b\sin 2\phi(l)$, instead of $s(l)$ and $\sin 2\phi(l)$ individually, because $s(l)$ and $\sin 2\phi(l)$ are too ambiguous and unreliable individually. For instance, $s(l)$ is zero for every straight-line segment, and $\sin 2\phi(l)$ tends to become unreliable as we flex a straight-line segment toward a complete circle. In the measurements $as(l) \pm b\sin 2\phi(l)$, each component mitigates the ambiguity/unreliability of the other. My choices of $s_1(l)$ and $s_2(l)$ are prompted in large part by my finding them to be more reliable and useful in practice than other pairs of shape measures based on the curvature ellipse I investigated.

I. Moving Coordinate Frame and Saturation

We now have a complete list of the signature characteristics we shall represent as functions of the normalized length l of the signature. These characteristics, each of which we shall compute over a window under variation of this window's position along the length of the signature, are $\overline{x}$, $\overline{y}$, T, s_1, and s_2, as defined in (1)–(5). Of these five characteristics, the center-of-mass coordinates $\overline{x}$ and $\overline{y}$ and the torque T exerted about the origin depend on the location of the origin of the coordinate frame in which we compute the signature characteristics, whereas the curvature-ellipse measures s_1 and s_2 do not depend on this location. On the other hand, only the torque T of the five signature characteristics does not depend on the orientation of the axes of the coordinate frame in which we compute the characteristics. Given this dependence of our signature characteristics on the location and orientation of our coordinate frame, we are faced with deciding how to choose our coordinate frame as our computation window slides along the length of the signature.

With regard to the choice of a coordinate frame for our computations of signature characteristics, one possibility is to use one or a few fixed globally computed coordinate frames for all our computations; we could, for instance, use a single globally computed coordinate frame that has its origin at the overall center of mass of the signature and its axes aligned with the global axes of maximum and minimum inertia of the signature—these axes of inertia being unique, barring degenerate cases. Another possibility with regard to the choice of a coordinate frame for our computations of signature characteristics is to use a coordinate frame whose position—and perhaps orientation—is computed locally along the signature and evolves as the computation window slides along the length of the signature. The latter alternative is more attractive than the former as the global computation of a coordinate frame has the severe drawback that it globalizes the effect of isolated discrepancies between signatures on their comparison. One example of isolated discrepancies between two signatures is a disparity in the sizes of corresponding gaps in the two signatures. It is important that we strive to localize the effects of isolated discrepancies between signatures on their comparison as such discrepancies are often produced inadvertently by

genuine signers. Such localization could serve to distinguish isolated discrepancies from distributed discrepancies in our comparisons of signatures, a distinction that is important because systematic distributed discrepancies are typically more indicative of a forgery than are isolated discrepancies. As we can localize the effects of isolated discrepancies between signatures on their comparisons only by adopting a moving coordinate frame that is computed locally, let us adopt such a frame of the type we shall discuss next. As an aside, a moving coordinate frame that is computed locally opens up the possibility of applying state-based approaches to recognition and verification, one such approach based on hidden Markov models being popular in speech recognition [16].

Let us attach our coordinate frame, which we use for our computations of signature characteristics over the sliding computation window, to the center of mass of the signature computed over another window that too is sliding along the length of the signature. However, for at least two reasons, let us align the axes of this moving coordinate frame permanently with the global axes of maximum and minimum inertia, rather than compute these axes locally. The first reason for aligning our coordinate-frame axes permanently with globally computed axes, rather than computing these axes locally, is one of robustness: Compared to the computation of the coordinate frame's origin locally, the computation of the coordinate frame's axes locally is not robust. The second reason for aligning our coordinate-frame axes permanently with globally computed axes is that, whereas, once we have begun writing, where we write next tends to depend on where we wrote last, the baseline and slant of our writing tends to be determined relatively globally.

Now, we have two windows that are sliding over the length of the signature—one window over which we compute the origin of our moving coordinate frame, and the other window over which we measure the center of mass, the torque, and the moments of inertia of the signature. Let us call the window over which we compute the origin of our moving coordinate frame the *coordinate-frame window*; we earlier named the window over which we compute the signature characteristics the computation window. As illustrated in Fig. 12, both windows slide in unison over the length of the signature with a fixed—but, in general, nonzero—displacement between their centers. Further, the two windows have fixed—but, in general, unequal—widths. Now, just as we weighted the computation window in Section V-E by a Gaussian centered over the window, let us weight the coordinate-frame window by its own Gaussian centered over the window.

As already indicated in Section V-E, a reasonable choice for the width of the computation window is a fraction of the length of an individual "character." If this width is too large, our measurements become nonlocal, and if it is too small, our measurements become nonrobust. The same reasoning applies to the width of the coordinate-frame window, except here we can get away by making the width a factor larger than the width of the computation window; a

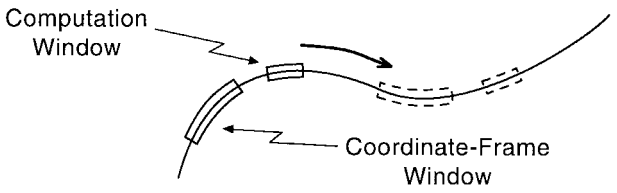

Fig. 12. Sliding computation and coordinate-frame windows.

broader coordinate-frame window provides greater stability to the coordinate frame, but does not sacrifice the localness of our measurements to the same extent as would a similar increase in the width of the sliding computation window. As far as the displacement between the two windows goes, the greater this displacement, the wider shall be the impact of a local discrepancy between any two signatures, eventually making our measurements nonrobust by introducing significant variations in the relative positions of the computation window and its coordinate frame even across genuine signatures. On the other hand, the smaller this displacement, the less robust shall be our measurement of torque, as we discussed in Section V-G. Again, a reasonable choice for the displacement between the two windows is a fraction of the length of an individual "character."

Now, assuming that a typical signature has ten "characters," a fraction of the length of a "character" is a few hundredths of the complete length of the signature. Under this assumption, we could fix the widths of our windows, and of the displacement between them, to be each a few hundredths of the complete length of the signature. Of course, a better strategy might be to analyze the overall complexity of the shape of each signature that we model—by determining how many "wiggles" it has—and then base our widths and displacement on this complexity. Alternately, we could try out a variety of widths and displacements when we model a signature, to discover, and subsequently use, the smallest widths and largest displacement that provide robust measurements.

In our discussion to this point, we have assumed implicitly that both our sliding windows, the computation window and the coordinate-frame window, span continuous curve segments. However, given that in our parameterization of the signature over the normalized distance traveled along the signature with pen down, we did not provide any special treatment to instances of pen up, our sliding windows will, on occasion, span gaps in curves, these gaps tending to be quite variable in size across multiple instances of the signature, and, often, inadvertent. Gaps along the length of a signature, in themselves, do not pose any conceptual hurdle to the motion of our sliding windows: As illustrated in Fig. 13, all we have to do whenever either of our sliding windows spans a gap is split the window, with its (Gaussian) weighting function, across the gap in the curve segment. Although such window splitting suffices to characterize a signature continuously across gaps in the signature, it is not sufficient for our purposes: Our measurements of signature characteristics in the vicinity of gaps in a signature could exhibit unusually large magnitudes

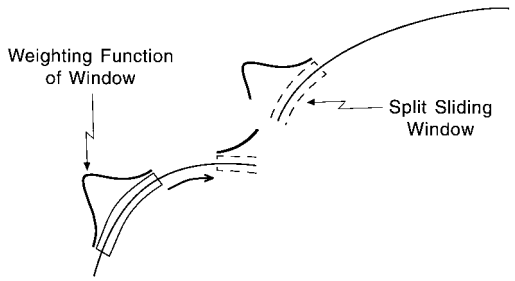

Fig. 13. Sliding window across gap in curve.

