IMPROVED CURVE TRACING IN IMAGES

Karthik Raghupathy and Thomas W. Parks

Dept. of Electrical and Computer Engineering
Cornell University, Ithaca, NY 14853
{kr83, twp3}@cornell.edu

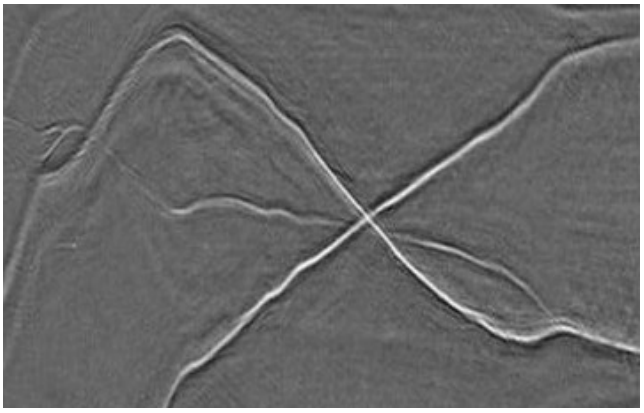
ABSTRACT

This paper begins with a description of the curve tracing problem and compares it with curve extraction, a fundamentally different problem. Issues concerning curve tracing are addressed. We discuss a good curve tracing algorithm proposed by Carsten Steger and improve this method to suit the goals of curve tracing. Results show that the improved algorithm emulates human perception by using both global and local features.

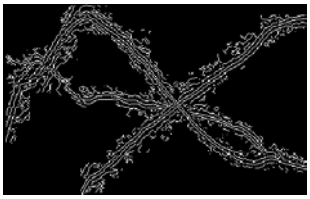
1. INTRODUCTION

Curve tracing is the operation of extracting **and** automatically classifying curves in images. A person can trace several curves simply by looking carefully at the image. For example, it is easy to visually trace three curves shown in Figure 1(a). Curve tracing has many applications in computer vision and image processing. In aerial images, curve tracing is used to extract curvilinear features such as roads, rivers and railroads [1]. Curve tracing is also used in medical imaging for the extraction of anatomical features such as blood vessels from an X-ray angiogram [2].

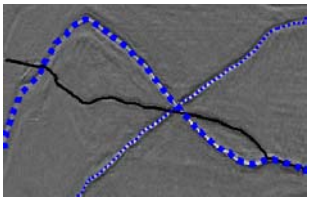
Curve tracing, however, is different from traditional curve extraction where curves are extracted but not necessarily classified. In curve extraction [3, 4, 5], the emphasis is on the extraction and not on the identification or labeling of a curve. A Canny edge detector [5] is a simple curve extraction algorithm. Figure 1(b) shows the result of curve extraction using a Canny edge detector. Many multi-resolution algorithms [3, 4] developed recently, such as curvelets and beamlets are very effective curve extraction algorithms and are used for compressing images with curves because they preserve the curves in a compressed image. Beamlets [3] are a collection of line segments at a range of locations, orientations, and scales, that give a multi-scale approximation to the collection of all line segments in an image. Polygonal curves in the $x - y$ plane are then built up by chaining beamlets together. Similarly, Curvelets [4] are a multiscale approach used to represent objects with edges and curves. These methods work better than edge detection algorithms when dealing with noisy images. However, all these methods would result in curves extracted but not labelled. These methods cannot be applied to the problem of curve tracing as they only extract or detect curves in an image. We want to identify an individual curve, label it, and follow it across the image, even if it crosses other curves, fades out, and then reappears. The result that we expect from a curve tracing algorithm is one that matches the identification done by a person. The desired result is shown in Figure 1(c).



(a) Original image



(b) Result of curve extraction



(c) Desired result

Fig. 1.

