Fast Stroke Matching by Angle Quantization

Luke Olsen* Faramarz F. Samavati Mario Costa Sousa

Dept. of Computer Science
University of Calgary

ABSTRACT

Determining similarity of two point sequences (*strokes*) is a fundamental task in gestural interfaces. Because the length of each stroke is arbitrary, mapping to a fixed-dimension feature space is often done to allow for direct comparison. In this paper, we propose a new feature space based on angle quantization. For each adjacent pair of points in a stroke, the vector between them defines an angle relative to a fixed axis. The sequence of these angles can be mapped to a k-dimensional feature space by quantizing the unit circle into k ranges, and taking a normalized count of the number of stroke angles in each range. The Euclidean distance between strokes in this feature space gives a measure of stroke similarity. The measure is scale invariant, and some degree of rotational invariance can be achieved with slight modification. Our method is shown to offer efficient and accurate gestural matching performance compared to traditional signal-processing and image-based methods.

Categories and Subject Descriptors

I.3 [**Computer Graphics**]: Methodology and Techniques—*Interaction techniques*; I.5 [**Pattern Recognition**]: Clustering—*Feature selection, similarity measures*

Keywords

Gesture recognition, interaction techniques

1. INTRODUCTION

The ways in which humans interact with computers are an active and important research topic. Historically, interfaces have been limited by the technology of the day, leading to functional but unintuitive WIMP-style (Window, Icon, Menu, Pointer) interfaces. A recent trend in human-computer interaction is the exploration of intuitive and discoverable interfaces [8]. Stroke-based or gestural interfaces are an example of such an interface.

*Corresponding author, olsenl@cpsc.ucalgary.ca.

IMMERSCOM 2007, October 10-12, Verona, Italy

DOI 10.4108/ICST.IMMERSCOM2007.2114

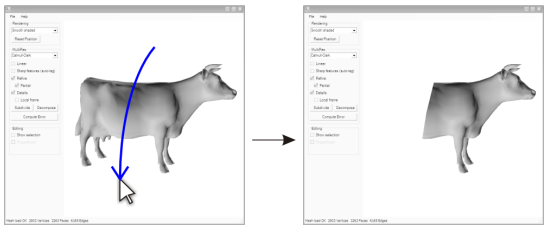


Figure 1: Gestural interfaces forgo menus in favor of intuitive gestures. Stroke matching is used to map input to the correct operation, eg. mesh cutting.

A fundamental problem in gestural interfaces is to determine the operation that corresponds to an input gesture (Fig. 1). The problem of determining similarity of point sequences appears in many applications such as trajectory analysis, character recognition, and motion planning. Because similarity is usually not treated as a binary relationship – i.e. two objects may be exactly the same, completely different, or anywhere in between the two extremes – similarity measures are used to quantify the amount of similarity between two objects in a consistent and intuitive manner. For stroke-based interfaces, the input is usually an ordered sequence of points $S = \{p_1, p_2, \ldots, p_n\}$ in window coordinates, acquired from an input device such as a mouse or tablet. It is difficult to compare such objects directly. For example, to compare two strokes one might try to compute the Euclidean distance between them by summing the point-wise distances. This approach is poor for several reasons: it is dependent on the coordinate system, the ordering, and the number of points. None of these factors play a role in the *intuitive* notion of similarity. As Sezgin et al. [19] observe, a recognition system should "respond to how an object looks, not how it was drawn."

Thus, similarity measures are commonly defined within a *feature space*, which the objects can be transformed to. For example, to recognize a handwritten character represented by a pixel grid, we might first extract some high-level information about the character – the number of bays and lakes, the width and height – and then compare these values to known characters. The tuple $\langle$ *bays, lakes, width, height* $\rangle$ represents the character in a 4-dimensional feature space.

