FINDING THE SET OF ALL MINIMAL NESTED CONVEX POLYGONS
(EXTENDED ABSTRACT)

J. Bhadury, *University of New Brunswick, Fredericton, N.B., Canada*
R. Chandrasekaran, *The University of Texas at Dallas, Richardson, TX , U.S.A.*

Abstract. Given two simple polygons P_{in} and P_{out}, with P_{in} completely contained in P_{out}, the Minimal Nested Polygon problem is to find a polygon P* that is completely contained in the annulus between P_{out} and P_{in}, contains P_{in} and has the minimum number of edges. Given P_{in} and P_{out}, there may be an infinite number of Minimal Nested Polygons. In this paper, we consider the case where P_{in} and P_{out} are convex and have attempted to characterize the set of all Minimal Nested Polygons by giving two different algorithms. The first partitions the annulus into disjoint regions to identify all points in the annulus that can be assumed to be the vertex of some Minimal Nested Polygon. The second algorithm identifies those points that can be assumed to lie on some Minimal Nested Polygon. The time taken by the algorithms is $O(n\Phi)$, where n is the total number of edges in P_{out} and P_{in} and Φ is the number of edges in P*.

1 Introduction

The Minimal Nested Polygon problem is defined as the following: given two simple polygons P_{in} and P_{out}, with P_{in} completely contained in P_{out} (i.e. $P_{in} \subset P_{out}$), find a nested polygon (i.e. one that is contained in the annulus between P_{in} and P_{out} and contains the inner polygon P_{in}) P*, that has the fewest number of edges. This problem has applications in robotics, collision avoidance, stock cutting etc. and is hence extensively studied - see [1],[2],[3],[5]. When P* is known to be non-convex, an $O(n)$ time algorithm is given in [3] to find it, where n is the total number of edges of P_{in} and P_{out}. However if P* is known to be convex, the best algorithm is in [1], which takes $O(n\log\Phi)$ time, where Φ is the number of edges in P* (for a characterization of when P* is convex see [4]). Given a pair of nested polygons P_{out} and P_{in}, P* is not unique (for example when P_{out} (respectively, P_{in}) is a large (respectively, small) triangle) - hence the question '*for a given pair of nested polygons P_{in} and P_{out} can the set of all Minimal Nested Polygons be characterized* ?'. This is important in applications where alternate Minimal Nested Polygons are needed because of one being preferable to another. In this paper we have attempted to address this question in the case where P_{in} and P_{out}, (and hence P*, by [4]) are convex. Two indices are first defined for every point in the annulus - the Polygon Index and the Turn Index respectively. Two algorithms are then given that identify all points in the annulus that can either be assumed to be the *vertex* of some P* or *lying* on some P*. The algorithms are shown to take $O(n\Phi)$ time.

2 Preliminaries

As mentioned before, P_{out} and P_{in} are assumed to be convex polygons, with $P_{in} \subset P_{out}$ and the Minimal Nested Polygon for P_{out} and P_{in} is assumed to be P*. n is assumed to represent the total number of edges of P_{in} and P_{out} and Φ, the total number of edges in P*. The entire annular region between P_{out} and P_{in} is referred to as the annulus and designated by [P_{out}-P_{in}]. The boundary of P_{out} (respectively, P_{in}) is referred to as $bd(P_{out})$ (respectively, $bd(P_{in})$) and (P_{out}-P_{in}) refers to all the points that are in the interior of the annulus - i.e. all points in [P_{out}-P_{in}] except those on $bd(P_{out})$ and $bd(P_{in})$. For two points $x,y \in$ [P_{out}-P_{in}], $[x \rightarrow y]$ is assumed to represent a ray from x in the direction of y. Let x and y be two points on $bd(P_{out})$, with the property that P_{in} is completely on one side of the line segment $[x,y]$ and let z be any point on $bd(P_{out})$ that is on the same side of $[x,y]$ as P_{in}. Then, in a clockwise traversal of $bd(P_{out})$ that begins at z, if x is encountered after y, then x is said to be **clockwise** of y. Further, x is said to be **atleast**

clockwise of y if either x and y are coincident or if x is clockwise of y.

For any point $v \in [P_{out}-P_{in}]$, $P(v)$ is defined as a nested polygon that passes through v and has the minimum number of edges and $T(v)$ as a nested polygon that has v as a vertex and has the minimum number of edges. $\|P(v)\|$ (called the **Polygon Index**) and $\|T(v)\|$ (the **Turn Index**) denote the number of edges in $P(v)$ and $T(v)$ respectively (obviously, $\|T(v)\| \geq \|P(v)\|$).

