

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

F. DEHNE and L. FICOCELLI

SCS-TR-137

May 1988

School of Computer Science, Carleton University
Ottawa, Canada K1S 5B6

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

F. DEHNE ¹ AND L. FICOCELLI ²

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 independant, 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 School of Computer Science, Carleton University, Ottawa, Ontario, Canada K1S 5B6. Research supported by the Natural Sciences and Engineering Research Council of Canada under grant A9173.

² Grande Prairie Regional College, Grande Prairie, Alberta, Canada T8V 4C4.

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,To].

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 $S = \{p_1, \dots, p_n\}$; 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, 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, 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, \dots, 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, \dots, 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 straight forward: 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.

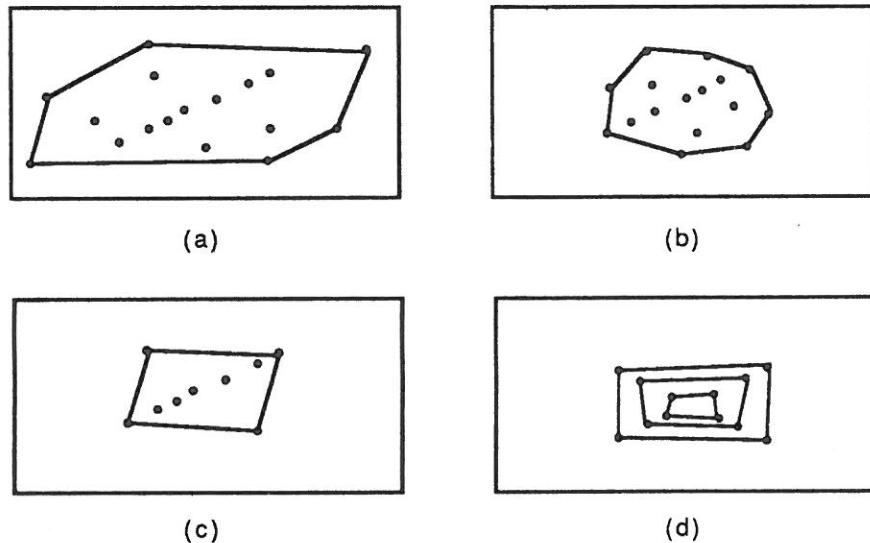


Figure 1: Using Convex Hulls for Noise Removal
(a) Initial Case for Line Segment (b) Subsequent Case for Line

Segments and Initial Case for Lines (c) A Case Where as Many Line
Points as Noise Points are Removed (d) A Worst Case Scenario

Figure 1 shows the most important cases which can occur while peeling h_1, \dots, 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 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 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 SIGNIFFICANT 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.

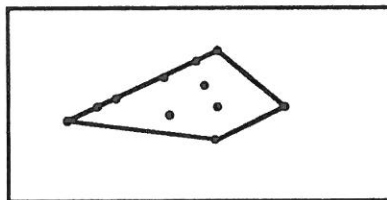


Figure 2: A Typical Hull h_k .

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, 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 $\frac{HP_i}{n}$ 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, \dots, p_{t-1} be the extreme points of h_k and let p and $a(p_i)$ denote the centroid of p_1, \dots, p_t and the angle of the polar coordinates of p_i with respect to center p ($0 \leq i \leq t-1$), 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 center 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 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 $\beta(p_i) > \frac{360^\circ}{HP_i}$.

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.

noise: $\frac{nN}{n}$ (in %)	n	100	200	300	
90		23	44	61	horizontal or vertical lines
85		55	79	96	
80		79	100	100	
75		92	100	100	
90		0	8	12	diagonal lines
85		15	57	71	
80		57	84	100	
75		80	100	100	
90		31	25	32	arbitrary line segments
85		45	48	41	
80		52	80	81	
75		85	95	95	

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 were tested since they apply, e.g., to the detection of the paths of nuclear particles in 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.