In this paper, we describe a stroke feature that is based on quantizing the unit circle into a small number of angle ranges and counting the number of stroke angles in each

range (see Fig. 5). Because the stroke-to-feature transformation appeals to the geometric nature of a stroke, it is easy to understand and implement, and the features have intuitive geometric properties not offered by other methods, such as rotational and multi-scale invariance.

The paper is organized as follows. Section 2 discusses related work in the field of stroke and shape similarity. Section 3 presents our proposed method of angle quantization. In Sec. 4, we show that our method can outperform other methods in stroke recognition tasks. Finally, Sec. 5 offers some directions for future work.

2. RELATED WORK

Many approaches have been proposed for recognizing sketched input. An early approach by Rubine [18] uses geometric properties to distinguish gestures, such as the initial angle and bounding box size. Graph-based techniques try to judge similarity from the spatial relationships between strokes, such as crossings and shared endpoints [12]. Other methods exploit domain-specific knowledge to derive higher-level understanding of strokes, such as labelling different elements [20] or building a diagrammatic representation [2].

Image-based techniques, such as Kara and Stahovich [9], operate on strokes rendered to pixel grids rather than point sequences to allow for missing or overlapping strokes. Area-based methods, such as [7], try to characterize the area enclosed by a stroke rather than the stroke itself. These approaches only use spatial information, ignoring other information such as stroke order, direction, and speed.

While such approaches offer robust performance, their complexity is too much for recognizing simple gestures that are usually constrained to single strokes and designed to be distinct and discoverable. On the other hand, while non-interactive applications might have loose constraints on efficiency, a gestural input system must recognize and perform the requested operation quickly. Therefore, simple and fast approaches are preferred.

Methods based on Fourier analysis are quite common, particularly in trajectory analysis [1, 15]. Its adoption is not surprising given the strong theoretical basis and broad library support, but the transform loses locality of features due to signal-sized waves. Wavelet methods [3] attempt to address this issue by using smaller waves in the transformation, but suffer from signal length restrictions.

Direction-based methods use feature spaces based on the vectors defined by adjacent elements of a point sequence. Kato et al. [10] consider *directional element features* for image-based character recognition, building a feature based on 4 possible directions that can be associated with each contour pixel (up, down, and diagonals). Li et al. [13] use compass directions (North, Northwest, etc.) to represent a trajectory as a series of direction-distance-interval tuples; similarity is determined by examining trajectory directions in corresponding intervals. Shim and Chang [21] explore a similar construction, but allow for arbitrary angles and alter the similarity measure to account for displacement as well.

Our angle quantization method is direction-based, but is unique in that it produces fixed-dimensional features that can be compared easily. A similar method – for image-based hand gesture recognition – has been proposed by Freeman and Roth [6]. Their method computes a feature from the histogram of gradient directions in an image. However, the dimensionality of their features are high, and they must be smoothed to achieve good results. Although some systems, such as [19], consider stroke speed for recognition, our method works best with regularly sampled (i.e. speed-invariant) strokes.

Below we discuss three existing methods in stroke or trajectory analysis, which will be used in Sec. 4 to evaluate our proposed method. They were chosen because they represent a variety of approaches – wave- and wavelet-based, image-based – and produce fixed-dimensional features with efficient Euclidean distance measures, suitable for interactivity. Because other direction-based methods produce variable-length features, we do not consider them in our evaluation.

Fourier Transform (FT)

For a discrete signal $f(x)$ sampled n times, the Fourier transform $F(f(x))$ decomposes f into n complex-valued frequency components [16]. The complexity of a signal is reduced by retaining only the first $m < n$ coefficients.

Following the method of Naftel and Khalid [15], a stroke S is converted to a feature in Fourier space by splitting it into x and y signals, S^x and S^y, and applying the Fourier transform to them independently: $X = F(S^x)$ and $Y = F(S^y)$. Retaining the first m coefficients of X and Y (real and imaginary) results in a $4m$-element feature. Because the "power" of a signal is typically concentrated in lower frequencies, m can be chosen to be quite small ($m \approx 5$) [1]. Figure 2 depicts the Fourier feature of a 5-point star (inset).

