# Computing the Largest Empty Convex Subset of a Set of Points

David Avis
David Rappaport
McGill University
805 Sherbrooke Street West
Montreal, Quebec, H3A 2K6

## Abstract

A largest empty convex subset of a finite set of points, S, is a maximum cardinality subset of S, that (1) are the vertices of a convex polygon, and (2) contain no other points of S interior to their convex hull. An $O(n^3)$ time and $O(n^2)$ space algorithm is introduced to find such subsets, where n represents the cardinality of S. Empirical results are obtained and presented. In particular, a configuration of 20 points is obtained with no empty convex hexagon, giving a partial answer to a question of Paul Erdös.

## 1. Introduction

Esther Klein showed that from any five points in the plane, no three collinear, it is always possible to select four of them that determine a convex quadrilateral. Miss Klein suggested the following problem: Determine, for a given integer $n \geq 3$, the smallest number $f(n)$, such that from every set of at least $f(n)$ points, it is always possible to select $n$ points forming a convex polygon. In a paper published in 1935 that was to shape an entire field of mathematics, Erdös and Szekeres [3] proved that

$$f(n) \leq \binom{2n-4}{n-2} + 1.$$

In a later paper [4] they showed that

$$f(n) \geq 2^{n-2} + 1.$$

An interesting account of the history of the former paper, and a reprint of it, is contained in the collection of papers by Erdös edited by Spencer [5]. The exact value of $f(n)$ is still unknown, except for the values $f(3)=3$, $f(4)=5$, $f(5)=9$, see Moser [9].



The corresponding problem in computational geometry is to design an efficient algorithm to find a largest convex subset of a set of $n$ points in the plane. An $O(n^3)$ algorithm for this problem was found by Chvátal and Klincsek [1].

In this paper, we will be concerned with the following related problem, also posed by Erdös. Given a set S, of n points in the plane, no three collinear, we are interested in computing a maximum cardinality subset of S, $\Phi(S)$, with the following properties:

(1) The points in $\Phi(S)$ lie on the vertices of a convex polygon.

(2) There are no points of S that lie interior to this polygon.

$\Phi(S)$ is the largest empty convex subset of S. We introduce a method for finding $\Phi(S)$ in the next section. Let $|\Phi(S)|$ denote the cardinality of $\Phi(S)$. Define g(k) as the smallest integer such that any set of g(k) points, no three collinear, contains an empty convex k-gon. It has been shown that $g(3)=3$, $g(4)=5$, and $g(5)=10$ [7]. For values of $k \geq 7$, Joe Horton has proved [8] that g does not exist. The value of g(6) remains an open problem. In this paper we exhibit a 20 point set with no empty convex 6-gon showing that $g(6) \geq 21$.

To compute $\Phi(S)$, Chvatal and Klincsek's [1] result is used. It will be seen, that by adding an appropriate pre-processing step, a simple modification of the algorithm to compute largest convex subsets, can be used to solve the problem at hand.

## 2. Preliminaries

Some fundamental objects will be defined. A *simple polygon* P is a simply connected subset of the plane whose boundary is a closed chain of line segments with adjacent edges intersecting at their endpoints and no two non-adjacent intersecting edges. A polygon will be represented as a sequence of vertices, such that the *interior* of the polygon lies to the right as the vertices are traversed. We will denote the edge e *inside* a polygon P if e does not intersect the exterior of P. Let y and z be points interior or on the boundary of P. Point y is said to be *visible* from z if the line segment (y,z) lies entirely inside P. A *convex*

*polygon* is a simple polygon such that every pair of points interior to the polygon are mutually visible. A convex polygon of k vertices will be known as a convex *k-gon*. A polygon P is a *fan* with *root* x, a vertex of P, if all vertices of P are visible from x. For convenience we label the points in S, $p_1, p_2, \ldots, p_n$ in order of increasing x-coordinates. By rotating the points about the coordinate axes if necessary, we may assume that no two points in S have the same x-coordinate.

Set $S_1 = S$ and for $i = 2, \ldots, n$ set $S_i = \{p_i, p_{i+1}, \ldots, p_n\}$. Define $\phi(S_i)$ as the largest empty convex subset of $S_i$ that includes the point $p_i$. Then we observe that $|\Phi(S)| = \max_i |\phi(S_i)|$. This follows by noting that (1) for every $i = 1, \ldots, n$, $\phi(S_i)$ can contain no point from the set $S - S_i$ in the interior of its convex hull due to the fact that the points in S are sorted and hence (2) if $p_i$ is the point in $\Phi(S)$ with smallest label then $\phi(S_i)$ is a largest empty convex subset of S.

