The Curve Filter Transform – A Robust Method for Curve Enhancement

Kristian Sandberg

Computational Solutions, LLC
1800 30th St, Suite 210B
Boulder, CO 80301
http://www.computationalsolutions.com

Abstract. In this paper we introduce the Curve Filter Transform, a powerful tool for enhancing curve-like structures in images. The method extends earlier works on orientation fields and the Orientation Field Transform. The result is a robust method that is less sensitive to noise and produce sharper images than the Orientation Field Transform. We describe the method and demonstrate its performance on several examples where we compare the result to the Canny edge detector and the Orientation Field Transform. The examples include a tomogram from a biological cell and we also demonstrate how the method can be used to enhance handwritten text.

1 Introduction

In this paper we improve upon previous work on using orientation fields to enhance curve-like objects in images. The method is based on the Orientation Field Transform (OFT, see [5],[7],[8]), but introduces a new type of transform that significantly improves the sharpness of the enhanced features while being less sensitive to noise.

The OFT was originally developed for enhancing membranes in tomograms of biological cells, but the method is general and can be applied to any image where curve-like objects need to be enhanced. The OFT has the property of not only enhancing curve-like structures, but also suppress point-like structures.

Other methods for enhancing curve-like structures includes the Hough transform (see [1] and references therein), but whereas the Hough transform works best to detect global objects, the OFT is local in nature. The Canny edge detector [3] can be used to detect general discontinuities, including edges of curve-like structures. However, it does not distinguish between point-like and curve-like structures, and requires significant blurring in noisy environments, which leads to poor localization of the edges.

Anisotropic diffusion methods (see e.g., [9]) are well suited for de-noising curve-like objects without blurring the edges. However, as noted in [6] and [2], these methods does not distinguish between point-like and curve-like objects. Anisotropic diffusion methods typically involves solving computationally expensive non-linear PDEs, and also tend to rely on tuning several parameters to achieve good results.

G. Bebis et al. (Eds.): ISVC 2010, Part II, LNCS 6454, pp. 107–116, 2010.

The introduction of one or more user controlled parameters can often improve the processed image. However, parameters can also offset the benefit of automized image enhancement. A user needs to have a good understanding of the effect of the parameters, and too many non-intuitive parameters is a common obstacle for wide-spread adaptation of automized image processing algorithms in laboratory settings.

The method in this paper only requires one parameter, which corresponds to the typical width of the targeted structure. Hence, the same method can easily be adapted to a broad variety of situations, and is easy for a new user to learn.

In this paper we introduce the Curve Field Transform (CFT), which uses the idea of orientation fields [4] and the OFT to generate what we will refer to as a curve field. A curve field assigns a weighted curve segment to each location in the image. Each such curve segment has a support that typically extends over many pixels in the image such that each pixel is covered by multiple curve segments from nearby pixels. The CFT sums the weights of all curve segments overlapping a given pixel. The CFT relies on computing the curve field in a way that captures the geometric structures of curve-like objects in the image.

The paper is outlined as follows. We begin by presenting the main idea of the method. In the following two sections we provide the details for generating the curve field and how to apply the CFT. We next provide several examples including a synthetic image, a slice from a tomogram of a biological cell, and a noisy image where we extract hand written text. We conclude the paper with a discussion of the method and a conclusion.

2 Idea

In this section, we outline the idea behind the CFT. In the following two sections, we provide a more detailed description of the method.

A *curve field* is the assignment of a parameterized curve segment to each location of an image. With each such curve we associate a (scalar) weight that indicates the "importance" of the curve segment. The lengths of the curve segments are typically fixed and chosen as roughly twice the width of a typical curve-like structure in the image.

The first step of our method is to generate a curve field with the following two properties:

- For a pixel located on a curve-like structure, the associated curve segment should align with the underlying structure, and have a large weight.
- For a pixel far away from a curve-like structure, the associated weight should be small.

We generate a curve field with these two properties using a two-step process. In the first step, we go through each pixel of the image and look for the direction along which the image contains the largest mean intensity. This generates what we will refer to as a primary curve field where the curve segment at each pixel can be represented by an angle and a weight that gives the mean intensity along the line (see Figure 1 a and b).

