Polynomial Approximation in Handwriting Recognition

[Extended Abstract]

Stephen M. Watt
Dept of Computer Science
University of Western Ontario
London, Ontario, Canada
Stephen.Watt@uwo.ca

ABSTRACT

Considering digital ink traces as plane curves provides a useful framework for handwriting recognition. Characters may be represented as parametric curves approximated by certain truncated orthogonal series, mapping symbols to the low-dimensional vector space of series coefficients. Many useful properties are obtained in this representation, allowing fast recognition based on small training sets. The beauty of this framework is that a single, coherent view leads to highly efficient methods with a high recognition rate. Furthermore, these truncated orthogonal series are subject to all the geometric techniques of symbolic-numeric polynomial algorithms.

Categories and Subject Descriptors

I.5.1 [**Pattern Recognition**]: Models; I.7.5 [**Document and Text Processing**]: Document Capture—*Optical character recognition*; G.1.2 [**Numerical Analysis**]: Approximation

General Terms

Algorithms

Keywords

Handwriting recognition, orthogonal series, approximation

1. INTRODUCTION

Machine-based handwriting recognition has been studied now for more than a century. In 1910, Hyman Goldberg proposed recognizing handwriting using electically conducting ink [1]. Since then, the subject of handwriting recognition has grown and flourished. Handwriting recognition is essential to major economic activities, such as cheque processing and mail sorting, and is a standard feature on many mobile electronic devices.

There is by now a vast literature on the subject of handwriting recognition by computer, divided between "off-line" and "on-line" recognition. Off-line recognition takes a static image of some handwriting and produces text. The input is typically an image which may involve background noise, digitization artifacts and distortion. On-line recognition takes motions and other events, such as button presses, pen up and pen down, and produces text. A variety of capture devices may be used, including digitizing tablets, screen overlays or cameras. The captured pen movements and related events may be called "digital ink" regardless of the source, and which may be stored and transmitted in a number of ways, including InkML [2]. On-line recognition is often regarded as an easier problem because the writing order is given, the identification of the input is evident and mis-recognitions can be corrected. On the other hand, processing time becomes a constraint and there is no forward context.

A problem of particular interest is that of mathematical handwriting recognition. High quality mathematical handwriting recognition would be useful for expression entry and editing in both document processing systems and mathematical software, such as computer algebra systems. We are therefore interested in on-line methods. The usual techniques used for natural languages cannot, however, be applied to written mathematics. There are a number of difficulties: First, a dizzying array of symbols from many different alphabets are used at once. Second, many similar characters, which must be distinguished, are written with just a few strokes. For example, Figure 1 shows a progression of symbols with similar features all written with one stroke. Third, the lay out is two dimensional and relative positioning matters. Moreover the symbols are typically in several sizes leading to ambiguous juxtapositions, as shown in Figure 2. (Are the first two symbols a^p or aP?) Fourth, in mathematical handwriting, there is not a useful fixed dictionary of words that may be used to rule out senseless letter combinations. Almost any sequence of symbols potentially has meaning. On the other hand, symbols tend to be well separated with a known orientation.

With so many symbols, each with variants, and new ones frequently added, the usual techniques of symbol recognition are difficult to apply. It is impractical do develop hand-tuned heuristics to recognize specific features for each symbol. Matching against a comprehensive database of models is too time consuming. Neural nets require massive amounts of training data not readily available for rarely used symbols. Instead, in a series of papers, we have developed a framework suitable for this setting, allowing high recognition rates at high speed with only modest amounts of training data.

SNC 2011, June 7-9, 2011, San Jose, California.

