Enhancement of Low-Contrast Curvilinear Features in Imagery

Mark J. Carlotto

***Abstract*—A new method is described for enhancing low-contrast curvilinear features in imagery that combines directional filtering with Fischler, Tenenbaum and Wolf's F^* algorithm for computing minimum cost paths. The method exploits a phenomenon called "the stability of lines over angle." The idea is that when a directionally filtered image contains a line plus noise, minimum cost paths tend to be aligned in the direction of the line with random jumps between parallel paths. When the input image contains noise only, the direction of minimum cost paths resemble random walks with drift. As the direction of the filter changes, minimum cost paths that follow true features persist and are more stable over angle than those that follow noise. Adding them up in an accumulator array over angle produces a larger number of votes along signal paths than along noise paths. This provides a means for enhancing trajectories of low-contrast features. Several examples illustrate the enhancement of forest trails in USGS aerial imagery, linear features on Mars, and roads in synthetic aperture radar imagery.**

***Index Terms*—Image enhancement, F^* algorithm, low-contrast curvilinear feature extraction, track-before-detect.**

I. Introduction

THE ability to enhance, track, and detect low-contrast curvilinear features is important in a variety of image analysis applications. One is in finding fine-scale cartographic and micro-terrain features such as trails and small streams in aerial imagery. These features are often not contained in cartographic databases and yet can be important in route planning, terrain analysis, and cross-country mobility assessment. Another is the detection of lineaments (e.g., fault lines, aligned volcanoes, and other surface features) in terrestrial and planetary satellite imagery. Lineaments are often very subtle, and sometimes overlaid by more recent surface processes making their detection and localization difficult. A third example is the extraction of roads and other similar features in synthetic aperture radar (SAR) imagery. The coherent nature of SAR creates speckle noise which limits the usefulness of conventional edge and line detectors.

Directional (matched) filters, Hough, and Radon transforms can be used to detect straight lines and other features of known shape. Jao *et al.* [7] describe a coherent spatial filtering approach for SAR. Copeland [1] developed a localized Radon transform for detecting short linear features such as ship wakes in SAR imagery. Curvilinear features present a greater challenge. If their shape is not known in advance, it is not possible to construct a matched filter.

Tracking techniques provide an alternative approach. McKeown and Denlinger [5] survey techniques for tracking high-contrast roads in images. Edge linkers segment the image into edges (or lines) and connect the segments together using a model of the feature. Region-based followers operate on a similar principle. Given a starting point and direction, a correlation tracker moves out by matching a surface model of the feature (e.g., the cross-sectional intensity profile) to the image perpendicular to the direction of advance, using a path model to control the search process.

Techniques where tracking follows detection do not work well in low-contrast situations where the feature of interest is at, or below, the noise level. Instead a track-before-detect approach can be used which accumulates evidence along alternative track (feature) hypotheses before making a detection decision. Track-before-detect techniques have been developed for moving target indication (MTI) detection in video data [8]–[10]. Samadani and Vesecky [6] describe a single image SAR curvilinear feature detection technique based on a track-before-detect approach, which uses maximum *a posteriori* estimation together with statistical models for speckle noise and the curve generation process to find the most probable estimate of the feature given the image data.

Motivated by the track-before-detect approach, a new technique for enhancing low-contrast curvilinear features in images is proposed that combines directional filtering with Fischler, Tenenbaum, and Wolf's F^* algorithm [3] for computing minimum cost paths. The F^* is a track-before-detect algorithm that was originally developed for detecting high-contrast roads in low-resolution imagery. Roads are extracted interactively by following a minimum cost path back from a user-specified end point to either a starting point or edge. Instead of using the F^* algorithm interactively for detection, a different application is described here in which it is used, in effect, as a filter to enhance low-contrast features. The goal is to do this automatically without knowledge of a feature's shape, direction, or endpoints.

The plan of the paper is as follows. Section II reviews the F^* algorithm and shows how its performance varies with the input signal to noise ratio (SNR). It describes how low-contrast features can be detected by using a directional filter before the F^* algorithm to increase the SNR. When the endpoints are not specified, the F^* can be used in a different way to enhance low-contrast features by computing minimum cost paths between all edge/point combinations on opposite sides of an image and adding up the resultant paths in an accumulator array. If the direction of the feature is unknown a directional filter bank (DFB) can be used before the F^* algorithm as described in Sec-

Manuscript received May 23, 2006; revised July 6, 2006. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Vicent Caselles.

M. J. Carlotto is with the General Dynamics—Advanced Information Sciences Division, Arlington VA 22209 USA (e-mail: markcarlotto@adelphia.net).

Digital Object Identifier 10.1109/TIP.2006.884949

tion III to increase SNR. One way to enhance low-contrast features is to place the DFB before the $F^*(\mathrm{DFB}+F^*)$ so that all paths are computed from the same directionally filtered set of costs. Another way applies the F^* algorithm to separate directionally filtered images. This new algorithm known as track enhancement using the stability of lines over angle (TESLA) is described in Section IV. Using simulated data, it is shown that TESLA outperforms $\mathrm{DFB}+F^*$ in enhancing low contrast curvilinear features. Additional examples are presented in Section V illustrating the extraction of forest trails in USGS aerial imagery, linear features on Mars, and roads in SAR imagery. Extending the technique to multiband data is discussed in Section VI. An Appendix contains a performance model for the F^* algorithm, which gives the expected number of errors along a track as a function of the SNR.

II. F^* Algorithm

Let $X=\{1,2,\ldots M\}$ be the nodes of a network. Define $c_{i,j}$ as the cost to go from node i to node j. Ford [11] developed the following iterative algorithm for finding the minimum cost path through a network.