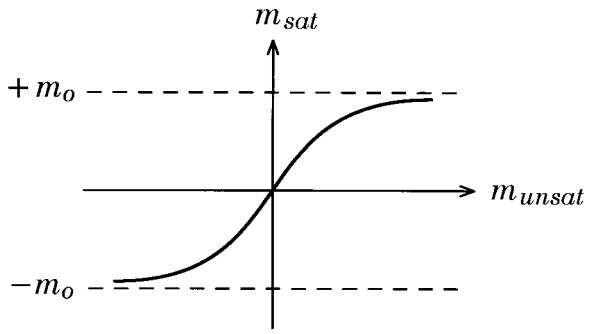

Fig. 14. Saturation function.

when the gaps are large, these large magnitudes posing the threat of disproportionately influencing comparisons of characteristic functions to their models.

In particular, whenever our coordinate-frame window is largely on one side of a large gap, and our computation window is largely on the other side of the gap, the center-of-mass coordinates and the torque exerted about the origin could all exhibit unusually large magnitudes. Recall, however, that our curvature-ellipse measures are independent of the position of the coordinate frame. Further, even when individual moments of inertia assume unusually large magnitudes owing to the computation window spanning a large gap, as we established in Section V-H, our curvature-ellipse measures remain well bounded. Note, however, that when our computation window does span a large gap, our curvature-ellipse measures will tend to have values similar to those for a straight-line segment spanning the large gap.

As already indicated, unusually large magnitudes of the center-of-mass coordinates and the torque exerted about the origin pose the threat of disproportionately influencing comparisons of the center-of-mass and torque characteristic functions to their models. We can circumvent this threat caused by large spatial gaps in the pen-down trajectories of signatures by *saturating* our measurements of the two center-of-mass coordinates and the torque exerted about the origin, employing, in each case, the following *saturation function*:

$$m_{\text{sat}} = m_o \tanh\left[\frac{m_{\text{unsat}}}{m_o}\right]$$

where m_{unsat} is the original unsaturated measurement, m_{sat} is the same measurement after saturation, and m_o—which is positive and chosen individually for each signature characteristic—determines the degree of saturation. This saturation function is illustrated in Fig. 14. When $|m_{\text{unsat}}| \ll |m_o|, |m_{\text{sat}}| \approx |m_{\text{unsat}}|$, but when $|m_{\text{unsat}}| \gg |m_o|, |m_{\text{sat}}| \approx |m_o|$, the signs of m_{sat} and m_{unsat} always being the same.

J. Weighted Cross Correlation and Warping

No signer is uniformly consistent along the entire length of the signer's signature. Further, the consistency of a signer at a particular location along a signature depends on the signature characteristic we examine. As a result, whenever we measure a characteristic of a signature along its length, we must also measure, as a function of the normalized length of the signature, the consistency of the characteristic across multiple instances of the signature. Doing so, for every characteristic function of a signature, we shall have a *consistency function* that provides a measure of the consistency of the characteristic function along its length. A natural choice for the consistency function of a characteristic function is the inverse standard deviation of the characteristic function at each point along its length. Let us adopt this choice. Once we have a consistency function to accompany the prototype of a characteristic function, whenever we compare a characteristic function to this prototype, we shall weight each of the two functions along its length by the consistency function of the prototype.

The question now is, how do we compare a characteristic function to its prototype? or equivalently, what is our measure of error in comparing a characteristic function to its prototype? Of the several error measures that are possible—for instance, the integral of the difference of squares—I have chosen (1 – *cross correlation*), where we compute the *cross correlation* between the characteristic function and its prototype while weighting each function by the consistency function of the prototype. The *weighted cross correlation* of two functions $f(l)$ and $h(l)$, each function weighted by the function $w(l)$, is by definition

$$\text{Cross Correlation} = \frac{\int w^2(l) f(l) h(l)\, dl}{\sqrt{\int w^2(l) f^2(l)\, dl \int w^2(l) h^2(l)\, dl}}.$$

If $f(l)$ is a characteristic function here and $h(l)$ is this characteristic function's prototype, and $w(l)$ is the prototype's consistency function, then for us, $f(l)$, $h(l)$, and $w(l)$ will be related as follows: $h(l) = E[f(l)]$ and $w(l) = 1/(E[(f(l) - E[f(l)])^2])^{1/2}$.

When we compute the various individual weighted cross correlations between a signature's characteristic functions and their models, we will allow all the characteristic functions of the signature—or, equivalently, all the models of these functions—to *warp* simultaneously along their lengths such that an overall error measure is minimized. This simultaneous warping of the individual functions must, of course, be constrained to be identical at identical abscissae along the lengths of all the functions because the abscissa of each function is the same length parameter l, whose each specific value corresponds to a specific physical location

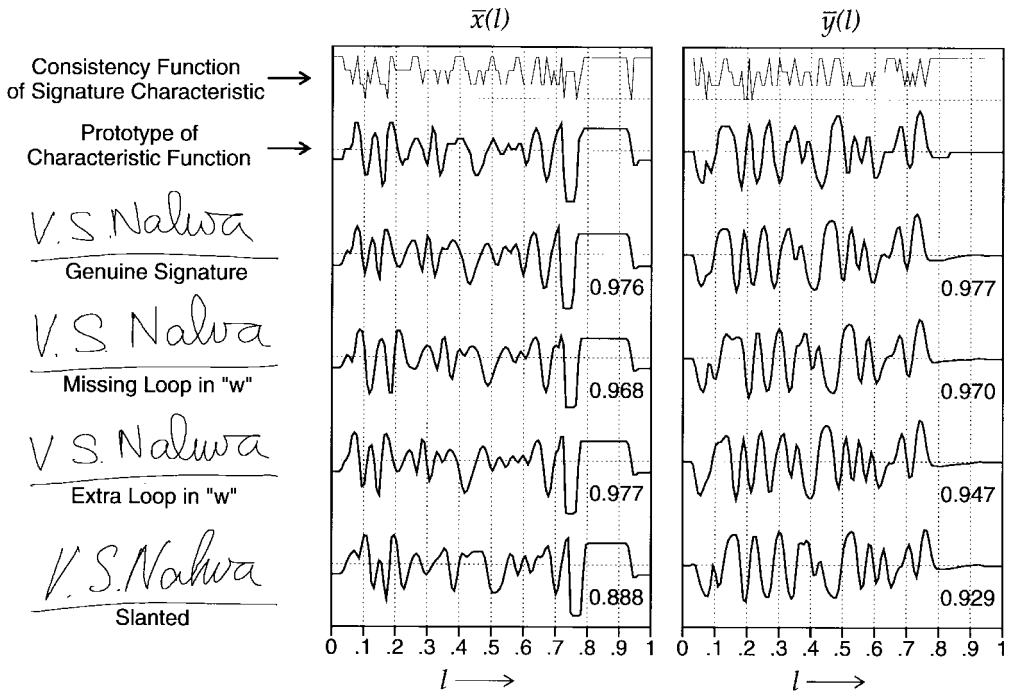

Fig. 15. Example illustrating characteristic functions of signatures.

along the signature. Warping allows us to accommodate instances of signatures that deviate from one another with respect to the fractional lengths of their various parts, such deviations being unavoidable even when all the signatures are produced by the original signer.

Time warping is common in speech recognition [16], and I have also seen it previously in signature verification [18]. However, I have not encountered length warping in the signature-verification literature. Length warping, of course, assumes parameterization over the signature's length, which I have not seen either in the signature-verification literature. Neither have I previously encountered the notion of the weighted cross correlation as we have discussed here.

