Digital Ink Compression via Functional Approximation

Vadim Mazalov and Stephen M. Watt

Ontario Research Center for Computer Algebra
Department of Computer Science
University of Western Ontario

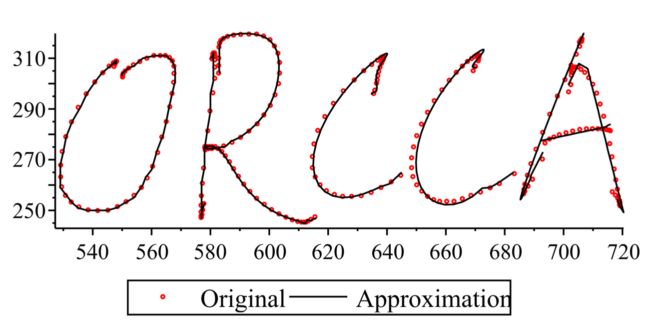
Introduction

In our work on symbol recognition we have found it useful to represent ink strokes in a functional form, as coefficients of truncated orthogonal series. This form has the property that the shapes of the curves are given quite succinctly. It is natural to ask how this form may be used for compression. This is the subject of this poster. A consequence of this work is that we almost directly do recognition on compressed ink.

Problem Statement

We ask whether it is feasible to apply the theory of functional approximation to describe a stroke up to some given threshold of the maximal point-wise error and root mean square error. If so, what is the compression one could expect as the result of such approximation?

To measure the quality of approximation independently of application and device, we have computed errors as a fraction of the height of the characters in a stroke.



	O	R	C	C	A
Max. pt-wise err., %	1	2	3	4	5
RMSE, %	0.33	0.67	1	1.33	1.67

Table: Different approximation thresholds.

Segmentation

The following segmentation has been tested

1. Fixed Degree Segmentation.
2. Fixed Length Segmentation.
3. Adaptive Segmentation (the combination of degree and coefficient size that gives the smallest resulting total size for each stroke and channel).

Compression

At a high level, our compression method takes the following steps for each stroke:

```
for all Segments do
  for all Channels do
    Compute the orthogonal series coefficients for
    the appropriate segmentation
  end for
end for
Compress the stream of coefficients
```

Reconstruction

```
Decompress the coefficient stream to obtain the
curve segments
for all Channels do
  for all Segments do
    Blend the curves on the overlaps to obtain the
    piecewise coordinate functions.
    Obtain traces by evaluating the coordinate func-
    tions with the desired sample frequency.
  end for
end for
```

On a given segment, the series coefficients are computed by numerical integration of the required inner products. The cost to compute the compression is linear in the number of trace sample points and in the number of coefficient size/approximation degree combinations allowed.

Parameterization

We tested two choices for curve parameterization widely used in pen-based computing: time and arc length.

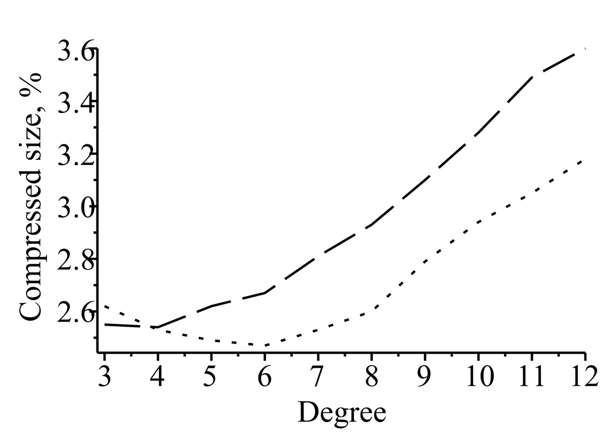


Figure: Compression for parameterization by time (dot) vs. arc length (dash) for series with integer coefficients.

Segment Blending

To make the transition between approximation pieces smooth, we blend the pieces by overlapping segments slightly and transitioning linearly from one segment to the next on the overlap.