We denote by $i_x$ and $i_y$ the x and y coordinates of a point $p_i$ in S. Let $s(p_i, p_j) = (j_y - i_y) / (j_x - i_x)$. A sequence of points $c_1, \ldots, c_m$ of S defines a *convex chain* of length m-1 if

$$s(c_1, c_2) \leq s(c_2, c_3) \leq \ldots \leq s(c_{m-1}, c_m)$$

or a *concave chain* of length m-1 if

$$s(c_1, c_2) \geq s(c_2, c_3) \geq \ldots \geq s(c_{m-1}, c_m)$$

The points $c_1$ and $c_m$ are endpoints of the chain. Concatenation of a convex chain and a concave chain with identical endpoints yields a convex polygon.

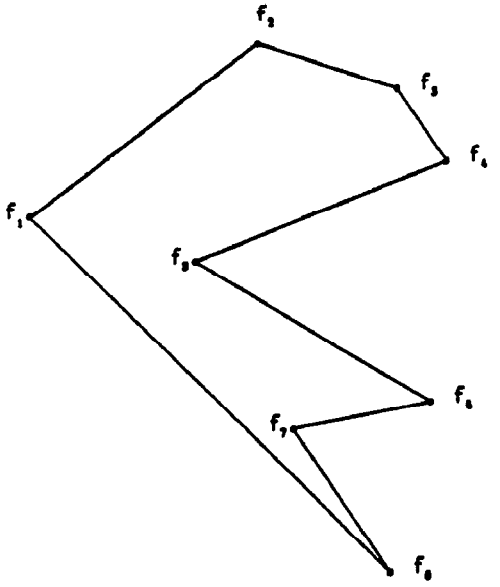
As a first step in computing $\phi(S_i)$ a set of legal edges $L(S_i)$ are found. Two vertices $p_j$, $p_k$, $i \leq j < k$, form a *legal edge* in $L(S_i)$ if and only if they form an edge of an empty convex subset of $S_i$, containing the point $p_i$.

The sets of legal edges are computed from a set of fans constructed on S. Define the *polar angle* of a point as the angle relative to a fixed *polar origin* and a fixed *polar line*. Let $p_i$ be the polar origin and let the line $y = i_y$ be the polar line. We define the fan $F(S_i) = f_i, f_{i+1}, \ldots, f_n$ where $f_i = p_i$ is the root and $f_{i+1}, f_{i+2}, \ldots, f_n$ is a permutation of the points $p_{i+1}, \ldots, p_n$ of S in descending order by polar angle. See figure (1).

The set $L(S_i)$ can now be expressed in terms of $F(S_i)$.

*Lemma:* An edge is in the set $L(S_i)$ if and only if it is inside the fan $F(S_i)$.

*Proof:* First we will show that every convex subset including $p_i = f_i$ with an edge of its convex hull




figure(1)

intersecting the exterior of $F(S_i)$ is not an empty convex subset. We need only consider edges of the form $(f_j, f_k)$ where $j,k \neq i$, since every edge $(f_i, f_k)$ is inside $F(S_i)$ by definition. Suppose $(f_j, f_k)$ is not inside $F(S_i)$. Observe that every convex polygon including $f_i$ and an edge $(f_j, f_k)$ which intersects the exterior of $F(S_i)$, contains a triangle T with vertices $f_i, f_j, f_k$. Since $(f_j, f_k)$ intersects the exterior of $F(S_i)$ while $(f_i, f_j)$ and $(f_k, f_i)$ do not, then T contains at least one point of S in its interior. Therefore $(f_j, f_k)$ is not a legal edge.

Conversely, if $(f_j, f_k)$ is inside $F(S_i)$, then the triangle $f_i, f_j, f_k$ is empty, hence $(f_j, f_k)$ is a legal edge.

□

Shamos [11] describes the visibility graph of a polygon. This graph has nodes corresponding to the vertices of a polygon and has two nodes connected by an edge if and only if the associated vertices are mutually visible. The task of computing $L(S_i)$ is reduced to constructing the fan $F(S_i)$ and then computing the visibility graph of $F(S_i)$.

