Some Defects in Finite-Difference Edge Finders

Margaret M. Fleck

Abstract— **This paper illustrates and explains various artifacts in the output of five finite difference edge finders (those of Canny, Boie and Cox, and three variations on that of Marr Hildreth) reimplemented with a common output format and method of noise suppression. These artifacts include gaps in boundaries, spurious boundaries, and deformation of region shape.**

Index Terms—**Boie-Cox edge finder, boundary representation, Canny edge finder, edge finder artifacts, edge finding, edge tracking, Marr-Hildreth edge finder.**

I. Introduction

MANY RECENT edge finders are based on finite differences of image intensities or derivatives implemented using finite differences. They differ in the orders of the differences used, as well as in the method of combining differences from different directions. Although numerous legends circulate about the relative merits of different algorithms, there have been few controlled studies. This paper illustrates and explains some differences in performance that are visible in nearest-pixel boundary representations, including artifacts such as gaps in boundaries, spurious boundaries, and large distortions of shape. Such differences will be of interest to researchers developing new edge finders, trying to choose among existing algorithms, or looking for problems to analyze theoretically.

Five algorithms will be examined. The Canny algorithm reports boundaries at peaks of the first difference in the gradient direction. The Boie-Cox algorithm is similar but picks the maximum first difference and uses a directional second difference for localization. The (slightly modernized) Marr-Hildreth algorithm locates boundaries at zero-crossings of the Laplacian, while also checking the sign of the first difference in the direction of the Laplacian's gradient. The directional Marr-Hildreth and the Max-Second algorithm replace parts of the Marr-Hildreth computation with directional differences.

Two published algorithms may differ in many ways, making it difficult to establish what causes a particular difference in behavior. For this reason, the algorithms in this paper have been reimplemented using a uniform method of suppressing noise (Section III)[1] and a uniform output representation (Section II). Obvious deficiencies in the algorithms have been repaired, and attempts have been made to pin down details left vague in their published descriptions so that the true worth of each method can be accurately assessed.

Manuscript received July 14, 1989; revised June 26, 1991. This work was supported by a Junior Research Fellowship funded by British Petroleum.

The author was with the Department of Engineering Science, University of Oxford, Oxford, England. She is now with the Department of Computer Science, University of Iowa, Iowa City, IA 52242.

IEEE Log Number 9104993.

[1] A separate study [1] compares several methods of suppressing noise.

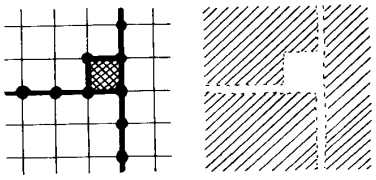

Fig. 1. Left: a cell complex with boundaries. Some boundaries lie on cells (shaded). Others lie on the sides or vertices between cells (thick lines and dots). Right: the region topology of the image modeled by deleting all points in the boundaries.

II. The Output Representation

The new implementations use a common output format, which allows on-cell boundaries, intercell boundaries, and combinations of the two. An image has been modeled as a complex of space-filling cells, as is illustrated in Fig. 1.[2] A piece of boundary can lie either on a cell, on the common side of two cells, or on the common vertex of four cells. The set of boundaries is, however, required to be a closed set, that is, 1) if a cell belongs to the boundaries, its sides must also belong to the boundaries, and 2) if a side belongs to the boundaries, its ends must also belong to the boundaries. Boundary thickness is used to encode uncertainty in location.

Image values are regarded as samples from an underlying continuous function, and thus, the output of an edge finder is regarded as a description of features in a continuous intensity surface. As in [2] and [4], the effect of boundaries on an image is modeled by deleting all points in the boundaries from the image (Fig. 1). The remaining open set is given the topology, metric, and differential structure that it inherits as a subspace of the image. This model of the regions in an image avoids the connectivity paradoxes found by [5]–[7], and it generalizes easily to other tesselations [2]. Further, boundary thickness does not affect the topology.

The choice between on-cell and intercell boundaries depends primarily on the edge operators used. If the algorithm uses a first or third difference mask centered between two cells or a second difference mask centered on a cell, boundaries should be placed between cells. Using on-cell boundaries requires complicated tests [8] to avoid a systematic bias in boundary locations. Output from other types of masks requires on-cell boundaries.

Some types of fine texture can only be represented accurately with intercell boundaries because on-cell boundaries use up cells that could otherwise represent bits of region. However, intercell boundaries alone cannot properly represent a boundary that falls near the middle of a cell. In the presence

[2] Formally, a specialized type of regular cell complex [2], [3].

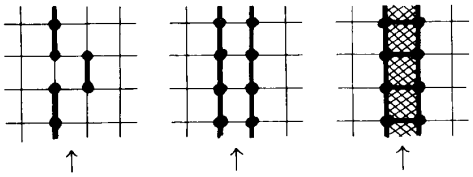

Fig. 2. Boundary lying right in the middle of a cell (arrows show intended boundary location) cannot be successfully represented using only intercell boundaries. Camera noise prevents a stable choice between the two closest locations, creating gaps (left). Choosing both locations creates an extra region (middle). Adding on-cell boundaries (right) ensures the correct region topology.