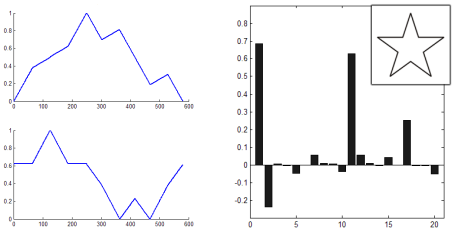


Figure 2: The FT feature of a star shape: (left) the x and y signals; (right) the corresponding feature, with $m = 5$.

According to a theorem of Parseval, the Euclidean distance between signals in the frequency domain is equal to their distance in the time domain [16]. Similarity of two strokes is therefore approximated by the Euclidean distance between their Fourier features.

Wavelet Decomposition (WD)

Wavelet decomposition of an n-sample signal $f(x)$ produces a coarse signal $c(x)$ with roughly half as many samples, and high-frequency details $d(\omega)$ in a wavelet basis, such that $f(x)$ can be reconstructed fully from $c(x)$ and $d(\omega)$. The complexity of a signal can be reduced by iteratively decompos-

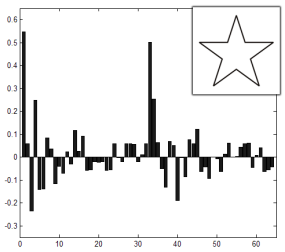


Figure 3: The WD feature of a star shape after 10 levels of decomposition, where $n_c = 1$, $n_d = 31$.

ing until $c(x)$ contains very few samples, and retaining only some elements of $d(\omega)$. For example, decomposing a signal with n samples j times produces an approximation with $n_c = \left\lfloor \frac{n}{2^j} \right\rfloor$ samples, and $(n - n_c)$ details. A k-dimensional feature is produced by retaining the n_c coarse samples and the $n_d = k - n_c$ most significant details.

For strokes, the x and y signals can be decomposed separately and then appended to form a single feature with $2(n_c + n_d)$ samples. However, for meaningful comparisons between strokes n_c must be the same for each, which requires that the original strokes are equal-length. Often this condition is assumed to be satisfied, eg. [3], but it is difficult to satisfy in gestural interfaces. Therefore, we resample each stroke to a fixed length prior to feature construction.

Characteristic Loci

The characteristic loci method of Glucksman [7] is an area-based method for character recognition. For a binary image I, each pixel is assigned a code based on how many lines are intersected by rays cast from the pixel in four directions: up, down, left, and right (Fig. 4). Variations of this method are still used in image-based character recognition [14]. For stroke comparisons, each stroke must be rasterized to a binary image, or acquired from the frame buffer rather than the pointing device.

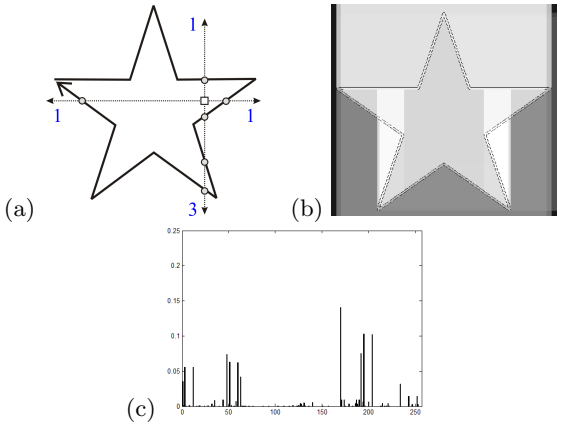


Figure 4: The Loci feature of a star shape: (a) each pixel is assigned a code based on four ray-line intersections; (b) a visualization of the codes; (c) a normalized count of each code produces a feature.