An $O(n^2)$ algorithm exists for computing the visibility graph of a simple polygon. An O(n) algorithm due to El-Gindy and Avis [2] for computing the visible vertices in a polygon from a point is repeated n times. At each iteration a vertex is used as the

viewpoint and all other vertices of the polygon visible from it are found. A new method for computing the visible vertices in fans is proposed. Although there is no improvement in the asymptotic complexity, the algorithm presented in section three exploits the structure found in fans resulting in a simpler procedure.

Once the legal edges $L(S_i)$ are found, $\phi(S_i)$ can be computed by modifying an algorithm due to Chvátal and Klincsek [1]. This algorithm computes largest convex subsets. Denote $\hat{\phi}(S_i)$ as the largest convex subset of $S_i$ that includes the point $p_i$. $\hat{\phi}(S_i)$ is not necessarily an empty convex subset.

The method used to compute $\hat{\phi}(S_i)$ is to find convex and concave chains of largest cardinality. We will say the chain of largest cardinality is the longest chain. Computing longest convex and concave chains is symmetric so only the convex case need be discussed.

The longest chain from $p_i$ to $p_k$ for $i < k \leq n$ will be found in stages. Denote C[j,k] as the length of the longest convex chain $(p_i,c_2),(c_2,c_3),\ldots,(p_j,p_k)$. The value of C[j,k] = m - 1 where m is the number of points in the chain, or C[j,k] = 0 if there is no such convex chain.

Initially set j = i and trivially set C[i,k] = 1 for all $i < k \leq n$. We then set j = i+1,i+2,..., n, and determine C[j,k] for all $j < k \leq n$. We will use two arrays RIGHT and LEFT to assist in the computation of C[j,k]. Copy the indices i,i+1,..., j-1 to the array LEFT and copy the indices j+1,j+2,..., n to the array RIGHT. Now sort the values in LEFT so that $s(p_{LEFT[1]},p_j) < s(p_{LEFT[2]},p_j) < \ldots < s(p_{LEFT[j-i]},p_j)$ and sort the values in RIGHT so that $s(p_j,p_{RIGHT[1]}) < s(p_j,p_{RIGHT[2]}) < \ldots < s(p_j,p_{RIGHT[n-j]})$ Fixing j, an algorithm to compute C[j,k] for all $j < k \leq n$ is described below in Pascal-like code.

Input: Values of C[h,k] for $i \leq h < j$ and the arrays LEFT and RIGHT.

Output: The row j of C[j,k].

```
1. L <= 1; LLIM <= j-i; RLIM <= n-j;

2. while C[LEFT[L],j] = 0
     do L <= L+1;
      {There is always an L < LLIM
       such that C[LEFT[L],j] > 0}

3. M <= C[LEFT[L],j]; R <= 1;

4. while
   (s(p_LEFT[L],p_j) > s(p_j,p_RIGHT[R]))
   and (R ≤ RLIM)
   do begin
     C[j,RIGHT[R]] <= 0; R <= R+1
   end;

5. while R ≤ RLIM
     do begin
       while
       (s(p_j,p_RIGHT[R]) > s(p_LEFT[L],p_j))
       and (L ≤ LLIM)
       do begin
         M <= max(M,C[LEFT[L],j]);
         L <= L+1
       end;
       C[j,RIGHT[R]] <= M+1;
       R <= R+1;
     end;
```

*Algorithm 2.1*

It is easy to see that algorithm 2.1 requires O(n) time. To compute C[j,k] for all $i < j \leq n$ the algorithm must be repeated O(n) times, resulting in $O(n^2)$ time and $O(n^2)$ space to store the table.

Similarly we may compute values D[j,k] for $i \leq j < k \leq n$ representing the length of the longest concave chains $(p_i,c_2),(c_2,c_3),\ldots,(p_j,p_k)$. Store the maximum value of column k of C[j,k] in MVEX[k] and store the maximum value of column k of D[j,k] in MCAVE[k] for all $i < k \leq n$. The value of $|\hat{\phi}(S_i)| = \max_k$ (MVEX[k] + MCAVE[k]). Let T be the value of k that maximizes the above sum.

To find $\hat{\phi}(S_i)$ we must reconstruct a convex chain $(p_i,c_2),(c_2,c_3),\ldots,(c_{MVEX[T]},p_T)$ of length MVEX[T]. Similarly we must reconstruct the corresponding concave chain. The method of reconstructing these chains are symmetric so only the convex case will be described. We will use the values C[j,k] for $i \leq j < k \leq n$ in the computation of the chain. Algorithm 2.2 succinctly describes the reconstruction procedure in Pascal-like code.

