FINDING THE CONVEX HULL OF

A SIMPLE POLYGON IN LINEAR TIME

S. Y. Shin
T. C. Woo

Department of Industrial & Operations Engineering
The University of Michigan
Ann Arbor, Michigan 48109

February 1985

Technical Report 85-3
Finding the Convex Hull of a Simple Polygon in Linear Time

S. Y. Shin
T. C. Woo

Department of Industrial & Operations Engineering
University of Michigan
Ann Arbor, Michigan 48109

November 1984

This work was supported in part by AFOSR under contract F4920-82-C-0089 and in part by IBM Data Systems Division.
Summary

A new linear algorithm for finding the convex hull of a simple polygon is given. Based on the original idea by Sklansky (7), our version is easy to understand. Adopting the form of CH-POL by Toussaint and Avis (9), the presentation is concise. As shown in the Appendix, a PASCAL implementation of the algorithm itself is only half a page long.

In the paper, we define a "zipper" as a non-self-intersecting, concave chain. Choosing an extreme vertex of the polygon as the initial zipper, we update it by classifying a vertex of the given polygon by one of three cases. Case 1: vertex of the given polygon is added to the zipper. Case 2: vertex of the given polygon is not added to the zipper. Case 3: zipper vertex is deleted. We show that, after a complete traversal of the given polygon, the zipper thus constructed is the convex hull.
Abstract

Though linear algorithms for finding the convex hull of a simply-connected polygon have been reported, not all are short and correct. A compact version based on Sklansky's original idea (7) and Bykat's counter-example (8) is given. Its complexity and correctness are also shown.

Keywords: Convex hull, linear algorithm, computational geometry
1. Introduction

There have been many reports on a linear algorithm for finding the convex hull of a simple polygon. Certain versions were prone to counter-examples. In particular, a recent version by Ghosh and Shyamasundar \(^1\) turned out to be incorrect \(^2, 3\). Ideally, an algorithm should be not only correct but also easy to implement. McCallum and Avis \(^4\), for example, reported a version using two stacks. Lee \(^5\) used one stack but the algorithm itself was two pages long. Recently, Graham and Yao \(^6\) reported a compact algorithm that is said to be similar in spirit to Lee's version. Both \(^5\) and \(^6\) included two types of pocket test. In this paper, we present a version employing only one pocket test.

Perhaps, the simplest version is still the one presented by Sklansky \(^7\) in 1972. After a counter-example by Bykat \(^8\), sufficiency condition was established by Toussaint and Avis \(^9\) in 1982 and by Orlowsky \(^10\) in 1983. Almost concurrently, Sklansky gave a modified version \(^11\) but it was later shown to be incorrect by Toussaint and El Gindy \(^12\). Our search for a simple, concise, and correct linear convex hull algorithm traces the following path. For simplicity, we adopt the ideas from the original version by Sklansky \(^7\). For conciseness, we follow the form of CH-POL by Toussaint and Avis \(^9\). For correctness, we use the notion of a pocket (or lobe) as in Graham and Yao \(^6\) (or Lee \(^5\)) with Bykat's counter-example \(^8\) in mind.
2. Preliminaries

Let $P$ be a simple polygon with $n$ vertices. Each vertex $V_i$, $i = 0, 1, 2, \ldots, (n-1)$, is represented by its $X$ and $Y$ coordinates, $(X_i, Y_i)$. Let $V_0$ be the vertex with the minimum $Y$ coordinate. If two or more vertices are tied then we choose among them the vertex with the minimum $X$ coordinate as $V_0$. Starting from $V_0$ and traversing the boundary $B(P)$ of $P$ in the clockwise order, we label the $j$th vertex from $V_0$ as $V_i$, where $i$ is $j$ modulo $n$. These vertices in sequence are maintained as a circular doubly linked list. Throughout this paper we assume the following:

1. The boundary $B(P)$ of a simple polygon $P$ is traversed in the clockwise order from $V_0$.
2. No three consecutive vertices are collinear.

**Definition 2.1:** $L(P_i, P_j)$ denotes a directed line segment joining two points $P_i$ and $P_j$ in the direction from $P_i$ to $P_j$.