To construct a feature, the number of occurrences of each code are summed. Consider an image with 2 bits allotted for each ray (for a maximum of 3 intersections): in this case, there are $2^{(4 \cdot 2)} = 256$ possible codes, corresponding to a 256-dimensional feature. For b bits per direction, a 16^b-dimensional feature space results. Knoll [11] notes, however, that some codes occur very infrequently in practice, so the feature complexity could be heuristically reduced.

3. ANGLE QUANTIZATION

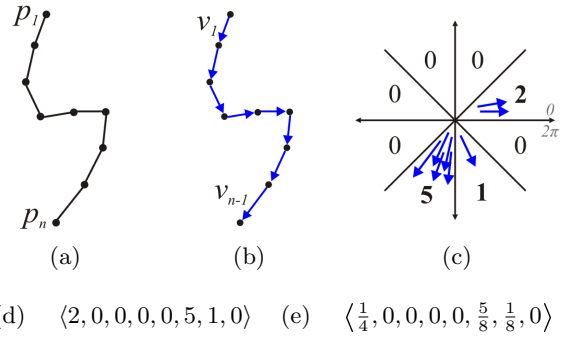
We now describe the details of our stroke feature based on angle quantization (AQ), which has the following properties:

- translation, scale, and multi-scale invariant;
- rotationally invariant under suitable distance measure;
- efficient to compute and compare; and,
- descriptive even with low-dimension features.

The construction is based on a directional stroke representation: every pair of points $\{p_i, p_{i+1}\}$ is replaced with a vector $v_i = p_{i+1} - p_i$. We then quantize the unit circle $[0, 2\pi]$ into k equal-size bins, and count the number of vectors in each bin. If bin i contains σ_i vectors, then the sequence of these sums forms a feature $\hat{Q}$:

$$\hat{Q} = \langle \sigma_1, \ldots, \sigma_k \rangle \ .$$

Figure 5 illustrates this construction for a simple stroke.



(d) $\langle 2, 0, 0, 0, 0, 5, 1, 0 \rangle$ (e) $\left\langle \frac{1}{4}, 0, 0, 0, 0, \frac{5}{8}, \frac{1}{8}, 0 \right\rangle$

Figure 5: Angle quantization of a stroke: (a) original stroke; (b) directional representation; (c) count of vectors in each angle range; (d) corresponding feature, $\hat{Q}$; (e) normalized feature, Q.

Because the directional representation depends only on relative point locations, AQ features are invariant under translation. To provide invariance to the sampling rate and scale, $\hat{Q}$ should be normalized to unit length, $Q = \hat{Q}/\|Q\|$ so that each coefficient in Q represents a percentage of stroke activity in an angle range (Fig. 5e).

The optimal number of bins, k, is application-dependent, but in our experiments $k = 16$ works well. (For reasons discussed below, multiples of two are preferred.) Larger values can make the quantization too sensitive to noise.

To compare two AQ features, we use the Euclidean, or norm-2, distance measure, $|\cdot|_2$:

$$d(Q_1, Q_2) = |Q_1 - Q_2|_2 = \sqrt{Q_1 \cdot Q_2} \ .$$

Multi-Scale Invariance

For regularly sampled strokes, the normalized AQ features are scale invariant. This is an important property in typical stroke recognition applications. A more interesting quality of the AQ feature space is *multi-scale* invariance. For an illustration, consider the strokes in Fig. 6. Based on Fourier features, S_1 and S_3 are more similar, but intuitively S_1 and S_2 are more similar because S_1 consists of several smaller copies of S_2.

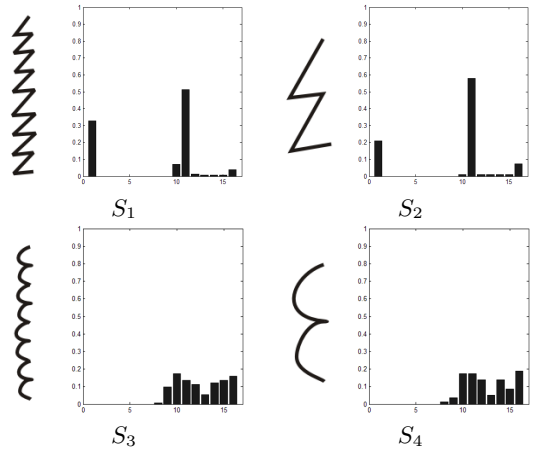


Figure 6: The AQ features are similar for multi-scale patterns.

