An Efficient Computational Geometry Method for Detecting Dotted Lines in Noisy Images

F. DEHNE[1] AND L. FICOCELLI[2]

[1] *School of Computer Science, Carleton University, Ottawa, Ontario, Canada K1S 5B6*

[2] *Grande Prairie Regional College, Grande Prairie, Alberta, Canada T8V 4C4*

***In this paper we present an efficient* O(n *log* n) *time, linear space, algorithm for detecting a line, or line segment, represented by a set of* n_L *collinear points contained in a rectangular window with an additional set of* n_N *independent, uniformly distributed random noise points;* $n = n_L + n_N$. *Empirical results show that the algorithm is very reliable for* $n_L/n_N \geq \frac{1}{3}$.**

Received January 1989, revised April 1989

1. INTRODUCTION

In this paper, we study the well-known problem of detecting dotted lines in noisy images; i.e. detecting a line, or line segment, represented by a set of n_L collinear points (referred to as *line points*) contained in a circular or rectangular window with an additional set of n_N independent, uniformly distributed random *noise points*. Several methods for solving this problem have been proposed in the literature.[3,5,8,13,15,17] However, most of them are based on the Hough Transform, which is computationally very costly. This motivated our research in studying more efficient methods for line detection; in particular, we considered the application of efficient computational geometry methods, since some of these tools have already been successfully applied to other image processing and statistics problems.[16,18]

We present an efficient $O(n \log n)$ time, $O(n)$ space algorithm for detecting dotted lines in noisy images; $n = n_L + n_N$. Our method is based on convex hull and peeling algorithms. Empirical studies have shown that the algorithm is very reliable for up to three times as many noise points as data points; i.e. $n_L/n_N \geq \frac{1}{3}$.

The remainder is organized as follows. Section 2 will review the concept of convex hulls and peelings and discuss some statistical properties. In Section 3 we shall present our algorithm and discuss its rationale; while Section 4 will analyse its time complexity and discuss some empirical results with respect to the accuracy of our method.

2. HULLS AND PEELING

One of the most extensively studied structures in computational geometry is the *convex hull* of a planar point set $S = \{p_1, \ldots, p_n\}$; i.e. the smallest convex set containing S. The points of S located on the border of the convex hull are referred to as *extreme points* of S; removing the extreme points and iterating this process for the remaining points until all points have been removed is called *peeling*. A variety of algorithms has been proposed for computing the convex hull;[12,14] for peeling a set S of n points, Chazelle[2] has presented an optimal $O(n \log n)$ time, linear space algorithm.

In addition to its efficient computation, several authors have also studied the properties, in particular the statistical properties, of convex hulls.[1,6,10,16,18] Tukey suggested (as reviewed in Ref. 10) that the convex hull could be used for getting a robust estimator for mean values in higher dimensions similar to computing trimmed means for one-dimensional data sets; i.e. a fraction of the upper and lower extreme points of the data set is removed before the mean value is computed, thereby making it less sensitive to exceptional values in the data set. For two-dimensional data sets, a portion of the first hulls in the peeling process may be considered as exceptions and deleted.

Except for removing exceptions, the deletion of the extreme points does not have a considerable effect on the centroid of a two-dimensional point set. In fact, for the peeling process our experiments also show that, for random point sets, the centroid remains reasonably stable until the number of points becomes too small. Furthermore, we placed a second smaller but denser compact and convex point set (mass) within the convex hull of the first set of points (noise). We found that, for successive peelings (of the entire point set), the centroid of the whole set migrates towards the centroid of the mass. This is intuitively obvious since, for most cases, each peeling will eliminate more points from the noise than from the mass. In addition, the successive hulls tend to converge towards an approximation of the mass; however, if the mass is located close to the border of the convex hull of the noise this effect is not as pronounced, since an increasing number of points are deleted from the mass by the peeling process.

3. ALGORITHM OVERVIEW

The above observation motivated our development of an algorithm for locating line segments in a field with random noise. The rationale for the algorithm is that, since peeled hulls tend to converge to a convex and compact mass, it is perhaps possible that they can be made to converge to a line which is convex but, unfortunately, very thin (see Ref. 4 for an exact definition of thinness).