For any point $v \in [P_{out}-P_{in}]$ a clockwise **greedy structure** $G(v)$ is defined, that is obtained as follows: (figure 1 shows $G(a)$ for point $a \in bd(P_{out})$): from v the clockwise tangent to P_{in} is drawn - this tangent is assumed to intersect $bd(P_{in})$ (or, in other words, is tangential to P_{in}) at a vertex denoted by ***Tgt(v)*** (in case it intersects two vertices, $Tgt(v)$ is defined as the more clockwise of these two vertices). The point of intersection of the ray $[v \rightarrow Tgt(v)]$ with $bd(P_{out})$ is defined as v_1. From v_1 this process is continued and successive points v_2, v_3 etc. (called the vertices of $G(v)$) are defined on $bd(P_{out})$ similarly until that vertex of $G(v)$ is reached where v becomes visible for the first time - it will be assumed throughout the paper that for any point v, this occurs on the $k^{th.}$ vertex of $G(v)$. Then another clockwise tangent is drawn from v_k to P_{in} to obtain the next vertex of $G(v)$ (i.e the point v_{k+1}) in a manner similar to the other vertices. The line segments $[v,v_1],[v_1,v_2],..[v_k,v_{k+1}]$ constitute $G(v)$ - and these line segments are referred to as the edges of $G(v)$. $\|G(v)\|$ is used to denote the number of edges in $G(v)$. If the sequence of *anticlockwise* tangents is taken from v, the resulting greedy structure is denoted by $G_a(v)$. The point of intersection of the line segments $[v,v_1]$ and $[v_k,v_{k+1}]$ is denoted by ***Int(v)***. A point $v \in bd(P_{out})$ is defined as a **tight point** if $v = v_\Phi$. For such a tight point, $G(v)$ is a closed polygon and is referred to as a **tight greedy polygon** for the tight point v.

For any point $v \in [P_{out}-P_{in}]$, consider the anticlockwise tangent from v to P_{in} - the vertex of P_{in} that this anticlockwise tangent from v intersects is denoted by ***Atgt(v)*** - if it intersects two vertices, the more clockwise of these two is chosen as $Atgt(v)$. Then the point of intersection of the ray $[v \rightarrow Atgt(v)]$ with $bd(P_{out})$ is denoted as ***Anti(v)*** - and since $P_{in} \subset P_{out}$, if $v \in bd(P_{out})$, v and $Anti(v)$ will occur on different edges of $bd(P_{out})$. For any point $v \in bd(P_{out})$ with $\|G(v)\| = \Phi$, the **slack cone** of v is defined as the entire region of the annulus bounded by the line segments $[v,v_{\Phi-1}]$, $[v,Anti(v)]$ and the section of $bd(P_{out})$ between $Anti(v)$ and $v_{\Phi-1}$ (including these boundaries themselves). For example in figure 1, if $\|G(a)\| = \Phi$ (and hence $a_k = a_{\Phi-1}$), then the slack cone of a is the triangle $[a, Anti(a), a_k]$. Note that slack cone for a point v is only defined if $v \in bd(P_{out})$ and $\|G(v)\| = \Phi$. For a point $v \in [P_{out}-P_{in}]$, the **projector** of v, denoted by ***Proj(v)*** is defined as the point of intersection of the ray $[v_1 \rightarrow v]$ with $bd(P_{out})$ - in figure 1, a is the projector of $Int(a)$. For all $v \in bd(P_{out})$, $Proj(v) = v$.

The following results are either known or easy to verify: **(i)** It is shown in [1] that for any $v \in bd(P_{out})$, $\Phi \leq \|G(v)\| \leq \Phi+1$. **(ii)** If v is a tight point then $\|G(v)\| = \Phi$. Further, since $P_{in} \subset P_{out}$, for a tight point v, $v_k = v_{\Phi-1} = Anti(v)$ and $v = v_{k+1} = v_\Phi$. **(iii)** For any point $v \in bd(P_{out})$, $\|G(v)\| = \Phi+1$ iff $Anti(v)$ (respectively, v) is clockwise of $v_{\Phi-1}$ (respectively, v_Φ). **(iv)** For any point $v \in [P_{out}-P_{in}]$, the edges $[v,v_1],[v_1,v_2],..,[v_k,v]$ represents $T(v)$. Hence $\|T(v)\| = \|G(v)\|$. Based on these, we now state the following results.

Lemma 1: *For any point* $v \in [P_{out}-P_{in}]$, $\Phi \leq \|G(v)\| \leq \Phi+2$ (hence, $\Phi \leq \|T(v)\| \leq \Phi+2$).
Lemma 2: *For any point* $v \in [P_{out}-P_{in}]$, $\Phi \leq \|P(v)\| \leq \Phi+1$.
Lemma 3: *For any* $v \in [P_{out}-P_{in}]$, $\|P(v)\| = \Phi$ *iff v is in the slack cone of a point* $x \in bd(P_{out})$.