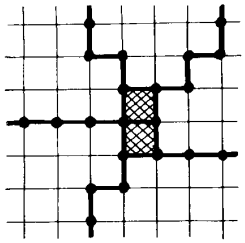

Fig. 3. Thickened boundaries make it easier to represent junctions that do not fit nicely into the image digitization.

of noise, the edge finder will be unable to make a stable choice between the two sides of the cell. Marking both or making an unstable choice between them causes mistakes in the region topology. This can be avoided by adding on-cell boundaries (Fig. 2). On-cell boundaries also allow a better approximation of junctions that fit poorly into the cell structure (Fig. 3).

Thinning would be required to make this output suitable for standard boundary trackers. It is easier to track region borders, as in [9]. Define a piece of region border to be a boundary vertex or side, together with a nonboundary cell of which it is a face. The boundaries in Fig. 4 generate three connected chains of regions borders. Region borders are easier to handle than boundaries because they always form thin, simple closed curves,[3] even when boundaries are thick, branch, or end abruptly. To track borders, first handle any isolated boundary vertices. Then, start at any cell side in the border. To move to the next cell side in the border, keeping the boundary to the left, the four continuations in Fig. 5 are tried in the order shown.

III. Method of Noise Suppression

All edge finders implemented in this paper use a common method of noise suppression, motivation for which can be found in [1]. Before processing, the input image is smoothed with a 2D Gaussian mask of standard deviation 1 cell.[4] Edge operators are then applied in ways that depend on the particular algorithm. The output of each operator is thresholded, i.e., responses below a given level are treated as if they were zero.

[3] Except where they run off the edges of the image. This special case can be avoided by encircling the image with a one-cell-wide band of nonboundary cells.

[4] Smoothed values are stored to the nearest 20th of an intensity unit.

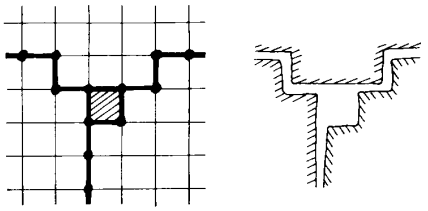

Fig. 4. Y-shaped boundary (left) generates three connected chains of region borders (right).

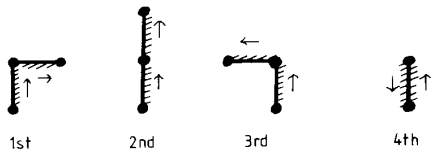

Fig. 5. Tracker can continue forward from a border cell side, keeping the boundary to its left, in one of four ways. They should be tried in the order shown.

No hysteresis, thinning, noise estimation, or similar algorithms were used, even if present in the original implementations.

Noise in our camera system has been measured [1] and can be approximated as additive Gaussian noise of standard deviation 1.0 intensity units (IU) (1.04 IU when quantization effects are included). The threshold for each operator is set at 5.0 times the expected deviation of noise in the operator's output when applied to camera noise. For finite differences, this can be found by convolving the finite difference mask with the digitized Gaussian mask used in smoothing, taking the root sum of squares of the mask weights, and multiplying by the standard deviation of camera noise (1.04). Nonlinear operators were approximated by linear ones; therefore, thresholds may not be precisely comparable. However, experience suggests that the artifacts discussed in this paper are not sensitive to exact threshold settings.

These algorithms differ only slightly in overall ability to separate real features from noise. All require that the response of either a second difference or a third difference operator be significantly different from zero (even Canny's algorithm; see below). A third difference operator has a worse signal-to-noise ratio overall. However, in the immediate vicinity of a step edge, second differences are near zero, whereas third differences are near their peak values. Thus, when actual detection thresholds are calculated (see [1]), third difference tests turn out to be as reliable and sometimes more reliable.

IV. Canny's Algorithm

Canny's algorithm [10], [11] marks boundaries at maxima in the magnitude of the image gradient in the direction of the gradient. The following algorithm is copied, except where noted, from Canny's original LISP implementation and gives almost identical output. Similar algorithms are described in [12]–[18].

Canny's code used the masks

$$\begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} -1 & -1 \\ 1 & 1 \end{bmatrix} \tag{1}$$

to compute the X and Y components of the gradient. However, this method creates a systematic bias in the boundary locations. Instead, the new implementation uses the mask $[-1, 0, 1]$ to compute first differences in four directions: H (horizontal), V (vertical), D_1, and D_2 (diagonal). The X and Y components of the gradient are computed by projecting the diagonal differences onto the axes[5]:

$$X = H + \frac{D_1 + D_2}{2}$$
$$Y = V + \frac{D_1 - D_2}{2}.$$

The amplitude of the gradient is then

$$A = \sqrt{X^2 + Y_2}.$$

At each cell, a value is then interpolated for a pseudo-cell in the gradient direction. Values are extracted from the two neighboring cells closest to the gradient direction: one from a horizontal or vertical neighbor (A_{hv}) and one from a diagonal neighbor (A_d). Let B be the larger of the magnitudes of X and Y, and let S the smaller. The pseudo-cell's value is then

$$A_g = \frac{(B - S)A_{hv} + SA_d}{B}.$$

A value A_{-g} for a pseudo-cell in the opposite direction is computed in the same way. A cell (x, y) is marked as a boundary if $A(x, y)$ is greater than a threshold T_1 (currently 3.00 IU), $A(x, y) \geq A_g(x, y)$, and $A(x, y) > A_{-g}(x, y)$.