Figure 1: Similar single-stroke characters

Figure 2: Juxtaposition ambiguity

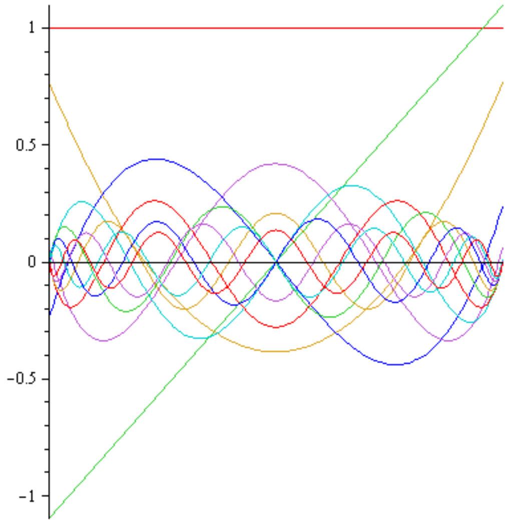

Figure 3: L-S polynomials on $[0, 1]$ for $\mu = 1/8$

2. FRAMEWORK

One of the main difficulties we find with other recognition methods is that the digital ink is thought of as a set of points, and subject to various *ad hoc* treatments, such as smoothing to eliminate device jitter and "re-sampling", i.e. interpolation to add or remove or equalize the distance between points. This is followed by various numerical heuristics to try to identify features such as cusps, self intersections, etc. The problem with this is that we are not treating the ink traces as what they really are, curves.

We consider an ink trace to be a segment of a plane curve $(x(\lambda), y(\lambda)), \lambda \in [0, L]$. Various parameterizations are possible, including parameterization by time (as the curve was traced), by arc length or by affine arc length. We have found over the course of various experiments that arc length is the most robust parameterization in most cases, which makes intuitive sense since this gives curves that look the same the same parameterization.

The next step is to realize that curves that are "almost the same" should be recognized as the the same symbol. We may therefore work with approximations in a finite dimensional function space:

$$x(\lambda) \approx \sum_{i=0}^{d} x_i B_i(\lambda), \qquad y(\lambda) \approx \sum_{i=0}^{d} y_i B_i(\lambda).$$

By appropriate choice of basis functions B_i, $i = 0, \ldots, d$, the approximations can be made arbitrarily close to x and y. After normalizing for position and scale, the coefficients represent the curve as a $(2d-1)$-dimensional point. If the basis functions are orthogonal with respect to a functional inner product, then we can obtain the coefficients (x_i, y_i) by numerical integration. One of the most appealing aspects to this is that this completely captures the curves in a manner independent of the resolution of the device and allows us to ask geometrically meaningful questions.

The first step in this approach used Chebyshev polynomials as the basis functions [3]. The non-linearity of the weight function $1/\sqrt{1-\lambda^2}$ required first capturing the entire curve, normalizing it, then computing the series coefficients. It was then shown that a Legendre polynomial basis allowed the coefficients to be calculated instantly on pen-up from moments of $x(\lambda)$ and $y(\lambda)$ integrated as the curve is written [4]. This real-time property is preserved using a Legendre-Sobolev basis, orthogonal with respect to the inner product

$$\langle f, g\rangle = \int_a^b f(\lambda)g(\lambda)d\lambda + \mu \int_a^b f'(\lambda)g'(\lambda)d\lambda,$$

for which orthogonal series approximate both the function and its derivative well [5]. An orthogonal polynomial basis may be obtained for this or any desired inner product by Gram-Schmidt orthogonalization of the monomial basis $\{1, \lambda, \lambda^2, \ldots\}$. The Legendre-Sobolev polynomials of degrees 0 to 12 for $a = 0, b = 1, \mu = 1/8$ are shown in Figure 3. As the representation by series coefficients is space efficient, it may be used for digital ink compression [6].

3. DISTANCE-BASED RECOGNITION

Representing handwritten mathematical symbols as points in such an inner product space has a number of pleasant properties. One property of using orthonormal series is that distances in the function space become Euclidean distances in the coefficient space:

$$||f - g|| \approx \sqrt{\sum_{i=0}^{d} (f_i - g_i)^2}$$

as the integrals of the $B_i B_{j \neq i}$ cross terms vanish by orthogonality. This allows the variational integrals approximated by elastic matching to be calculated extremely quickly.

We find that classes of like symbols form clusters, and that the classes are convex even at low dimension [7, 8]. This implies that linear homotopies between points in the same class should have intermediate points that remain in the same class. Indeed this is observed, as illustrated in Figure 4. This property allows us to classify points based on distances to convex hulls of sets of points, rather than to particular point or their averages, which is more robust when the training sets are small. A comparison of recognition methods using elastic matching with dynamic time warping (re-parameterization), and Euclidean and Manhattan distances in the orthogonal series coefficient space [9] shows that for many choices of dimension d and coefficient size,

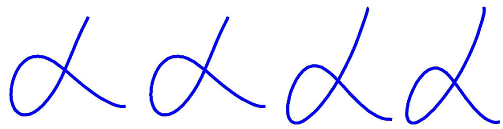

Figure 4: Linear homotopy stays within the class

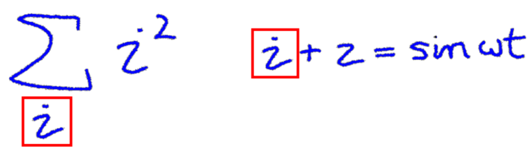

Figure 5: Context dependency: i or $\dot{z}$

the coefficient-based methods give similar results to elastic matching, but may be computed much faster. Alternatively to convex hulls of the known points, one may use the polyhedra bordered by the planes of SVM ensembles [10] and these may additionally make use of other features [11].