VI. Algorithm

As I indicated in the introduction, not everyone produces consistently shaped signatures. A signature is particularly likely to be shaped inconsistently when it is produced ballistically, rather than deliberately (see Section IV). Whereas, given sufficient motivation—perhaps, simply the convenience of not having to sign more than once—many signers might produce consistent shapes, some signers might even then be unable to do so. It is for this reason that we cannot rely on the local shapes of signatures alone for signature verification, even though such a reliance is preferable when possible. Hence, we must invoke two models for each signature: one local and purely shape based, and the other global and based on both shape and time.

In particular, my algorithm has three distinct components—normalization, description, and comparison—each of which I outline broadly next. *Normalization* makes the algorithm largely independent of the orientation and aspect of a signature; the algorithm is inherently independent of the position and size of a signature. *Description* generates the five characteristic functions of the signature. *Comparison* computes a net measure of error between the signature characteristics and their prototypes.

A. Normalization

1) Fit a polygon to the ordered set of samples of the online signature, and keep a count of the total number of pen-down samples, a number proportional to the total pen-down time under uniform temporal sampling.
2) Compute the jitter (Section V-B). There is no further need for the original samples.
3) Compute the global axes of maximum and minimum inertia of the signature through the signature's global center of mass, and then rotate the signature to normalize the orientation of these axes.
4) Compute the aspect (Section V-C) of the signature from the fitted polygon, and then scale the signature either vertically or horizontally to normalize its aspect.

B. Description

1) Parametrize the signature over its length l (Section V-D), measured along the fitted and normalized polygon as a fraction of the total length.
2) Compute a moving coordinate frame (Section V-I).
3) In the moving coordinate frame, measure, as a function of l, the following signature characteristics over a sliding computation window (Section V-E): the coordinates $\bar{x}(l)$ and $\bar{y}(l)$ of the center of mass (Section V-F), the torque $T(l)$ exerted about the origin (Section V-G), and the curvature-ellipse measures $s_1(l)$ and $s_2(l)$ (Section V-H). All these computations can be conveniently performed over a discrete l that is uniformly sampled.

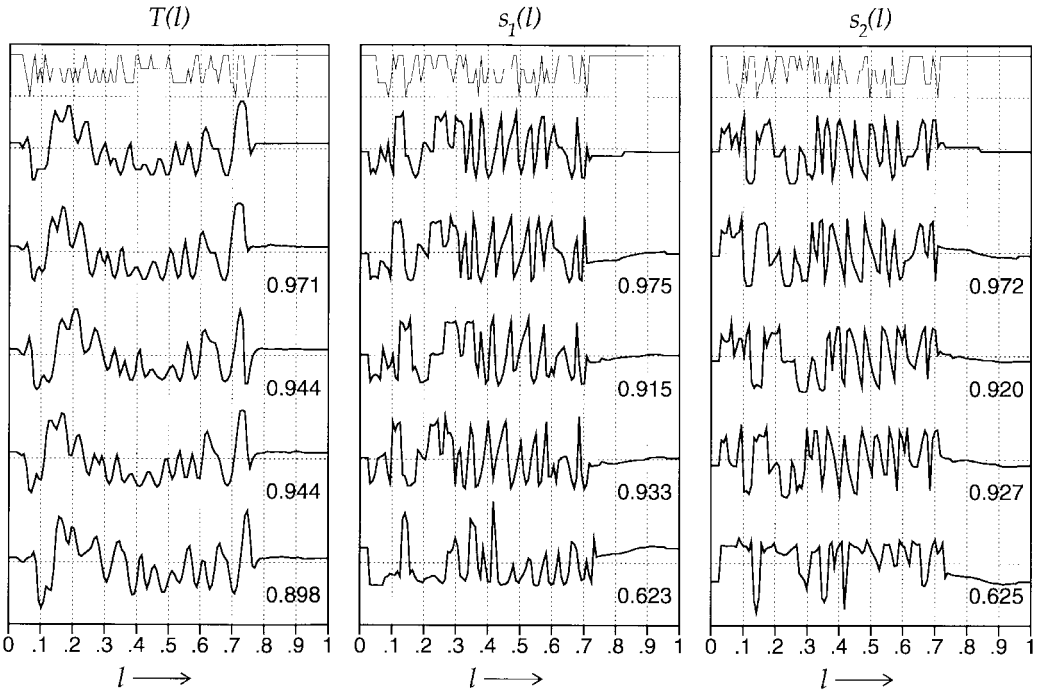

Fig. 15. *(Continued.)* Example illustrating characteristic functions of signatures.

4) Saturate the characteristic functions (Section V-I) and normalize each function to have a zero mean.

C. Comparison

1) Simultaneously warp the five characteristic functions to maximize the sum of the weighted cross correlation of each function with respect to its model (Section V-J), and retain a measure of the total warping performed.
2) Compute the error between each characteristic function and its model by subtracting from 1.0 the weighted cross correlation between the two functions; then normalize each such error by first subtracting its mean from it and then dividing the resultant by the error's standard deviation; finally, bias each normalized error and then threshold it so as to make it 0.0 if it is negative.
3) Compute the root mean square (rms) (Section V-A) of the individual biased and thresholded errors between each of the five characteristic functions and their models (C-2) to arrive at the *net local error*.
4) Compute the rms (Section V-A) of the differences between the following four global entities and their means, after first normalizing each difference by the entity's standard deviation, to arrive at the *net global error*: jitter (A-2), aspect (A-4), warping (C-1), and the total number of pen-down samples (A-1).
5) Compute the weighted and biased harmonic mean (Section V-A) of the net local error (C-3) and the net global error (C-4)—the weights and biases reflecting the overall spatial consistency of the signature across its multiple instances—to arrive at the *net error*, which provides us a measure of the discrepancy between the signature being verified and its model and whose comparison against a threshold determines whether we accept or reject the signature being verified.

VII. Example

I now illustrate the algorithm we discussed in Section VI—specifically, the nature of the characteristic functions that lie at the core of this algorithm—through an example. This pedagogically contrived example, shown in Fig. 15, has four signatures, all shown in the left-most column: Proceeding from top to bottom, the first signature is a typical genuine, the second signature has a loop missing from its "w," the third signature has an extra loop in its "w," and the fourth signature is written with a slant. Shown alongside each signature, in a row, are the characteristic functions of that signature: Proceeding from left to right, shown in sequence are $\overline{x}(l)$, $\overline{y}(l)$, $T(l)$, $s_1(l)$, and $s_2(l)$, with the result of the weighted cross correlation of each characteristic function with its prototype indicated at the lower right of the function. The prototype of each characteristic function is shown immediately above the four corresponding characteristic functions of the four signatures, and immediately above each prototype is shown the consistency function of the signature characteristic, this function bounded below by zero.

As is clear from the figure, local deviations in the shape of a signature from its typical genuine instance lead to locally identifiable deviations in the characteristic functions from their prototypes, and systematic deviations in the shape of a signature from its typical genuine instance lead to distributed deviations in the characteristic functions from their prototypes. In particular, note that the characteristic functions of the two signatures with the extra and missing

loop in "w" each differ from its prototype roughly within the interval of l between 0.5 and 0.6; because of the discrepancy between the shapes of the signatures, the characteristic functions of these two signatures are neither aligned with each other along the l axis, nor are they aligned with their prototypes (until we warp the l axis).

VIII. Database Results

The three databases on which I tested my implementation of the algorithm we discussed in Section VI were compiled by Bell Laboratories and are proprietary. Let us call these databases *DB1*, *DB2*, and *DB3*, calling their union *DB*. Before we examine the various results on these databases, let us spend a few moments on their backgrounds.

