AN EFFICIENT COMPUTATIONAL GEOMETRY METHOD FOR DETECTING DOTTED LINES IN NOISY IMAGES

F. DEHNE
School of Computer Science, Carleton University, Ottawa, Ontario, Canada K1S 5B6

L. FICOCELLI
Grande Prairie Regional College, Grande Prairie, Alberta, Canada TB8 4C4

ABSTRACT

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 $n_N/n \leq 75\%$.

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 [CT, DH2, GL, MA, RT, ST]. 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 [S,T0].

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 75% noise; i.e., $n_N/n \leq 75\%$.

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 will then present our algorithm and discuss its rational; 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 [PS, LP]; for peeling a set $S$ of $n$ points, Chazelle [Ch] 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 [Ca, E, H, S, Tu]. Tukey suggested (as reviewed in [H]) 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 of 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 then 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 [DH1] for an exact definition of thinness).

Experiments showed that, although lines do not behave as good with respect to peeling as a compact mass, there are some similarities. In particular, as we will point out in the following Section 3.1, in 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, then the other line points which have been peeled off by previous hulls can be recovered by a postprocessing 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 the following sections 3.1, 3.2, and 3.3 we will discuss Phase 1, Phase 2, and Phase 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 nice 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.

![Figure 1:](image)

Figure 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 Points as Noise Points are Removed (d) A Worst Case Scenario

Figure 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 (Figure 1a); therefore, only noise is removed. When the concentric hull becomes 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 Figure 1b.

For lines, the second case will occur immediately.

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

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, e.g., the line is located close to a border of the window.

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

Figure 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,
- 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 $\frac{1}{ANA_i} := \frac{A_i}{A_{i+1}} \cdot \frac{HP_i}{HP_i}$

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

the reciprocal value of the normalized average area of $h_i$.

Therefore, our algorithm detects hulls $h_k$ by selecting from all hulls $h_i$ the hull which maximizes $a_i := \frac{HP_i}{n} \cdot \frac{1}{ANA_i}$.

Our experimental results (see Section 4) show that this measure is very reliable; Appendix A shows some typical plots of $\frac{HP_i}{n}$ 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 $p_0, ..., p_{t-1}$ be the extreme points of $h_k$ and let $p$ and $a(p)$ denote the centroid of $p_1, ..., p_t$ and the angle of the polar coordinates of $p_i$ with respect to center $p$ (Dist-1), respectively; finally, let $\beta(p) := |a(p_{i+1} \mod t) - a(p)|$ denote the difference of the angle of the polar coordinates (with center $p$) of $p_i$ and its successor $p_{i+1} \mod t$.

Our method for eliminating noise points of $h_k$ is very simple; the rational for it is that the angle (with respect to center $p$) between noise points is larger than the angle between line points (see, e.g., Figure 2):

- Compute for each extreme point $p_i$ of $h_k$ the value $\beta(p_i)$.
- Discard all $p_i$ with $\beta(p_i) > \frac{360^0}{HP_i}$.

Although it is very simple, it turns out that this method is very accurate in that it deletes most of the noise points and nearly none 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 [Ch] 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, e.g., algorithms based on the Hough Transform.

On the other hand, the proposed methods proved to be very reliable. Figure 3 summarizes some performance results obtained from extensive testing with randomly generated data sets.

<table>
<thead>
<tr>
<th>noise in %</th>
<th>100</th>
<th>200</th>
<th>300</th>
</tr>
</thead>
<tbody>
<tr>
<td>90</td>
<td>23</td>
<td>44</td>
<td>61</td>
</tr>
<tr>
<td>85</td>
<td>79</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>75</td>
<td>92</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>60</td>
<td>0</td>
<td>8</td>
<td>12</td>
</tr>
<tr>
<td>55</td>
<td>57</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>50</td>
<td>80</td>
<td>100</td>
<td>100</td>
</tr>
</tbody>
</table>

Figure 3: Percentage of Correct Answers For Random Data Sets

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 where tested since they apply, e.g., to the detection of the paths of nuclear particles in Physics; for these cases, all answer 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 $H_{P_i}$ and $A_{N_i}$ 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.

REFERENCES


[To] G.T. Toussaint, "Pattern recognition and geometrical complexity", in Proc. 5th Int. Conf. on Pattern Recognition, 1980, pp. 1324-1347.
### APPENDIX A: SAMPLE PLOTS

<table>
<thead>
<tr>
<th>Total Number of Points (TP)</th>
<th>Noise (% of TP)</th>
<th>Data</th>
<th>Convex Hull Peeling</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Hull Points Selected ($h$)</td>
<td>Post-Processed Points</td>
</tr>
<tr>
<td>100</td>
<td>60</td>
<td><img src="image1" alt="Plot 1" /></td>
<td><img src="image2" alt="Plot 2" /></td>
</tr>
<tr>
<td>200</td>
<td>80</td>
<td><img src="image5" alt="Plot 5" /></td>
<td><img src="image6" alt="Plot 6" /></td>
</tr>
<tr>
<td></td>
<td></td>
<td><img src="image9" alt="Plot 9" /></td>
<td><img src="image10" alt="Plot 10" /></td>
</tr>
</tbody>
</table>

**Horizontal Line**

**Horizontal Line Close to a Border**