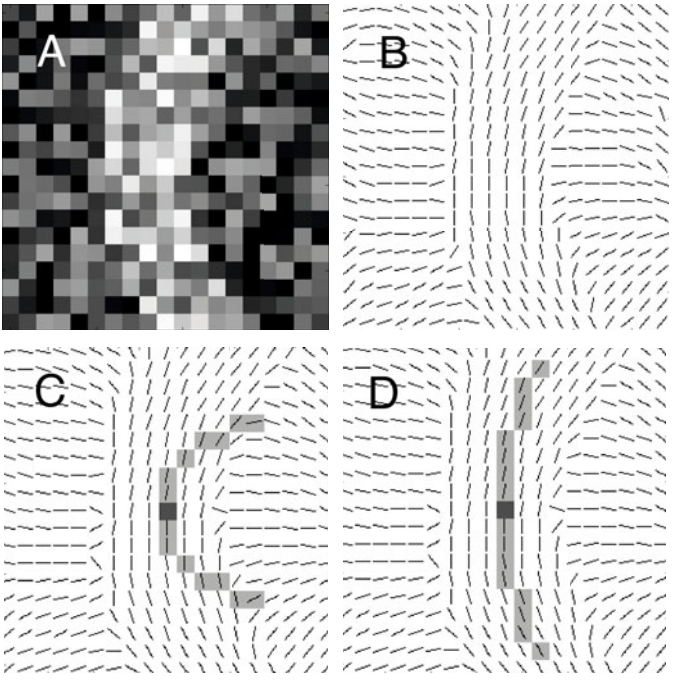

Fig. 1. a) Original image. b) Illustration of the primary curve field. c) Measuring the alignment along the gray curve (high curvature). d) Measuring the alignment along the gray curve (small curvature). In this example, the curve in d) aligns best with the underlying primary curve field, and the secondary curve field at the center pixel is therefore assigned the gray curve in d). Note that in the above figures, we have for clarity plotted each curve segment of the primary curve field as having a length of one pixel. In practice, each curve extends over several pixels. Curve segments have also been plotted with uniform intensity, although in practice the intensity (weight) may vary betweens curve segments (see also Figure 2 below).

Fig. 2. Illustration of the CFT. The lines illustrate the secondary curve field for the image in Figure 1a. (For clarity, we only plot straight curves.) In this figure, each curve segment is plotted with an intensity that reflects the weight associated with the curve segment. The CFT is computed by at each pixel, summing the weight for each curve segment that intersect the pixel.

In the second step, we look for curves along which the primary curve field from the first step shows strong alignment. We record the curve along which we find the best net alignment, and record the net alignment value as the weight (see Figure Figure 1 c and d). The new curve and weight from this step is referred to as the *secondary curve field*.

Once the secondary curve field has been generated, we apply the CFT as follows: At each pixel we look for all curve segments that overlap a pixel, and sum the weights from these curves (see Figure 2).

3 Generating the Curve Field

We denote a curve field at location $\mathbf{x} = (x, y)$ as $\mathcal{F}(x, y) = \{w(x, y), \mathbf{r}(x, y, t)\}$ where w is a (scalar) weight and $\mathbf{r}(x, y, t)$ denotes a parameterized curve centered at (x, y) and parameterized with parameter $t \in [-\frac{L}{2}, \frac{L}{2}]$ such that $\mathbf{r}(x, y, 0) = (x, y)$. Here L denotes the length of the curve and is typically chosen as roughly twice the width of a typical curve-like structure in the image. For clarity, we will sometimes omit the t argument and simply write $\mathbf{r}(x, y)$.

We first generate what we will refer to as the *primary curve field* by using the method in [5] or [7]. In this section, we will for simplicity review the simpler method in [5] but note that the somewhat more complicated method in [7] gives significantly better results.