A. Database 1 (DB1)

1) This database was created using a Bell Laboratories in-house developmental LCD writing tablet with a tethered pen. The spatial resolution of the tablet was uniformly about 0.08 mm, or about 300 dots/in, along both the horizontal and vertical directions, and the temporal sampling rate of the tablet was about 300 samples/s. Although this tablet provided the pen pressure in addition to the pen position, the pen pressure was used solely to determine whether the pen was in contact with the writing surface.
2) The total number of genuine signatures was 904 from 59 different signers. Fifteen signers were women and eight were left handed, and all of these signers were Bell Laboratories employees, ranging in age from 19 to 66. After first being allowed to get accustomed to using the writing tablet, each signer was asked to provide ten genuine signatures in a first session. At the end of each signature, each signer was allowed to delete and redo the signature if, in the opinion of the signer, the signature was not "OK." After an interval of at least a week, each genuine signer was called back for a second session to provide either five more, or, in four cases, nine or ten more, genuine signatures.
3) The total number of forgeries was 325, with either five or ten forgery attempts for each of the 59 different genuine signatures. These forgeries were performed by 32 willing participants from among those who provided the genuine signatures. Each forger was shown copies of the genuine signatures to be forged and allowed to forge one or more of them either five times, or, in two cases, ten times after first being allowed to practice the signature. Cash rewards were used to motivate the forgers.
4) There were actually 60 genuine signers who participated in the creation of the database, but all the signatures related to one signer were subsequently removed from the database when it was realized that this signer had expanded his first initial, which he had used in his first set of signatures, into its full form in the second set.

B. Database 2 (DB2)

1) This database was created using an NCR 5990 LCD writing tablet with a tethered pen. The spatial resolution of the tablet was uniformly about 0.08 mm, or about 300 dots/in, along both the horizontal and vertical directions, and the temporal sampling rate of the tablet was uniformly 200 samples/s. Although this tablet provided the pen position not only when the pen was touching the writing surface, but also when the pen was in the vicinity of this surface, the pen-up information was not used.
2) The total number of genuine signatures was 982 from 102 different signers. These signers were all from Bell Laboratories and NCR, and all signatures from each signer were collected in a single session outside a cafeteria.
3) The total number of forgeries was 401, with these forgeries collected in the same sessions as the genuine signatures. Each forger was shown a copy of the signature to be forged on a computer screen, and the effort made by a forger to reproduce the displayed signature varied from negligible to substantial, including prior practice.
4) There were originally many more genuine signatures and forgeries in the database, but subsequently only those genuine signatures were retained that had roughly 80–120% of the strokes of their typical specimen, and only those forgeries were retained that were attempts to replicate the shape of the genuine signature and that had roughly 80–120% of the strokes of the genuine signature. This pruning of the database was not a deliberate attempt to rid it of its so-called goats, which we discussed in Section IV, but rather an attempt to ensure that estimates of performance subsequently achieved on the database would be more realistic than they might otherwise be. Further, this pruning was supervised not by researchers developing signature-verification software, but rather by NCR, which manufactured the writing tablets used and was seeking to bundle such software with these tablets.

C. Database 3 (DB3)

1) The same as for DB2.
2) The total number of genuine signatures was 790 from 43 different signers. These signers were all from Wright State University, and all signatures from each signer were collected in a single session.
3) The total number of forgeries was 424, these forgeries collected in the same fashion as for DB2.
4) The same as for DB2.

The details I have provided for each database are what I have been able to glean from their various accounts. I was not a participant in the creation of any of the databases. From accounts of the creation of these databases, it seems safe to conclude that DB1 was created in the most

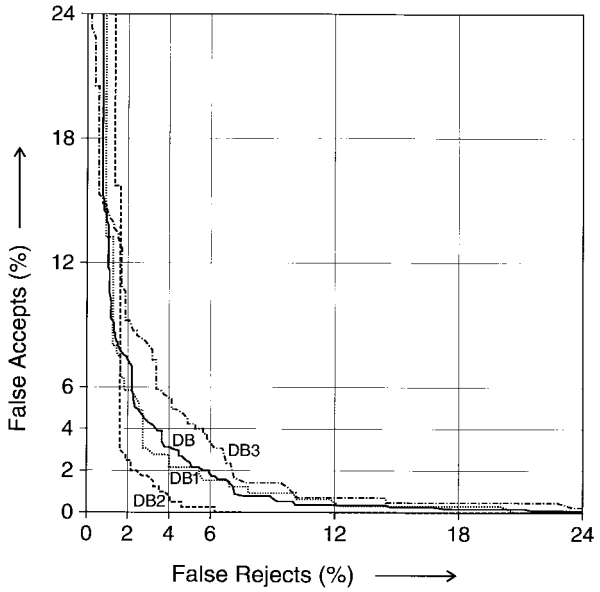

Fig. 16. Error tradeoff curves when modeling with first six genuine signatures.

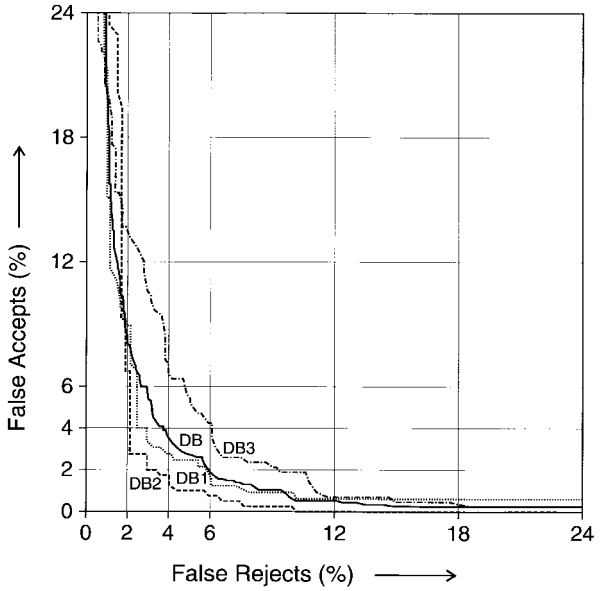

Fig. 17. Error tradeoff curves when modeling with first five genuine signatures.

carefully controlled fashion, and that DB3 was created in the least carefully controlled fashion. Owing to the varying circumstances of their creation, I shall report not only the error tradeoff curve (see Section III) for the three databases collectively, but also the error tradeoff curves for the three databases individually.

In Fig. 16, I show the error tradeoff curves for the three databases individually and collectively, labeling the last curve DB. Each curve was generated using the first six genuine signatures of each signer to build a model of the signer's signature, this model requiring about 600 bytes of storage after some straightforward compression. Under these circumstances, DB1 provides a test set of 550 genuines from 59 signers in addition to 325 forgeries, DB2 provides a test set of 370 genuines from 102 signers in addition to 401 forgeries, and DB3 provides a test set of 532 genuines from 43 signers in addition to 424 forgeries. For the curve labeled DB in Fig. 16, then, the total number of genuines tested is 1452 from 204 signers, and the total number of forgeries tested is 1150.

I used six genuine signatures to build each signature model because, at least on these databases, as I increased the number of signatures for modeling up to six, there were tangible, albeit increasingly smaller, improvements in the various error tradeoff curves. Figs. 17 and 18 illustrate the error tradeoff curves when, instead of the first six signatures, we use the first five and four signatures, respectively, of each signer to model the signer's signature. On these databases, I did not see a tangible improvement in the error tradeoff curves as I increased the number of signatures for modeling up from six; also, the test set of genuines became increasingly smaller, which made the results less reliable.