Suppose S_2's feature is $\hat{Q}_2 = \langle \sigma_1, \ldots, \sigma_k \rangle$. If S_1 contains m copies of S_2, each scaled by $\frac{1}{n}$, then $\hat{Q}_1 = m \cdot \langle \frac{1}{n}\sigma_1, \ldots, \frac{1}{n}\sigma_k \rangle$, or $\hat{Q}_1 = \frac{m}{n}\hat{Q}_2$. After normalization, $Q_1 = Q_2$. Thus, AQ features match our intuition that S_1 and S_2 are similar.

Rotational (In)variance

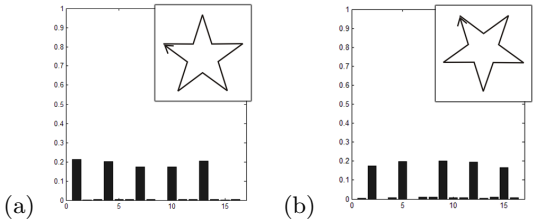


Figure 7: (a) The AQ feature of a star; (b) when the shape is rotated, the feature is shifted.

Scale and translational invariance are necessary properties, but the desirability of rotational invariance is application-dependent. For pure shape recognition it would be sensible, but orientation may be important in gestural interfaces (eg. different orientations of a stroke may specify variations of a single operation or even different operations entirely).

AQ features are not rotationally invariant. Consider Fig. 7: the feature after rotation is different than the original, because potentially every stroke vector falls into a different angle range. The effect of rotation is predictable, though: a rotation of $\pm 2\pi/k$ corresponds to a shift of ± 1 element in Q. A rotation by a non-integral multiple of $2\pi/k$ could result in an "impure" shift due to only some vectors changing bins, so k should not be chosen too small. To have rotational invariance, then, the similarity measure can be modified to

$$d_\theta(Q_1, Q_2) = \min_{i \in [1,k]} \{|Q_1 - \text{shift}(Q_2, i)|_2 + p(i)\} \ ,$$

where *shift* performs an integral shift-with-wraparound and $p(i)$ is an optional function to penalize the rotation.

Directional independence can also be realized by shifting, as a stroke reversal is equivalent to each stroke vector being rotated by π. For example, if the stroke of Fig. 5a were reversed, the feature in Fig. 5d would be shifted by 4 elements, i.e. $\hat{Q} = \langle 0, 5, 1, 0, 2, 0, 0, 0 \rangle$. In general, a reversal is equivalent to a shift by $k/2$ elements (for even k).

4. EVALUATION

To evaluate the utility of AQ features in gesture recognition, we compare it against the FT, WD, and Loci methods in a typical usage scenario: matching an input stroke against a set of known set of gestures. For the set of gestural inputs shown in Fig. 8, we acquired a training set R with 204 gestures, and a query set U of 49 strokes. The sets were acquired from 3 users and exhibit common input variations, such as scale, speed, and starting point (eg. a circle is not always drawn from the top).

For each gesture type, we construct a template by averaging the features of all training examples from that type. Methods such as Hidden Markov Models [17] or Bayesian statistics [4] are often used to isolate outliers in the training data, but we were able to achieve good results without them. Figure 8 illustrates the templates for each method.

Recognition success is measured by finding the nearest template to each query stroke and comparing the result to the ground truth; the success rate is the number of correctly recognized strokes, N_C, divided by the total, $|U|$. The time required to recognize all strokes in the query set is also reported for each method. Table 1 summarizes the results of these experiments.