A good curve tracing algorithm, proposed by Steger [6], uses a local approach to curve tracing. It is local in the sense that the algorithm classifies pixels as belonging to a

This work is supported by Lockheed Martin Naval Electronics and Surveillance Systems and by Center for Electronic Imaging Systems

curve or not, and then links curve points together. Section 2 describes this algorithm briefly. The result of curve tracing using this approach is shown in Figure 2. As can be observed, some curves in the image have been traced but have not been identified as a person would identify them. Curve orientation at junction points is not accurate and this leads to wrong-curve following. The term wrong-curve following refers to the fact that the algorithm follows a curve at a junction contrary to how the human eye would do so. This is illustrated in Figure 2 where the algorithm follows the big dotted and the small dotted curves erroneously at the junction. We also observe that the two solid curve segments running horizontally through the image are perceptually just one curve that fades out and emerges again. However, the algorithm traces this single curve as two separate curves. Another problem is that the algorithm does not trace out the faint curves in the upper left portion of the image. These problems arise from the local nature of the algorithm. As a result, the algorithm cannot track faint curves that fade out or disappear and emerge again. In addition, at junctions the algorithm does not use any global information to perform curve tracing.

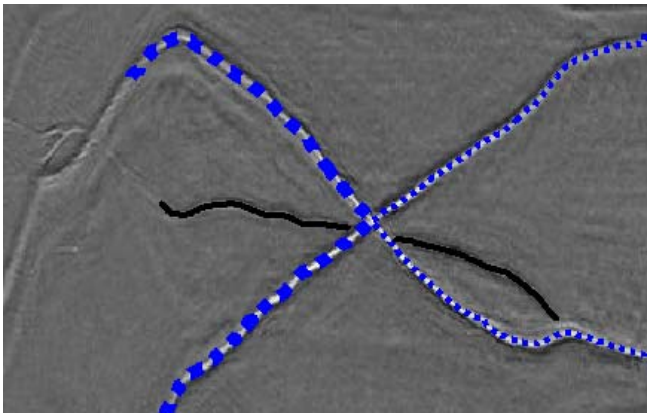


Fig. 2. Result of curve tracing using [6]

There are several issues that need to be taken into account when developing a curve tracing algorithm. Some of them include (i) Noisy images (ii) Multiple and possibly intersecting curves (iii) Disappearance and re-emergence of a curve (this is perceptually the same curve) and (iv) Tracking curves that fade out. In this paper, we present modifications of the local algorithm [6] by incorporating global features to address the above mentioned issues.

2. LOCAL METHOD

We begin with a method described by Steger [6] and modify the algorithm suitably. In accordance with Stegers method, the first step involves classifying pixels as being curve points or not depending on whether they lie on some curve in the image. The next step is the linking of curve points found in the first step to form curves.

2.1. Classification of curve points

Curvilinear structures in a 2D image can be modelled as curves $s(t)$ that exhibit a characteristic 1D line profile in the direction perpendicular to the line (refer Figure 3). The 1D line profile is characterized by a vanishing first derivative. Let the direction perpendicular to the curve be $n(t)$. Thus, at a curve point, the first directional derivative in the direction $n(t)$ should vanish and the second directional derivative should be of large absolute value. A pixel in an image is classified as a curve point if the first derivative along $n(t)$ vanishes within a unit square centered around the pixel. The direction, $n(t)$, is computed by finding the eigenvector corresponding to the maximum absolute eigenvalue of the Hessian matrix. The Hessian matrix consists of partial derivatives of the image and is computed after convolving the image with a Gaussian smoothing kernel [6].

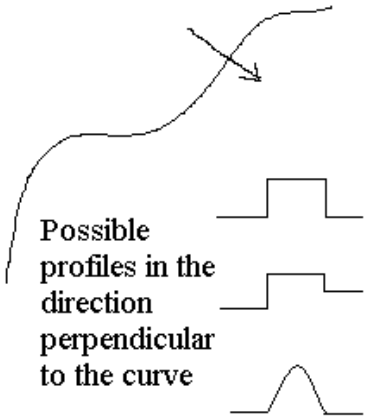


Fig. 3. Classification of curve points

2.2. Linking of curve points

Linking proceeds by using the orientation information obtained in the classification step and joins curve points appropriately to form curves. Given a curve point, three neighboring pixels compatible with the current orientation of the curve are examined. The choice of the next curve point to be added is based on the distance between sub-pixel locations and the angle difference of the orientation of the curve at the two points. The linking terminates when there are no more curve points in the neighborhood of the current pixel.