Input: Values of C[j,k] for all $i < j \leq k \leq n$.

Output: The indices of the points in a convex chain of length MVEX[T] stored in the array CHAIN.

```
1. CHAIN[1] <= i; M <= MVEX[T];
   CHAIN[M+1] <= T;

2. find a point p_v such that
   C[v,CHAIN[M]] = M;
```

```
        then set
          CHAIN[M] <= v; M <= M-1;

3. while M > 1
   do begin
    find a point p_v such that
    C[v,CHAIN[M+1]] = M and
    s(p_v ,p_CHAIN[M+1]) < s(p_v ,p_CHAIN[M+2])
      then set
        CHAIN[M] <= v; M <= M-1
   end;
```

*Algorithm 2.2*

Algorithm 2.2 requires $O(n^2)$ time. For further details regarding the computation of $\dot{\phi}(S_i)$ see [1].

We may now return to the problem of computing $\phi(S_i)$. Observe that if we have a convex chain comprised entirely of legal edges, and a concave chain comprised entirely of legal edges, with identical endpoints, then the interior of the resulting convex polygon is empty. When computing values C[j,k] for $i \leq j < k \leq n$, to ensure that only legal edges are considered, we copy into array LEFT the subset U, of indices i,i+1,..., j-1, such that if u is in U then $(p_u, p_j)$ is a legal edge. We now sort the values in LEFT as before. Similarly we copy into the array RIGHT the subset W, of indices j+1,j+2..., n such that if w is in W then $(p_j, p_w)$ is a legal edge. Sort the values in RIGHT as before. The procedures 2.1 and 2.2 may be applied as before with the exception that LLIM and RLIM are assigned the values |U| and |W| respectively.

Recall $\Phi(S) = \max_i |\phi(S_i)|$, therefore $O(n^3)$ time and $O(n^2)$ space suffice to compute $\Phi(S)$. These algorithms have been implemented. In section four some empirical results relating $|\Phi(S)|$ to |S| are presented and discussed.

## 3. Visibility in Fans

Given a fan $F = f_1, f_2, \ldots, f_n$ with root $f_1$ an algorithm is shown to compute the vertices of F visible from a viewing position that is itself a vertex in F. Let the viewing position be denoted $f_p$. Define the *left fan* as the vertices of F left of the line segment $(f_1, f_p)$. The *right fan* is similarly defined. To find the vertices of F visible from $f_p$, first find the vertices visible in the left fan followed by the vertices visible in the right fan.

A primitive function performed by the following algorithm is the *turn* function:

$$\text{turn}(f_i, f_j, f_k) = (k_x - i_x)(j_y - i_y) - (k_y - i_y)(j_x - i_x)$$

If turn is positive (negative,zero) the turn is denoted as left(right,no). Another function accept($f_i$) accepts $f_i$ as visible and puts $f_i$ on the visible vertex list. Define a(b,c,d) as the clockwise angle between the rays (c,b) and (c,d) at the vertex c.

An algorithm VIS-LFAN that computes the visible vertices in a left fan is described below in Pascal-like code.

Input: The vertices of a fan $F = f_1, f_2, \ldots, f_n$ and a viewing position $f_p$ a vertex of F.

Output: A subset of F, the vertices in the left fan visible from $f_p$.

```
0. Extend the line segment (f_p , f_1)
   and find its intersection with
   the boundary of F.

   if the ray (f_p ,f_1) intersects F
     at an edge (f_i ,f_i+1)
       then term <= i
       else term <= 0;

1. u <= p-1;
   if u > term then accept(f_u );
   v <= p-2;


2. while( v > term)
    do begin
     if turn(f_p ,f_u ,f_v ) is left
       then begin
         accept(f_v );u <= v
       end;
       v <= v-1
    end;
```

*Algorithm VIS-LFAN*

*Lemma:* A vertex accepted by VIS-LFAN is visible from $f_p$.

*Proof:* We prove the following statement inductively: at every step of VIS-LFAN the edge $(f_p, f_u)$ is inside F. Initially the hypothesis is true since $(f_p, f_{p-1})$ is a fan edge.

Every time a triple $f_p, f_u, f_v$ is a left turn, $f_v$ is accepted. It will be shown that this implies that $f_1, f_v, f_u, f_p$ lie on the vertices of a convex quadrilateral inside F.