Even with the best individual character recognition, ambiguities remain. Consider the two expressions shown in Figure 5. Even though the symbol shown in the red box is exactly the same in both cases, in one case it should be recognized as "i" and in the other as "$\dot{z}$". It is therefore useful to be able to use our distance-based methods in conjunction with contextual information, such as frequency information for mathematical n-grams [12, 13, 14]. In this regard, a useful distance-based confidence measure may be obtained for classification using either distances to convex hulls of classes or to separating planes in SVM ensembles [15].

To this point, we have discussed classification based on functional approximation of the coordinate curves $x(\lambda)$ and $y(\lambda)$. If the orientation of the characters is not certain, or if they are deformed in other ways, then it is natural to seek methods that are invariant under these transformations. We have seen that similar methods may be applied to integral invariants of the coordinate curves to classify symbols with unknown rotation or shear [16, 17].

4. ON TO APPROXIMATE POLYNOMIALS

The objects with which we are working, these truncated series, are polynomials with approximate coefficients in a non-monomial basis. Given the vast body of work in approximate polynomial algebra, e.g. [18, 19, 20, 21, 22], we ask what ideas may be brought to bear from this community. This points to several directions for investigation.

It is easy in this representation to find all the critical points of digital ink traces. Many of these, such as self intersection, number of local maxima, etc, have long been used as features for recognition. The usual methods for detecting these features depend significantly on device resolution. With rapidly evolving technology, this means that newly adjusted algorithms become necessary, and these cannot use archival data directly. Instead, finding these critical points from the polynomial approximations is robust against changes in device resolution.

We give an example. Consider the digital ink trace of a lower case letter d, shown in Figure 6(a). The trace consists of about 300 data points sampling the x and y coordinates and pressure at a uniform frequency. Figure 6(b) shows an approximation in parametric form $(x(\lambda), y(\lambda))$ for $\lambda \in [0, 1]$, with x and y being Legendre Sobolev series over $\lambda \in [0, 1]$ with $\mu = 1/8$ up to degree 12. Figure 6(c) shows the critical points found by solving $x'(\lambda) = 0$ and $y'(\lambda) = 0$. This is achieved by univariate polynomial root finding, retaining real roots in the interval $[0, 1]$.

The key point is that it is meaningful to perform useful analysis efficiently on the traces as curves, rather than considering the trace as a collection of discrete points. To illustrate this concretely, consider the critical point on the top of the body of the letter d. This is the second critical point from the beginning of the trace. Finding the critical point from the polynomial approximation can be done by a fast Newton iteration. This takes into account the effect of all the sample points on the local behaviour. In contrast, consider trying to find this local maximum from the discrete sample points. A portion of the trace data is shown in Figure 7, magnified more vertically than horizontally. We see that the local maximum should occur somewhere near the five sample points with approximately equal y value, but have to construct heuristics to compute a value. For example, in deciding which points are at the maximum, what error tolerance should be used? If several points are within the error tolerance of being the maximum, whereabouts in that point set should the maximum be taken? Should the maximum be taken as the maximum value achieved by one of the sample points, or should the maximum of a local spline be used? All this is avoided by working with curves instead of points.

Some operations can be more natural on implicit curve models. This is obtained directly from the parametric representation as $Resultant(X - x(\lambda), Y - y(\lambda), \lambda)$. For the example, the implicit polynomial obtained this way to approximate the example trace is plotted in Figure 6(d). This polynomial of degree 12 in X and Y has 91 terms. (A plotting artefact loses part of the tail.)

5. FUTURE DIRECTIONS

There appears to be a fertile ground for further work in this area. It should be useful to maintain the perspective that perturbations are to be minimized in the Legendre-Sobolev space instead of with respect to polynomial norms in a monomial basis. Thus we would want to compute resultants, SVDs, etc, in this basis, rather than perform ill-conditioned conversions. Some new results are required in this area in order to proceed.