3. IMPROVEMENTS - INCORPORATING GLOBAL FEATURES

As mentioned earlier, the local method suffers from two main problems (i) Curve following at junctions and (ii) Tracing faint curves that fade out or disappear and re-emerge again. The ideal curve tracing algorithm should emulate human perception. To do so, we have to remember that a human observer has an overall perception (a global view) as well as a local perception of an image. Thus the ideal curve tracing algorithm has to incorporate features from both the local as well as the global structure. We propose the following improvements to the linking algorithm of [6] to make it more global in nature.

3.1. Curve following at junctions

The linking algorithm proposed by Steger produces results that lead to wrong-curve following at junctions. To correct this problem, we modify the algorithm to disallow large bending of curves at junctions. Wrong-curve following at a junction is characterized by a large change in the curve orientation. Thus, when a large change in orientation of the curve is encountered while linking, the modified algorithm prevents this by looking ahead in the current direction and finding a more suitable curve point to be added to the existing curve. Looking ahead is done by using the curve orientation obtained prior to the junction and continuing on in this direction to find a more "suitable" curve point. A suitable curve point is defined as that which minimizes the angle difference between the curve orientation prior to arriving at the junction and the curve orientation just after the junction. This emulates human perception of intersecting curves and leads to accurate curve following at junctions.

3.2. Tracing faint curves

Steger's linking algorithm terminates when there are no more curve points in the immediate vicinity of the current curve point. This is clearly too local and hence the algorithm cannot track curves that (i) fade out or (ii) disappear and re-emerge but are perceptually the same curve.

3.2.1. Curves that fade

In order to tackle the problem of tracing curves that fade out (e.g. the faint curves in the upper left portion of the image), we use ideas from the Radon transform [7, 8]. The Radon transform is a global approach to curve extraction. Curves that are to be extracted are modelled and this is followed by calculation of the inner product of the image with all possible curves represented by the model. A particular example is the first order Radon Transform [9] that extracts lines out of an image. Salient curves are extracted from the image by locating inner product maxima in the parameter space. The Radon transform can be used effectively to extract faint curves from noisy images. Radon's original paper [8] deals with the extension of the Radon Transform to higher order polynomials and other general functions. In related work [10], we show how to use the Radon transform for curve tracing.

We modify the linking algorithm as follows; Once a curve has been declared to have ended (i.e. there are no more curve points in the vicinity), we look for the possibility of a faint curve by calculating the inner product of the image with various curves (we chose straight lines) starting from the last known curve point. Let θ_o be the average orientation of the curve just prior to it being declared as terminated. We use this average orientation information to limit our search over inner products. That is, we only compute the inner product of the image with lines whose orientations belong to the set $[\theta_o - \theta, \theta_o + \theta]$ where 2θ defines the size of the interval over which we search for the maximum inner product. If the maximum inner product is higher than a certain threshold, then this would indicate the presence of a curve that is fading out. As a result, we can trace a curve even if it fades.

3.2.2. Disappearance and re-emergence of curves

We modify the linking algorithm by predicting the possible path of a curve once it has been declared to have ended. This is done again my making use of the curve extracted prior to it being declared as terminated. Thus, when the algorithm is not able to find any curve points in the vicinity of the current curve point, three possibilities arise: (i) The curve actually terminates or (ii) The curve fades (this case has been tackled in the previous section) or (iii) The curve disappears and re-emerges later. To account for these possibilities, we first determine the possible path of the curve by using the average orientation of the curve just prior to it being declared as terminated. Figure 4 illustrates this process. Having found the average orientation of the curve over the last few pixels, it is extrapolated in this average direction to give the possible path of the curve. If the curve actually fades out and re-emerges, then the end of the extrapolated curve should be near the re-emergence of the curve. Thus, a search for a suitable curve point is done in the neighborhood of the end of the predicted curve. A suitable curve point is defined using two metrics (i) The difference in curve orientation at that point and the average orientation extracted as above and (ii) The difference between the strength of the curve at that point and the average strength of the curve nearby. In doing so, we are trying to find a curve whose characteristics are similar to the one extracted hitherto. If no suitable curve point is found, then the curve is declared as terminated.