Figs. 6–8 show the output of this algorithm on a real camera image and a test pattern (described fully in the Appendix). The output for the real image looks noisy. Boundaries for the synthetic image wobble, and there are spurious boundaries in the slope. T_1 is often set higher to disguise these problems, but they actually reflect a flaw in Canny's design. His algorithm marks a cell as a maximum if its amplitude is larger than that of its neighbors *without checking that these differences are higher than those expected from random noise.* This makes the algorithm slightly more sensitive to faint edges than, e.g., a Marr-Hildreth [19] but causes it to report spurious and unstable boundaries wherever there is a nontrivial slope in intensities, such as on smoothly shaded objects and on blurred boundaries.

These problems can be eliminated with a better peak detector. Just adding a threshold to Canny's peak detector eliminates real boundaries if they lie approximately halfway between two cells. Instead, the new implementation interpolates two additional pseudo-cell value A_{2g} and A_{-2g} using values 2 cells from the starting location. A cell (x, y) is then marked as a boundary if

$$\begin{aligned} A(x, y) &> T_1 \\ A_g(x, y) - A(x, y) &\leq T_2 \\ A_{-g}(x, y) - A(x, y) &\leq T_2 \\ A(x, y) - A_{2g}(x, y) &> T_3 \quad \text{and} \\ A(x, y) - A_{-2g}(x, y) &> T_3. \end{aligned}$$

T_2 and T_3 are currently set at 3.75 and 5.1 IU.

Fig. 9 shows the output of Canny's algorithm with the improved peak detector. It produces gaps at the sharp corners

[5]The $\frac{1}{2}$ weight on the diagonal contributions is the product of two weights of $\frac{1}{\sqrt{2}}$. One of these corrects for the longer intercell distance used in the diagonal differences. The second is the appropriate weight for projecting the diagonal vectors onto the two axes, i.e, $\sin(45°) = \cos(45°)$.

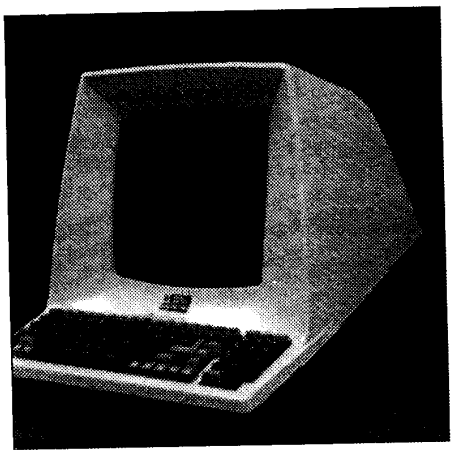

Fig. 6. 256 by 256, 8-b image of a terminal.

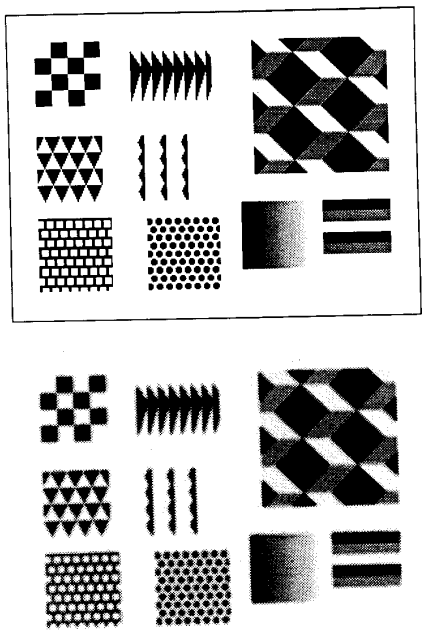

Fig. 7. 300 by 225 synthetic test pattern (top) and the degraded version of the image given to the edge finders (bottom). The patterns are named: (left) square checkerboard, triangular checkerboard, bricks, (middle) triangles, wavy bars, dots, (right) Clark pattern slope, staircase. For more details, see Appendix.

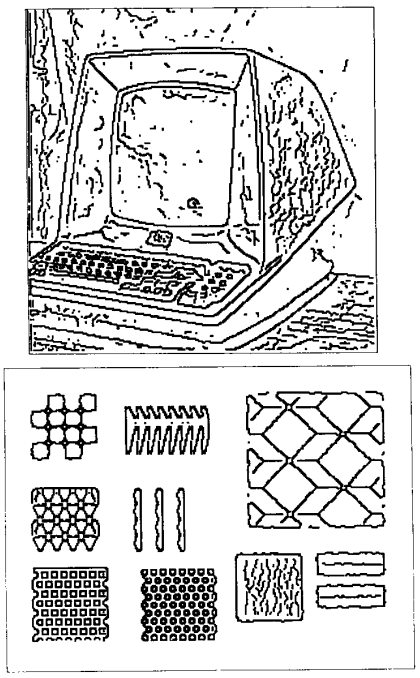

Fig. 8. Output of Canny's algorithm on the two test images.