Method	**FT**	**WD**	**Loci**	**AQ**
Success (%)	75.5	79.6	95.9	100
Time (s)	1.49	2.58	6.65	1.28

Table 1: Gesture recognition success rate and timing for all strokes in the query set U. The feature dimensions are as indicated in Fig. 8.

The AQ method is able to recognize all query strokes successfully, and does so in the least amount of time. Looking at the templates, we can see that each gesture produces a very unique template: for example, the square's feature has four peaks, while the circle's has none. This diminishes the chance of misclassification, as even a poor example of a gesture should be closer to its template than any other.

The Loci method is able to achieve a high recognition rate as well although we notice that it has difficulty distinguishing between closed convex objects, such as the circle and the square. Each interior point in these objects has a single intersection in every direction, and so the corresponding code tends to dominate the features.

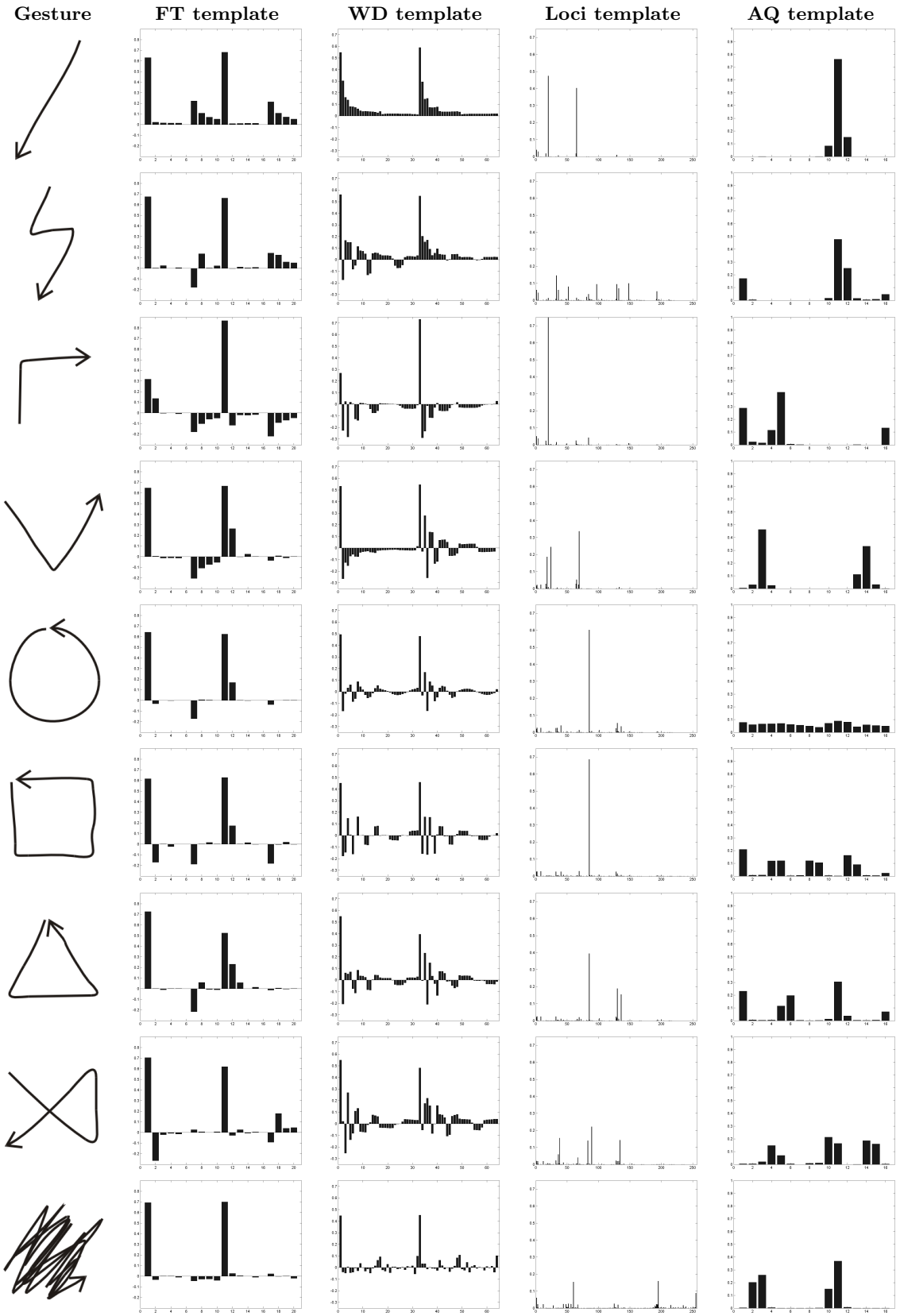


Figure 8: A set of gestures (left) and their corresponding feature templates for the FT, WD, Loci, and AQ methods. Feature dimensions are 20 ($m = 5$), 64 ($n_c = 1$,$n_d = 31$), 256 ($b = 2$), and 16 ($k = 16$), respectively.

The FT method is quite efficient, but provided relatively poor recognition. Observation reveals that most templates are dominated by the zero-th frequency components, which makes the method prone to misclassification because the templates for each gesture type are too close to reliably distinguish between them. The WD method achieves a recognition rate close to FT. Like the Loci method, closed objects produce similar features in the WD method, and are likely to be confused. The FT and WD methods are both sensitive to the initial position of a stroke.

The efficiency of the Loci method is hindered by two aspects: stroke rasterization, and high-dimensional features. The WD method also requires high-dimensional features, and though wavelet decomposition is more efficient than Fourier analysis ($O(n)$ versus $O(n \log n)$), this gain is offset by the resampling step. FT and AQ are fast because there is an efficient mapping from stroke to feature, and because they can use low-dimensional features.