The results of the above mentioned improvements are

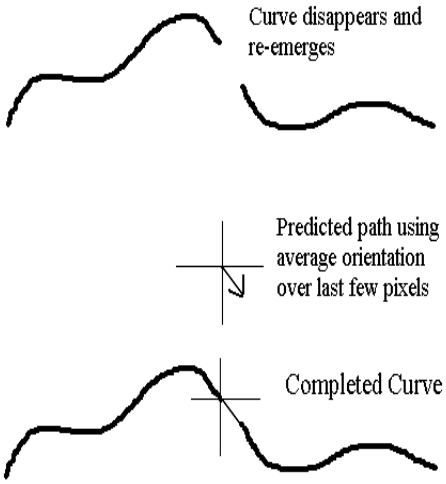


Fig. 4. Tracing re-emergent curves

shown in Figure 5. Note that the problem of wrong-curve following at junctions and the problem of tracing faint curves have been solved for this example.

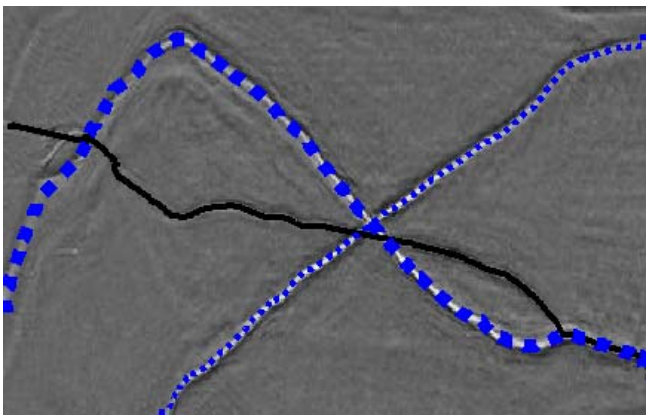


Fig. 5. Results of improved curve tracing algorithm

4. CONCLUSIONS

An improved curve tracing algorithm that incorporates global features in a local algorithm has been proposed. The problem of wrong-curve following at junctions has been solved by disallowing large bending of curves. To trace a curve that disappears and re-emerges, we look ahead and predict the possible path of the curve when it disappears. The problem of tracing curves that fade out has been solved by using ideas from the Radon Transform. Comparing Figure 5 and Figure 2, we observe that the proposed algorithm results in improved curve tracing.

5. ACKNOWLEDGEMENTS

The authors would like to thank Dr. Thomas J. Barnard from Lockheed Martin for introducing us to this problem and for his helpful suggestions towards solving this problem.

6. REFERENCES

[1] A. Baumgartner, C. Steger, C. Wiedemann, H. Mayer, W. Eckstein, and H. Ebner, "Update of roads in gis from aerial imagery: Verification and multi-resolution extraction," *International Archives of Photogrammetry and Remote Sensing*, 1996.

[2] C. Coppini, M. Demi, R. Poli, and G. Valli, "An artificial vision system for x-ray images of human coronary arteries," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, February 1993.

[3] D. Donoho and Xiaoming Huo, "Applications of beamlets to detection and extraction of lines, curves and objects in very noisy images," in *Proceedings of NSIP*, 2001.

[4] E. J. Candes and D. L. Donoho, *Curvelets - a suprisingly effective nonadaptive representation for objects with edges*, Vanderbilt University Press, 1999.

[5] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, November 1998.

[6] Carsten Steger, "An unbiased detector of curvilinear structures," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, February 1998.

[7] Peter Toft, *The Radon Transform - Theory and Implementation*, Ph.D. thesis, Informatics and Mathematical Modelling, Technical University of Denmark, 1996.

[8] S. R. Deans, *The Radon Transform and some of its applications*, John Wiley and Sons, 1983.

[9] A. K. Jain, *Fundamentals of Digital Image Processing*, Prentice Hall, 1998.

[10] Karthik Raghupathy and Thomas W. Parks, "Curve tracing in images," in *Proceedings of IEEE Western New York Image Processing Workshop*, October 2003.