It is clear from the various error tradeoff curves that each such curve depends highly on the nature of the database on which the curve was produced. For instance, in Fig. 16, the equal-error rates for DB1–DB3 are about

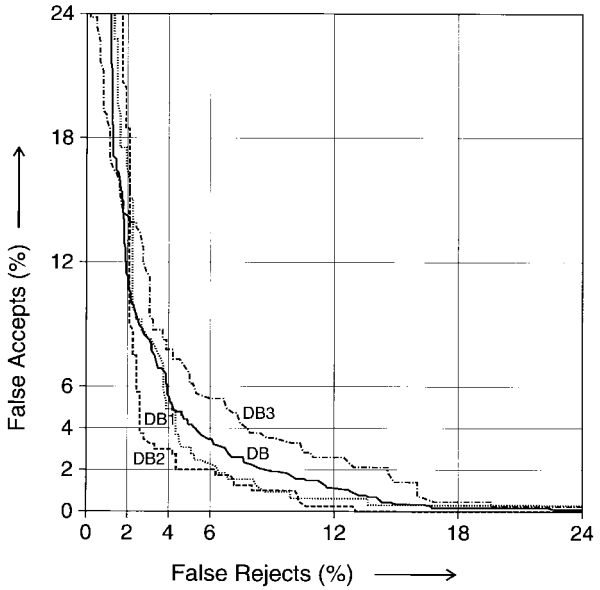

Fig. 18. Error tradeoff curves when modeling with first four genuine signatures.

3%, 2%, and 5%, respectively. The error tradeoff curves for DB2 are clearly the best, perhaps because the forgeries were not well motivated, and the error tradeoff curves for DB3 are clearly the worst, perhaps because not only were the forgeries not well motivated, but also because the acquisition of genuine signatures was largely unsupervised and from college students, who are likely to have relatively unpracticed signatures. Note here also that the arrangement used for the creation of forgeries in DB2 and DB3 is not ideal for successful forgery as any forger who tries to replicate the shape of a signature displayed on a computer screen without first memorizing the shape of this signature, would need to pause on several occasions to look up at the displayed signature for guidance. I am speculating.

The equal-error rate of my implementation's net error tradeoff curve in Fig. 16 is about 3.6%. An operating point that I consider reasonable for many credit card transactions is at about 1% false accepts, which corresponds to about 7% false rejects on the net error tradeoff curve in Fig. 16. At this operating point, statistically, approximately one out of every 100 forgeries will be accepted and approximately one out of every 14 genuine signatures will be rejected—an instance of rejection requiring either a fresh signature, or some other action. *In practice, of course, the point on the error tradeoff curve at which we operate in a particular situation must depend on the relative penalties we would incur for committing the two types of errors in that situation*. For instance, a substantial potential financial loss from accepting a forgery would favor an operating point with a low false-accept rate, whereas a high likelihood of customer annoyance from rejecting a genuine signature provided by the customer, such annoyance posing the risk of loosing the customer, would favor an operating point with a low false-reject rate. Perhaps, then, the signature-verification system should provide a confidence measure—say, in the range ± 100—rather than a *yes* or a *no*, leaving the final decision of whether to accept a signature to the system operator.

As I indicated in Section III, it is not just the percentage of false rejects that is important, but also the visual similarity of the false rejects to genuine signatures. Because of the shape-based nature of my scheme, we would expect its false rejects to be visually dissimilar to genuine signatures. This expectation seems to be largely met in Fig. 19, where I show each of the false accepts (forgeries accepted) and false rejects (genuines rejected) at the equal-error point of the error tradeoff curve for DB1 in Fig. 16. In Fig. 19, for each misclassified signature, I show first (topmost) the original that was determined by the system to be most representative of the six originals used to model the signature. Immediately below this particular original, I show two additional originals if, in my judgment, these additional originals help us understand why a particular subsequent signature was either falsely accepted or falsely rejected. Below each set of originals I show the one or more misclassified signatures pertinent to that set, providing to the lower right of each misclassified signature, the final (scaled) numerical error put out by my implementation. The larger this error, the lower the degree of match of the signature to the six original signatures. The threshold demarcating genuines from forgeries at the equal-error point for DB1 in Fig. 16 is 0.59. At this threshold, the total number of forgeries accepted is ten, and the total number of genuines rejected is 15, all shown in Fig. 19, each of these numbers roughly 3% of the total number of forgeries and genuines available for testing in DB1, respectively.

Note, here, that the equal-error point is unlikely to be our operating point in practice. In particular, I have been using a threshold of 0.50, rather than 0.59, to distinguish between genuines and forgeries in demonstrations. This threshold corresponds to about 7% false rejects and 1% false accepts on the error tradeoff curve for DB in Fig. 16.

Let us now examine in turn the sets of signatures in Fig. 19, whose originals are produced by 12 different individuals.

- Signer 4 has a very simple, but inconsistent, signature, and although the first two forgeries shown accepted each seem to resemble at least one original closely, the low error for the last forgery accepted surprises me. At a threshold of 0.50, however, this last forgery would be rejected.
- Signer 6 has a genuine rejected because of this genuine's apparent visual dissimilarity to its original.
- Signer 7 has a genuine rejected because of this genuine's apparent visual dissimilarity to its original.
- Signer 14 seems to have an inconsistent signature, especially around the "D" and the "B." As a result, it has a forgery accepted, but with an error greater than 0.50.
- Signer 18 has a genuine rejected because of this genuine's apparent visual dissimilarity to its original.
- Signer 19 has a signature that, on close visual examination, is revealed to be relatively consistent only around "Ell," these first three characters then greatly influencing the final error. All the genuines rejected are rejected at least in part because of the insertion of a middle initial in each of them, and from among the forgeries accepted, the second appears to be the most visually similar to the originals and it alone has an error less than 0.50.
- Signer 33 has two genuines rejected, apparently because of the dissimilarity of the first word of these genuines to the first word of their originals.
- Signer 38 has a forgery accepted that looks quite similar to the original, at least at first glance, but this acceptance is with an error greater than 0.50.
- Signer 40 has a genuine rejected because of this genuine's apparent visual dissimilarity to its original. This signer also has a forgery accepted because of this forgery's apparent visual similarity to the original, but this acceptance is with an error greater than 0.50.
- Signer 46 has two genuines rejected, apparently because of the dissimilarity of the shape, size, and position of the "S" and the shape of the "y" in each rejected genuine with respect to its originals. This variability in the "S" and the "y" could, of course, be weighted out if we had originals from this signer exhibiting this variability.
- Signer 48 has a genuine rejected because of this genuine's apparent visual dissimilarity to its original.
- Signer 54 has three genuines rejected because of the obvious discrepancies between the last word of these genuines and their original. Notice how well the errors here correlate with the apparent visual discrepancies.

I indicated in Section IV that the performance of a verification system on a database is typically limited by the database's goats, which are generally few in number and might vary greatly from one database to another. For DB1, at the equal-error point, notice that Signers 4, 19, and

