A Method for Calculating Similarity among Glyphs using Cross Correlation

Tao He[1], Jie Feng[1], Bingfeng Zhou[1], Zhigang Fan[2], Francis Kapo Tse[2]
1. Institute of Computer Science and Technology, Peking University, Beijing, China
2. Xerox Corporation, Norwalk, CT, USA

Abstract

In this paper, a method for measuring similarity among glyph contours is proposed. A predominance measure of each contour point is defined and computed at the first place, then we adopt cross correlation between glyph contour pairs as the metric of similarity. The proposed method has several attractive properties: firstly, it is invariable to scales and rotations. Secondly, it is automatically normalized. Thirdly, it can be accelerated by FFT algorithm. These characteristics are of great use in many document image processing procedures, especially in applications where small deviations or rotations is a commonplace. Experiments show that our method can effectively figure out degree of similarity among glyph contours.

I. Introduction

The project stems from a text vectorization task, where the objective is to represent glyphs of a character in mathematical curve forms. In the course of this application, similarity computation among glyphs is needed as an intermediate processing. In this paper, we employ cross correlation as a metric to weigh the degree of similarity among glyphs.

Cross correlation is a basic concept in image processing. It often serves as an image similarity metric. A typical application is to locate the best match portion between two images. It has also found applications in a broad range of computer vision, medical image processing tasks such as object recognition, clustering, image matching, etc. [5-7] In medical image processing, a basic image similarity-based method consists of a transformation model which is applied to the reference image coordinates to locate their corresponding coordinates in target image space. Cross correlation is the simplest but effective measure that well fits in this kind of problems. It has several advantages over other commonly used similarity metrics such as mutual information, mean square difference [8, 9], etc.

- It is automatically normalized between [0,1], with 1 denotes completely same signals;
- It can compare signals of different size, which is a commonplace in many applications;
- Cross correlation is invariant to rotation and small shift of input signals.
- The normalized cross correlation algorithm can be accelerated using FFT.

The other aspect of this problem to capture the shape features of the glyphs, which is served as inputs of cross correlation. Glyphs of a character could be represented in many forms, such as chain-code, polygon approximation, signatures, boundary segments and skeletons. Because the primary focus of vectorization for text is on the representation of the shape characteristics and the construction of a feature function that reflects these shape characteristics, chain-code representation is a handy choice [1, 2]. The research on chain-code based method was pioneered by Freeman [1] and thereafter many researchers have done great effort to extend this method. Bribiesca [3] gives a thorough study on recent development of chain code scheme and its applications. Freeman [1] states in general that a coding scheme for any object contours should satisfy three objectives: (a) it must faithfully preserve the information of interest; (b) it must permit compact storage and convenient for display; and (c) it must facilitate any required processing and manipulation.

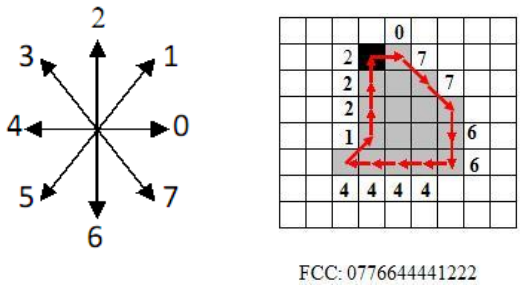
The proposed algorithm creatively adopt cross correlation as the similarity measure among glyph contours and it defines a metric called cornerity as the signal that captures the shape characteristics of glyphs. The algorithm works as follows: we obtain binary data from the input image by using optimum automatic thresholding methods and then glyph contours are identified with a contour tracing algorithm that uses an 8-connective path template to link boundary pixels. While the glyph contours are obtained, we define and compute a metric that faithfully describes the intrinsic feature of glyph contours, which is named as the cornerity [4]. It is then served as the cross correlation signal.

The paper is organized as follows. In section II, we introduce chain code representation of glyph contours. In section III, the definition and computation of cornerity for contours is described in detail. Section IV focuses on cross correlation, explaining why and how it could serve as a reasonable similarity metric for glyph contours. Experimental results and are given in section V and conclusions are drawn in section VI.