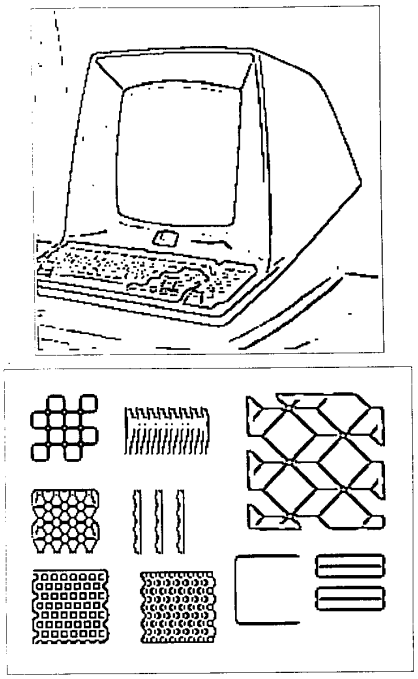

Fig. 9. Output of Canny's algorithm with improved peak detection on the two test images.

in the triangles, at the ends of the wavy bars, in the pattern of dots, and at some junctions. There are spurious ring-like boundaries around junctions of four or more regions and spurious dots in the triangles (perhaps related). Output using Canny's original peak detector (Fig. 8) is similar but has fewer broken corners.

V. Boie-Cox Algorithm

Boie and Cox [20], [21] have proposed an algorithm similar to Canny's but they replace the peak detector with second difference zero crossings and the gradient direction with the direction of maximum first difference (cf. also [16], [22]). Published descriptions of this algorithm are missing vital details, which were inferred from the authors' code.

The new code computes first and second differences in four directions, using the masks $[-1, 0, 1]$ and $[-1, 1, 1, -1]$. Let M be the amplitude of the maximum first difference, F and S the first and second difference values in some direction, and (dx, dy) a one-pixel displacement in the same direction.[6] For a cell (x, y) to be marked as a boundary in this direction, the first difference must be nontrivial and close to the maximum:

$$|F(x,y)| > T_1 \quad \text{and} \quad M(x,y) - |F(x,y)| \leq T_2.$$

The cell must also be at a second-difference zero-crossing, i.e., one of the following, or one of the similar expressions with $<$ and $>$ reversed must hold:

1) $S\left(x + \frac{dx}{2}, y + \frac{dy}{2}\right) > T_3$ and $S\left(x - \frac{dx}{2}, y - \frac{dy}{2}\right) < -T_3$, or
2) $|S(x + \frac{dx}{2}, y + \frac{dy}{2})| \leq T_3$, $S\left(x + \frac{3dx}{2}, y + \frac{3dy}{2}\right) < -T_3$ and $S\left(x - \frac{dx}{2}, y - \frac{dy}{2}\right) > T_3$, or
3) $|S\left(x - \frac{dx}{2}, y - \frac{dy}{2}\right)| \leq T_3$, $S\left(x - \frac{3dx}{2}, y - \frac{3dy}{2}\right) < -T_3$ and $S\left(x + \frac{dx}{2}, y + \frac{dy}{2}\right) > T_3$.

Currently T_1 and T_2 are 1.95 IU, and T_3 is 2.25 IU. In the original Boie and Cox design, T_3 is zero. Just like the missing threshold on Canny's peak detector, this causes wobbly boundaries and spurious boundaries on slopes (Fig. 10). Boie and Cox set T_1 very high to compensate.

As Fig. 11 shows, this edge finder produces rings around junctions of four or more regions, like Canny's algorithm. It breaks only a few sharp corners, but others have short spines sticking out of them. There are spurious boundaries in the staircase pattern at some junctions in the Clark pattern and along the bottom edge of the terminal keyboard. Except for the broken corners, these faults occur even when T_3 is zero (Fig. 10).

VI. Marr-Hildreth Algorithm

The Marr and Hildreth algorithm ([23], [24], cf. also [25]) locates boundaries at the zero-crossings of the Laplacian of the image. The original version of this algorithm did not contain sufficient safeguards to prevent spurious boundaries [26], [27] or to prune responses due to noise. Thus, a clean, updated version is required to illustrate its full potential.

The Laplacian is equal to the sum of the directional second differences in all directions. The new implementation

[6] I.e., a vector of unit length horizontally or vertically, length $\sqrt{2}$ diagonally.

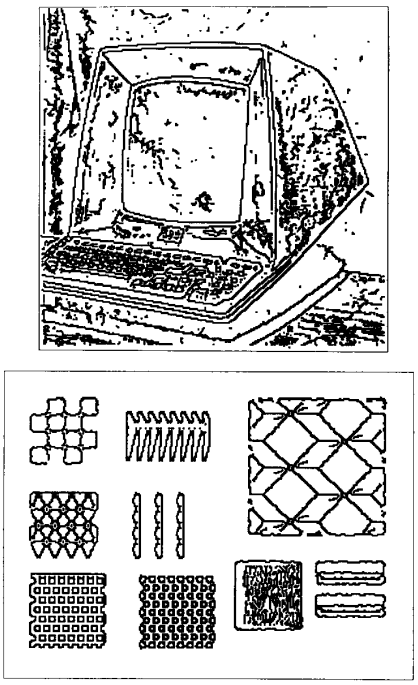

Fig. 10. Output of Boie and Cox's algorithm on the two test images.