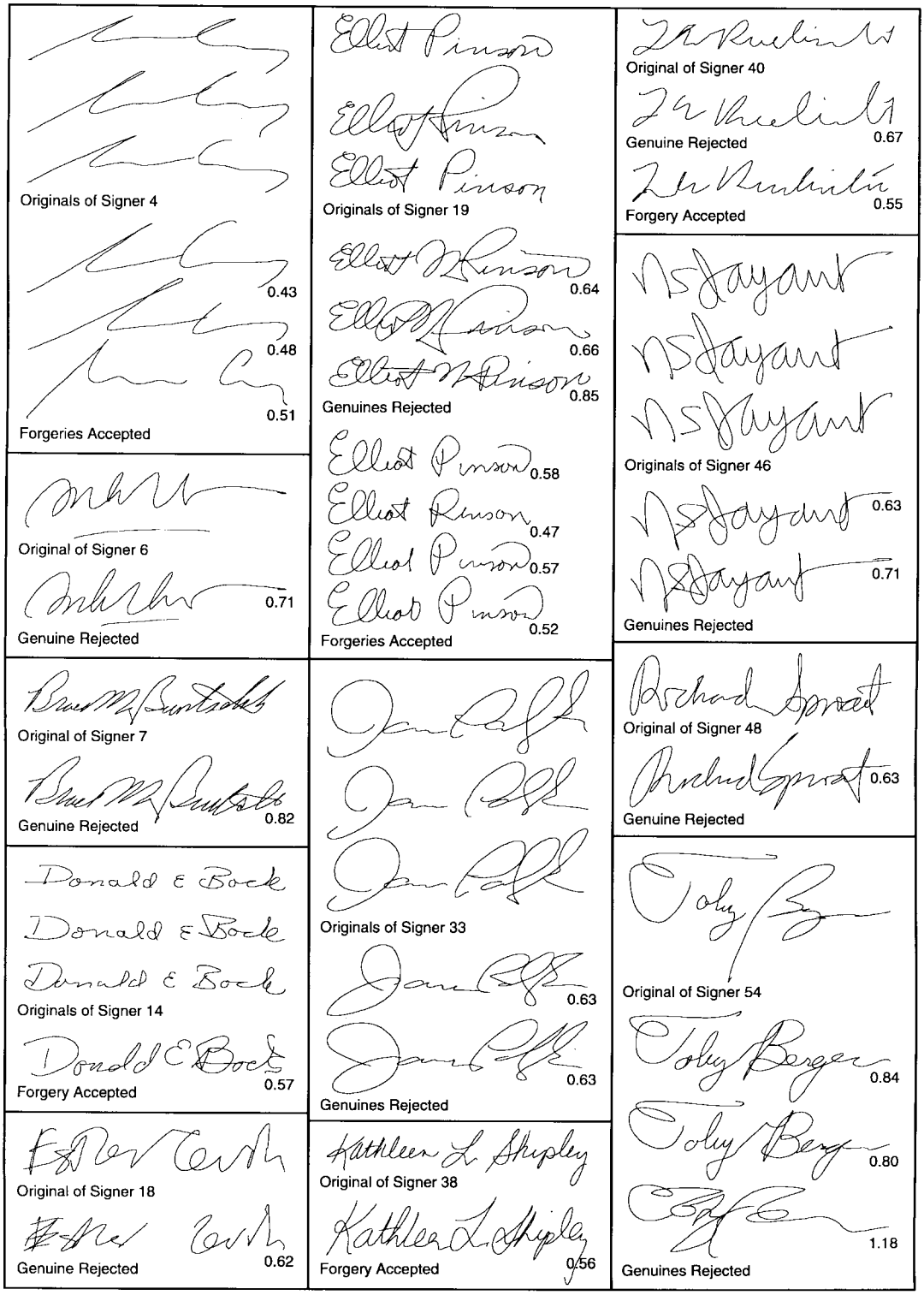

Fig. 19. Errors at the equal-error point for database DB1 when modeling with first six genuine signatures.

54—just three out of the 59 signers in all—had their signatures account for seven of the ten forgeries accepted and for six of the 15 genuines rejected. At the equal-error point for the combined complete database, DB, it turns out that the signatures of ten signers out of 204 signers in all, about 5%, account for 35 of the 41 total forgeries accepted, about 85%, and for 21 of the 53 total genuines rejected, about 40%; in arriving at these numbers, I included every signer whose genuines and forgeries led to three or more errors.

I have shown you only misclassified signatures to this point. In Fig. 20, I show all the (correctly classified)

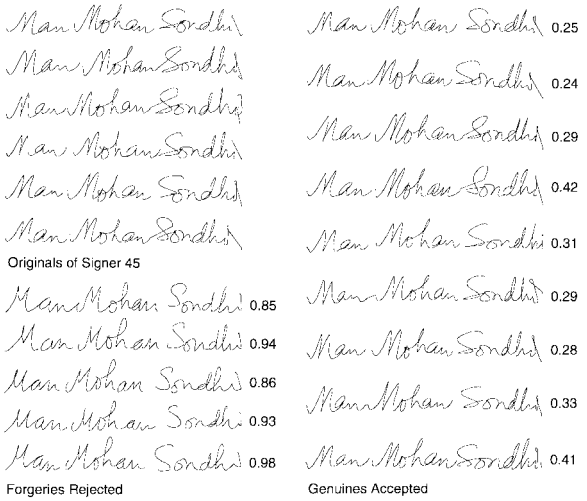

Fig. 20. All signatures pertinent to one signer from database DB1.

signatures pertinent to one particular signer from DB1. In the figure, on the left on top are the first six signatures provided by this signer, these signatures used to create a model of the signer's signature. Once again, the topmost signature from among the six is the one that was determined to be most representative of the six originals. I have shown all six signatures so that you may discover intrasigner variations among them. Below the six originals, I show five attempted forgeries, and, to the right of these originals and forgeries, I show nine genuines, indicating alongside each classified signature, to its right, the (scaled) numerical error put out for that signature by my implementation. In this example, the errors for the genuines and the forgeries are clearly bimodal, this bimodality facilitating straightforward discrimination between the genuines and the forgeries. Notice that one significant distinction between the forgeries and the six originals is the way in which the "a's" are looped. In comparing the various signatures visually, note not only variations between the forms of corresponding characters, but also between their sizes and their orientations relative to the rest of the signature of which they are a part.

A final point I would like to make is this: For any given database, perhaps a composite of multiple individual databases, we can always fine tune a signature-verification system to provide the best overall error tradeoff curve for that database—for the three databases here, I was able to bring my overall equal-error rate down to about 2.5%—but we must always ask ourselves, does this fine tuning make *common sense* in the real world? If the fine tuning does not make common sense, it is in all likelihood exploiting a peculiarity of the database. Then, if we do plan to introduce the system into the marketplace, we are better off without the fine tuning.

IX. A Signature-Verification System

In the marketplace, it is desirable that automatic signature verification be feasible not only over the network, but also on site—with little delay and at low cost. Automatic on-site verification of handwritten signatures is now feasible for two reasons: 1) high-fidelity digital models of signatures, each such model requiring at least of the order of a few hundred bytes of storage,[5] can now be stored on smart cards, and 2) the computation required to verify a signature, given the signature's model, can now easily be accomplished in real time on a relatively inexpensive personal computer. A *smart card*, although often designed to look and feel just like an ordinary credit card, offers secure internal data storage unlike currently used credit cards, whose storage is restricted to the magnetic stripe at the back of the card. As a result, whereas ordinary currently used credit cards typically provide only a few hundred bytes of storage, smart cards provide at least several kilobytes of storage. Data transfer to and from a smart card is accomplished through a smart-card reader.

An alternative to signature verification that is completely on site is to store all the signature models in a central database, and then perhaps transmit each signature to be verified to the site of this database to perform the verification there; such verification could be performed, for instance, in conjunction with a credit-worthiness check. Storing the signature models in a central database has the disadvantage that, to verify signatures, we would have to access this database, which in all likelihood would be located remotely. However, central storage would allow us to update, gradually and transparently to signers, the model of each signer's signature as this signature evolves over time. With regard to the net risk due to possible tampering of stored signature models, it is unclear which storage alternative poses a greater risk: Whereas a central database can probably be made more secure against tampering than can individual smart cards, any compromise in the security of a database is likely to lead to more dire consequences. We could, of course, store signature models both on smart cards and in a central database simultaneously, retrieving, in each instance of signature verification, the model of the signature from the source that better suits the constraints and demands of the particular situation.

Let me now describe a particular system for automatic on-site signature verification that executes in real time the algorithm we discussed in Section VI. This system, shown in the photograph in Fig. 21, has four principal hardware components: a notebook personal computer, an electronic writing tablet, a smart card, and a smart-card reader. The writing tablet and the smart-card reader are

[5] I arrived at this rough estimate for the storage required for a high-fidelity digital model of a typical on-line handwritten signature—of the order of a few hundred bytes—through the following thumbnail argument. Assume that a signature has ten "characters" each of whose properties we must represent at ten locations to construct a high-fidelity model of the signature. Irrespective of the particular properties of each "character" we choose to represent at each location, for our model to have a high fidelity, we must capture at least the average and the variance of the position, the orientation, and the curvature of the signature at each location across multiple instances of the signature. Then, for a high-fidelity digital model of a signature, we inherently need several bytes to represent the properties of the signature at each location, and, therefore, several hundred bytes to represent the properties of the signature at all locations.