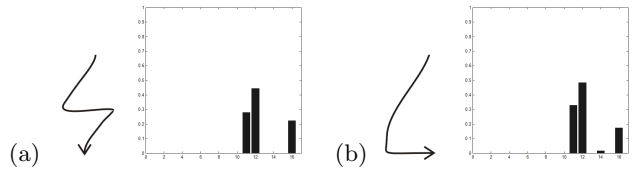


Figure 9: Loss of locality in the AQ method: the features of (a) and (b) are almost the same.

These experiments indicate that our AQ method is well-suited to simple gesture recognition tasks. There are some drawbacks to the method that could hinder its usefulness in other applications, though. First, the locality of stroke features is lost, allowing possible ambiguities in unconstrained input (Fig. 9) that can be avoided in gesture recognition. Second, the transform's coordinate independence may be undesirable in domains such as trajectory analysis.

5. CONCLUSION & FUTURE WORK

In this paper, we have presented a method for extracting low-dimensional features from arbitrary-length point sequences based on angle quantization. The feature is efficient, descriptive, and offers useful geometric properties such as multi-scale and rotational invariance.

In a gesture recognition experiment, low-dimensional AQ features were shown to offer high recognition rates and fast operation, outperforming Fourier, wavelet, and area-based techniques.

For unconstrained recognition tasks, the angle quantization technique may suffer from ambiguities. A future direction of this work is investigating possible disambiguations, such as augmenting the AQ feature with speed information. Adding spatial information, eg. start and end points, to the feature could also make the feature suitable for coordinate-dependent applications like trajectory analysis.

Finally, more sophisticated classification tools could be used within our feature space to achieve better performance. Improvements could be made in both the template construction (eg. [4, 17]) and in the distance metric (eg. [5, 22]).

Acknowledgments

The authors would like to thank Min Xin, Ehud Sharlin, and the anonymous reviewers for their helpful comments. This work was supported by the National Science and Engineering Research Council of Canada.

6. REFERENCES

[1] R. Agrawal, C. Faloutsos, and A. Swami. Efficient similarity search in sequence databases. In *Proc. of the 4th International Conference of Foundations of Data Organization and Algorithms*, 1993.