Consider the sequence of vertices $f_u, f_{u-1}, \ldots, f_{v+1}$. Every point in this sequence lies to the right of $(f_u, f_v)$. Recall that the vertices of F are in polar

order. Since v > term and the angle $a(f_{term+1}, f_1, f_v) \leq \pi$ (we will consider $a(f_1, f_1, f_v)$ to be equal to $\pi$), then $a(f_v, f_1, f_v) < \pi$. This implies $f_1$, $f_v$, $f_v$ lie on the vertices of a triangle, $T_1$, inside F. It can also be shown that $f_1$, $f_v$, $f_p$ are vertices of a triangle, $T_2$, inside F. $T_1$ and $T_2$ are adjacent. It is clear that $a(f_p, f_v, f_v)$ is convex. The angle $a(f_v, f_1, f_p)$ can be shown to be convex by using the polar order in fans. Therefore $f_1$, $f_v$, $f_v$, $f_p$ lie on the vertices of a convex quadrilateral inside F, implying $f_v$ is visible from $f_p$.

□

*Lemma:* If a vertex is not accepted by algorithm VIS-LFAN then the vertex is not visible from $f_p$.
*Proof:* If $\text{turn}(f_p, f_v, f_v)$ is left then v is accepted. Observe that if $\text{turn}(f_p, f_v, f_v)$ is right then $f_1$, $f_v$, $f_v$, $f_p$ is a quadrilateral with concave vertex $f_v$. Therefore triangle $f_1$, $f_v$, $f_p$ contains the point, $f_v$ implying $f_v$ is not visible from $f_p$. Exiting when reaching the point $f_{term}$ is justified by the polar order of the vertices in fans.

□

Using the above results we have:
*Theorem:* Algorithm VIS-LFAN finds the vertices visible from a vertex of a fan in O(n) time.

To find the visibility graph of the fan VIS-LFAN may be applied to the vertices $f_n, f_{n-1}, \ldots, f_2$ sequentially, resulting in an $O(n^2)$ algorithm. Note that step 0 is required only if a fan has its root at a concave vertex. In the application considered in this paper this case never arises.

The preceding method can be generalized for solving various other visibility problems in special classes of polygons. These can be found in [10].

## 4. Empirical Results

The algorithms described in sections 2 and 3 have been programmed. The following experiments have been performed.

An input of a random set of points was used and $|\Phi(S)|$ was computed. The experiment was repeated for different set sizes. A table in figure (4) shows a breakdown of the results. It is interesting to note that using this method we have obtained a set of 15 points with $|\Phi(S)| = 5$. A total of 380,000 experiments were performed with random sets of 16 points, and no set without empty 6-gons were found.

A different approach was then tried. Taking a point set with 15 points and no empty 6-gons a new point was tried to see if it caused the formation of an empty 6-gon. If no 6-gon was found then the point was added to the set, and the experiment was repeated. In this way a set of 20 points with no empty 6-gons was found. However to date that is the largest sized point set found such that $|\Phi(S)| = 5$. This configuration of points is shown in figure (2).

Harborth [7] shows a set of 9 points with no empty convex pentagons. As a result of these experiments a set of 9 points with $|\Phi(S)| = 4$ was generated figure (3). Goodman and Pollack [6] define an "order type" for sets of points. It can be shown that the configuration of 9 points shown in figure (3) differs from Harborth's example in "order type". We can conclude that a configuration of 9 points such that $|\Phi(S)| = 4$ is not unique.

Some timings are listed in figure (5). The timings represent CPU time required to perform 100 experiments, denoted in minutes and seconds.

The programs used to perform these experiments were written in the C language. The experiments were run on a VAX 750 computer running UNIX. These facilities were provided at the McGill School of Computer Science, Computational Geometry Laboratory.

## References


[1] V. Chvátal, G. Klincsek, *Finding Largest Convex Subsets*, Congresus Numeratium, Vol. 29 (1980) pp. 453-460.

[2] H. El-Gindy, D. Avis, *A Linear Algorithm for Computing the Visibility Polygon from a Point*, J. Algorithms 2(June 1981) pp. 186-197.

[3] P. Erdös, G. Szekeres *A Combinatorial Problem in Geometry*, Compositio Math. 2 (1935) pp. 463-470. (Also appears in [5]).