II. Chain-Code Representation of Glyph

In the 8-connected encoding scheme introduced by Freeman [1], a link denotes the direction between two points (pixels). These links are marked with digits {0, 1,

2, 3, 4, 5, 6, 7} as shown in Figure 1a. Each link can be considered as the angular direction, in multiples of 45 degree that we must move to go from one contour pixel to the next. The contour tracking algorithm begins by finding a start pixel (black pixel in figure 1b), and traverses the object boundary along the 8-connective path. Coordinates of boundary pixels are recorded in a clockwise (or counterclockwise) direction and the algorithm terminates when it returns to the start point. Figure 1b shows an example of the freeman chain code (FCC) with 8-connected path template.



a *b*

Figure 1. a: 8-connected freeman chain code convention
b: an 8-connected freeman chain code example

III. Cornerity: Intrinsic Characteristic of Glyph

Once we have obtained the chain-coded glyph contours, the next step is to find a metric that embodies each contour's intrinsic features. We call this metric ***cornerity***.

Assume that we have got the simple closed ***n***-link chain codes: $A_n = C_{i=1}^{n} a_i = a_1 a_2 \cdots a_n$. (1)

1. Define L_{is} as the moving line segment spanning ***s*** chain links and terminating on the node to which the link is directed: $L_{is} = \{a_j, j = i-s+1, \cdots, i\}$ (2)

The ***x*** and ***y*** component of L_{is} are given by

$$X_{is} = \sum_{j=i-s+1}^{i} a_{jx}, \qquad a_{jx} \in \{-1,0,1\}.$$
$$Y_{is} = \sum_{j=i-s+1}^{i} a_{jy}, \qquad a_{jy} \in \{-1,0,1\}. \quad (3$$

Where a_{jx} , a_{jy} are the ***x*** and ***y*** components, respectively, of the chain link a_j .

The angle L_{is} makes with the ***X***-axis is given by

$$\theta_{is} = \begin{cases} \tan^{-1}\frac{Y_{is}}{X_{is}}, & if\,|X_{is}| \geq |Y_{is}| \\ \cot^{-1}\frac{X_{is}}{Y_{is}}, & if\,|X_{is}| < |Y_{is}| \end{cases} \quad (4$$

2. Define the incremental curvature δ_{is} as twice the mean over two adjacent angular differences.

$$\delta_{is} = 2\left[\frac{(\theta_{i+1s} - \theta_{is}) + (\theta_{is} - \theta_{i-1s})}{2}\right] = \theta_{i+1s} - \theta_{i-1s} \quad (5)$$

The incremental curvature is a smoothed version of point's curvatures on the curve; the larger value of ***s,*** the heavier the smoothing. It is recommend to set a number ranges from 5 to 11 as the value of ***s*** [4].

3). Define the forward and backward arms [4], t_{i1} and t_{i2}, as the lengths(or number) of contour points to either side of the current contour point, with incremental curvature fluctuates in a small margin :

$$\Delta = \tan^{-1}\frac{1}{s-1} \quad (6)$$

4). **Cornerity** is defined as follows:

$$K_i = \sqrt{t_{i1}} \times \sum_{j=i}^{i+s} \delta_{js} \times \sqrt{t_{i2}} \quad (7)$$

Informally, cornerity can be considered as a variable that reflects the sharpness of a contour point. The sharpness of a contour point consists of two components. One is the incremental curvature at this point, which reflects the abruptness of discontinuity along the contour points. The other component is its influence on neighboring contour points, which were defined as forward and backward arms.

Freeman and Davis dubbed in their work [4] a way to detect corner points in planar curve. It could be as well used to compute cornerity of contour points. The algorithm works as follows.

Firstly, scanning the glyph contour with a moving line segment, L_{is} , which connects the endpoints of ***s*** consecutive links. The amount of smoothing that is done to the curve increases with the number of links, ***s***, which is used to initialize the scanning line segment.

The moving line segment creates an angle with the ***X***-axis. The incremental curvature is then formulated as twice the mean over two adjacent angular differences. With the line segment spanning through all the glyph contour points, each point's incremental curvature is recorded.

Cornerity for each contour point is compute according to Equation 7. Figure 2 illustrates the cornerity of a glyph contour for alphabetic letter 'E'. The horizontal direction shows the points on glyph contours and vertical direction represents each contour point's cornerity value. The mono-dimensional signal has a size that equals to the points in contours, and the magnitude of each contour point in the signal is expressed by the metric defined in the past paragraphs.