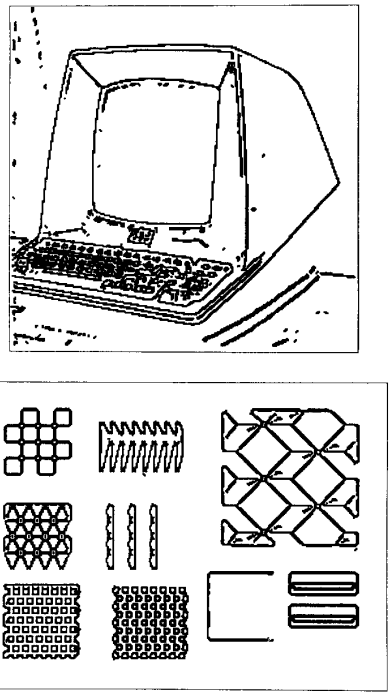

Fig. 11. Output of Boie and Cox's algorithm with thresholding of second differences on the two test images.

computes this using the mask:

$$\begin{pmatrix} \frac{1}{\sqrt{2}} & 1 & \frac{1}{\sqrt{2}} \\ 1 & -4-2\sqrt{2} & 1 \\ \frac{1}{\sqrt{2}} & 1 & \frac{1}{\sqrt{2}} \end{pmatrix}.$$

Laplacian responses below a noise threshold T_2 (currently 3.85 IU) are reset to zero.

The Laplacian is well suited to localizing boundaries but not to detecting them. First, some zero-crossings of the Laplacian do not correspond to sharp changes in intensity [26], [27]. Second, the Laplacian amplitude often drops below T_2 in the middle of a boundary, sometimes for multiple cells if the boundary is blurred. It is difficult to distinguish these zeros from zeros in the middle of uniform-intensity regions. (Many implementors must have encountered this problem, but it seems to have received no theoretical attention.)

Therefore, the new implementation adds a second detection map to the algorithm. It computes the gradient of the Laplacian, as in [23], but goes on to compare its sign with that of the image first difference in the same direction (computed by adapting Canny's pseudo-value code). The detector responds at a cell if the amplitudes of the first and third differences are over noise thresholds T_1 and T_3 (currently 1.95 and 11.95 IU) and the two differences have opposite signs. A cell is then marked as a boundary if the detector responds and the Laplacian is zero. The common side of two cells is marked as an intercell boundary if the detector responds at both cells, and they have Laplacian responses of opposite, nonzero signs.

This method of combining first and third differences is an extension of the proposals in [17], [26]–[29]. It would also be possible to take both differences in the image gradient direction [17] or take a vector combination [27]. Some researchers may have been discouraged from using such methods by the poor results presented in Clark's paper [27]. Clark does not explain why he found it necessary to use such large smoothing masks (standard deviation 10 cells). A direct implementation of his mathematics, however, would involve examining the sign of the product of the first and third differences. This would be substantially more sensitive to noise than testing the two signs individually.

The new detection map may have uses beyond the immediate problem of edge detection. For well-separated boundaries, the edges of the detector regions mark the peaks and troughs of the second difference, which is believed by some [30] to be useful in stereo matching. The width of the response regions could be used in estimating boundary blur [31]. Finally, the detector map can help separate the second difference responses of roof edges from those of nearby step edges, making them easier to identify [32].

Fig. 12 shows the output of the new algorithm. It produces slight distortions of boundary shape near the junctions in the Clark pattern. Some junctions in the Clark pattern, the hexagonal checkerboard, and the staircase have small gaps. There is a spurious boundary inside the end of the slope, and the regions in the brick pattern have been rounded.

VII. The Directional Marr-Hildreth and Max-Second Algorithms

The final two algorithms illustrate ways of adapting the Marr-Hildreth algorithm to use directional differences. The

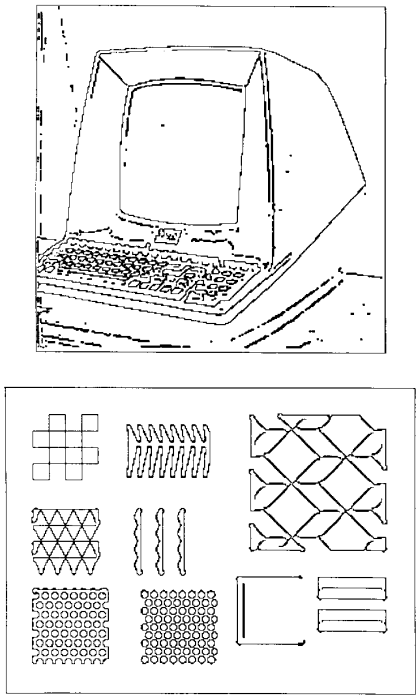

Fig. 12. Output of the Marr and Hildreth algorithm on the two test images.

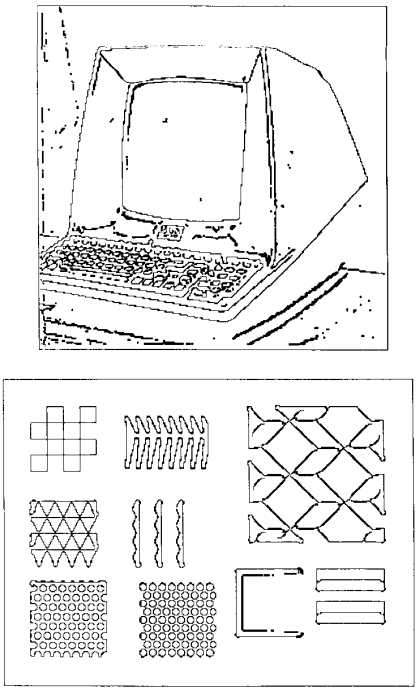

Fig. 13. Output of the directional Marr and Hildreth algorithm on the two test images.

first, the directional Marr-Hildreth, continues to use the Laplacian for location. The detection map, however, is obtained by computing the directional first and third differences in four directions using the masks $[-1, 0, 1]$ and $[-1, 2, 0, -2, 1]$. Values below T_1 and T_3 (currently 1.95 and 3.05 IU) are reset to zero. The detector responds if there is some direction in which the first and third differences have opposite nonzero signs.