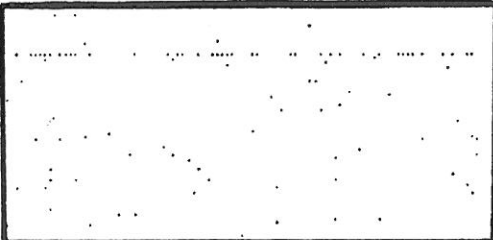
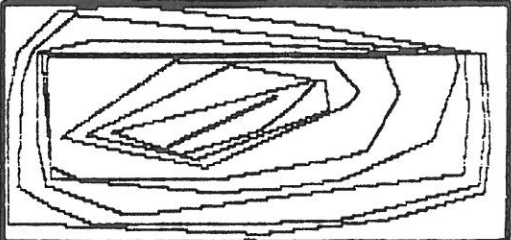


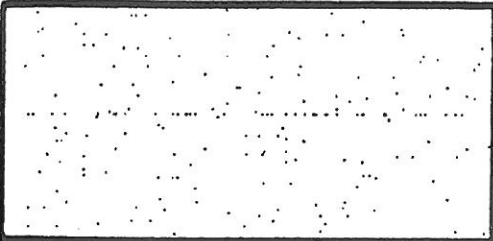
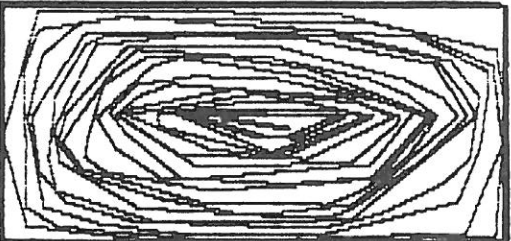
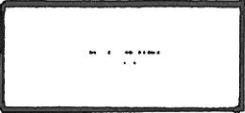
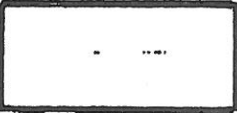

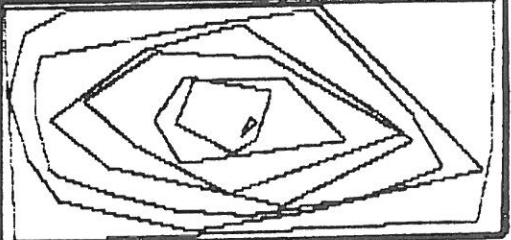

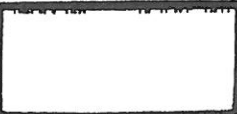
Appendix A shows some sample plots. For each experiment, the original point set, all hulls determined in Phase 1 as well as the values $\frac{HP_i}{n}$ and ANA_i for each hull h_i , the hull h_k selected in Phase 2, and the remaining points of h_k after Phase 3 are shown.

REFERENCES

- [Ca] H. Carnal, "Die konvexe Hülle von n rotationssymmetrisch verteilten Punkten", Zeitschr. Wahrscheinlichkeitstheorie und verw. Geb. 15, 1970, pp. 169-176.
- [Ch] B. Chazelle, "Optimal algorithms for computing depth and layers", in Proc. 21st Allerton Conf. Comm. Control Comput., 1983, pp. 427-436.
- [CT] M. Cohen, G. Toussaint, "On the detection of structures in noisy images", Pattern Recognition 9, 1977, pp. 95-98.
- [DH1] R.O. Duda, P.E. Hart, "Pattern classification and scene analysis", Wiley, New York, 1973.
- [DH2] R.O. Duda, P.E. Hart, "Use of the Hough Transformation to detect lines and curves in pictures", Comm. ACM 15:1, 1972, pp. 11-15.
- [E] O. Efron, "The convex hull of a random set of points", Biometrika 52:3,4, 1965, pp. 331-343.