1) Initialize the values of the nodes as follows: $x_i = \begin{cases} 0, & i=0 \\ \infty, & i\neq 0 \end{cases}$ where x_0 is the starting node.
2) For each connected pair of nodes, if $x_i - x_j > c_{i,j}$, we replace the value of the first node: $x_i = x_j + c_{i,j}$; otherwise, it is left alone. This is repeated for all pairs of connected nodes until no value changes. At that point, the value at a node is the minimum path cost to get to that node from the starting node.
3) The minimum cost path from the starting node to any node in the network is obtained by moving from that node backwards to the starting node in the direction of decreasing path cost.

The F^* algorithm [3], a variant of Rosenfeld and Pfaltz's distance transform [4], is a 2-D implementation of Ford's algorithm. It was originally developed for tracking high contrast features such as roads in optical imagery, and has subsequently been augmented to include line contrast and curvature [13], and extended to 3-D data [12].

The first step in the F^* algorithm computes the minimum cost to get to all pixels from a set of starting pixels. A series of alternating top-to-bottom/bottom-to-top passes over two arrays, $x_{i,j}$ and $c_{i,j}$ are performed, where i and j are the row and column indices, respectively. The first array, which stores the path cost, is set to zero at the starting location of the feature, and to a large value everywhere else. The second array stores the costs, which are derived from the input image. In the top-to-bottom pass, each row is first processed left to right by

$$x_{i,j}=\min\left\{\begin{array}{c} c_{i,j}+x_{i-1,j-1},\\ c_{i,j}+x_{i,j-1},\\ c_{i,j}+x_{i+1,j-1},\\ c_{i,j}+x_{i-1,j},\\ x_{i,j}\end{array}\right\} \tag{1}$$

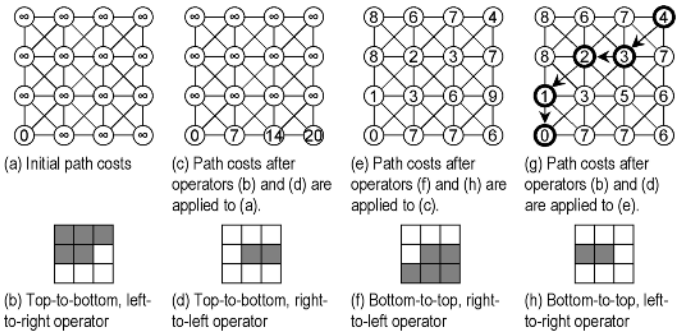

Fig. 1. F^* algorithm processing sequence.

and then right to left by

$$x_{i,j}=\min\left\{\begin{array}{c} c_{i,j}+x_{i+1,j},\\ x_{i,j}\end{array}\right\}. \tag{2}$$

In the bottom-to-top pass, each row is processed right to left by

$$x_{i,j}=\min\left\{\begin{array}{c} c_{i,j}+x_{i-1,j+1},\\ c_{i,j}+x_{i,j+1},\\ c_{i,j}+x_{i+1,j+1},\\ c_{i,j}+x_{i+1,j},\\ x_{i,j}\end{array}\right\} \tag{3}$$

and then left to right by

$$x_{i,j}=\min\left\{\begin{array}{c} c_{i,j}+x_{i-1,j},\\ x_{i,j}\end{array}\right\}. \tag{4}$$

F^* iterations are repeated until no $x_{i,j}$ value changes.

Consider the following cost array:

6	4	5	1
7	1	1	4
1	3	3	3
0	7	7	6

Fig. 1 illustrates the operation of the F^* algorithm on this array, represented as a network. The array of path costs is initialized (a), and processed by the top-to-bottom, left-to-right (b) and right-to-left (d) operators to produce the path cost array (c). The shaded squares are those cells (pixels) within the window that contribute to the update. This is then processed by the bottom-to-top, right-to-left (f) and left-to-right (h) operators to produce the path cost array (e). One more application of operators (b) and (d) produce the final array of path costs (g). The algorithm stops after the next iteration in which none of the values changes. The minimum cost path from the starting node to the edge nodes is indicated in Fig. 1(g).

Next, consider the effect of noise on the algorithm. Fig. 2 shows four realizations of a constant amplitude horizontal line in additive white Gaussian noise (AWGN) at SNRs of -3, 0, 3, and 10 dB (a)–(d). The images have been contrast-stretched for display purposes. The F^* algorithm was applied to each image, assuming the left starting point of the line and right destination

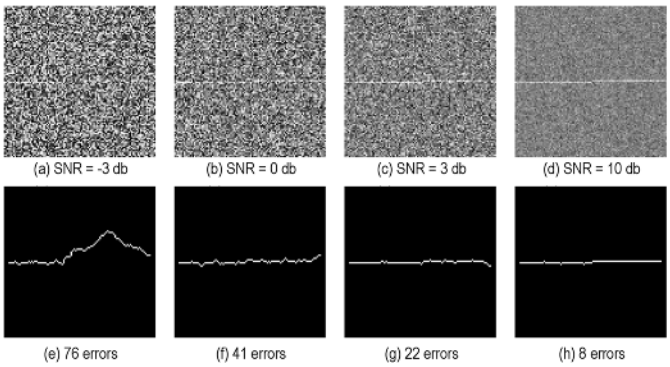

Fig. 2. F^* algorithm results at four different SNRs (point-to-point).