The Max-Second algorithm uses the same detection map as the directional Marr-Hildreth. For location, however, second differences are computed in four directions using the mask $[1, -2, 1]$, Diagonal differences are divided by $\sqrt{2}$. The maximum positive P and minimum negative N differences at each cell are then extracted. If $P + N$ is greater than a threshold T_2 (2.10 IU), the cell in the location map is assigned the value P.[7] If $P + N < -T_2$, it is assigned the value N. Otherwise, it is set to zero.

Figs. 13 and 14 show the output of the two variant algorithms on the test images. These are almost identical to the output of the Marr-Hildreth algorithm. The only significant difference is that these algorithms generate spurious boundaries along the sides of the slope pattern as well as along its end.

VIII. Spurious Boundaries

The most noticable edge finder artifacts are reports of boundaries in regions entirely lacking sharp intensity changes.

[7] In other words, P must not only be larger than N, but this difference must be distinguishable from the effects of noise.

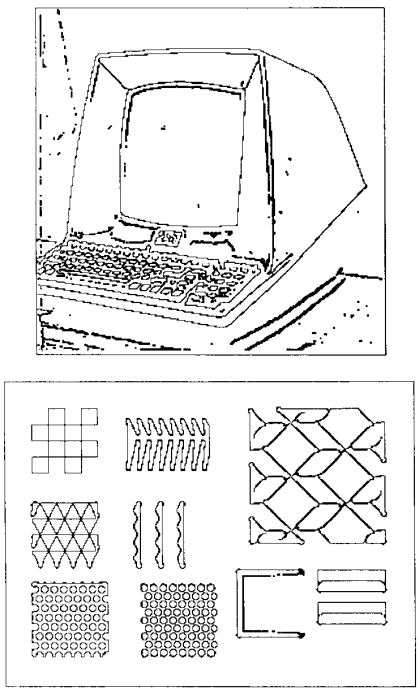

Fig. 14. Output of the Max-Second algorithm on the two test images.

The Boie-Cox algorithm (as well as the original Marr-Hildreth code) produces spurious boundaries in staircase intensity patterns and at staircase-like region junctions for reasons de-

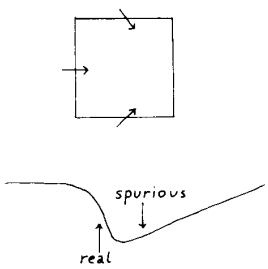

Fig. 15. Straight across the end of the slope in the test image and diagonally across its sides, the smoothed intensities have roughly the shape shown. The transition to zero in the directional second differences occurs while the first and third differences still have opposite, nonzero signs, generating a spurious boundary in addition to the real one.

scribed in [16], [19], [27], and [33]. The other algorithms prevent this by comparing the sign of the first difference to the sign of the third difference or ensuring that the magnitude of the first difference is a local maximum.

It is less well known that spurious boundaries can occur on slopes. We have seen that the original versions of the Canny and Boie-Cox algorithms generate spurious and unstable boundaries throughout the slope because they (effectively) examine the signs of second differences without ensuring that these signs are reliable. The three variants of the Marr-Hildreth algorithm produce spurious boundaries inside the end of the slope, and two produce them inside its edges. As Fig. 15 illustrates, these are not proper zero crossings, but rather transitions to zero in which the detector responds after the location map has already become zero. It is unclear how to avoid these artifacts since similar response patterns occur at real but faint boundaries in other images.

IX. Sharp Corners

The five edge finders show a variety of artifacts at sharp corners: distortions of shape, gaps, and spurs. Theoretical analysis has been carried out only for the Canny and Marr-Hildreth methods and only for noise-free images. Zero crossings of the Laplacian were found [26] to go through the true corner but bulge out on the sides. Maximum errors[8] were $\sqrt{2}$ cells for an infinitely acute corner and 0.9 cells for corners more than 30°, which is barely large enough to observe in nearest pixel representations. The gradient peak boundary [19] follows the sides of the corner closely but is flattened in at the end, missing the true corner entirely. The maximum deviation is large: about 2.3 cells for a 30° corner and unbounded in the worst case.

When noise is present, the Canny and Boie-Cox algorithms break some sharp corners. There seem to be two distinct causes. First, the second difference in the gradient direction, i.e., straight out the corner, may have too low an amplitude to be reliably distinguished from zero. The three Marr-Hildreth variations avoid this problem by using third differences and by taking their second differences in several directions, including across the corner.

[8]The current implementation uses smoothing of standard deviation 1 cell applied twice—once in the camera and once in the edge finder—giving a total standard deviation of $\sqrt{2}$ cells.

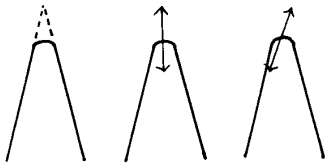

Fig. 16. On a sharp corner (left, dotted line), gradient peak algorithms return a flattened corner (left, solid line). Locations near the tip of the curve are amplitude peaks in the gradient direction (middle). However, small errors in computing the gradient direction (right) can make the peak detector compare the magnitude at the tip to a magnitude along the side of the corner, which is much higher.