**Definition 2.2:** An edge $E(V_i, V_{i+1})$ of $P$ is a directed line segment $L(V_i, V_{i+1})$ joining two adjacent vertices $V_i$ and $V_{i+1}$ on $B(P)$. A chain $C(V_i, V_j)$ is a sequence of edges $E(V_i, V_{i+1}), E(V_{i+1}, V_{i+2}), \ldots, E(V_{j-1}, V_j)$ on $B(P)$ in the clockwise order.

**Definition 2.3:** A vertex $V_i$ of $P$ is **extreme** if $V_i$ cannot be expressed as a convex combination of other vertices in $P$, i.e., $V_i$ is extreme if and only if $V_i \neq \sum_{j \neq i} \alpha_j V_j$, $\sum_{j \neq i} \alpha_j = 1$, and $\alpha_j \geq 0$.

**Definition 2.4:** The **convex hull** $\text{CH}(P)$ of $P$ is the smallest convex polygon containing $P$. 

2
Definition 2.4: necessarily implies that every vertex of CH(P) is an extreme vertex of P. Hence, one way to find CH(P) is to discard all non-extreme vertices. To characterize a non-extreme vertex, we employ the notion of a pocket.

**Definition 2.5:** A pocket PKT(V_i, V_j) is one or more regions bounded by L(V_i, V_j) and C(V_i, V_j) such that all points in C(V_i, V_j) are on or to the right of L(V_i, V_j).

We state an interesting property of a pocket due to Graham and Yao (6).

**Lemma 2.1:** Let V_r be in a PKT(V_i, V_j). If V_r is neither V_i nor V_j, then V_r is not an extreme vertex of P.
3. Property of Zipper

A pocket PKT\((V_i, V_j)\) is said to be maximal with respect to \(C(V_0, V_q)\) if \(C(V_i, V_j)\) is not contained in another pocket PKT\((V_k, V_m)\), where \(0 \leq i < j \leq q\), and \(0 \leq k < m \leq q\). Let an ordered list \((Z_0, Z_1, Z_2, ..., Z_j)\) be the sequence of all vertices in \(C(V_0, V_q)\) such that PKT\((Z_i, Z_{i+1})\), \(0 \leq i < j\), is maximal with respect to \(C(V_0, V_q)\). The sequence of line segments \((L(Z_0, Z_1), L(Z_1, Z_2), ..., L(Z_{j-1}, Z_j))\) is said to be a zipper ZPR\((V_0, V_q)\) as illustrated in Figure 3.1.

<Insert Figure 3.1>

In this section, we show that ZPR\((V_0, V_q)\) is concave and non-self-intersecting. Our first lemma forms the basis for showing this property. In its proof and in all subsequent discussions, we use the following notations.

\(V_{q+1}: \) the most recently visited vertex in \(P\).

\(V_q: \) the previous (counter-clockwise) vertex of \(V_{q+1}\) in \(P\).

\(Z_j: \) the vertex that is most recently added into ZPR\((V_0, V_q)\).

\(Z_{j-1}: \) the previous vertex of \(Z_j\) in ZPR\((V_0, V_q)\).

\(V_*: \) the previous vertex of \(Z_j\) in \(P\).

Lemma 3.1: Let ZPR\((V_0, V_q)\) = \((L(Z_0, Z_1), L(Z_1, Z_2), ..., L(Z_{j-1}, Z_j))\) and \(0 < q < n\). Any vertex \(V_k\) in the chain \(C(Z_r, V_q)\) must be to the right of \(L(Z_i, Z_{i+1})\), \(0 \leq i \leq r \leq j\), if \(V_k \neq Z_{i+1}\).
Figure 3.1: Vertices of a polygon $P$ and $ZPR(V_0,V_q)$
[Proof] The proof will be by the induction on the subscript $i$ of a zipper vertex $Z_i$ in $ZPR(V_0, V_q)$. Let $Z_{-1}$ be a point on the horizontal line containing $Z_0$ such that $Z_{-1}$ lies to the right of $Z_0$. Let $L_1$ be the line containing $L(Z_{i-1},Z_i)$, $i=0,1,2,...,j$. $L_1$ partitions the plane into two half planes. Let $LHP_1$ be the half plane to the left of $L(Z_{i-1},Z_i)$ and $RHP_1$ be the other.