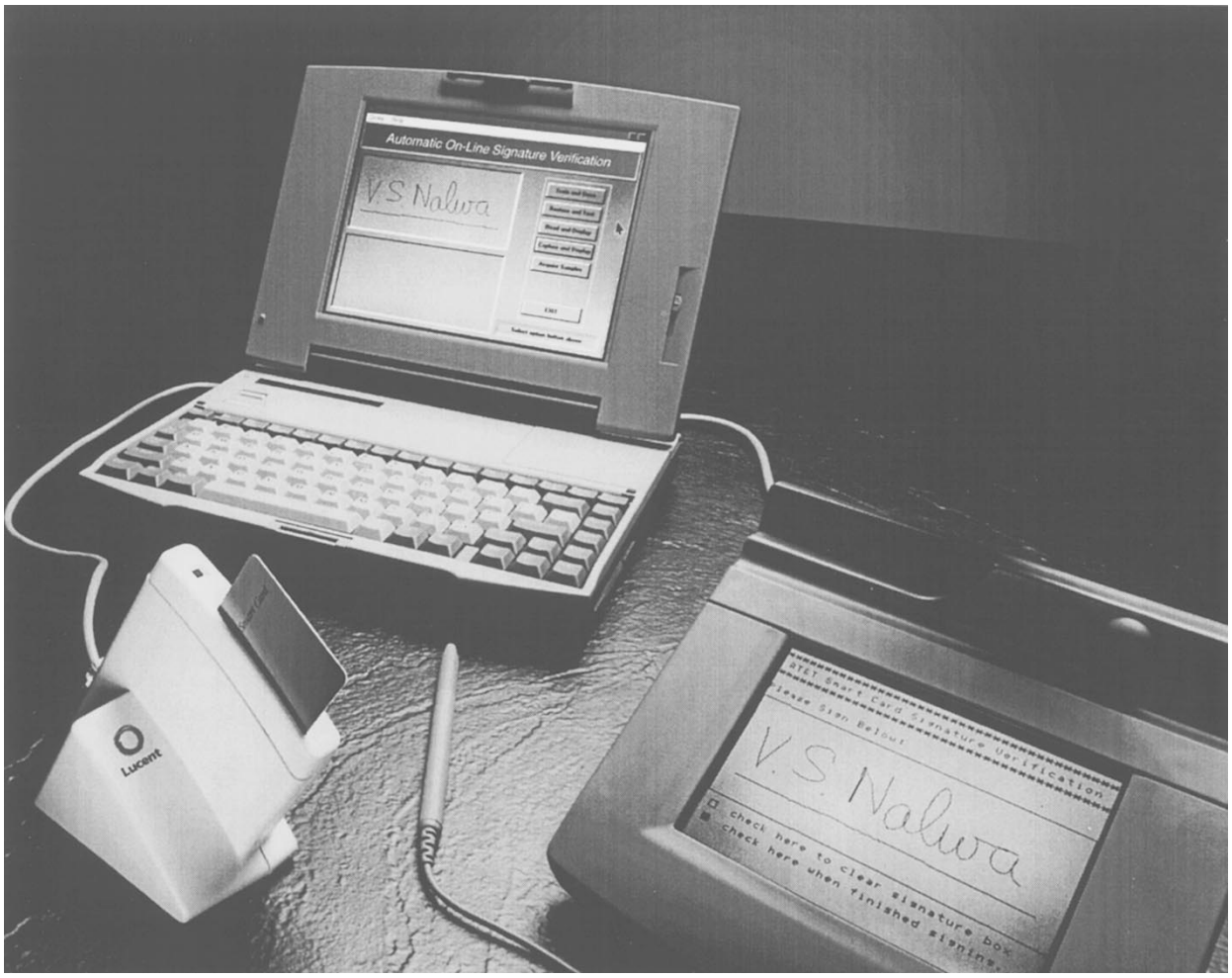

Fig. 21. A signature-verification system.

both connected to the notebook computer through serial ports; one serial port is standard on the notebook computer and another serial port is configured through a PCMCIA-slot adapter. The writing tablet captures on-line signatures and sends them to the notebook computer, which creates signature models, storing each model on a smart card from which we can subsequently retrieve the model for on-site signature verification. The notebook computer is based on the Intel 486 DX2/50 MHz microprocessor, and the electronic writing tablet is an LCD writing tablet with a maximum uniform spatial resolution of 512 dots/in (or about 0.05 mm) along both the horizontal and vertical directions, and with a uniform temporal sampling rate of 200 samples/s. However, owing to on-tablet signature smoothing and compression, these spatial and temporal sampling rates of the writing tablet were not available to us uniformly.

We tested the described system most extensively using six signatures to model each signature; this choice is based on the database results we discussed in Section VIII. The model of each signature we stored on a smart card required about 600 bytes, after some straightforward compression. Although we do not require a visual sample of a signature for automatic verification, we stored one such sample signature on the smart card too, again after straightforward compression; the storage requirement for a specimen signature was typically between a few hundred bytes and a kilobyte, depending on the signature. Such a sample signature allows the human operating the system not only to override the automatic result, but also show the signer why a particular signature was rejected. Note that even though our signature model is relatively unique to a signature, we cannot effectively recover a specimen signature from its model. Thus potential forgers cannot recreate a card owner's signature from just the model stored on the card: They need access to the specimen signature stored on the card if the card is all they have to work with. Hence, we should consider the interesting possibility of encoding the specimen signature we store on the card on the basis of a PIN, which is known only to the rightful owner of the smart card, and which is necessary to decode the signature for visual display.

Modeling a signature from its six specimens takes about 20 s, and verifying a signature takes about 2 s. After verification, a window pops up displaying an answer between -100 and $+100$, a positive value indicating acceptance of a signature as a genuine, and a negative value indicating its rejection—the larger the magnitude, the greater the

acceptance or rejection. The threshold zero, which distinguishes forgeries from genuines here, corresponds to 0.50 on the scale of Figs. 19 and 20, and it corresponds roughly to 7% false rejects and 1% false accepts on the error tradeoff curve for DB in Fig. 16. Typically, we observed a correlation between the magnitude of the answer and the degree of apparent visual similarity or dissimilarity between the signature being verified and its previously stored specimen. Anecdotally, if the model acquired was good, which was the case roughly nine out of every ten times, the quality of verification greatly impressed users of the system. In the roughly one out of every ten times that the performance of the system was not as impressive, remodeling often corrected the situation. This anecdotal observation in laboratory settings is in line with the observation in Section IV, supported by Section VIII, that the goats of a system typically limit its performance.

In any event, some signers will always be too inconsistent to allow us to build a model of their signatures that will allow us to both accept genuine signatures and reject forgeries reliably. Based on the database results of the preceding section, this number is probably at around 5%. We have three choices for such signers: We can attempt to force them to be consistent—something not always possible—we can insist that these signers use an additional mechanism of identification, such as a PIN, or we can altogether abandon the use of signatures for such signers, taking recourse to other mechanisms of identification and authorization.

X. Conclusion

What is primarily lacking in this investigation of automatic on-line signature verification is a field test of a real-time on-line on-site system based on the signature-verification algorithm we discussed. In my judgment, the necessary conditions that I laid out for such an undertaking in Section III have been met: Most users of the described system in a laboratory setting have expressed satisfaction at its performance, and the error tradeoff curves for the system on three different databases seem reasonable. These are my suggestions for a field trial.