However, even if all thresholds are tuned low, Canny's algorithm still breaks corners. Ideally, the magnitude at the tip of the reported boundary is a maximum in the gradient direction. However, points along the sides of the corner lie very close to this gradient direction and have magnitudes much higher than that at the tip (Fig. 16). As noted in [15], a slight error in computing the gradient direction, e.g., due to camera noise, can cause the tip not to be marked as a peak.

Finally, the Boie-Cox algorithm sometimes produces spurs protruding from a sharp corner. This has not received detailed theoretical attention. However, the problem seems to be that the second-difference zero crossings from different directions occur in noticably different places, but the first differences in these directions are all at or near the maximum amplitude. This effect is markedly worse if intercell boundaries are used.

X. Junctions

When three or more regions touch at or near a common point, the local configuration will be referred to as a junction. A theoretical analysis in [19], assuming no noise, predicts distortions of shape at three-region junctions for both the Canny and the Marr-Hildreth algorithms, as well as a gap separating one boundary from the other two. These effects seem to be pronounced only for the Marr-Hildreth variations and, even then, gaps are often closed because second differences fall below the noise threshold.

Junctions involving more than three regions have not been formally analyzed. Although they produce connected boundaries for the square checkerboard, all five algorithms produce frequent gaps on more complex junctions, e.g., in the Clark pattern or the triangular checkerboard. In addition, the Canny and Boie-Cox algorithms produce ring-like structures around many junctions involving more than three regions. These are visible in some previously published results [17] but rarely commented on. To understand why they occur, consider the square checkerboard. The gradient must be zero at the vertex, near zero in the middle of each region, but significantly different from zero along the diagonal line segment joining these two points (Fig. 17). Since the gradient direction for points on the diagonal lies along the diagonal, there must be a gradient peak somewhere along it.

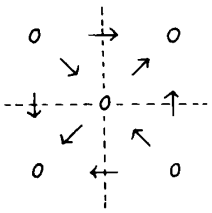

Fig. 17. Sketch of the gradient directions for a vertex in the square checkerboard pattern. Along a diagonal path from the middle of a square to the vertex, there must be a peak in the gradient amplitude.

XI. Conclusions

In the previous sections, we have seen that these closely related algorithms all produce various unwanted artifacts:

- Spurious and unstable boundaries on slopes (Canny, Boie-Cox)
- spurious but stable boundaries in staircase patterns (Boie-Cox) or near the edges of slopes (Marr-Hildreth, Max-Second, directional Marr-Hildreth)
- gaps or spines at sharp corners (Canny, Boie-Cox)
- small deformations at sharp corners (all algorithms)
- rings at junctions (Canny, Boie-Cox)
- gaps and small distortions at junctions (all algorithms).

Which artifacts seem most obtrusive will depend on the intended application. All the algorithms are easy to implement, run quickly, and produce clean outputs.

Perhaps the most surprising result of this study is the similarity of the algorithm outputs. In particular, the modernized Marr-Hildreth algorithm seems to perform at least as well as Canny's. Some of the differences between previous implementations seem to be due to differences in thinning, hysteresis, or subpixel interpolation algorithms, as well as differences in threshold settings. A few may be due to small differences in the shape of the smoothing filters or the derivative approximations [10], [11], [14], [20], [21], but there is no direct evidence to support this. Very different methods of suppressing noise (e.g., [1], [17]) could clearly have larger effects.

However, it is discouraging to uncover major differences caused by design bugs or unclear description of algorithm details. Adding a second-difference threshold markedly improves the performance of the Canny and Boie-Cox algorithms. The new detection map improves Marr-Hildreth performance far beyond that which might be supposed from Clark's results. Canny's method of pseudo-cell value computation is not published, nor is the fact that the Boie and Cox algorithm works extremely poorly if intercell boundaries are used. Could such details perhaps be of more practical significance than the exact shape of the optimal smoothing filter?

Appendix
Details of Test Pattern

The test pattern is a 300 by 225, 8-b image, with white (255) background. To simulate real imaging conditions, its contrast was reduced to 90% (range [13 242]), it was blurred with a Gaussian mask of standard deviation 1 cell, and finally, Gaussian noise of standard deviation 1.0 IU was added.

The patterns in the left-hand two columns are two valued (0 and 255), and the true boundaries lie between pixels. The gaps in the dot pattern (middle bottom) vary slightly in size. The pattern in the upper right-hand corner is a simulation of a pattern from [27], using the values 0, 127, and 254. Boundaries lie on cells, and these cells have intermediate values. To the left below the Clark pattern is an even slope from 0 to 255: values in adjacent columns differ by 5 IU. To its right are bars at values 0, 128, and 255, forming a staircase intensity pattern whose true boundaries lie between pixels.

Acknowledgment

This paper has been much improved by comments from M. Brady, D. Forsyth, and A. Noble.