TABLE I
COMPARISON OF PREDICTED AND ACTUAL F^* ALGORITHM PERFORMANCE

SNR (db)	Predicted P_{error}	Measured P_{error}
-3	.8	.76
0	.4	.41
3	.2	.22
10	.04	.08

edge are known. The path computed by the F^* algorithm is a series of points $\mathbf{x} = (x_0, \ldots, x_{K-1})$ where $x_k = (i_k, j_k)$ are the row and column indices of the kth point. The computed path can also be represented as an image

$$F = f_{i,j} = \begin{cases} 1, & i,j \in \mathbf{x} \\ 0, & \text{otherwise} \end{cases} . \tag{5}$$

The path images computed by the F^* algorithm are shown in Fig. 2(e)–(h). We can estimate the probability of making an error along the path as a function of SNR (Appendix A). At a level of confidence of $p = 0.995$

$$P_{\text{error}} = \frac{(140.2 - 100)}{100 \times 10^{SNR/10}} \tag{6}$$

where the correct path is $K = 100$ pixels long. Predicted and actual values show good agreement (Table I).

In an automated system, the endpoints are not available. If we know the line of interest runs in a particular direction, one strategy for finding it is as follows.

1) Mark one set of points (e.g., an edge of the image) as possible end points (set costs to zero).
2) Compute the path costs to this set of points, and then find the minimum cost path from each point in the other set of (starting) points (e.g., along the opposite edge of the image) to the nearest end point. Let $F(p, E)$ be the minimum cost path image computed from a start point $p \subset S$ to a set of end points E. The accumulated result over all start points is $\sum_{p \subset S} F(p, E)$. Repeat in opposite direction. In general, paths computed in one direction differ from those computed in the other direction, especially near the edges. They differ because the algorithm itself is not symmetrical, computing paths from each point along one edge to the nearest point on the other edge, and vice versa.
3) In order to capture all paths, we sum the minimum cost path images in both directions between a pair of edges $\sum_{p \subset S} F(p, E) + \sum_{q \subset E} F(q, S)$.

If E and S are the left and right edges of an image, then the sum represents possible horizontal paths; if they are the top and bottom edges, then the sum represents vertical paths; other edge pairs (e.g., top and right) give diagonal paths.

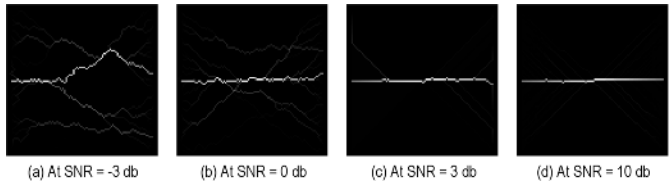

Fig. 3. Edge-to-edge F^* algorithm results.

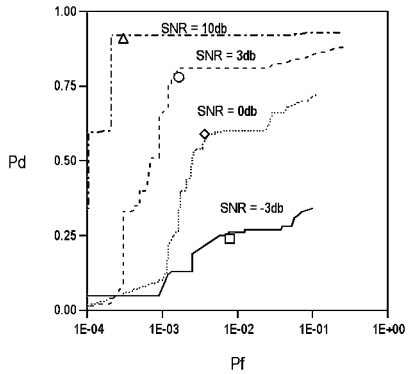

Fig. 4. Receiver operating characteristic (ROC) for F^* algorithm.

Fig. 3 shows what happens when the exact starting point is unknown. The path of interest runs between the left and right edges. If the edges are P pixels in length, $2P$ paths are generated which, when added together, produce an accumulated path image containing a range of possible values between 0 and $2P$. The value at a point in the accumulated path image is the number of paths that go through that point. Locations with large values are likely to lie along the feature we seek. Using the accumulated path image as a detection statistic, the receiver operating character (ROC) curve in Fig. 4 plots the probability of detection (Pd) versus the probability of false alarm (Pf) against truth as a function of the detection threshold $0 \leq t \leq 2P$. Pixels above threshold along the true line count as detections; others are false alarms.

Also plotted in Fig. 4 are the (P_d, P_f) values for the F^* algorithm assuming the starting point is known (Fig. 2), where $P_d = 1 - P_{\text{error}}$, $P_f = (K/M)P_{\text{error}}$, and M is the number of pixels in the image. These values occur at the "knee" of the ROC curves, where the slope changes from $\partial P_d / \partial P_f > 1$ to $\partial P_d / \partial P_f < 1$, generally regarded as the operating point that gives the best performance.

III. DIRECTIONAL FILTERING

Originally, small isotropic (nondirectional) filters were used for tracking high contrast features using the F^* algorithm [3].

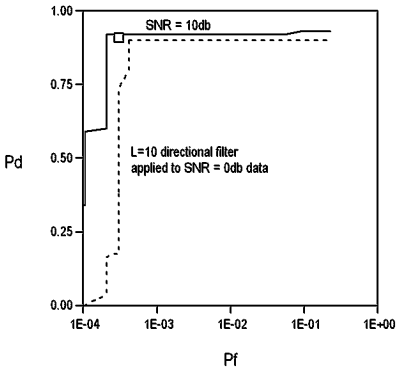

Fig. 5. Directional filtering improves performance of F^* algorithm.

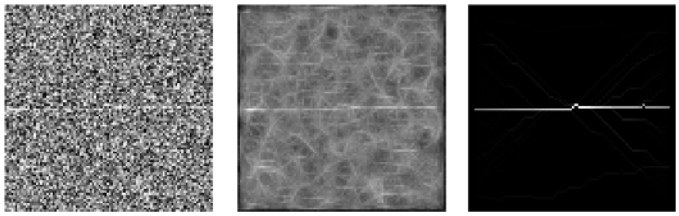

Fig. 6. Directional filtering before the F^* algorithm ($L = 10$ pixels, $45^\circ \leq \theta \leq 135^\circ$).