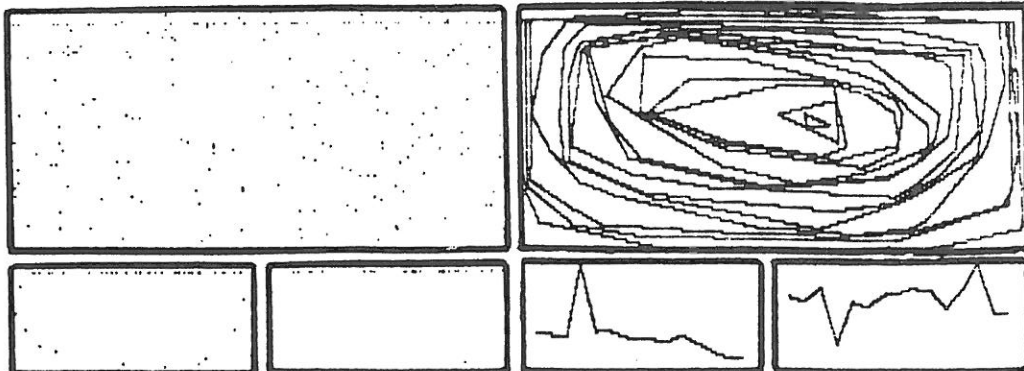
- [FD] J.R. Fram, E.S. Deutsch, "On the quantitative evaluation of edge detection schemes and their comparison with human performance", IEEE Trans. on Computers 24:6, 1975, pp.616-628.
- [GL] F. O'Gorman, M.B. Clowts, "Finding picture edges through collinearity of picture points", IEEE Trans. on Computers 25:4, 1976, pp. 449-556.
- [GW] R.C. Gonzales, P. Wirtz, "Digital image processing", 2nd edition, Addison-Wesley, 1987.
- [H] P.J. Huber, "Robust statistics: a review", The Annals of Mathematical Statistics 43:4, 1972, pp.1041-1067.
- [L] M.D. Levine, "Feature extraction: a survey", Proc. of the IEEE, Vol. 57:9, 1969, pp. 1391-1403.
- [LP] D.T. Lee, F.P. Preparata, "Computational Geometry - a survey", IEEE Trans. on Computers 33:L12, 1984, pp.1072-1101.
- [MA] A. Mitchie, J.K. Aggarwal, "Detection of edges using range information", IEEE Trans. PAMI 5:2, pp.174-178.
- [PS] F.P. Preparata, M.I. Shamos, "Computational Geometry, an introduction", Springer-Verlag, 1985.
- [RT] A. Rosenfeld, M. Thurston, "Edge and curve detection for visual scene analysis", IEEE Trans. on Computers 20:5, 1971, pp. 562-569.
- [S] M.I. Shamos, "Geometry and statistics: problems at the interface", in: Recent Results and New Directions in Algorithms and Complexity, J.F. Traub (Ed.), Academic Press, 1976.
- [ST] S.P. Smith, A.K. Tain, "Testing for uniformity in multidimensional data", IEEE Trans. PAMI 6:1, 1984, pp. 73-80.
- [To] G.T. Tousaaint, "Pattern recognition and geometrical complexity", in Proc. 5th Int. Conf. on Pattern Recognition, 1980, pp. 1324-1347.

APPENDIX A: SAMPLE PLOTS

Total Number of Points (TP)	Noise (% of TP)	<div> <div>data</div> <div>convex hull peelings</div> </div>			
		hull points selected (h_k)	post processed points	HP ₁ r_2	ANA i
100	60	<u>horizontal line</u>			
					