References

[1] M. Fleck, "Multiple widths yield reliable finite differences," to be published in *IEEE Trans. Patt. Anal. Machine Intell.*
[2] ——, "Boundaries and topological algorithms," Ph.D. thesis, Mass. Inst. Technol.; AI Lab. TR-1065, 1988.
[3] J. R. Munkres, *Elements of Algebraic Topology.* Menlo Park, CA; Addison-Wesley, 1984.
[4] M. M. Fleck, "Representing space for practical reasoning," *Image Vision Comput.*, vol. 6, pp. 75–86, 1988.
[5] T. Pavlidis, *Structural Pattern Recognition.* Berlin: Springer-Verlag, 1977.
[6] J. P. Hayes, "Naive physics 1: Ontology for liquids," in *Formal Theories of the Commonsense World* (J. R. Hobbs and R. C. Moore, Eds.). Norwood, NJ: Ablex, 1985.
[7] E. Davis, "Shape and function of solid objects: Some examples," Tech. Rep. 137, Dept. of Comput. Sci., Courant Inst. Math. Sci., New York Univ., NY, 1984.
[8] A. Huertas and G. Medioni, "Detection of intensity changes with subpixel accuracy using Laplacian-Gaussian masks," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. PAMI-8, pp. 651–664, 1986.
[9] J. A. Noble, "Descriptions of image surfaces," D. Phil thesis, Dept. Eng. Sci., Univ. of Oxford, 1989.
[10] J. F. Canny, "Finding edges and lines in images," M.S. thesis, Mass. Inst. Technol.; AI Lab. TR-720, 1983.
[11] ——, "A computational approach to edge detection," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. PAMI-8, pp. 679–698, 1986.
[12] F. Bergholm, "Edge focusing," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. PAMI-9, pp. 726–741, 1987.
[13] R. Deriche, "Using Canny's criteria to derive a recursively implemented optimal edge detector," *Int. J. Comput. Vision*, vol. 1, pp. 167–187, 1987.
[14] ——, "Fast algorithms for low-level vision," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 12, no. 1, pp. 78–87, 1990.
[15] A. F. Korn, "Toward a symbolic representation of intensity changes in images," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 10, pp. 610–625, 1988.
[16] M. A. Gennert, "Detecting half-edges and vertices in images," in *Proc. IEEE Conf. Comput. Vision Patt. Recogn.*, 1986, pp. 552–557.
[17] R. M. Haralick, "Digital step edges from zero crossings of second directional differences," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. PAMI-6, pp. 58–68, 1984.
[18] L. A. Spacek, "Edge detection and motion detection," *Image Vision Comput.*, vol. 4, pp. 43–56, 1986.
[19] E. DeMicheli, B. Caprile, P. Ottonello, and V. Torre, "Localization and noise in edge detection," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 11, pp. 1106–1117, 1989.
[20] R. A. Boie, I. J. Cox, and P. Rehak, "On optimum edge recognition using matched filters," in *Proc. IEEE Conf. Comput. Vision Patt. Recogn.*, 1986, pp. 100–108.
[21] R. A. Boie and I. J. Cox, "Two dimensional optimal edge recognition using matched and Wiener filters for machine vision," in *Proc. Int. Conf. Comput. Vision*, 1987, pp. 450–456.
[22] D. E. Pearson and J. A. Robinson, "Visual communication at very low data rates," *Proc. IEEE*, vol. 73, pp. 795–812, 1985.

[23] E. C. Hildreth, "The detection of intensity changes by computer and biological vision system," *Comput. Vision Graph., Image Processing,* vol. 22, pp. 1–27, 1983.
[24] D. Marr and E. C. Hildreth, "Theory of edge detection," in *Proc. Roy. Soc. London B,* vol. 207, pp. 187–217, 1980.
[25] R. A. Young, "Simulation of human retinal function with the Gaussian derivative model," in *Proc. IEEE Conf. Comput. Vision Patt. Recogn.,* 1986, pp. 564–569.
[26] V. Berzins, "Accuracy of Laplacian edge detectors," *Comput. Vision Graph. Image Processing,* vol. 27, pp. 195–210, 1984.
[27] J. J. Clark, "Authenticating edges produced by zero crossing algorithms," *IEEE Trans. Patt. Anal. Machine Intell.,* vol. 11, pp. 43–57, 1989.
[28] F. Ulupinar and G. Medioni, "Refining edges detected by a LoG operator," in *Proc. IEEE Conf. Comput. Vision Patt. Recogn.,* 1988, pp. 202–207.
[29] J. S. Chen and G. Medioni, "Detection localization and estimation of edges," *IEEE Trans. Patt. Anal. Machine Intell.,* vol. 11, pp. 191–198, 1989.
[30] J. E. W. Mayhew and J. P. Frisby, "Psychophysical and computational studies towards a theory of human stereopsis," *Artificial Intell.,* vol. 17, pp. 349–385, 1981.
[31] R. J. Watt and M. J. Morgan, "The recognition and representation of edge blur: Evidence for spatial primitives in human vision," *Vision Res.,* vol. 23, pp. 1465–1477, 1983.
[32] M. M. Fleck, "Spectre: An improved phantom edge finder," in *Proc. Fifth Alvey Vision Conf.,* 1989, pp. 127–132.
[33] Y. Lu and R. C. Jain, "Behavior of edges in scale space," *IEEE Trans. Patt. Anal. Machine Intell.,* vol. 11, pp. 337–356, 1989.

Margaret M. Fleck was born in Madison WI, in 1961. She received the B.A. degree from Yale University in 1982 in linguistics and the M.S. and Ph.D. degrees from the Massachusetts Institute of Technology in 1985 and 1988, both in computer science.

From 1988 to 1991, she worked at the Department of Engineering Science, Oxford University, as a junior research fellow. In 1991, she joined the Department of Computer Science, University of Iowa, as an assistant professor. Her research interests include computer vision (both low-level and high-level) and computational linguistics.