Given the relationship between SNR and accuracy, increasing SNR prior to applying the F^* algorithm should improve performance. In AWGN, summing along the signal increases the SNR by the length of the filter L. The ROC curves in Fig. 5 show that after filtering the 0-dB data Fig. 2(b) with an $L = 10$ pixel filter in the direction of the line, the performance of the F^* algorithm on that data is close to that of the 10-dB data Fig. 2(h).

If the direction of the feature is unknown and/or if it exhibits significant curvature, filtering must be performed over a range of possible directions. Let

$$z_{i,j}^q = z_{i,j} * d_{i,j}^q \tag{7}$$

be the output of the qth filter

$$d_{i,j}^q = \delta\left(i - l\sin\theta_q, j - l\cos\theta_q | - \frac{L}{2} \leq l < \frac{L}{2}\right) \tag{8}$$

which sums along lines at an angle θ_q, where δ is the Dirac delta function. To compute the maximum response over a range of angles one can take the maximum value across the filters pixel by pixel over those angles

$$z_{i,j}^D = \max_q \left\{z_{i,j}^q\right\}. \tag{9}$$

Fig. 6 shows the results of preprocessing an image (a) through a directional filter bank (b) prior to F^* processing (c). As in the example in Fig. 2, we assume the image contains a line originating from a known starting point along the left edge running in any direction between 45° and 135° to the right edge (angles are measured relative to north). A bank of 18 filters

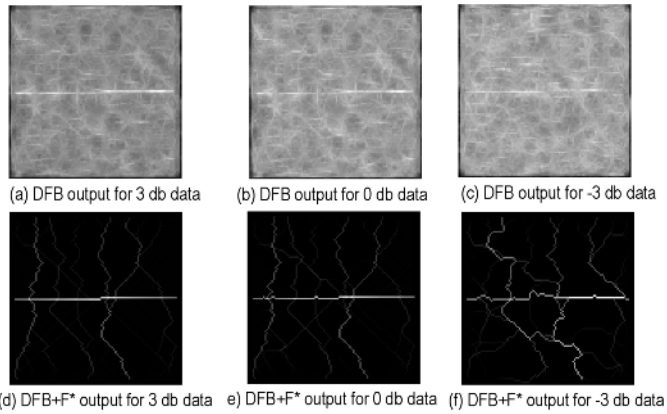

Fig. 7. Comparison of DFB and $\mathrm{DFB} + F^*$ results.

($\Delta\theta = 5^\circ$), $L = 10$ pixels were used. The performance using a directional filter bank (DFB) before the F^* algorithm (known as the $\mathrm{DFB} + F^*$) is comparable to that obtained using a single direction filter.

When the endpoints are not known we must use the edge-to-edge version of the F^* algorithm. In addition, if the direction of the line is unknown, one must search over all angles. One strategy is to filter over all directions, and compute possible paths between all 6 pairs of edges. Fig. 7(a)–(c) are the DFB outputs for the 3-, 0-, and −3-dB data. A bank of 36 filters ($\Delta\theta = 5^\circ$), $L = 10$ pixels long were used from 0° to 180°. Fig. 7(d)–(f) are the corresponding accumulated path images ($\mathrm{DFB} + F^*$). In addition to finding the path of the true line, the algorithm also finds spurious paths through noise. The number of these paths increases as the SNR decreases.

IV. Track Enhancement Using the Stability of Lines Over Angle (TESLA)

Instead of combining directional filter outputs into a single image before F^* processing, an alternative method (TESLA) involves applying the F^* algorithm to each directionally filtered image and combining F^* outputs in an accumulator array. The method exploits a phenomenon which we call "the stability of lines over angle." The idea is this: When the input image contains a line plus AWGN, minimum cost paths tend to be aligned in the direction of the line with random jumps between parallel paths. When the input image contains noise only, the direction of minimum cost paths resemble random walks with drift. As the direction of the filter changes, minimum cost paths that follow true features persist and are more stable over angle than those that follow noise. Adding them up in an accumulator array over angle produces a larger number of votes along signal paths than along noise paths. This phenomenon provides a means for enhancing trajectories of low-contrast features. Fig. 8(a) shows F^* paths for a 90° (horizontal) line in AWGN using filters from 0° to 90° at 10° increments. Note how the paths converge as the angle of the filter approaches that of the line. Noise, on the other hand, is less stable over angle Fig. 8(b).

The TESLA algorithm (Fig. 9) combines multiple F^* outputs, each computed at a given direction. For each direction, the input image is rotated so that the filter direction is horizontal. If the background is correlated, spatial whitening is performed

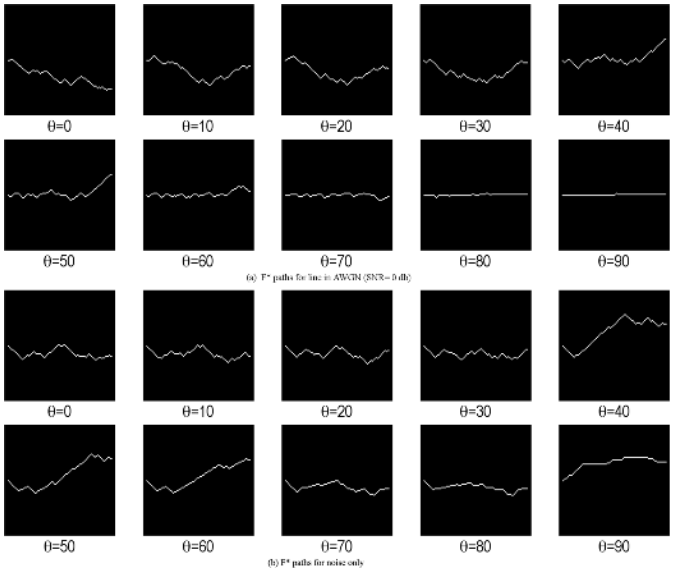

Fig. 8. F^* paths for different directional filters.