Experiments showed that, although lines do not behave as well with respect to peeling as a compact mass, there are some similarities. In particular, as we will point out in Section 3.1, it turns out that the hulls do not converge to the line but in general delete more noise points than line points and will, eventually, converge to some remainder of the line points (with several additional

noise points). Then this remainder will be peeled off by one hull. If it is possible to detect when this happens and separate, in this hull, the line points from the noise points, the other line points which have been peeled off by previous hulls can be recovered by a post-processing step.

Therefore, the general structure of our algorithm is the following.

Phase 1. Peel off the entire point set; store the created concentric hulls $h_1, \ldots, h_m$.

Phase 2. Detect which hull, h_k, has a 'significant' number of line points.

Phase 3. Delete noise points from h_k.

Phase 4. Recover the line points peeled off by $h_1, \ldots, h_{k-1}$.

In Sections 3.1, 3.2 and 3.3. we shall discuss Phases 1–3 of the algorithm in detail. Phase 4 is straightforward: once a subset of line points (with only very few noise points) has been extracted, all points close to the linear approximation can be determined by one linear search, thereby recovering all points on the original line or line segment.

3.1 The effect of peeling on collinear data points (Phase 1)

The basic idea for Step 1 of the algorithm is that, although lines do not behave as well as compact masses with respect to peeling, hulls h_1 to h_{k-1} will have a 'filtering' effect in that they remove more (in some worst cases, at least as many) noise points than line points.

Fig. 1 shows the most important cases which can occur while peeling $h_1, \ldots, h_{k-1}$. For line segments, there will be in general no line point contained in the first hulls (Fig. 1*a*); therefore, only noise is removed. When the concentric hulls become smaller, they will eventually contain at most two line points (otherwise, a hull contains all remaining line points and will be considered as containing a significant number of line points; see Section 3.2) and at least two noise points as depicted in Fig. 1*b*. For lines, the second case will occur immediately.

In the worst case, the number of removed noise and line points is identical; see Fig. 1*c*. A worst-case scenario (which can be easily detected and dealt with separately) is shown in Fig. 1*c*.

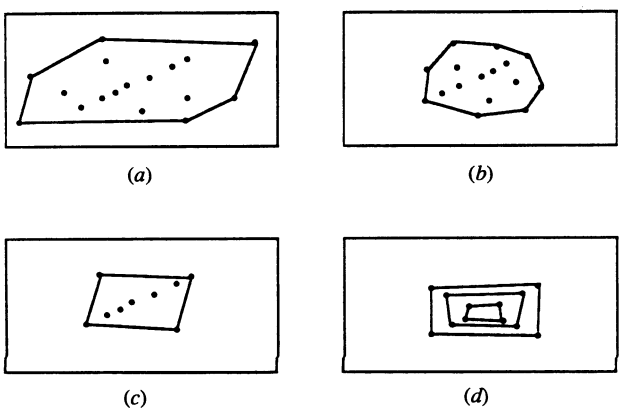


Fig. 1. Using convex hulls for noise removal. (*a*) Initial case for line segment. (*b*) Subsequent case for line segments and initial case for lines. (*c*) A case where as many line points as noise points are removed. (*d*) A worst-case scenario.

3.2 Detection of hull h_k with a significant number of line points (Phase 2)

At some stage of Phase 1, all points on one side of the line points have been removed; all remaining line points will be contained in the next convex hull and removed in the subsequent step. This hull, h_k, which contains a significant number of line points (except for the above worst case), is detected in Phase 2 of the algorithm. Note that k may be very small (i.e. h_k is peeled off very early) if, for example, the line is located close to a border of the window.

A typical example of a hull h_k is shown in Fig. 2.

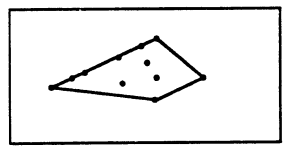


Fig. 2. A typical hull h_k.

Empirical studies with a large number of random data sets have shown that the hull h_k can be characterized by having one of the following two properties:

- an increased number of points on the hull boundary, or
- a small relative area (relative to the previous hull) per boundary point.