- The signatures acquired and verified must be purposeful, as for gaining access to a facility.
- The signatures of each participant in the trial must be verified over an extended period of time, as individual signatures tend to vary from day to day.
- The signatures must be from individuals who can easily be contacted during and after the trial for their feedback on the system, individuals such as employees of an organization.
- After a signer provides a signature for verification, but prior to its automatic verification, the signer must be asked to rate the quality of the signature—perhaps from *very good* to *very bad*. This information would help us establish the correlation, if any, between the perception of signers of the quality of their signatures and the answers generated by the system, such correlation being important for successfully introducing the system into the marketplace, as we discussed in Section III.
- In the event of signature rejection by the verification system, we must have a clear fall-back procedure, such as manual verification of the signature by comparing it against its stored visual sample, this sample being accessible only by entering a password or a PIN known only to the original signer.

Further, it would help to conduct the field trial in multiple phases, at least two, the experience of conducting one phase allowing us to modify the succeeding phase.

Topics in on-line signature verification that deserve our further attention include the following:

- better models for signatures whose instances are not shaped consistently or for which we have fewer than six instances to build a model;
- acquisition of instances of a signature used to build its model over multiple sessions, rather than over a single session, to obtain a more representative variety of instances of the signature;
- invocation of multiple models for individuals with multiple distinct signatures;
- statistically well-founded procedures for determining the parameters of a model from the relatively few instances of a signature available to model the signature;
- automatic adaptation of models to signatures as they evolve over time;
- theoretically sound statistical framework to exploit fully each of the various individual error measures generated from comparing the characteristic functions of a signature to their prototypes;
- partial matching of signatures, highlighting discrepancies if they are specific;
- identification of problem signers, including those who are unusually inconsistent or have signatures that are trivial to forge;
- comparison of signatures to their models at multiple or personalized resolutions, rather than at a single common resolution.

Over and above these issues, we must also further investigate the usefulness of pen dynamics during on-line signature production in automatic on-line signature verification. Such dynamics might include not only velocities and forces, but also the varying orientation of the pen, and the way in which a signer grasps a pen.

Acknowledgment

E. Pednault created the writing-tablet interface that first made it possible for me to conduct live experiments, such experiments crucial to the progress of my effort since the start. D. Weimer, with his uncommon intuition and artistic skills, highlighted the inadequacy of my early developmental efforts through live experiments. J. Bromley provided me two of the three databases on which I tested various implementations of my algorithm, and W. Turin provided me the third database. W. DePope unhesitatingly addressed my innumerable personal-computer questions,

and R. Carlisle ported my C code to a personal computer that was coupled to a writing tablet and to a smart-card reader, configuring the signature-verification system I described.

L. Rabiner and N. Jayant supported this effort through a critical phase in 1993, and A. Netravali and P. Henry provided me an environment for its unhindered completion in 1994, when I first reported this work in an internal Bell Laboratories Technical Memorandum. I am especially grateful to A. Netravali for his unwavering support.

REFERENCES

[1] G. Chrystal, *Algebra: An Elementary Text-Book for the Higher Classes of Secondary Schools and for Colleges*, pt. 1, 7th ed. New York: Chelsea, 1964.
[2] H. D. Crane and J. S. Ostrem, "Automatic signature verification using a three-axis force-sensitive pen," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-13, no. 3, pp. 329–337, May–June 1983.
[3] J. J. Denier van der Gon and J. Ph. Thuring, "The guiding of human writing movements," *Kybernetik*, vol. 2, no. 4, pp. 145–148, Feb. 1965.
[4] I. Evett and R. N. Totty, "A study of the variation in the dimensions of genuine signatures," *J. Forensic Sci. Soc.*, vol. 25, pp. 207–215, 1985.
[5] W. R. Harrison, *Suspect Documents: Their Scientific Examination*. New York: Praeger, 1958 (reprinted by Nelson-Hall, Chicago).
[6] T. Hastie, E. Kishon, M. Clark, and J. Fan, "A model for signature verification," in *Proc. 1991 IEEE Int. Conf. on Syst., Man, Cybern.*, vol. 1, Charlottesville, VA, Oct. 1991, pp. 191–196.
[7] F. Leclerc and R. Plamondon, "Automatic signature verification: The state of the art—1989–1993," *Int. J. Patt. Recognit. and Artificial Intell.*, vol. 8, no. 3, pp. 643–660, June 1994.
[8] M. M. Lipschutz, *Differential Geometry*. New York: McGraw-Hill, 1969.
[9] G. Lorette and R. Plamondon, "Dynamic approaches to handwritten signature verification," in *Computer Processing of Handwriting*, R. Plamondon and C. G. Leedham, Eds. Singapore: World Scientific, 1990, pp. 21–47.
[10] J. Mathyer, "The expert examination of signatures," *J. Criminal Law, Criminol. Police Sci.*, vol. 52, pp. 122–133, 1961.
[11] V. S. Nalwa, *A Guided Tour of Computer Vision*. Reading, MA: Addison-Wesley, 1993.
[12] W. Nelson and E. Kishon, "Use of dynamic features for signature verification," in *Proc. 1991 IEEE Int. Conf. on Syst., Man, Cybern.*, vol. 1, Charlottesville, VA, Oct. 1991, pp. 201–205.
[13] A. S. Osborn, *Questioned Documents*, 2nd ed. Albany, NY: Boyd, 1929 (reprinted by Nelson-Hall, Chicago).
[14] J. R. Parks, "Automated personal identification methods for use with smart cards," in *Integrated Circuit Cards, Tags and Tokens: New Technology and Applications*, P. L. Hawkes, D. W. Davies, and W. L. Price, Eds. Oxford, U.K.: BSP, 1990, pp. 92–135.
[15] R. Plamondon and G. Lorette, "Identity verification from automatic processing of signatures: Bibliography," in *Computer Processing of Handwriting*, R. Plamondon and C. G. Leedham, Eds. Singapore: World Scientific, 1990, pp. 65–85.
[16] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*. Englewood Cliffs, NJ: Prentice-Hall, 1993.
[17] G. Salmon, *A Treatise on Conic Sections*, 6th ed. New York: Chelsea, 1954.
[18] Y. Sato and K. Kogure, "Online signature verification based on shape, motion, and writing pressure," in *Proc. 6th Int. Conf. on Patt. Recognit.*, Munich, Germany, Oct. 1982, pp. 823–826.
[19] T. K. Worthington, T. J. Chainer, J. D. Williford, and S. C. Gundersen, "IBM dynamic signature verification," in *Computer Security: The Practical Issues in a Troubled World*, J. B. Grimson and H.-J. Kugler, Eds. Amsterdam: North-Holland Elsevier, 1985, pp. 129–154.
[20] P. Zhao, A. Higashi, and Y. Sato, "On-line signature verification by adaptively weighted DP matching," *IEICE Trans. Informat. Syst.*, vol. E79-D, no. 5, pp. 535–541, May 1996.

Vishvjit S. Nalwa received the B.Tech. degree in electrical engineering from the Indian Institute of Technology, Kanpur, in 1983. He received the M.S. and Ph.D. degrees in electrical engineering from Stanford University, Stanford, CA, in 1985 and 1987, respectively.

Since 1987, he has been with Bell Laboratories, Holmdel, NJ. His research interests span the processing and capture of signals with geometric interpretations, as in computer and human vision. He is the author of *A Guided Tour of Computer Vision* (Addison-Wesley, 1993). He holds several U.S. and international patents. He is currently an Associate Editor of IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE.

Dr. Nalwa received the First Prize for Academic Excellence in the Core Curriculum and the General Proficiency Prize for the Best Graduating Student in Electrical Engineering, both jointly, at the Indian Institute of Technology, Kanpur. He received the Information Systems Laboratory Fellowship from Sanford University. He has since won several awards for his research.