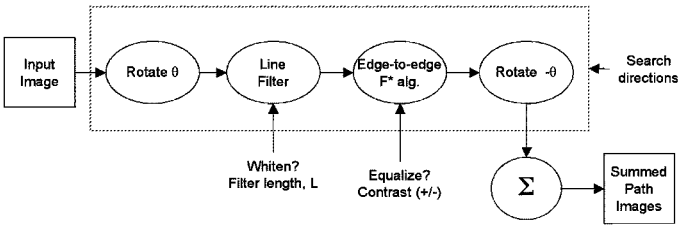

Fig. 9. TESLA processing flow.

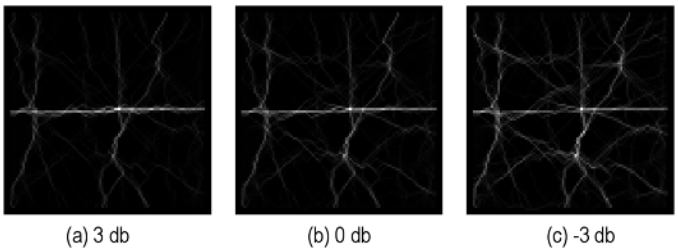

Fig. 10. TESLA summed path images at different input SNRs.

before the directional filter is applied. For positive-contrast features, the filtered image is contrast-reversed so that bright features have low cost. If the image is highly cluttered (i.e., contains other competing features), histogram equalization is performed. The edge-to-edge version of the F^* algorithm is applied to this rotated, filtered, and equalized image. The resultant accumulated path image is rotated back in the opposite direction and summed with other accumulated path images from other directions.

Fig. 10 shows the TESLA accumulated path images for the same −3-, 0-, and 3-dB data. Again, $L = 10$ pixel filters spaced $\Delta\theta = 5°$ apart from 0° to 180° were used. ROC curves (Fig. 11) compare $\mathrm{DFB} + F^*$ and TESLA enhancement performance. The performance of the $\mathrm{DFB}+F^*$ drops off significantly below

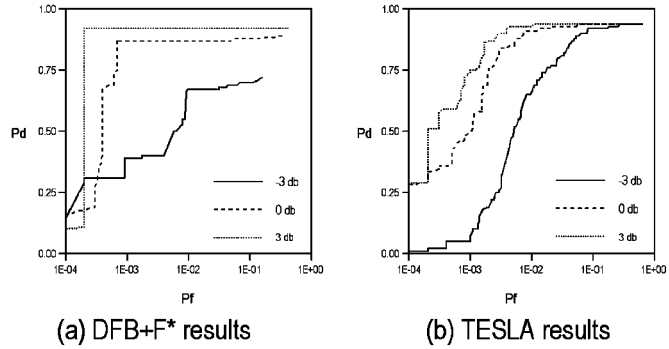

Fig. 11. Comparison between DFB, $\mathrm{DFB} + F^*$, and TESLA algorithms.

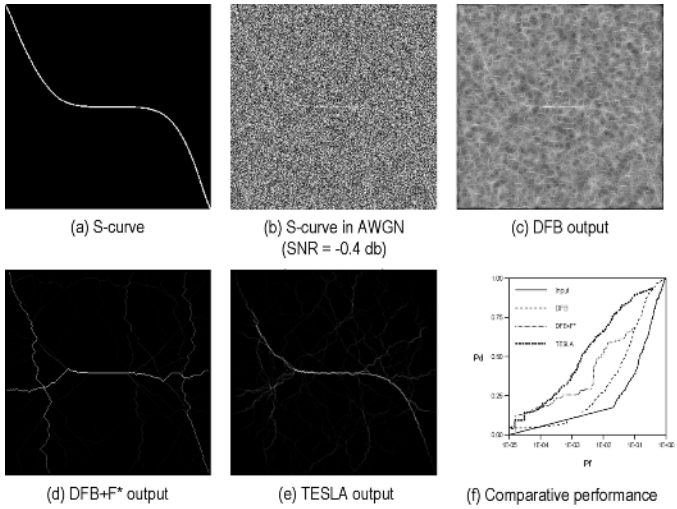

Fig. 12. Simulated S-curve.

0 dB (a), while that of the TESLA algorithm degrades more gracefully (b).

The strategy of using directional filters to increase SNR before using the F^* algorithm breaks down to some degree with curved features. In places of high curvature, directional filtering provides little if any enhancement along the curve. Fig. 12 is an S-curve (a) in AWGN (b). The SNR is -0.4 dB. A comparison of DFB (c) $\mathrm{DFB}+F^*$(d) and TESLA (e) results for $L = 10$ and $\Delta\theta = 5°$ shows DFB provides little enhancement of the curve, $\mathrm{DFB}+F^*$ misses the two bends, while TESLA responds to the entire curve. A comparison of ROC curves (f) shows TESLA has significantly better performance than $\mathrm{DFB}+F^*$. For a constant false alarm rate (CFAR) detector operating at $\mathrm{Pf} = 0.01$, the TESLA Pd is 25% higher than $\mathrm{DFB}+F^*$. Fig. 13 provides another example for a curve with a loop. In this case, TESLA's Pd is 53% higher. TESLA, thus, appears to better enhance low-contrast curvilinear features than DFB or $\mathrm{DFB} + F^*$.

V. Image Examples

In most practical image exploitation situations, we often do not know the signal amplitude, other than that it is larger or smaller than the background (i.e., positive or negative contrast).