$i = 0$: Since $V_0$ is extreme, $V_0$ coincides with $Z_0$. By the way in which $V_0$ is chosen, the $Y$ coordinate of $V_0$ is not greater than the $Y$ coordinate of any other vertex in $P$. Therefore, $C(V_0, V_q)$ cannot pass through $LHP_0$. Now, $RHP_0$ is partitioned by $L_1$ into two regions, $RHP_0 \cap LHP_1$ and $RHP_0 \cap RHP_1$. We need to show that $C(Z_1, V_q)$ cannot be in $RHP_0 \cap LHP_1$. Suppose that some vertices in $C(Z_1, V_q)$ are in $RHP_0 \cap LHP_1$. Let $W$ be the vertex in $C(Z_1, V_q)$ such that $C(Z_0, W)$ is to the right of $L(Z_0, W)$. Clearly, $PKT(Z_0, W)$ contains $C(Z_0, Z_1)$, which contradicts the maximality of $PKT(Z_0, Z_1)$. Suppose that the lemma is true for $i = m-1 < j-2$.

$i=m$: We need to show that $C(Z_{i+1}, V_q)$ cannot be in $R = \bigcap_{p=0}^{m} RHP_p \cap LHP_{m+1}$ as shown in Figure 3.2

<Insert Figure 3.2>

Suppose that some vertices in $C(Z_{m+1}, V_q)$ are in $R$. Let $W$ be the vertex in $C(Z_{m+1}, V_q)$ such that $C(Z_m, W)$ is to the right of $L(Z_m, W)$. $PKT(Z_m, W)$ contains $C(Z_m, Z_{m+1})$, which contradicts the maximality of $PKT(Z_m, Z_{m+1})$. 


Figure 3.2: \( C ( V_{m+1}, V_q ) \) cannot be in \( \bigcap_{p=0}^{m} RHP_p \cap LHP_{m+1} \).
As illustrated in Figure 3.3, the property described in Lemma 3.1 does not necessarily hold true unless \( V_0 \) is an extreme vertex of \( P \). We next state the lemmas characterizing a \( \text{ZPR}(V_0, V_q) \), the proofs of which are direct consequences of Lemma 3.1.

\[ \text{<Insert Figure 3.3>} \]

**Lemma 3.2:** Let \( \text{ZPR}(V_0, V_q) = (L(Z_0, Z_1), L(Z_1, Z_2), \ldots, L(Z_{j-1}, Z_j)) \). The internal angle \( \text{ANGLE}(Z_i, Z_{i+1}, Z_{i+2}) \) between two consecutive line segments \( L(Z_i, Z_{i+1}) \) and \( L(Z_{i+1}, Z_{i+2}) \), \( 0 \leq i \leq j-2 \), is strictly between zero and 180 degrees.

**Lemma 3.3:** A \( \text{ZPR}(V_0, V_q) \) is not self-intersecting.

Finally, we show that a zipper vertex \( Z_k \) cannot be in a pocket \( \text{PKT}(Z_i, Z_{i+1}) \) if \( k \neq i \) and \( k \neq i+1 \). We use this property to update \( \text{ZPR}(V_0, V_q) \).

**Lemma 3.4:** Let \( \text{ZPR}(V_0, V_q) \) be \( (L(Z_0, Z_1), L(Z_1, Z_2), \ldots, L(Z_{j-1}, Z_j)) \).

Then, \( Z_k \cap \text{PKT}(Z_i, Z_{i+1}) = Z_k \) if \( k = i \) or \( i+1 \)

\[ \emptyset \quad \text{otherwise} \]

for all \( 0 \leq i < j \) and \( o \leq k \leq j \).

[Proof] Suppose that \( Z_k \cap \text{PKT}(Z_i, Z_{i+1}) \neq \emptyset \) for some \( k \neq i \) and \( k \neq i+1 \). Then either \( P \) is not simple or \( V_0 \) is not an extreme point.
Figure 3.3: Lemma 3.1 does not hold true if $V_0$ is not extreme.
4. Updating of Zipper

Consider the relationship between two line segments \( L(Z_{j-1}, Z_j) \) and \( E(V_*, Z_j) \). As illustrated in Figure 4.1, the vertex \( V_{q+1} \) can be in any one of the four quadrants formed by the extensions of these two line segments. The quadrants are:

- **Qla**: to the right of \( L(Z_{j-1}, Z_j) \) and to the right of \( E(V_*, Z_j) \)
- **Qlb**: to the right of \( L(Z_{j-1}, Z_j) \) and to the left of \( E(V_*, Z_j) \)
- **Q2a**: to the left of \( L(Z_{j-1}, Z_j) \) and to the right of \( E(V_*, Z_j) \)
- **Q2b**: to the left of \( L(Z_{j-1}, Z_j) \) and to the left of \( E(V_*, Z_j) \)