Let us first consider a family of straight lines $\mathbf{r}_\theta(x, y, t)$ of length L indexed by the *orientation angle* θ such that

$$\mathbf{r}_\theta(x, y, t) = (x + t\cos\theta)\ \hat{\mathbf{i}} + (y + t\sin\theta)\ \hat{\mathbf{j}},\ t \in [-\frac{L}{2}, \frac{L}{2}].$$

Let $I(x, y)$ denote the image. The primary curve field $\mathcal{F}_1$ is given by $\mathcal{F}_1 = \{w(x, y), \mathbf{r}_{\tilde{\theta}}(x, y)\}$ where

$$w(x, y) = \max_\theta \int_{-\frac{L}{2}}^{\frac{L}{2}} I(\mathbf{r}_\theta(x, y, s))\ ds$$

and

$$\tilde{\theta}(x, y) = \arg\max_\theta \int_{-\frac{L}{2}}^{\frac{L}{2}} I(\mathbf{r}_\theta(x, y, s))\ ds.$$

An example of a primary curve field is illustrated in Figure 1a and b.

In order to generate what we will refer to as the *secondary curve field*, we let $\{\mathbf{r}_k(x, y, t)\}_k$ denote a generic family of parameterized curves indexed by k. In order to define more complicated curves, these curves may include more parameters than the orientation angle θ used above. We can, for example, for each tangent angle θ also vary the curvature for the curve (see [8] for details).

Let w_1 and θ_1 denote the weight and orientation angle of the primary curve field $\mathcal{F}_1$. We now consider a curve $\mathbf{r}_k(\mathbf{x}, t)$ and introduce the *alignment integral*

$$\Omega[\mathcal{F}_1](\mathbf{x}, k) = \int_{-\frac{L}{2}}^{\frac{L}{2}} w_1(\mathbf{r}_k(\mathbf{x}, s)) \cos\left(2(\theta_1(\mathbf{r}_k(\mathbf{x}, s)) - \nu_k(s))\right) \left\| \frac{d\mathbf{r}_k}{ds} \right\| ds \quad (1)$$

where ν_k denotes the tangent vector $\frac{d\mathbf{r}_k}{dt}$ mapped to the interval $[0, 180°)$.[1]

Using the alignment integral we define the secondary curve field $\mathcal{F}_2$ as $\mathcal{F}_2(x, y) = \{w(x, y), \mathbf{r}_{\tilde{k}}(x, y)\}$ where

$$\tilde{k} = \arg\max_k |\Omega[\mathcal{F}_1](x, y, k)|$$

and

$$w(x, y) = \Omega[\mathcal{F}_1](x, y, \tilde{k}).$$

This alignment integral can be compared to a work integral over a vector field in physics represented in polar coordinates (see [8] for more details).

In order to interpret the alignment integral, we note that the integrand in (1) gives a large positive response if the tangent of $\mathbf{r}_k(\mathbf{x}, s)$ aligns with the curve segment of the underlying primary curve field. The curve $\mathbf{r}_{\tilde{k}}(x, y, t)$ therefore represents the curve that aligns best with the underlying primary curve field (see Figure 1c and d).

We can visualize the secondary curve field as a curve segment of length L assigned to every pixel, with a "density" (weight) associated with the curve segment. The shape of the curve segment typically mimics the underlying structure of the original image in a neighborhood of a pixel. However, in areas without any significant curve-like structures, the shape of the curve does not matter, as the weight will be close to zero at such locations.[2]

The reason for introducing the secondary curve field is that it measures correlations in orientations, which are less sensitive to noise than correlations in intensity.

4 The Curve Field Transform

Once the secondary curve field has been computed, the weight of the curve segment associated with each pixel is added to the output image for all pixels along the curve segment. Formally, we define the curve filter transform as

$$\mathcal{C}[\mathcal{F}](\mathbf{x}) = \sum_{\mathbf{x}' \in \mathcal{D}} \chi_{\mathbf{r}(\mathbf{x}')}(\mathbf{x}) w(\mathbf{x}')$$

where $\mathcal{D}$ denotes the image domain, $\{w(\mathbf{x}'), \mathbf{r}(\mathbf{x}')\}$ is the secondary curve field, and χ denotes the characteristic function defined by

$$\chi_{\mathbf{r}(\mathbf{x}')}(\mathbf{x}) = \begin{cases} 1, \mathbf{x} \in \mathbf{r}(\mathbf{x}') \\ 0, \mathbf{x} \notin \mathbf{r}(\mathbf{x}') \end{cases}.$$

[1] As opposed to vectors, orientation angles lack a sense of "backward/forward", and are only defined for angles in the range $[0, 180°)$. If the tangent vector has an angle in the interval $[180°, 360°)$, we map such angles to $[0, 180°)$ by subtracting $180°$.