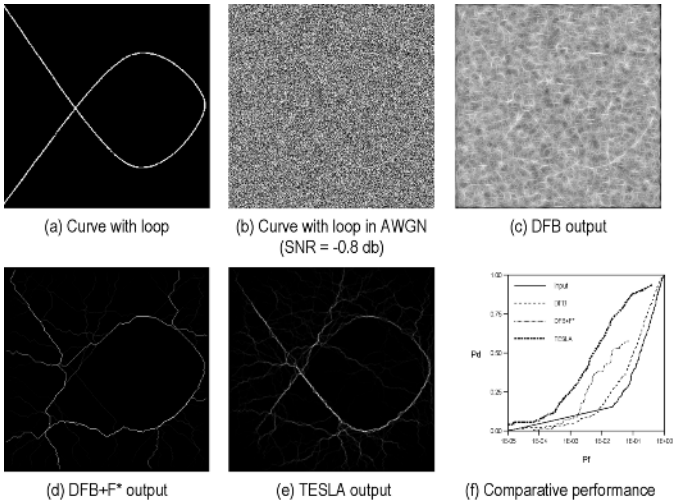

Fig. 13. Simulated curve with loop.

For a positive contrast signal, large values of z_k can be assigned low costs; e.g.,

$$c(\xi_k) = 1 + \frac{\left(b^2 - z_k^2\right)}{m}. \tag{10}$$

For example, the values $b = m = 255$ map the input range $0 \leq z < 256$ to the range of costs For negative contrast signals (29) can be flipped; i.e.,

$$c(\xi_k) = 1 + \frac{\left(z_k^2 - b^2\right)}{m}. \tag{11}$$

We have also found that, in practice, using histogram-equalized costs produces much better results for closely spaced features, and for highly cluttered backgrounds, as originally suggested by Fischler *et al.* [3].

Several examples are now presented to illustrate the use of the TESLA algorithm on collected imagery. The first involves finding trails in aerial imagery over a hilly and wooded area on Cape Ann, north of Boston, in Gloucester, MA (Fig. 14). A known trail (dotted line) runs from east of Goose Cove Reservoir, northeast up past a large rock formation known as Whale's Jaw, and north up to the town of Rockport (a). Portions of the trail are visible in an aerial image (b). The true location of the trail was measured using a GPS receiver (c). The output from the TESLA algorithm (d) finds the entire trail plus a few spurious paths of lower value. The following algorithm parameters (Fig. 9) were used: whitening = on, contrast = positive, histogram equalization = on, filter length = 5, and search direction 0°–180° in 5° increments.

The second example takes us from Gloucester to Mars. Fig. 15(a) is a portion of a Mars Global Surveyor (MGS) image in eastern Arabia Terra near 16.5° N latitude, 311.4° W longitude.[1] The image shows a variety of natural features including small craters, buttes and mesas left by erosion of

[1]http://www.msss.com/mars_images/moc/7_30_98_devil_rel/index.html

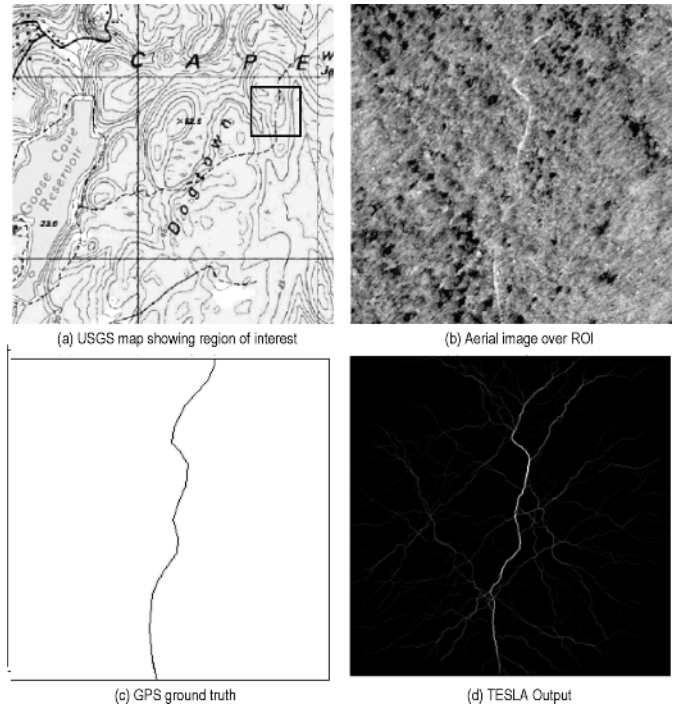

Fig. 14. Enhancing trails in USGS image.

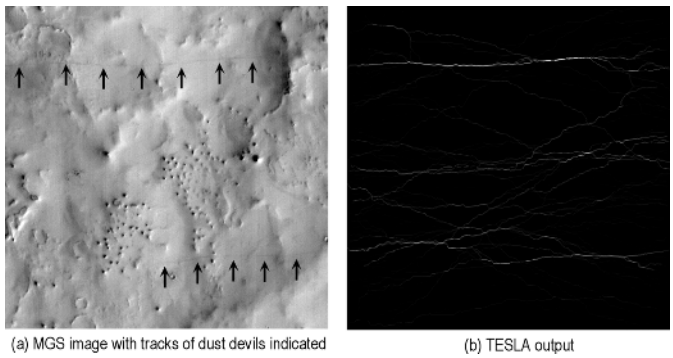

Fig. 15. Enhancing tracks of dust devils on Mars.

the surrounding terrain, small dunes and drifts, and a mantle of dust that varies in thickness. In the image two dark lines extending several kilometers/miles across the image are indicated that are thought to be tracks left by dust devils traveling over the Martian surface. Fig. 15(b) is the output of the TESLA algorithm showing the tracks of two dust devils as well as a number of weaker paths through clutter (crater rims, ridges, etc.). The following parameters were used: whitening = on, contrast = negative, histogram equalization = on, filter length = 10, and search direction 80°–100° in 5° increments.