Two of the main ideas in symbolic-numeric algorithms for polynomials are that of backward error and semi-definite programming. With backward error, we may ask questions such as whether there is a near by polynomial (i.e. requiring perturbation by less than a given bound) that has certain properties, e.g. singularity, factorization, etc. With semi-definite programming we can ask questions about the least perturbations required.

These tools provide a most useful opening for symbolic-numeric computation in handwriting recognition. Rather than *ad hoc* numerical techniques based on sample points, we have a framework in which to ask well-posed questions about nearby curves and have a systematic, meaningful approach to answering them.

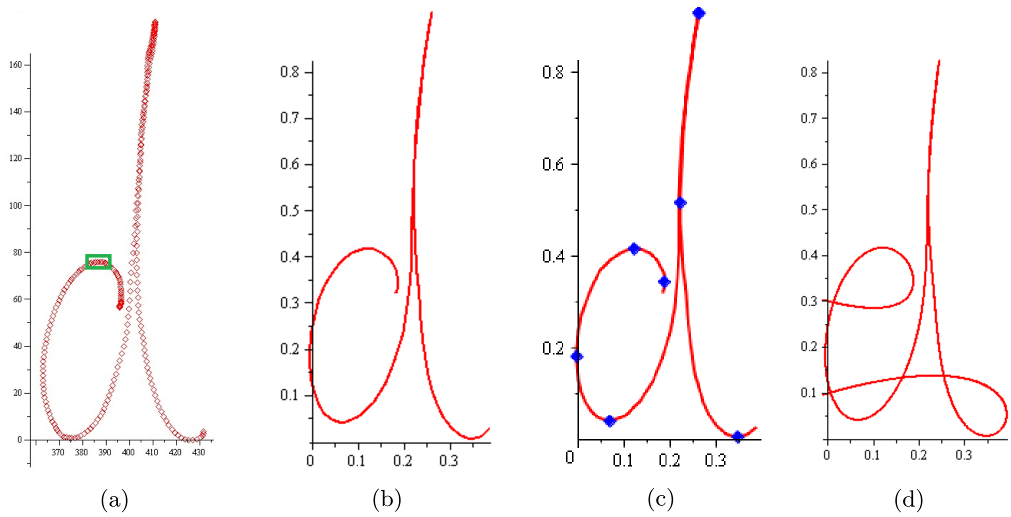

Figure 6: Analysis of an input symbol:
(a) Trace data (green box magnified in Figure 7), (b) Parametric approximation $(x(\lambda), y(\lambda))$ $\lambda \in [0, 1]$, (c) Critical points computed from $(x(\lambda), y(\lambda))$, (d) Implicit approximation $P(X, Y) = 0$.

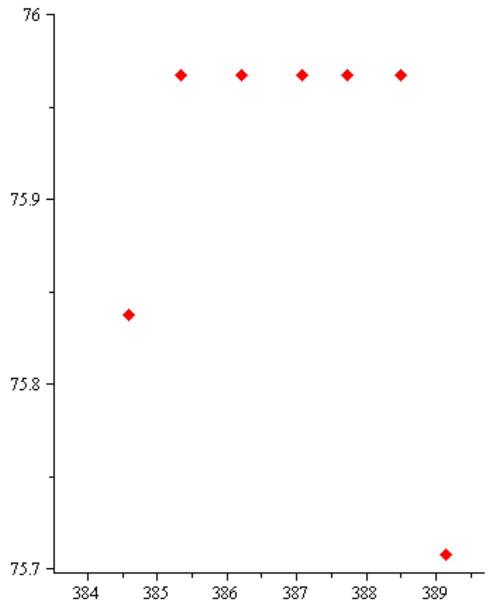

Figure 7: Top of d body, magnified more vertically

6. REFERENCES

[1] Goldberg, H.E: US Patent No. 1,117,184. Filed November 23, 1910. Granted November17, 1914.

[2] Watt, S.M. and Underhill, T. (eds): Ink Markup Language (InkML). W3C Proposed Recommendation 10 May 2011 `http://www.w3.org/TR/2011/PR-InkML-20110510` .

[3] Char, B.W. and Watt, S.M.: Representing and Characterizing Handwritten Mathematical Symbols Through Succinct Functional Approximation. pp. 1198–1202, Proc. Int'l Conf. Document Analysis and Recognition (ICDAR), IEEE Press 2007.