[2] This assumes that the weights are computed according to the method in [7].

Although $\mathbf{r}(\mathbf{x}, t)$ was defined as a vector valued function in Section 3, when using it as an argument to χ we will treat it as a set defined by the values $\mathbf{r}(\mathbf{x}, t),\ t \in [-\frac{L}{2}, \frac{L}{2}]$.

In practice, we compute the output of the CFT as follows. We first discretize the curve parameter t as the array `t[0],t[1],...,t[N-1]`, and then discretize the curve field by introducing the two arrays `r[x,y,t[k]]` and `w[x,y]`. The CFT is now computed by the following algorithm:

```
1. output = 0
2. for all pixels (x,y):
       for k=1,...,N
           output[r[x,y,t[k]]] += w[x,y]
```

The outer loop loops over the discretized curve field {`w[x,y],r[x,y,t]`}, while the inner loop loops over all pixel that each curve segment overlaps. For each pixel that the curve segment `r[x,y,t]` overlaps, we add the weight `w[x,y]` of the curve segment to the output.

5 Results

In this section we apply the method described above, but use the method from [7] to compute the weights and orientation angles for the primary curve field.

In our first example we demonstrate the method for a synthetically generated test image consisting of an S-curve and a point with noise added (Figure 3a). The

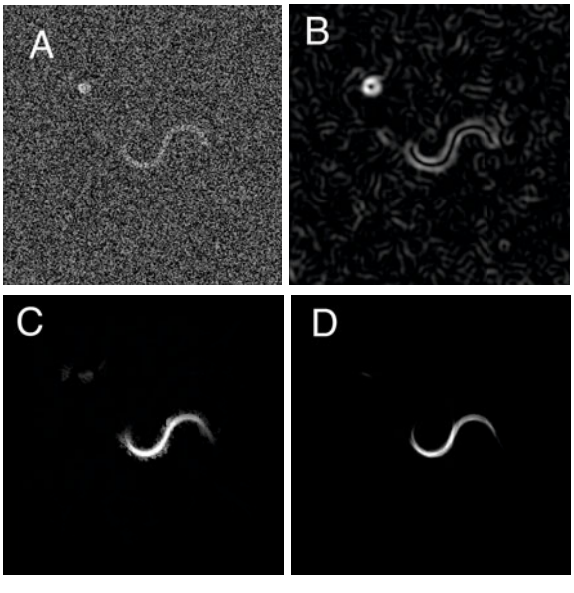

Fig. 3. a) Original image. b) Result using the Canny edge detector. c) Result using the OFT. d) Result using the CFT.

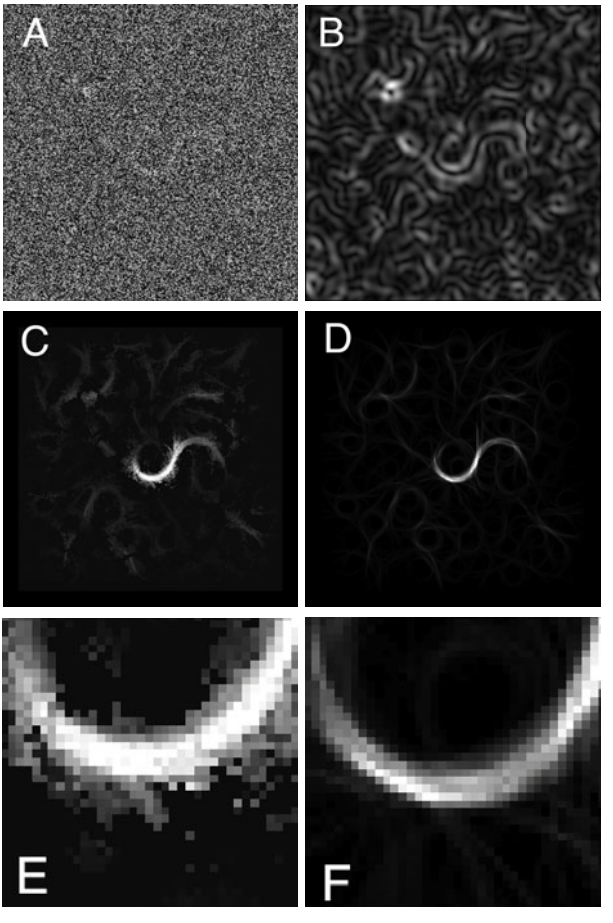

Fig. 4. a) Original image. b) Result using the Canny edge detector. c) Result using the OFT. d) Result using the CFT. e) Close up of Figure 4c (OFT method). f) Close up of Figure 4d (CFT) method. We note that the CFT gives significantly sharper result.