200	80				
		<u>horizontal line close to a border</u>			
100	60				

200

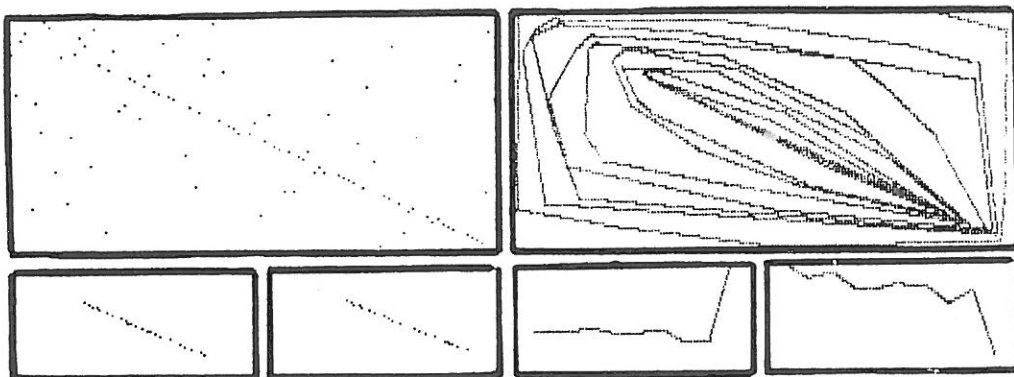
80



diagonal line

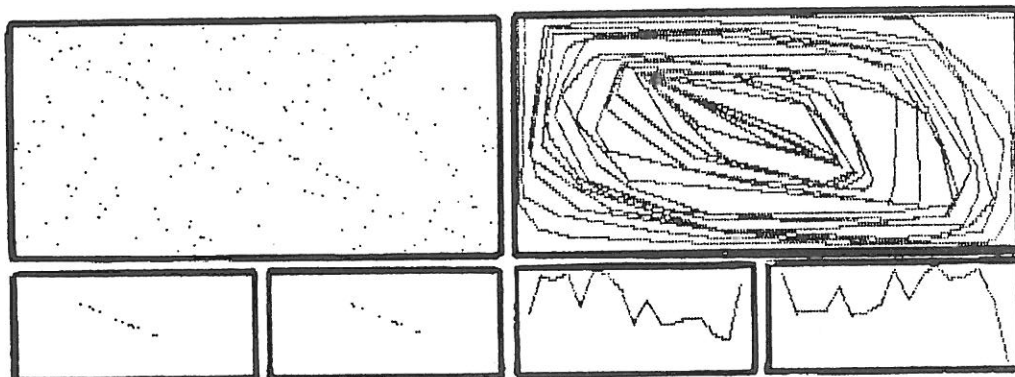
100

60



200

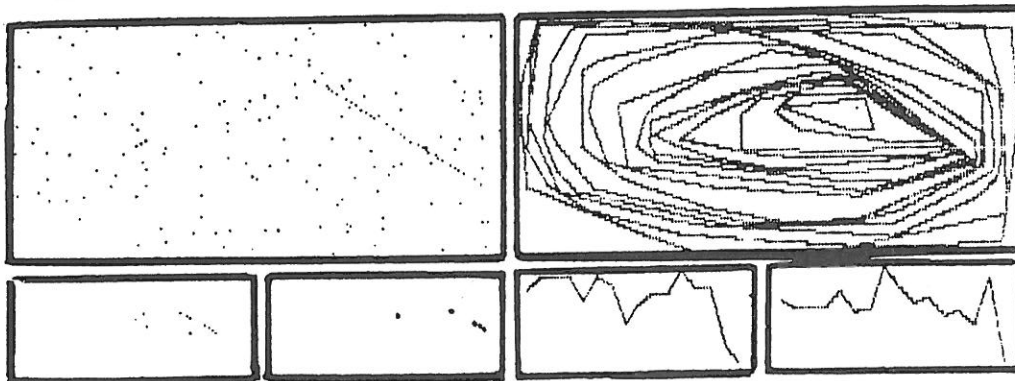
80

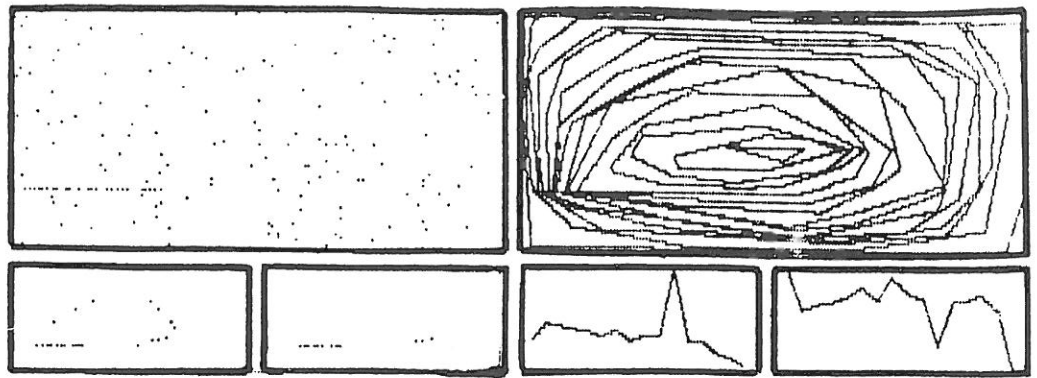


arbitrary line segment

200

80





Carleton University, School of Computer Science
Bibliography of Technical Reports
Publications List (1985 -->)

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

- SCS-TR-66 **On the Futility of Arbitrarily Increasing Memory Capabilities of Stochastic Learning Automata**
_____ B.J. Oommen, October 1984. Revised May 1985.
- SCS-TR-67 **Heaps in Heaps**
_____ T. Strothotte, J.-R. Sack, November 1984. Revised April 1985.
- SCS-TR-68 **Partial Orders and Comparison Problems**
out-of-print M.D. Atkinson, November 1984. See Congressus Numerantium 47 ('86), 77-88
- SCS-TR-69 **On the Expected Communication Complexity of Distributed Selection**
_____ N. Santoro, J.B. Sidney, S.J. Sidney, February 1985.
- SCS-TR-70 **Features of Fifth Generation Languages: A Panoramic View**
_____ Wilf R. LaLonde, John R. Pugh, March 1985.
- SCS-TR-71 **Actra: The Design of an Industrial Fifth Generation Smalltalk System**