[4] Golubitsky, O. and Watt, S.M.: Online Stroke Modeling for Handwriting Recognition. pp. 72-80, Proc. 18th Annual Int'l Conf. Computer Science and Software Engineering (CASCON 2008), IBM Canada 2008.

[5] Golubitsky, O. and Watt, S.M.: Online Computation of Similarity between Handwritten Characters. pp. C1-C10, Proc. Document Recognition and Retrieval XVI (DRR XVI), SPIE and IS&T 2009.

[6] Mazalov, V. and Watt, S.M.: Digital Ink Compression via Functional Approximation. pp. 688-694, Proc. Proc. 12th Int'l Conf. Frontiers in Handwriting Recognition (ICFHR 2010), IEEE Computer Society 2010.

[7] Golubitsky, O. and Watt, S.M.: Online Recognition of Multi-Stroke Symbols with Orthogonal Series. pp. 1265-1269, Proc. 10th Int'l Conf. Document Analysis and Recognition (ICDAR 2009), IEEE Computer Society 2009.

[8] Golubitsky, O. and Watt, S.M.: Improved Classification through Runoff Elections. pp. 59–64, Proc. Int'l Workshop on Document Analysis Systems (DAS 2010), ACM 2010.

[9] Golubitsky, O. and Watt, S.M.: Distance-Based Classification of Handwritten Symbols. Int'l Journal on Document Analysis and Recognition, Vol. 13, No. 2, pp. 133-146, Springer 2010.

[10] Keshari, B. and Watt, S.M.: Online Mathematical Symbol Recognition using SVMs with Features from Functional Approximation. Electronic Proc. Mathematical User-Interfaces Workshop 2008 (MathUI 2008), Activemath.org 2008.

[11] Keshari, B. and Watt, S.M.: Hybrid Mathematical Symbol Recognition Using Support Vector Machines. pp. 859–863, Proc. Int'l Conf. Document Analysis and Recognition (ICDAR), IEEE Press 2007.

[12] So, Clare M. and Watt, S.M.: Determining Empirical Properties of Mathematical Expression Use. pp. 361-375, Proc. Fourth Int'l Conf. Mathematical Knowledge Management (MKM 2005), LNCS 3863, Springer 2005.

[13] Watt, S.M.: An Empirical Measure on the Set of Symbols Occurring in Engineering Mathematics Texts. pp. 557-564, Proc. 8th IAPR Int'l Workshop on Document Analysis Systems (DAS 2008), IEEE Computer Society 2008.

[14] Smirnova, E. and Watt, S.M.: Context-Sensitive Mathematical Character Recognition. pp. 604–610, Proc. IAPR Int'l Conf. Frontiers in Handwriting Recognition (ICFHR 2008), Cenparmi, Concordia University 2008, ISBN 1-895193-03-6.

[15] Golubitsky, O. and Watt, S.M.: Confidence Measures in Recognizing Handwritten Mathematical Symbols. pp. 460-466, Proc. Conf.s on Intelligent Computer Mathematics 2009, LNAI 5625, Springer 2009.

[16] Golubitsky, O., Mazalov, V. and Watt, S.M.: Orientation-Independent Recognition of Handwritten Characters with Integral Invariants. pp. 252-261, Proc. Joint Conf. of ASCM 2009 and MACIS 2009 (ASCM 2009), COE Lecture Note Vol. 22, Kyushu U. ISSN 1881-4042.

[17] Golubitsky, O., Mazalov, V. and Watt, S.M.: Towards Affine Recognition of Handwritten Mathematical Characters. pp. 35–42, Proc. Int'l Workshop on Document Analysis Systems (DAS 2010), ACM 2010.

[18] Watt, S.M. and Stetter, H.J. (eds): Symbolic-Numeric Algebra for Polynomials. J. Symbolic Computation Special Issue, Vol. 26 No 6. 1998.

[19] Wang, Dongming and Zhi, Lihong (eds): Symbolic-Numeric Computation (SNC 2005). Birkhäuser 2007.

[20] Watt, S. and Verschelde, J.: Proc. 2007 Conf. Symbolic Numeric Computation (SNC 2007). ACM 2007.

[21] Kai, Hiroshi and Sekigawa, Hiroshi (eds): Proc. 2009 Conf. Symbolic Numeric Computation (SNC 2009). ACM 2009.

[22] Moreno Maza, M. (ed): Proc. 2011 Conf. Symbolic Numeric Computation (SNC 2011). ACM 2011.