<Insert Figure 4.1>

If \( V_{q+1} \) is in Qlb, it is also in \( PKT(Z_{j-1}, Z_j) \). By Lemma 2.1, \( V_{q+1} \) and its clockwise vertices in \( PKT(Z_{j-1}, Z_j) \) can be deleted. Otherwise, we need to show if the existing zipper vertices are to be deleted or kept to advance to \( V_{q+1} \). The following three lemmas as illustrated in Figure 4.2 are useful for the updating of \( ZPR(V_0, V_q) \).

<Insert Figure 4.2>

**Lemma 4.1:** Let \( ZPR(V_0, V_q) = (L(Z_0, Z_1), L(Z_1, Z_2), ..., L(Z_{j-1}, Z_j)) \) and \( V_q = Z_j \neq V_0 \). All pockets \( PKT(Z_i, Z_{i+1}) \), \( 0 \leq i < j \), are maximal with respect to \( C(V_0, V_{q+1}) \), if \( V_{q+1} \) is in Qla.

[Proof] If \( V_{q+1} = V_0 \), then \( ZPR(V_0, V_q) \) together with \( E(V_q, V_{q+1}) \) forms a convex polygon since \( V_{q+1} = V_0 \) and \( ZPR(V_0, V_q) \) is concave and non-self-
Figure 4.1: Possible locations of vertex $V_{q+1}$
(a) Illustration of Lemma 4.1

(b) Illustration of Lemma 4.2

(c) Illustration of Lemma 4.3

Figure 4.2. Updating Zipper Vertices
intersecting. Therefore, the result follows immediately.

Let us consider the case for $V_{q+1} \neq V_0$. Since ZPR($V_0,V_q$) implies that PKT($Z_1,Z_{i+1}$), $o \leq i < j$, is maximal with respect to C($V_0,V_q$), all we need to show is that E($Z_j,V_{q+1}$) is PKT($Z_j,V_{q+1}$) and is maximal with respect to C($V_0,V_{q+1}$). First we show E($Z_j,V_{q+1}$) $\cap$ PKT($Z_1,Z_{i+1}$) $\neq$ E($Z_j,V_{q+1}$) for any $o \leq i < j$. By Definition 2.5, $V_{q+1}$ cannot be in PKT($Z_{j-1},Z_j$) since $V_{q+1}$ is in Q1a. From Lemma 3.4, $Z_j$ cannot be in PKT($Z_1,Z_{i+1}$) for any $o \leq i < j-1$. Therefore, E($Z_j,V_{q+1}$) $\cap$ PKT($Z_1,Z_{i+1}$) $\neq$ E($Z_j,V_{q+1}$) for any $o \leq i < j$.

Finally, there does not exist a vertex $V_r$ in C($V_0,V_w$) such that C($V_r,V_{q+1}$) and L($V_r,V_{q+1}$) form a pocket PKT($V_r,V_{q+1}$) since ZPR($V_0,V_q$) is concave and P is simple. Hence, the result follows.

**Lemma 4.2:** Let ZPR($V_0,V_q$) = (L($Z_0,Z_1$), L($Z_1,Z_2$), ..., L($Z_{j-1},Z_j$)). If C($V_q,V_r$), $r > q$, is in PKT($Z_{j-1},Z_j$), then $V_{r+1}$ is also in PKT($Z_{j-1},Z_j$) unless $V_{r+1}$ is to the left of L($Z_{j-1},Z_j$).

**[Proof]** Since P is simple, C($V_q,V_{r+1}$) can get out of PKT($Z_{j-1},Z_j$) only through L($Z_{j-1},Z_j$).

**Lemma 4.3:** Let ZPR($V_0,V_q$) = (L($Z_0,Z_1$), L($Z_1,Z_2$), ..., L($Z_{j-1},Z_j$)). Then PKT($Z_{j-1},Z_j$) is not maximal with respect to C($V_0,V_{q+1}$), if $V_{q+1}$ is in quadrant Q2a or Q2b.