Let A_i and HP_i denote the area and the number of extreme points of hull h_i, respectively, and let

$$ANA_i := \frac{A_i}{A_{i-1} HP_i}$$

denote the *normalized average area* of hull h_i.

Our studies (see Section 4 and Appendix A) show that hull h_k can be characterized by either a significant increase in

$$\frac{HP_i}{n},$$

the relative number of extreme points of hull h_i, or

$$\frac{1}{ANA_i},$$

the reciprocal value of the normalized average area of h_i.

Therefore, our algorithm detects hull h_k by selecting from all hulls h_i the hull which maximizes

$$\alpha_i := \frac{HP_i}{n} * \frac{1}{ANA_i}.$$

Our experimental results (see Section 4) show that this measure is very reliable; Appendix A shows some typical plots of (HP_i/n) and ANA_i.

3.3 Deleting noise points from h_k (Phase 3)

Once the hull h_k has been computed, the next stage of the algorithm is to eliminate extreme points of h_k which are noise points.

Let $p_0, \ldots, p_{t-1}$ be the extreme points of h_k and let p and $a(p_i)$ denote the centroid of $p_1, \ldots, p_t$ and the angle of the polar coordinates of p_i with respect to centre p $(0 \leqslant i \leqslant$

$t-1$), respectively; finally, let $\beta(p_i) := |a(p_{i+1 \bmod t}) - a(p_i)|$ denote the difference of the angle of the polar coordinates (with centre p) of p_i and its successor $p_{i+1 \bmod t}$.

Our method for eliminating noise points of h_k is very simple; the rationale for it is that the angle (with respect to centre p) between noise points is larger than the angle between line points (see, for example, Fig. 2).

- Compute for each extreme point p_i of h_k the value $\beta(p_i)$.
- Discard all p_i with $\beta(p_i) > (360°/HP_i)$.

Although it is very simple, it turns out that this method is very accurate in that it deletes most of the noise points but hardly any of the line points of h_k (see Section 4 and Appendix A).

4. TIME COMPLEXITY OF THE ALGORITHM AND EMPIRICAL RESULTS

The most time-consuming step of the algorithm is the peeling process in Phase 1. As we have already indicated in Section 2, Chazelle[2] has presented an optimal $O(n \log n)$ time, linear space algorithm for peeling a set S of n points. It is easy to see that all other steps can be executed in linear time. Hence the entire algorithm has a time complexity and space requirement of $O(n \log n)$ and $O(n)$, respectively; therefore, this method is much more efficient than, for example, algorithms based on the Hough Transform.

On the other hand, the proposed methods proved to be very reliable. Table 1 summarizes some performance results obtained from extensive testing with randomly generated data sets.

Table 1. Percentage of correct answers for random data sets

	n_L+n_N			
n_L/n_N	100	200	300	
10/90	23	44	61	Horizontal or vertical lines
15/85	55	79	96	
20/80	79	100	100	
25/75	92	100	100	
10/90	0	8	12	Diagonal lines
15/85	15	57	71	
20/80	57	84	100	
25/75	80	100	100	
10/90	31	25	32	Arbitrary line segments
15/85	45	48	41	
20/80	52	80	81	
25/75	85	95	95	

It shows three classes of tests:

- randomly generated dotted horizontal or vertical lines with additional random noise,
- randomly generated dotted diagonal lines with additional random noise, and
- randomly generated arbitrary dotted line segments with additional random noise.

The first two cases were tested since they apply, for example, to the detection of particle paths in nuclear physics; for these cases, all answers were correct for up to 75–80% noise.

For arbitrary line segments, the performance was slightly inferior. However, for 75% noise a 95% correctness rate could still be achieved; for 70%, no incorrect answer was found.

Appendix A shows some sample plots. For each experiment, the original point set, all hulls determined in Phase 1 as well as the values (HP_i/n) and ANA_i for each hull h_i, the hull h_k selected in Phase 2, and the remaining points of h_k after Phase 3 are shown.