[2] C. Alvarado and R. Davis. Sketchread: a multi-domain sketch recognition engine. In *UIST '04: Proc. of the 17th annual ACM symposium on User interface software and technology*, pages 23–32, 2004.

[3] K.-P. Chan and A. Fu. Efficient time series matching by wavelets. In *Proc. of 15th International Conference on Data Engineering*, 1999.

[4] K. Cheung, D. Yeung, and R. Chin. A bayesian framework for deformable pattern recognition with application to handwritten character recognition. *IEEE Tranactions on Patter Analysis and Machine Intelligence*, 20(12):1382–1388, 1998.

[5] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines*. Cambridge University Press, 2000.

[6] W. Freeman and M. Roth. Orientation histograms for hand gesture recognition. In *IEEE Intl. Wkshp. on Automatic Face and Gesture Recognition*, 1994.

[7] H. Glucksman. Classification of mixed-font alphabetics by characteristic loci. In *1967 Digest of 1st Annual IEEE Computer Conference*, 1967.

[8] T. Igarashi. Freeform user interfaces for graphical computing. In *Lecture Notes in Computer Science*, volume 2733/2003, pages 39–48. Springer, 2003.

[9] L. Kara and T. Stahovich. An image-based, trainable symbol recognizer for hand-drawn sketches. *Computers & Graphics*, 29(4):501–517, 2005.

[10] N. Kato, M. Suzuki, S. Omachi, H. Aso, and Y. Nemoto. A handwritten character recognition system using directional element feature and asymmetric mahalanobis distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21:258–262, 1999.

[11] A. Knoll. Experiments with characteristic loci for recognition of handprinted characters. *IEEE Transactions on Computers*, C-18, 1969.

[12] W. Lee, L. Kara, and T. Stahovich. An efficient graph-based symbol recognizer. In *2006 Eurographics Workshop on Sketch Based Interfaces and Modeling*, 2006.

[13] J. Li, M. Özsu, and D. Szafron. Modeling of moving objects in a video database. In *Proc. of the 1997 Intl. Conference on Multimedia Computing and Systems*, pages 336–343, 1997.

[14] S. Mori, C. Suen, and K. Yamamoto. Historical review of ocr research and development. *Proceedings of the IEEE*, 80, 1992.

[15] A. Naftel and S. Khalid. Motion trajectory learning in the dft-coefficient feature space. In *IEEE International Conference on Computer Vision Systems 2006 ICVS '06*, 2006.

[16] A. Oppenheim and R. Schafer. *Digital Signal Processing*. Prentice-Hall, 1975.

[17] G. Rigoll, A. Kosmala, J. ROttland, and C. Neukirchen. A comparison between continuous and discrete density hidden markov models for cursive handwriting recognition. In *Proceedings of International Conference on Patter Recognition*, volume 2, pages 205–209, 1996.

[18] D. Rubine. Specifying gestures by example. In *SIGGRAPH '91: Proceedings of the 18th annual conference on Computer graphics and interactive techniques*, pages 329–337, 1991.

[19] T. M. Sezgin, T. Stahovich, and R. Davis. Sketch based interfaces: early processing for sketch understanding. In *PUI '01: Proceedings of the 2001 workshop on Perceptive user interfaces*, pages 1–8, 2001.

[20] D. Sharon and M. van de Panne. Constellation models for sketch recognition. In *EG SBIM 2006: Eurographics Workshop on Sketch Based Interfaces and Modeling*, 2006.

[21] C.-B. Shim and J.-W. Chang. Spatio-temporal representation and retrieval using moving object's trajectories. In *MULTIMEDIA '00: Proceedings of the 2000 ACM workshops on Multimedia*, pages 209–212, 2000.

[22] J. Ye, Z. Zhao, and H. Liu. Adaptive distance metric learning for clustering. In *Proceedings of Computer Vision and Pattern Recognition 2007*, pages 1–7, 2007.