Returning to earth, we conclude with an example illustrating the enhancement of small roads and trails in SAR imagery [Fig. 16(a)]. The image is a desert scene in the western US. Roads are relatively smooth and so have a low backscatter relative to desert scrub (bright areas). Changes in background brightness are caused by variations in the local incidence angle (topography). The TESLA output [Fig. 16(b)] captures most of the roads in the image (whitening = off, contrast = negative, histogram equalization = on, filter length = 20, and search direction 0°–180° in 5° increments).

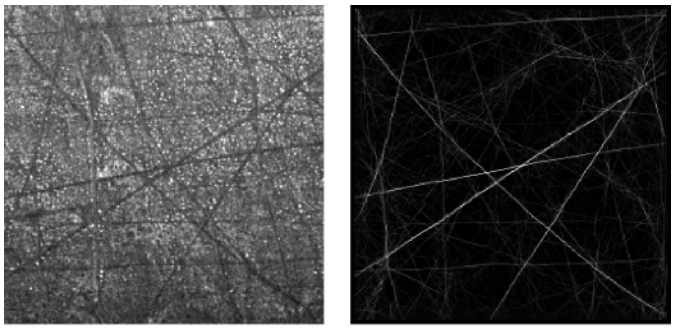

Fig. 16. Extracting roads from SAR imagery.

The run-time of the algorithm is linear $(r^2 = 0.976)$ in the number of pixels times the number of directions, $\text{time(s)} = 37 \times \#\text{megapixels} \times \#\text{directions} + 9.7$ for a C implementation on a 933-MHz Mac PowerPC G4 processor. For the example in Fig. 14, the run-time was about 2 min.

VI. Summary

Methods that combine directional filtering with Fischler, Tenenbaum and Wolf's F^* algorithm for computing minimum cost paths were explored as a means for enhancing low-contrast curvilinear features in images. A simple method of combining the two uses a directional filter bank before the F^* algorithm $(\text{DFB} + F^*)$. A new algorithm (TESLA) applies the F^* algorithm to individual cost images from each directional filter, and adds the resultant minimum cost paths in an accumulator array. On simulated images of curved features in AWGN, TESLA shows 25%–50% increase in the probability of detection at a constant $(\text{Pf} = 0.01)$ false alarm rate over $\text{DFB} + F^*$. Several imagery examples involving the extraction of forest trails in USGS aerial imagery, linear features on Mars, and roads in SAR imagery show promising results.

For constant amplitude features in AWGN, the cost at a pixel in the image is the normalized distance between the pixel value and the feature amplitude. The cost of a track of length K has a χ^2 distribution with K degrees of freedom. A means for estimating the number of errors along a track as a function of the input SNR was derived and shown to agree with experiment.

Extending F^* and TESLA to multiband data is straightforward. Assuming a known signal spectrum $\mathbf{a}$, the cost at a pixel

$$c(\xi_k) = [\mathbf{z}_k - \mathbf{a}]^T \mathbf{C}^{-1} [\mathbf{z}_k - \mathbf{a}] \tag{12}$$

where $\mathbf{z}_k$ is the pixel value at location k along the track, and $\mathbf{C}$ the spectral covariance of the noise. For M spectral bands, the total cost along a path of length K has a χ^2 distribution with $K \times M$ degrees of freedom.

Appendix
Performance Model

Assume a constant amplitude signal in additive white Gaussian noise

$$z_k = \begin{cases} a + \eta_k, & \text{signal} \\ \eta_k, & \text{noise} \end{cases} \tag{A1}$$

where the η_k are Gaussian with zero-mean and variance σ^2. The Viterbi algorithm [2] finds the shortest path through a graph (trellis) representing all possible state transitions. For (A1), the "length" of a state transition is

$$\begin{aligned} \lambda(\xi_k) &= -\ln\left[\frac{1}{\sigma\sqrt{2\pi}} e^{-(z_k - a)^2/2\sigma^2}\right] \\ &= \ln \sigma\sqrt{2\pi} + \frac{(z_k - a)^2}{2\sigma^2}. \end{aligned} \tag{A2}$$

The first term is the same for all transitions and can be ignored. The second term is the normalized (Mahalanobis) distance. We define the cost of a transition to be proportional to this second term

$$c(\xi_k) = \frac{(z_k - a)^2}{\sigma^2}. \tag{A3}$$

The expected value of the cost of a pixel along a path is

$$E[c(\xi_k)] = \begin{cases} 1, & \text{signal} \\ \frac{1+a^2}{\sigma^2,} & \text{noise.} \end{cases} \tag{A4}$$

The total cost along a track of length K is

$$y = \sum_{k=0}^{K-1} c(\xi_k) \tag{A5}$$

which has a has a chi-square (χ^2) density, $p(y)$, with K degrees of freedom. The expected value of the total cost of a track without errors is K, which is the minimum cost. Errors along the path add to the total cost. To compute the number of errors along the path at a level of significance, p, we determine the value of such that

$$\int_0^t p(y)dy = p \tag{A6}$$

where

$$t = K + N\left(\frac{a^2}{\sigma^2}\right) \tag{A7}$$

is the cost of a path containing N errors. Solving for N gives

$$N = \frac{(t - K)}{\left(\frac{a^2}{\sigma^2}\right)} = \frac{(t - K)}{S} \tag{A8}$$

where $S = a^2/\sigma^2$ is the input signal-to-noise ratio.