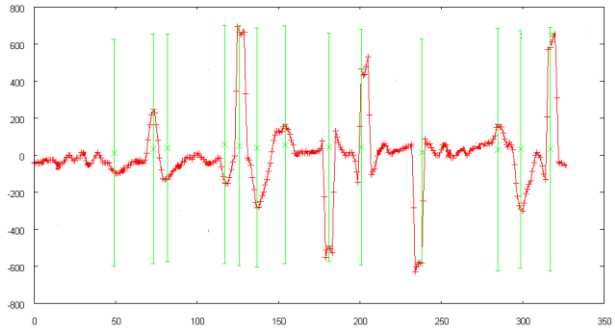


Figure.2 Cornerity value of contour points for letter 'E'

IV. Cross Correlation

Distance between two sequences is one of the most common measures used in computer algorithms for sequence analysis [8]. It has found applications in various areas. However, former researches have shown that most distance metrics are neither robust to small shape deformations of sequences nor to shifts on the indexing axis [6]. Cross correlation is a standard method for estimating the degree to which two signals are related (the more two signals are related, the less the distance between them). It is an intuitive metric for similarity measure and has already been applied to image classification and object recognition [5 - 7].

In section III, we have obtained the Cornerity of glyph contours. It could be views as a mono-dimensional signal. In this section, we will apply cross correlation as a similarity measure for these signals, and show the advantages it has over other distance metrics.

Let κ_a, κ_b be the signal (cornerity) of contour ***a*** and ***b*** respectively, and ***i*** is a parameter that represents the relative shifts between the two signals. The cross correlation function between two glyph contours can be expressed by Equation 8.

$$C_{ab}(i) = \frac{\sum_{l=1}^{n} \kappa_a(l+i)\kappa_b(l)}{\sqrt{\sum_{l=1}^{n} \kappa_a(l)^2} \sqrt{\sum_{l=1}^{n} \kappa_b(l)^2}} \qquad (8)$$

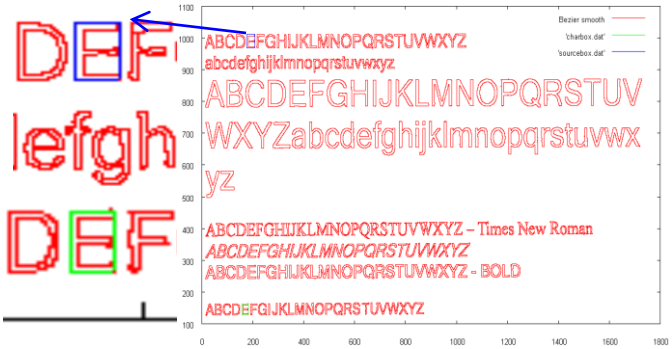
This metric computes pixel-wise cross correlation and normalizes it by the square root of the auto-correlation of the signals. More difference between the signals results in smaller measuring values, and vice versa. There are several advantages of using cross-correlation metric to measure the similarity (distance) between two glyph contours:

1. Normalization. The cross-correlation function is normalized to [0, 1] automatically. When two contours are the same, the cross-correlation value equals to 1. This value decreases when the similarity between the two contours decreases.
2. Invariable to contour size. Number of points on contours is usually different from each other; even the same alphabetic letter could have different contour representation due to digitization errors. Using cross correlation as the similarity metric, these differences make no effects on the result.
3. Invariable to rotations. Since the glyph contour adopts cornerity as its decisive feature, it doesn't depend on the absolute locations of each contour. Instead, it depends on the relative location of contour points.
4. Fast computation. Cross correlation resembles convolution in expression form. While it is well-known that a convolution in the time domain is equivalent to a multiplication in the frequency domain after a Fourier transformation, we can use Fast Fourier Transform (FFT) to calculate the cross-correlation function. When the number of contour points is large, the improvement will show great efficiency.

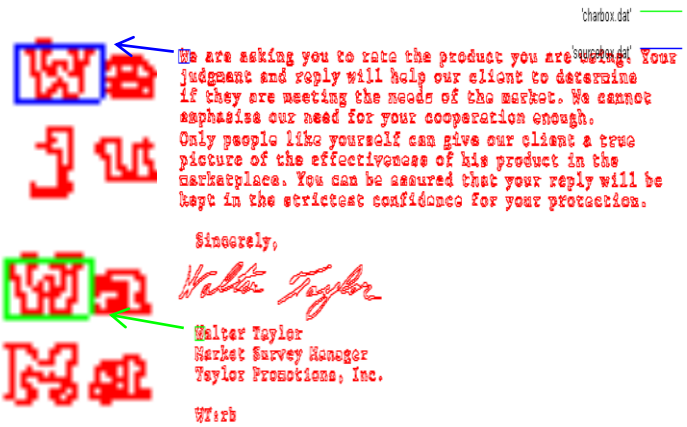
V. Experimental Results

Given a scanned document, we first do some preprocessing (denoisy, binarization, etc), and then select a character as template, surrounded by blue box in Figure 3. Our goal is to find a character that is most similar to the template.