**[Proof]** $\angle(Z_{j-1},Z_j,V_{q+1})$ is greater than or equal to 180 degrees since $V_{q+1}$ is in Q2a or Q2b. Since ZPR($V_0,V_q$) is concave and non-self-intersecting, there must exist a vertex $Z$ in ZPR($V_0,V_q$) such that L($Z,V_{q+1}$) and C($Z,V_{q+1}$) form a pocket PKT($Z,V_{q+1}$). Clearly, PKT($Z,V_{q+1}$) contains C($Z_{j-1},Z_j$).
5. The Algorithm and Its Analysis

Our linear algorithm for finding the convex hull of a simple polygon $P$ takes $V_i$, $i=0,1,...,n-1$, as input and constructs a $ZPR(V_0,V_q)$ with vertices $Z_j$.

Algorithm 5.1

Step 0. $Z_0 \leftarrow V_0$, $Z_1 \leftarrow V_1$, $j \leftarrow 1$, $q \leftarrow 1$.

while ($V_q$ ≠ $V_0$) do;

Step 1. if $V_{q+1}$ is to the right of $L(Z_{j-1},Z_j)$, then do;

Step 1a. if $V_{q+1}$ is to the right of $E(V_*,Z_j)$

then $j \leftarrow j+1$, $Z_j \leftarrow V_{q+1}$, $q \leftarrow q+1$.

Step 1b. else while ($V_{q+1}$ is on or to the right of $L(Z_{j-1},Z_j)$) do;

$q \leftarrow q+1$

end

end

Step 2. else do;

while ($Z_j$ ≠ $V_0$ and $Z_{j-1}$ is not to the right of $L(Z_j,V_{q+1})$) do;

$j \leftarrow j-1$

end.

$j \leftarrow j+1$, $Z_j \leftarrow V_{q+1}$, $q \leftarrow q+1$.

end

end

Step 3. Stop.

We show the correctness of Algorithm 5.1 with the following lemma.

Lemma 5.1: Algorithm 5.1 constructs $ZPR(V_0,V_q)$ correctly.
[Proof] The proof will be by induction on the number of times Step 1 is reached. Initially, the statement is trivially satisfied by Step 0 of the algorithm. Suppose that the lemma is true when Step 1 is executed m times.

Then, there are three cases:

(1) Case 1a: $V_{q+1}$ is in Qla

(2) Case 1b: $V_{q+1}$ is in Qlb

(3) Case 2: $V_{q+1}$ is in Q2a or Q2b

Case 1a: $V_{q+1}$ qualifies as a zipper vertex if $PKT(Z_j, V_{q+1})$ is maximal with respect to $C(V_0, V_{q+1})$. Since $V_{q+1}$ is in quadrant Qla, by Lemma 4.1, $PKT(Z_j, V_{q+1})$ is maximal. Indeed, Step 1a takes $V_{q+1}$ as the new $Z_j$. Since the correct vertex is added to the zipper the next time Step 1 is reached, the induction holds. Now, Lemma 4.1 requires the precondition that $V_q$ equals $Z_j$. This precondition is satisfied iteratively after executing Step 1a or Step 2. After executing Step 1b, though $V_q \neq Z_j$, the control must go to Step 2 because $V_{q+1}$ cannot be to the right of $L(Z_{j-1}, Z_j)$. Hence, the precondition for Lemma 4.1 is always satisfied.

Case 1b. Because $V_{q+1}$ is in quadrant Qlb, by Definition 2.5, $V_{q+1}$ is in $PKT(Z_{j-1}, Z_j)$. Therefore, $V_{q+1}$ should not be a zipper vertex. Furthermore, by Lemma 4.2, all the subsequent vertices in $PKT(Z_{j-1}, Z_j)$ should not be in the zipper $ZPR(V_0, V_q)$ either. This is precisely what Step 1b does. Since no zipper vertex is added, the next time Step 1 is reached, $ZPR(V_0, V_q)$ is still correct.
Case 2: Step 2 deletes $Z_j$ since $\text{PKT}(Z_{j-1}, Z_j)$ is not maximal with respect to $C(V_0, V_{q+1})$ by Lemma 4.3. The old $Z_{j-1}$ becomes the new $Z_j$. This process is repeated until either $Z_j = Z_0$ or $Z_{j-1}$ is to the right of $L(Z_j, V_{q+1})$. At that point $\text{PKT}(Z_j, V_{q+1})$ is maximal with respect to $C(V_0, V_{q+1})$, because $\text{ZPR}(V_0, V_{q+1})$ is concave and non-self-intersecting. Hence, the lemma is true.