_____ David A. Thomas, Wilf R. LaLonde, April 1985.
- SCS-TR-72 **Minmaxheaps, Orderstatisticstrees and their Application to the Coursemarks Problem**
_____ M.D. Atkinson, J.-R. Sack, N. Santoro, T. Strothotte, March 1985.
- SCS-TR-73 **Designing Communities of Data Types**
_____ Wilf R. LaLonde, May 1985.
Replaced by SCS-TR-108
- SCS-TR-74 **Absorbing and Ergodic Discretized Two Action Learning Automata**
out-of-print B. John Oommen, May 1985. See IEEE Trans. on Systems, Man and Cybernetics, March/April 1986, pp. 282-293.
- SCS-TR-75 **Optimal Parallel Merging Without Memory Conflicts**
_____ Selim Akl and Nicola Santoro, May 1985
- SCS-TR-76 **List Organizing Strategies Using Stochastic Move-to-Front and Stochastic Move-to-Rear Operations**
_____ B. John Oommen, May 1985.
- SCS-TR-77 **Linearizing the Directory Growth in Order Preserving Extendible Hashing**
_____ E.J. Otoo, July 1985.
- SCS-TR-78 **Improving Semijoin Evaluation In Distributed Query Processing**
_____ E.J. Otoo, N. Santoro, D. Rotem, July 1985.

Carleton University, School of Computer Science
Bibliography of Technical Reports