point has twice the intensity of the S-curve. The goal is to enhance the S-curve while suppressing the point in order to illustrate the geometrical aspect of the method. In Figure 3 b-d we compare the result of using the Canny edge detector, the OFT algorithm from [8], and the CFT introduced in this paper. For the result using the Canny edge detector, we used a Gaussian blurring kernel combined with the Sobel edge detector mask.[3] We tuned the width of the Gaussian kernel to produce a result that was a compromise between denoising and good edge

[3] The Canny edge detector typically includes an additional step where a thinning operation is applied to the edge. As this paper focus on the enhancement step rather than the segmentation problem, we have omitted the thinning step, but note that this step could be applied to both the Canny method and the CFT method.

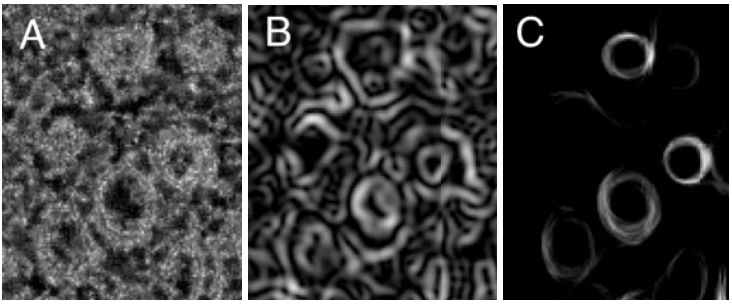

Fig. 5. a) Original section from a tomographic slice of a T-lymphocyte. b) Result after applying the Canny edge detector. c) Result after applying the CFT.

location. From the result in Figure 3 we note that the Canny edge detector cannot distinguish between the point and the curve, but detects the edges of both objects. Despite the inherent denoising in the Canny algorithm, we note that it is more sensitive to noise than both the OFT and the CFT methods, as it picks up more false positives from the background. Comparing Figure 3c and d, we also note that the CFT method produces a sharper image than the OFT method.

In Figure 4 we show the same comparison, but now with more noise added. We see that the CFT is less sensitive to the noise than the other methods, while also producing a sharper image than the OFT method. This is illustrated further in Figure 4e and f, where we have enlarged the bottom of the S-curve for the OFT and the CFT results from Figure 4c and d. We note how the curve is significantly sharper for the CFT method.

In the next example (Figure 5), we look at a section from a tomographic slice of a T-lymphocyte. This slice is a challenging data where the circular structures of interest are surrounded by complex structures that is a problem for most segmentation algorithms. Although the Canny edge detector does pick up the four main structures in the image, it cannot separate them from the complex structures in the background. The CFT enhances the structures of interest, while suppressing the surrounding structures. This is a most important property, as there is a huge interest in segmentation methods for biological cells [6].

In our final example, we consider a handwritten text with a few dots added to simulate ink stains (Figure 6a). The text was digitized at low resolution such that the width of the line is only 1-2 pixels wide. In Figure 6b we have added noise to the image. The noise combined with the coarse digitization makes this a challenging enhancement problem. We apply the CFT to the noisy image in Figure 6b and observe that the method suppresses the points and noise while capturing the text. Suppressing the points is important for hand recognition algorithms, which may be offset by such point-like objects.