Figure 3a shows the results of applying the proposed method to a high quality (600dpi) scanned document. In this figure, there are alphabetic letters of different fonts (Arial, time new roman, courier new), different sizes (12pt, 14pt, 28pt), and different styles (common, italic, bold). The selected letter (template) is Arial, 14pt, common, as shown in the enlarged portion of figure 3a.



a. High quality document



b. Low quality document

Figure 3. Test results for the algorithm. Template is surrounded by blue box, and the green box indicates a letter that is most similar to the template. Right part of the figure shows the whole view of the test results and left side portion of the figure show the magnification of our interest regions. They two part are linked together by arrows.

The algorithm first extracts the glyph's contour, and then computes cornerity for each contour point. After that, cornerity of all contour points is forming a one dimensional signal which is served as the input signal for cross correlation. Then, the cross correlation value is computed between each contour and the template. The

letter which is most similar to the template is bounded by green box, which is Arial, 12pt, common.

From the experimental results, we can find that font and style variance makes the contour vary more than other factors, especially different in the relative position of contour points. The cornerity of the contour (or signal) is deviate even further from the template and then results in a much small cross correlation value.

Figure 3b shows another test document with relatively low scan quality (200dpi), and small font size (8pt). The experimental result shows that the algorithm still works well even under such an unfavorable circumstance. In this respect, it is an effective and stable algorithm.

VI. Conclusions

To sum up, this paper provides a simple and robust procedure for calculating similarity among glyph contours and experimental results are given to illustrate the effectiveness and stability of the proposed algorithm. The algorithm integrates object representation, feature description, and similarity metric choice together gracefully. Experimental results show that this method is effective and rather intuitive, and it works well on both high quality and low quality scanned document samples.

Acknowledgements

This work is supported by the National Natural Science Foundation of China. (Grant NO. 60973054)

References

1. H. Freeman, On the Encoding of Arbitrary Geometric Configurations, IRE Trans. Elec. Computer, EC(10) , pp. 260-268. (1961).
2. H. Freeman, Computer Processing of Line-Drawing Image, Computer Surveys, no. 6, pp. 57-97. (1974).
3. E. Bribiesca, A new chain code, Pattern Recognition, vol. 32, no. 2, pp. 235-251. (1999).
4. H. Freeman and L.S. Davis, A Corner Finding Algorithm for Chain-coded curves, IEEE Trans. Comput. C-26, 297-303. (1977).
5. Yifeng, Wu, Hudson, Kevin, An Image Clustering Method Based on Cross-Correlation of Color Histograms, Proc. SPIE, Vol. 5682, 204 (2005)
6. F Zhao, Q Huang, W Gao, Image Matching by Normalized Cross-Correlation, 2006 IEEE International Conference on Acoustics, (2006)
7. K Briechle, D Hanebeck, Template matching using fast normalized cross correlationuka, Proceedings of SPIE, (2001)
8. J.P. Lewis, Fast Normalized Cross Correlation, In Vision Interface , pp. 120-123.(1995)
9. G.P. Penney, J. Weese, J.A. Little etc., a Comparison of Similarity Measures for Use in 2D-3D Medical Image Registration, MICCAI'98, vol. 1496. Pp. 1153-1161(1998)

Biography

Tao He received his B.E. degree in Software Engineer from Harbin Institute of Technology, China in 2007. He is currently pursuing a M.S. degree in Computer Applications at Peking University, China. His research interest is in document image processing, including binarization, object segmentation, corner detection and image vectorization.

Jie Feng is an assistant researcher of the Institute of Computer Science & Technology at Peking University. She earned her PH.D. from the Center for Information Science at Peking University in 2005. She has been concentrating on computer vision, digital geometry processing, image-based modeling and rendering since her graduation. Her research interests include 3D mesh model compressing, multi-resolution modeling, image-based modeling and relighting.

Bingfeng Zhou, Ph.D., professor, Ph.D. supervisor. His research interests include graphics simulation of robot kinematics, geometry models and CAD/CAM, color image processing, multimedia system and image special effects, digital image halftone, image based rendering and modeling, virtual reality, etc.