5. CONCLUSION

In this paper we have presented an efficient $O(n \log n)$ time, $O(n)$ space algorithm for detecting dotted lines in noisy images; i.e. detecting a line, or line segment, represented by a set of n_L collinear points contained in a circular or rectangular window with an additional set of n_N independent, uniformly distributed random noise points. Our method is based on efficient computational geometry methods: convex hull construction and peeling. Empirical studies have shown that the algorithm is very reliable for up to three times as many noise points as data points; i.e. $n_L/n_N \geqslant \frac{1}{3}$.

While the detection of one single line or line segment is important for a number of applications (for example, detection of particle paths in nuclear physics), there are obviously also applications which require several lines or line segments to be detected. The generalization of our method for the solution of such cases is currently under investigation.

Acknowledgement

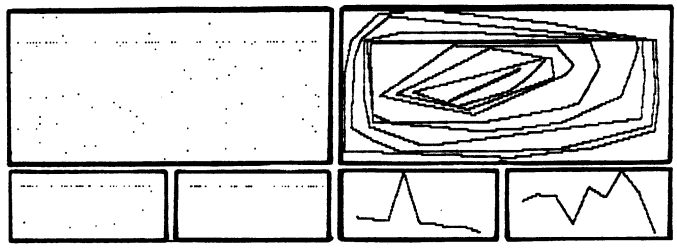
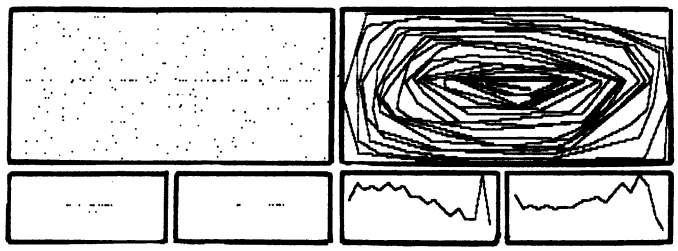
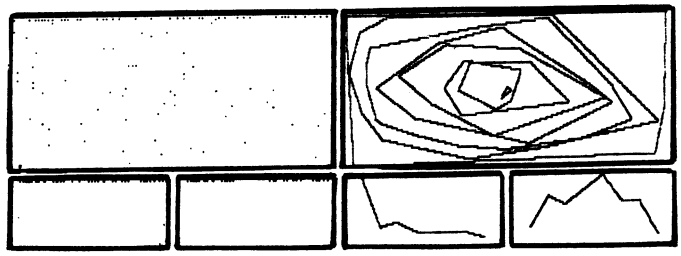
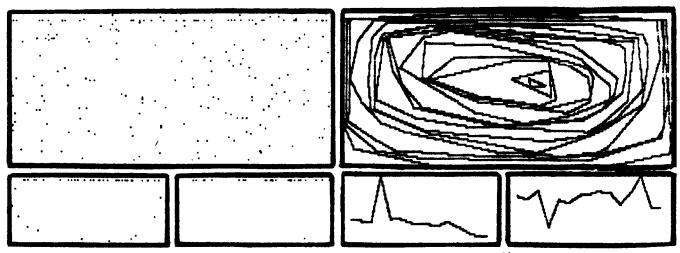
Research supported by the Natural Sciences and Engineering Research Council of Canada under grant A9173.