- SCS-TR-79 **On the Problem of Translating an Elliptic Object Through a Workspace of Elliptic Obstacles**
_____ B.J. Oommen, I. Reichstein, July 1985.
- SCS-TR-80 **Smalltalk - Discovering the System**
_____ W. LaLonde, J. Pugh, D. Thomas, October 1985.
- SCS-TR-81 **A Learning Automation Solution to the Stochastic Minimum Spanning Circle Problem**
_____ B.J. Oommen, October 1985.
- SCS-TR-82 **Separability of Sets of Polygons**
_____ Frank Dehne, Jörg-R. Sack, October 1985.
- SCS-TR-83 **Extensions of Partial Orders of Bounded Width**
out-of-print M.D. Atkinson and H.W. Chang, November 1985. See *Congressus Numerantium*, Vol. 52 (May 1986), pp. 21-35.
- SCS-TR-84 **Deterministic Learning Automata Solutions to the Object Partitioning Problem**
_____ B. John Oommen, D.C.Y. Ma, November 1985
- SCS-TR-85 **Selecting Subsets of the Correct Density**
out-of-print M.D. Atkinson, December 1985. To appear in *Congressus Numerantium*, Proceedings of the 1986 South-Eastern conference on Graph theory, combinatorics and Computing.
- SCS-TR-86 **Robot Navigation in Unknown Terrains Using Learned Visibility Graphs. Part I: The Disjoint Convex Obstacles Case**
_____ B. J. Oommen, S.S. Iyengar, S.V.N. Rao, R.L. Kashyap, February 1986
- SCS-TR-87 **Breaking Symmetry in Synchronous Networks**
_____ Greg N. Frederickson, Nicola Santoro, April 1986
- SCS-TR-88 **Data Structures and Data Types: An Object-Oriented Approach**
_____ John R. Pugh, Wilf R. LaLonde and David A. Thomas, April 1986
- SCS-TR-89 **Ergodic Learning Automata Capable of Incorporating Apriori Information**
_____ B. J. Oommen, May 1986
- SCS-TR-90 **Iterative Decomposition of Digital Systems and Its Applications**
_____ Vaclav Dvorak, May 1986.
- SCS-TR-91 **Actors in a Smalltalk Multiprocessor: A Case for Limited Parallelism**
_____ Wilf R. LaLonde, Dave A. Thomas and John R. Pugh, May 1986
- SCS-TR-92 **ACTRA - A Multitasking/Multiprocessing Smalltalk**
_____ David A. Thomas, Wilf R. LaLonde, and John R. Pugh, May 1986
- SCS-TR-93 **Why Exemplars are Better Than Classes**
_____ Wilf R. LaLonde, May 1986
- SCS-TR-94 **An Exemplar Based Smalltalk**
_____ Wilf R. LaLonde, Dave A. Thomas and John R. Pugh, May 1986
- SCS-TR-95 **Recognition of Noisy Subsequences Using Constrained Edit Distances**
_____ B. John Oommen, June 1986

Carleton University, School of Computer Science
Bibliography of Technical Reports

- SCS-TR-96 **Guessing Games and Distributed Computations In Synchronous Networks**
J. van Leeuwen, N. Santoro, J. Urrutia and S. Zaks, June 1986.
- SCS-TR-97 **Bit vs. Time Tradeoffs for Synchronous Elections**
M. Overmars and N. Santoro, February 1988.
- SCS-TR-98 **Reduction Techniques for Distributed Selection**
N. Santoro and E. Suen, June 1986.
- SCS-TR-99 **A Note on Lower Bounds for Min-Max Heaps**
A. Hasham and J.-R. Sack, June 1986.
- SCS-TR-100 **Sums of Lexicographically Ordered Sets**
M.D. Atkinson, A. Negro, and N. Santoro, May 1987.
- SCS-TR-102 **Computing on a Systolic Screen: Hulls, Contours, and Applications**
F. Dehne, J.-R. Sack and N. Santoro, October 1986.
- SCS-TR-103 **Stochastic Automata Solutions to the Object Partitioning Problem**
B.J. Oommen and D.C.Y. Ma, November 1986.
- SCS-TR-104 **Parallel Computational Geometry and Clustering Methods**
F. Dehne, December 1986.
- SCS-TR-105 **On Adding *Constraint Accumulation* to Prolog**
Wilf R. LaLonde, January 1987.
- SCS-TR-107 **On the Problem of Multiple Mobile Robots Cluttering a Workspace**
B. J. Oommen and I. Reichstein, January 1987.
- SCS-TR-108 **Designing Families of Data Types Using Exemplars**
Wilf R. LaLonde, February 1987.
- SCS-TR-109 **From Rings to Complete Graphs - $\Theta(n \log n)$ to $\Theta(n)$ Distributed Leader Election**
Hagit Attiya, Nicola Santoro and Shmuel Zaks, March 1987.
- SCS-TR-110 **A Transputer Based Adaptable Pipeline**
Anirban Basu, March 1987.
- SCS-TR-111 **Impact of Prediction Accuracy on the Performance of a Pipeline Computer**
Anirban Basu, March 1987.
- SCS-TR-112 **ϵ -Optimal Discretized Linear Reward-Penalty Learning Automata**
B.J. Oommen and J.P.R. Christensen, May 1987.
- SCS-TR-113 **Angle Orders, Regular n-gon Orders and the Crossing Number of a Partial Order**
N. Santoro and J. Urrutia, June 1987.
- SCS-TR-115 **Time is Not a Healer: Impossibility of Distributed Agreement in Synchronous Systems with Random Omissions**
N. Santoro, June 1987.