When $V_{q+1}$ coincides with $V_0$, Step 3 terminates the algorithm, and the lemma is still true by the induction hypothesis.

Since $\text{ZPR}(V_0, V_q)$ is concave and non-self-intersecting, it must form a convex polygon $P_C$ containing $P$ if $V_q = V_0$. Since every vertex of $P_C$ is a vertex of $P$, it is clear that $P_C$ is the smallest convex polygon containing $P$. By Definition 2.4, $P_C$ must be the convex hull of a simple polygon $P$.

**Theorem 5.1:** Algorithm 5.1 finds the convex hull of a simple polygon $P$ with $n$ vertices in $O(n)$ time.

[Proof] The algorithm moves forward, except in Step 2, until $V_0$ is revisited. Step 2 is executed at most a total of $n-3$ times.
6. Concluding Remarks

Algorithm 5.1 removes the vertices that cause self-intersection (8) in CH-POL (9). It is shorter than the version by Graham and Yao (6) when both the Left Hull and the Right Hull are taken into account.

Acknowledgement

The authors wish to thank J. D. Wolter and H. C. Lee for their critical reading of the manuscript and their constructive suggestions. J. D. Wolter implemented Algorithm 5.1 in several languages. His version in PASCAL is supplied in the Appendix.
References


2. R. Shyamasundar, Note on a Linear Time Algorithm for Obtaining the Convex Hull of a Simple Polygon, private communication, August 28, 1984.


Appendix

program main (input, output);

var  X, Y: array [0..50] of real;       {coordinates of points}
    Y, Z: array [0..50] of integer;    {polygon and hull}
    q, j: integer;                     {index into polygon and hull}
    n: integer;                        {number of vertices}
    i: integer;                        {loop index}

{ Is point p to the left of Line (a,b)? } function left (p, a, b: integer): boolean;
begin
  left := (Y[p] - Y[a])*(X[b] - X[a]) > (X[p] - X[a])*(Y[b] - Y[a]);
end;

{ Is point p to the right of Line (a,b)? } function right (p, a, b: integer): boolean;
begin
  right := (Y[p] - Y[a])*(X[b] - X[a]) < (X[p] - X[a])*(Y[b] - Y[a]);
end;

{ Read in the Polygon }
procedure readin;
var  i: integer;
    W: array [0..50] of integer;
    mx, my: real;
    mi: integer;
begin
{ Read in the number of points }
  repeat
    write(' Number of points? ');
    read(n);
  until (n > 3) and (n < 50);

  mx := le38;
  my := le38;

{ While reading in vertices, find an extremal one }
  for i := 0 to n-1 do begin
    write(' ', i+3, ': ');
    read(X[i], Y[i]);
    W[i] := i;
    if (Y[i] < my) or ((Y[i] = my) and (X[i] < mx))
       then begin
       mx := X[i];
       my := Y[i];
       mi := i;
      end;
  end;
{ Reorder with an extreme vertex first }
V[n] := W[mi];
for i := 0 to n-1
do begin
   V[i] := W[mi];
   mi := (mi + 1) mod n;
end;
end;

begin
{ Get the polygon, and echo it back }
readin;
writeln(' Polygon:');
for i := 0 to n-1 do
   writeln(' ',l:3,'= ',X[V[i]]:10:5,', ',Y[V[i]]:10:5);

{Step 0}
q := 1;
j := 1;
Z[0] := V[0];
Z[1] := V[1];

while (q < n) do
   if right ( V[q+1], Z[j-1], Z[j] )
      then
         if right ( V[q+1], V[q-1], V[q] )
            then begin
               { Step 1a }
               j := j + 1;
               q := q + 1;
               Z[j] := V[q];
            end
      else begin
         { Step 1b }
         while not left ( V[q+1], Z[j-1], Z[j] ) do
            q := q + 1;
      end
   else begin
      { Step 2 }
      while j > 0 and not right ( Z[j-1], Z[j], V[q+1] ) do
         j := j - 1;
      j := j + 1;
      q := q + 1;
      Z[j] := V[q];
   end;

{ Print the hull }
writeln(' Hull:');
for i := 0 to j-1 do
   writeln(' ',l:3,'= ',X[Z[i]]:10:5,', ',Y[Z[i]]:10:5);