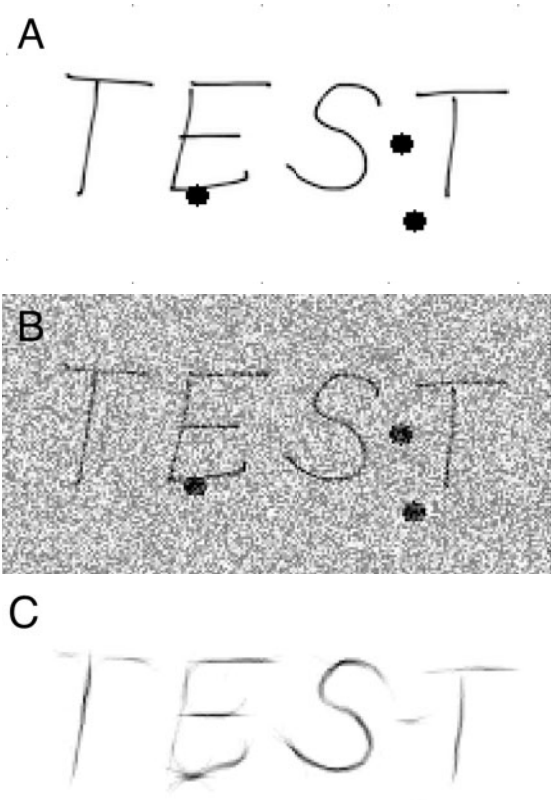

Fig. 6. a) Original hand-written note with simulated ink stains added. b) Same as the top image, but with noise added. c) Enhanced image after applying the CFT to the noisy image in b).

6 Discussion

The examples in the previous section illustrate a most important property of the CFT: The ability to enhance curves based on geometry rather than contrast. This ability makes the method ideal in situations where an image has to be automatically segmented, as many segmentation algorithms are hurt by the presence of high contrast point-like objects. The reason why the CFT works so well in this respect, is that focuses on orientations rather than intensity information in the image.

We also note that the CFT is forgiving for high noise levels. One reason for this is that the main operations are based on addition, rather than notoriously unstable subtraction operations used in typical edge detection methods. The method also does not require any pre-processing in terms of denoising, but denoising is built-in to the algorithm. The nature of the denoising is also anisotropic, which prevents blurring of the edges.

Finally, we stress that the proposed method only requires one parameter to be tuned. The parameter is related to the typical thickness of the structures to be enhanced and the method is therefore trivial to learn for a new user. Also,

the same method works well for a wide variety of problems. In this paper we applied the same method for enhancing cell membranes in a tomogram for a biological cell, and to enhance the text on a hand written note. Despite these quite different areas of application, the same algorithm could be used for both applications. In fact, the method is quite insensitive to variations in the one and only parameter used by the algorithm, and we were able to use the same parameter value for both applications.

7 Conclusion

We have presented a new method for enhancing curve-like structures in images. The new method improves on the previously suggested method using the Orientation Field Transform (OFT), but is less sensitive to noise and gives sharper images than the OFT. We have demonstrated the method's excellent performance in enhancing curves in noisy images, and its ability to suppress high contrast point-like objects.

Acknowledgment

The author would like to thank the Boulder Laboratory for 3-D Electron Microscopy of Cells at University of Colorado for providing the tomogram used in the paper.

References

1. Ballard, D.H.: Generalizing the Hough Transform to Detect Arbitrary Shapes. Pattern Recognition 13, 111–112 (1981)
2. Brega, M.: Orientation Fields and Their Application to Image Processing. Master's Thesis, University of Colorado at Boulder (2005)
3. Canny, J.: A Computational Approach to Edge Detection. IEEE Trans. Pattern Analysis 8, 679–698 (1986)
4. Gu, J., Zhou, J.: A novel model for orientation field of fingerprints. In: Proc. 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 493–498 (2003)
5. Sandberg, K., Brega, M.: Segmentation of thin structures in electron micrographs using orientation fields. J. Struct. Biol. 157, 403–415 (2007)
6. Sandberg, K.: Methods for image segmentation in cellular tomography. In: McIntosh, J.R. (ed.) Methods in Cell Biology: Cellular Electron Microscopy, vol. 79, pp. 769–798. Elsevier, Amsterdam (2007)
7. Sandberg, K.: Curve enhancement using orientation fields. In: Bebis, G. (ed.) ISVC 2009, Part 1. LNCS, vol. 5875, pp. 564–575. Springer, Heidelberg (2009)
8. Sandberg, K.: The Generalized Orientation Field Transform. In: Barneva, R.P., et al. (eds.) Object Modeling, Algorithms, and Applications, pp. 107–112. Research Publishing (2010)
9. Weickert, J.: Anisotropic Diffusion in Image Processing. Teubner-Verlag, Stuttgart (1998)