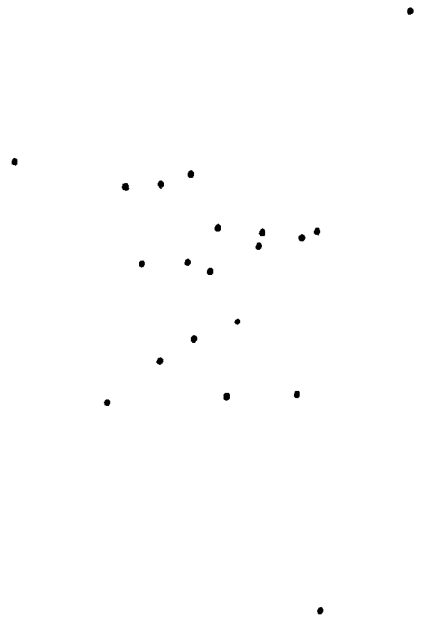
[4] P. Erdös, G. Szekeres *On Some Extremum Problems in Elementary Geometry* Ann. Univ Sci. Budapest 3-4 (1960/1) PP 53-62. (Also appears in [5]).

[5] P. Erdös, *Paul Erdös: The Art of Counting* M.I.T. Press, Ed. J.H. Spencer, 1973.

[6] J.E. Goodman, R. Pollack, *Geometric Sorting*, New York Academy of Sciences, Annals of Discrete Geometry and Convexity (1982).

[7] H. Harborth, *Konvex Funfecke in ebenen Punkmengen*, Elem. Math. 33 (1978) 116-118.

[8] J. D. Horton, *Sets with no Empty Convex 7-gons*, C. Math. Bull. 26 (Dec 1983) 482-484.



*20 Points with No Empty 6-gons*

*Coordinates :*

-671,1973; 189,8; 295,1797; 452,1167; 596,1829; 634,372; 850,1194; 856,1921; 922,565; 1043,1120; 1098,1485; 1210,88; 1287,721; 1450,1349; 1477,1461; 1808,125; 1818,1424; 1945,1487; 2053,-1641; 2697,3312.

figure (2)

*9 Points with No Empty Convex 5-gons*

*coordinates:*

-979,414; -969,273; -811,-239; -717,616; -128,976; -2,588; 23,-958; 199,-38; 312,-444.

figure (3)

| \|S\| | # exp. | \|Φ(S)\| | frequency |
|---|---|---|---|
| 7 | 10000 | 4 | 821 |
| | | 5 | 6266 |
| | | 6 | 2661 |
| | | 7 | 252 |
| 9 | 50000 | 4 | 1 |
| | | 5 | 11240 |
| | | 6 | 28090 |
| | | 7 | 9416 |
| | | 8 | 1186 |
| | | 9 | 67 |
| 14 | 10000 | 5 | 1 |
| | | 6 | 1571 |
| | | 7 | 5351 |
| | | 8 | 2516 |
| | | 9 | 495 |
| | | 10 | 58 |
| | | 11 | 8 |
| 15 | 120000 | 5 | 1 |
| | | 6 | 10398 |
| | | 7 | 62504 |
| | | 8 | 37101 |
| | | 9 | 8637 |
| | | 10 | 1218 |
| | | 11 | 135 |
| | | 12 | 4 |
| | | 13 | 2 |
| 16 | 380000 | 6 | 16778 |
| | | 7 | 180304 |
| | | 8 | 139923 |
| | | 9 | 36548 |
| | | 10 | 5743 |
| | | 11 | 662 |
| | | 12 | 40 |
| | | 13 | 2 |
| 20 | 20000 | 6 | 13 |
| | | 7 | 2162 |
| | | 8 | 5174 |
| | | 9 | 2162 |
| | | 10 | 412 |
| | | 11 | 62 |
| | | 12 | 8 |

*Empirical Results*

figure (4)

| \|S\| | CPU time |
|---|---|
| 10 | 0:15.4 |
| 15 | 0:54.0 |
| 20 | 2:10.3 |

*Sample Timings for 100 Experiments*

figure (5)

[9] W. Moser, *Research Problems in Discrete Geometry*, Department of Mathematics, McGill University, (1981) (problem 29).

[10] D. Rappaport, G. T. Toussaint, *A Simple Linear Hidden-Line Algorithm for Star-Shaped Polygons*, McGill University Tech. Report no. SOCS 83.23, November 1983. (To appear in Pattern Recognition Letters.)

[11] M.I. Shamos, *Problems in Computational Geometry*, Carnegie Melon University, 1977.