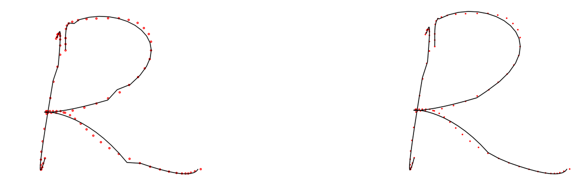


Figure: Example of blending.

Experimental Setting

The highest resolution device's specifications were: 512 pressure levels, 2540 dpi resolution and 133 pps max data rate. The sampling error of the device was $\pm.02$ in and the resolution of the monitor was 94 dpi. Therefore, the absolute sampling error, as the stroke is rendered on the screen, is $\approx \pm 2$ pixels. Error, relative to the height of writing, is $\approx 2.5\%$. Different individuals were asked to write various parts of regular text to ensure variations in the length of strokes and writing styles. Overall, we obtained 108,094 points split in 1,389 strokes.

Compressed size reported for the experiments is obtained by comparing the compressed size of the entire database to the original size, reporting it as a fraction between 0% and 100%.

Compression of Textual Traces

One set of experiments used a textual representation of trace data, which is relevant to XML-based standards.

$$\lambda_0; c^1_{00}, c^1_{01}, \ldots, c^1_{0d_{01}}; \ldots; c^N_{00}, c^N_{01}, \ldots, c^N_{0d_{0N}}$$
$$\lambda_1; c^1_{10}, c^1_{11}, \ldots, c^1_{1d_{11}}; \ldots; c^N_{10}, c^N_{11}, \ldots, c^N_{1d_{1N}} \quad \ldots$$
$$\lambda_D$$

where λ_i is the initial parameterization value of piece i in the stream, N is the number of channels (such as x and y coordinates of points, pressure, etc.) and d_{ij} is the degree of approximation of the piece i for j-th channel. Pen-based devices typically provide three channels: x, y coordinates of points and pen pressure p.

Compression of Binary Traces

We represented the sequence of approximation coefficients in an exponential format as ab where a and b are two's complement binary integers, standing for significand and a power of 10 respectively. We used the adaptive segmentation scheme and chose stroke-wise approximation parameters for each input channel separately. Compression packets for each stroke i took the form

$$b_i; d_i; \lambda_1; c_{10}, \ldots, c_{1d_i}, \lambda_2; c_{20}, \ldots, c_{2d_i} \;\ldots\; \lambda_D$$

where b_i is the number of bits, d_i degree, λ_j initial value of parameterization of piece j and $c_{j0}, c_{j1}, \ldots, c_{jd_i}$ are coefficients.

Comparison with Second Differences

A stroke may be represented by the values of the first two points and a sequence of second differences.

B\E,%	0.0	0.6	1.1	1.5	2.0	2.5	3.1	3.5
C	–	7.50	6.22	5.93	5.26	5.14	4.87	4.65
L	–	9.22	6.97	6.32	5.64	5.25	5.20	5.04
L-S	–	12.64	11.21	10.19	8.67	8.55	8.26	7.51
△2	23.35	–	–	–	–	–	–	–

(a) binary coefficients

B\E,%	0.0	0.6	1.1	1.5	2.0	2.5	3.1	3.5
C	–	3.07	2.61	2.31	2.05	1.90	1.80	1.72
L	–	3.41	2.86	2.53	2.26	2.08	2.00	1.91
L-S	–	9.36	7.27	6.25	5.51	4.98	4.64	4.49
△2	8.64	–	–	–	–	–	–	–

(b) binary coefficients, compressed

Table: Compressed size (%) for binary representation with different pointwise error limits (E) and bases (B): Chebyshev (C), Legendre (L) and Legendre-Sobolev (L-S). The lossless second difference method is shown for comparison (△2).

References

[1] Microsoft Inc. Ink serialized format specification.

[2] S. M. Watt and T. Underhill (editors). *Ink markup language (InkML)*. W3C Working Draft, 2010.

[3] J. Ziv and A. Lempel. A universal algorithm for sequential data compression. *IEEE Transactions on Information Theory*, 23:337Ű343, 1977.

[4] D. A. Huffman. A method for the construction of minimum-redundancy codes. In Proc. of the *I.R.E.*, pp. 1098Ű1102, 1952.