2 Partitioning *bd*(P_{out}) Into *Critical Intervals*

Consider a point v on *bd*(P_{out}) such that $\|G(v)\| = \Phi+1$ (and hence v is clockwise of v_{Φ}). As v is moved clockwise on *bd*(P_{out}), along the edge of P_{out} that it lies on, all vertices and edges of *G(v)* move clockwise too - a direct consequence of the fact that *bd*(P_{out}) and *bd*(P_{in}) are continuous, and $P_{in} \subset P_{out}$. During this movement of v, the following four events (heretofore referred to as events I through IV) that can occur will be of interest to us: **(I)** An edge of *G(v)* can encounter a new vertex of P_{in}. **(II)** A vertex of *G(v)* can encounter a new vertex of P_{out}. **(III)** v can encounter the next vertex of P_{out}. **(IV)** v can encounter a tight point on *bd*(P_{out}). We now give an algorithm to partition *bd*(P_{out}) into intervals that are "small" enough such that if the point v is restricted to move inside an interval, none of events I through IV will occur.

<u>*Algorithm Partition - bd(P_{out})*</u>

1. Every edge of P_{in} is extended to intersect with *bd*(P_{out}) and the two points of intersection are considered critical points. Every vertex of P_{out} is also considered a critical point.
2. For every critical point v found in Step 1, find the greedy structures *G(v)* and $G_a(v)$ and the vertices of these two structures for this critical point are also included as critical points - for each critical point v store the following: the value of k; the points *Anti(v)*, *Tgt(v)* and *Atgt(v)*; all the vertices of *G(v)* (including v_{k+1}); the points $Tgt(v_j)$, $Atgt(v_j)$ and functions $\delta v_j(d)$, for $1 \le j \le k+1$.
3. The critical points obtained above in Steps 1 and 2 partition *bd*(P_{out}) into disjoint intervals - for each interval do the following: check if there exists any tight point within this interval by checking if there is a solution to the quadratic $v(d) = v_{\Phi}(d)$ in this interval (there can be atmost 2 tight points per interval). If a tight point exists, then for this tight point v, draw the tight greedy polygon and this tight point and the vertices of its associated tight greedy polygon are also included as critical points. For each such tight point v, store all parameters in Step 2. ◇

It can be shown that *Partition-bd(P_{out})*, takes $O(n\Phi)$ time and produces a total of $O(n\Phi)$ critical points *bd*(P_{out}) that partition it into as many intervals (heretofore referred to as *critical intervals*) such that any two points within an interval have the same Turn Index. Further, note that a tight point $v \in$ *bd*(P_{out}) represents a 'crossing over' of the two points v and v_{Φ}. Hence, if there exist two points $x,y \in$ *bd*(P_{out}) such that $\|G(x)\|$ and $\|G(y)\|$ are not equal, then there must exist atleast one tight point on the section of *bd*(P_{out}) between x and y. This leads to:

Corollary 4: *After Algorithm Partition-bd*(P_{out}) *is over, for every critical interval* [*E,F*]. ***(i)*** *If E is not a tight point then Turn Index in (E,F)* $= \|G(E)\|$. ***(ii)*** *If Turn Index in the interval (E,F) is* Φ *then it is guaranteed that* $\|G(E)\| = \|G(F)\| = \Phi$.

3 Partitioning of the Annulus Based On $\|T(v)\|$

Now we give a polynomial time algorithm based on lemma 1, to partition the annulus according to $\|T(v)\|$. The algorithm is based on the idea of moving a point v clockwise on *bd*(P_{out}) and tracing the locus of the point *Int(v)*.

<u>*Algorithm Partition - $\|T(v)\|$*</u>

1. Find a P* and Φ for P_{in} and P_{out} using the algorithm in [1].
2. Partition *bd*(P_{out}) into critical intervals using *Partition-bd*(P_{out}).
3. For each critical interval [*E,F*] (assume that *F* is clockwise of *E*) do {
   3.1 Retrieve *G(E)* and *G(F)* and identify *Int(E)* and *Int(F)*, *Tgt(E)* and *Tgt(F)*.
   3.2 Obtain the values of $E^1, E^2, \tan\theta$ and $E_j^1, E_j^2, \tan\theta_j$ for $j=1,k,k+1$. Retrieve the functions $\delta E_j(d)$, $j=1,k,k+1$. Using them obtain the locus of *Int(v)* within this interval [*E,F*] as a point v is moved from *E* to *F*.