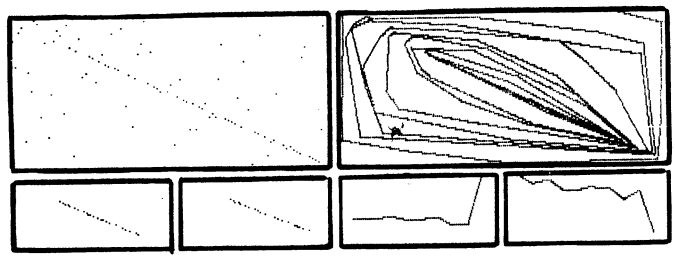
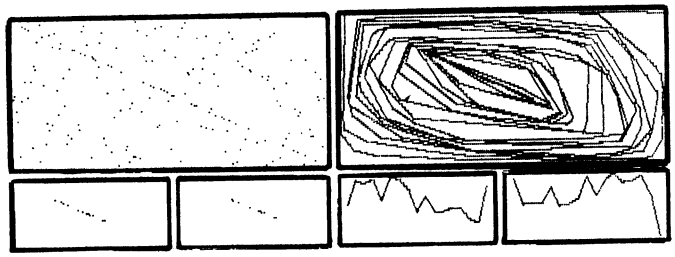
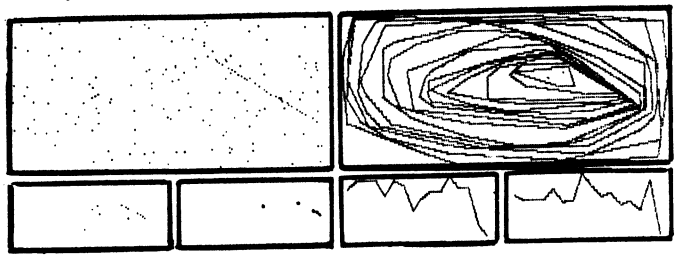
REFERENCES

1. H. Carnal, Die konvexe Hülle von n rotationssymmetrisch verteilten Punkten. *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete* **15**, 169–176 (1970).
2. B. Chazelle, Optimal algorithms for computing depth and layers. In *Proceedings of the 21st Allerton Conference on Communication and Control with Computers* (1983), 427–436.
3. M. Cohen and G. Toussaint, On the detection of structures in noisy images. *Pattern Recognition* **9**, 95–98 (1977).
4. R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. Wiley, New York (1973).
5. R. O. Duda and P. E. Hart, Use of the Hough Transformation to detect lines and curves in pictures. *Comm. ACM* **15** (1), 11–15 (1972).
6. O. Efrow, The convex hull of a random set of points. *Biometrika* **52** (3, 4), 331–343 (1965).
7. J. R. Fram and E. S. Deutsch, On the quantitative evaluation of edge detection schemes and their comparison with human performance. *IEEE Trans. on Computers* **24** (6), 616–628 (1975).
8. F. O'Gorman and M. B. Clowts, Finding picture edges through collinearity of picture points. *IEEE Trans. on Computers* **25** (4), 449–556 (1976).
9. R. C. Gonzales and P. Wirtz, *Digital Image Processing* (2nd edn). Addison-Wesley, New York (1987).
10. P. J. Huber, Robust statistics: a review. *The Annals of Mathematical Statistics* **43** (4), 1041–1067 (1972).
11. M. D. Levine, Feature extraction: a survey. *Proc. of the IEEE* **57** (9), 1391–1403 (1969).

12. D. T. Lee and F. P. Preparata, Computational Geometry – a survey. *IEEE Trans. on Computers* **33** (L12), 1072–1101 (1984).
13. A. Mitchie and J. K. Aggarwal, Detection of edges using range information. *IEEE Trans. PAMI* **5** (2), 174–178.
14. F. P. Preparata and M. I. Shamos, *Computational Geometry, an Introduction*. Springer, New York (1985).
15. A. Rosenfeld and M. Thurston, Edge and curve detection for visual scene analysis, *IEEE Trans. on Computers* **20** (5), 562–569 (1971).
16. M. I. Shamos, Geometry and statistics: problems at the interface. In *Recent Results and New Directions in Algorithms and Complexity*, edited J. F. Traub. Academic Press, New York (1976).
17. S. P. Smith and A. K. Tain, Testing for uniformity in multidimensional data. *IEEE Trans. PAMI* **6** (1), 73–80 (1984).
18. G. T. Toussaint, Pattern recognition and geometrical complexity. In *Proc. 5th Int. Conf. on Pattern Recognition*, pp. 1324–1347 (1980).

APPENDIX A: SAMPLE PLOTS

Total number of points (TP)	Noise (% of TP)	data / convex hull peelings / hull points selected (h_k) / post processed points / $\frac{HP_i}{n}$ / ANA_i
		horizontal line
100	60	
200	80	
		horizontal line close to a border
100	60	
200	80	

Total number of points (TP)	Noise (% of TP)	
		diagonal line
100	60	
200	80	
		arbitary line segment
200	80	
200	80