This model can be used to estimate the number of errors along a directionally filtered path. Let ϕ be the angle between a linear

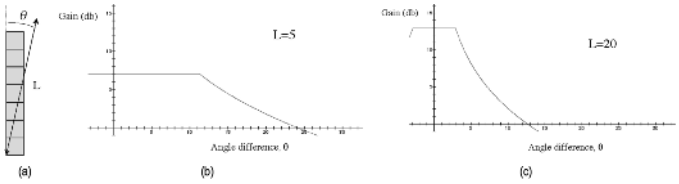

Fig. 17. (a) Directional filter and gain versus angle for (b) $L = 5$ and (c) $L = 20$.

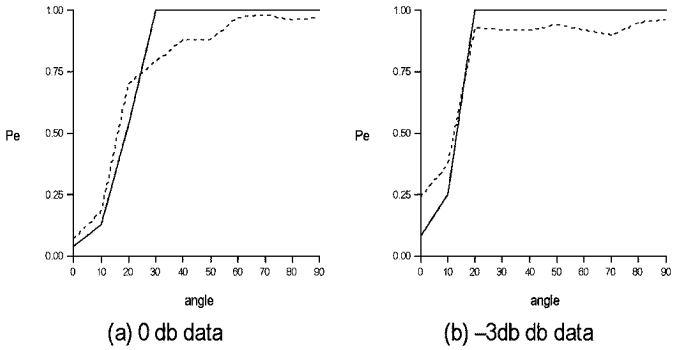

Fig. 18. (Solid) Predicted and (dotted) measured errors as a function of angle.

feature, and a directional filter of length . The SNR at the output of the filter as a function of angle is (Fig. 17)

$$S(\phi) = \frac{[a \max(1, \min(L, \cot \phi))]^2}{L\sigma^2}. \tag{A9}$$

Inserting this into (A8) provides an estimate of the number of errors as a function of angle

$$N(\phi) = \frac{t - K}{S(\phi)}. \tag{A10}$$

This is an approximation since the output from the DFB is not white but correlated in the filter direction. Fig. 18 plots the predicted and actual error probabilities (number of errors divided by $K = 100$) versus angle for the 0- and -3-dB data (Fig. 2) in 10° increments. As the angle, increases the model (A10) over-estimates the number of errors. Generally, though, the number of errors decreases with angle, implying that more votes will tend to be accumulated at locations along the line.

REFERENCES

[1] A. C. Copeland, G. Ravichandran, and M. M. Trivedi, "Localized radon transform-based detection of ship wakes in SAR images," *IEEE Trans. Geosci. Remote Sens.*, vol. 33, no. 1, pp. 35–45, Jan. 1995.

[2] G. D. Forney, "The Viterbi algorithm," *Proc. IEEE*, vol. 61, no. 3, pp. 268–278, Mar. 1973.

[3] M. A. Fischler, J. M. Tenenbaum, and H. C. Wolf, "Detection of roads and linear structures in low-resolution aerial imagery using a multisource knowledge integration technique," *Comput. Graph. Image Process.*, vol. 15, pp. 201–233, 1981.

[4] A. Rosenfeld and J. J. Pfaltz, "Sequential operations in digital picture processing," *J. Assoc. Comput. Mach.*, vol. 13, no. 4, pp. 471–494, Oct. 1966.

[5] D. M. McKeown, Jr and J. L. Denlinger, "Cooperative methods for road tracking in aerial imagery," in *Proc. Computer Vision and Pattern Recognition*, pp. 662–672.

[6] R. Samadani and J. F. Vesecky, "Finding curvilinear features in speckled images," *IEEE Trans. Geosci. Remote Sens.*, vol. 28, no. 4, pp. 669–673, Jul. 1990.

[7] J. K. Jao, C. F. Lee, and S. Ayasli, "Coherent spatial filtering for SAR detection of stationary targets," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 35, no. 2, pp. 614–625, Apr. 1999.

[8] S. M. Tonissen and R. J. Evans, "Performance of dynamic programming techniques for track-before-detect," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 32, no. 4, pp. 1440–1451, Oct. 1996.

[9] I. S. Reed, R. M. Gagliardi, and L. B. Stotts, "A recursive moving-target-indication algorithm for optical image sequences," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 26, no. 3, pp. 432–440, May 1990.

[10] Y. Barniv, "Dynamic programming solution for detecting dim moving targets," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 21, no. 1, pp. 144–156, Jan. 1985.

[11] L. R. Ford, "Network Flow Theory," Rand Tech. Rep. P-923, 1956.

[12] M. Styner, T. Coradi, and G. Gerig, "Brain morphometry by distance measurement in a non-Euclidean, curvilinear space," in *Proc. Image Processing in Medical Imaging*, vol. 1613, Lecture Notes in Computer Science (LNCS), pp. 364–369.

[13] N. Merlet and J. Zerubia, "New prospects in line detection by dynamic programming," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 18, no. 4, pp. 426–431, Apr. 1996.

Mark J. Carlotto was born on May 16, 1954 in New Haven, CT. He received the B.S., M.S., and Ph.D. degrees in electrical engineering from Carnegie Mellon University, Pittsburgh, PA, in 1977, 1979, and 1981, respectively.

He has held a variety of positions in academia and industry. From 1981 to 1993, he was a member of the technical staff at The Analytic Sciences Corporation (TASC), Reading, MA. He was also an Assistant Adjunct Professor with the College of Engineering, Boston University, Boston, MA, from 1981 to 1983. In 1993, he joined Pacific-Sierra Research (PSR), Arlington, VA, and remained with PSR until they were purchased by Veridian Systems in 2001. In 2003, Veridian was acquired by General Dynamics (GD). He is currently a senior staff scientist in GD's Advanced Information Systems Division. His research interests involve the development of automated image processing/understanding techniques for reconnaissance and remote sensing applications.