Carleton University, School of Computer Science
Bibliography of Technical Reports

- SCS-TR-116 **A Practical Algorithm for Boolean Matrix Multiplication**
M.D. Atkinson and N. Santoro, June 1987.
- SCS-TR-117 **Recognizing Polygons, or How to Spy**
James A. Dean, Andrzej Lingas and Jörg-R. Sack, August 1987.
- SCS-TR-118 **Stochastic Rendezvous Network Performance - Fast, First-Order Approximations**
J.E. Neilson, C.M. Woodside, J.W. Miernik, D.C. Petriu, August 1987.
- SCS-TR-120 **Searching on Alphanumeric Keys Using Local Balanced Tree Hashing**
E.J. Otoo, August 1987.
- SCS-TR-121 **An $O(\sqrt{n})$ Algorithm for the ECDF Searching Problem for Arbitrary Dimensions on a Mesh-of-Processors**
Frank Dehne and Ivan Stojmenovic, October 1987.
- SCS-TR-122 **An Optimal Algorithm for Computing the Voronoi Diagram on a Cone**
Frank Dehne and Rolf Klein, November 1987.
- SCS-TR-123 **Solving Visibility and Separability Problems on a Mesh-of-Processors**
Frank Dehne, November 1987.
- SCS-TR-124 **Deterministic Optimal and Expedient Move-to-Rear List Organizing Strategies**
B.J. Oommen, E.R. Hansen and J.I. Munro, October 1987.
- SCS-TR-125 **Trajectory Planning of Robot Manipulators in Noisy Workspaces Using Stochastic Automata**
B.J. Oommen, S. Sitharam Iyengar and Nicte Andrade, October 1987.
- SCS-TR-126 **Adaptive Structuring of Binary Search Trees Using Conditional Rotations**
R.P. Cheetham, B.J. Oommen and D.T.H. Ng, October 1987.
- SCS-TR-127 **On the Packet Complexity of Distributed Selection**
A. Negro, N.Santoro and J. Urrutia, November 1987.
- SCS-TR-128 **Efficient Support for Object Mutation and Transparent Forwarding**
D.A. Thomas, W.R. LaLonde and J. Duimovich, November 1987.
- SCS-TR-129 **Eva: An Event Driven Framework for Building User Interfaces in Smalltalk**
Jeff McAffer and Dave Thomas, November 1987.
- SCS-TR-130 **Application Frameworks: Experience with MacApp**
John R. Pugh and Cefee Leung, December 1987.
- SCS-TR-131 **An Efficient Window Based System Based on Constraints**
Danny Epstein and Wilf R. LaLonde, March 1988.
- SCS-TR-132 **Building a Backtracking Facility in Smalltalk Without Kernel Support**
Wilf R. LaLonde and Mark Van Gulik, March 1988.
- SCS-TR-134 **Separating a Polyhedron by One Translation from a Set of Obstacles**
Otto Nurmi and Jörg-R. Sack, December 1987.
- SCS-TR-135 **An Optimal VLSI Dictionary Machine for Hypercube Architectures**
Frank Dehne and Nicola Santoro, April 1988.

Carleton University, School of Computer Science
Bibliography of Technical Reports

SCS-TR-136

Optimal Visibility Algorithms for Binary Images on the Hypercube
Frank Dehne, Quoc T. Pham and Ivan Stojmenovic, April 1988.

SCS-TR-137

An Efficient Computational Geometry Method for Detecting Dotted Lines in Noisy Images
F. Dehne and L. Ficocelli, May 1988.