3.3 Partition the area of the annulus $[P_{out}\text{-}P_{in}]$ between the line segments $[E,Tgt(E)]$,$[F,Tgt(E)]$ and the interval $[E,F]$ into two sets - the 'outer' set that is bounded by the critical interval $[E,F]$, the line segments $[E,Int(E)]$ and $[F,Int(F)]$, and the locus of $Int(v)$ from $Int(E)$ to $Int(F)$. The 'inner' set is bounded by $[Int(E),Tgt(E)]$ and $[Int(F),Tgt(E)]$, and the locus of $Int(v)$ from $Int(E)$ to $Int(F)$. If $E = Int(E)$, the two sets are defined similarly. See figure 2.

3.4 If $E_\Phi \neq E$ then {

3.4.1 The following points are labelled with Turn Index = $\|G(E)\|$: all points in the interior of the outer set, all points on $bd(P_{out})$ in the interval $[E,F)$, all points on the locus of $Int(v)$ (except $Int(F)$), and all points on the line segment $[E,Int(E)]$.

3.4.2 The following points are labelled with Turn Index = $\|G(E)\| + 1$: all points in the interior of the inner set and all points on the line segment $(Int(E),Tgt(E)]$. } **end if**

3.5 If $E_\Phi = E$, (i.e. E is a tight point) **then** {

3.5.1 Retrieve the function $\delta E_\Phi(d)$. By examining its first two derivatives at $d = 0$, determine whether, for a point inside the interval, the Turn Index is Φ or $\Phi + 1$.

3.5.2 The following points receive a label of Turn Index = Turn Index in (E,F): all points on $bd(P_{out})$ in the interval (E,F); all points on the locus of $Int(v)$ (except $Int(F)$) and all points in interior of the outer set.

3.5.3 The following points are labelled with Turn Index = Turn Index in $(E,F) + 1$: all points on the line segment $(E,Tgt(E)]$; and all points in the interior of the inner set.

3.5.4 The point E is labelled with a Turn Index equal to Φ. } **end if** } **end for** ◇

It can be shown that Algorithm *Partition-*$\|T(v)\|$ takes $O(n\Phi)$ time and partitions the annulus into as many disjoint regions with the property that all points within the same region have the same Turn Index. Hence, for any point $v \in [P_{out}\text{-}P_{in}]$, there exists a P* passing through v with v as its vertex iff $\|T(v)\| = \Phi$ and hence this partitioning identifies all points in the annulus that can be assumed to be the vertex of some Minimal Nested Polygon for P_{out} and P_{in}.

4 Identifying Points In The Annulus With $\|P(V)\| = \Phi$.

Now we address the following question: given any point $x \in [P_{out}\text{-}P_{in}]$, is there a P* passing through x, with or without a vertex at x, and if so, produce it. This issue is addressed now by developing a scheme based on lemma 3 that identifies all points in the annulus with a Polygon Index of Φ.

After partitioning $bd(P_{out})$ using *Partition-bd*(P_{out}), for every critical interval $[E,F]$ with Turn Index equal to Φ, we move a point v from E to F and find the entire region swept out by the slack cone of the point v (see figure 3) - and because v remains within a critical interval, $v_{\Phi-1}$ remains atleast clockwise of $Anti(v)$ and the slack cone exists for each point in the interval. The region swept out is bounded by the following - the interval $[E,F]$, the section of $bd(P_{out})$ between $Anti(E)$ and $F_{\Phi-1}$ and the two envelopes formed by the line segments $[v,Anti(v)]$ and $[v,v_{\Phi-1}]$ as v moves from E to F - called the 'inner' and the 'outer' envelopes respectively. The inner envelope is given by the pair of straight lines $[Anti(E),Atgt(E)]$, $[Atgt(E),F]$. The outer envelope can be found on a case by case basis. Because E and F are adjacent critical points, they will lie on the same edge of P_{out} and the same is therefore true of the pair of points $E_{\Phi-1}$ and $F_{\Phi-1}$ - however these two pairs of points may all be collinear or not; this gives rise to the following two cases: __Case (A)__: When $E_{\Phi-1}$ and $F_{\Phi-1}$ are not on the same edge as E and F (as in figure 3). Here the upper envelope is the pointwise maximum of the line segment $[v,v_{\Phi-1}]$ as v is moved from E to F. __Case (B)__: When the points $E_{\Phi-1}$,$F_{\Phi-1}$,E and F are collinear. Here the upper envelope is the section of $bd(P_{out})$ between $Anti(E)$ and F. Based on this, the algorithm is as below.

Algorithm - $\|P(v)\|$

1 Find a P* and Φ for P_{in} and P_{out} using the algorithm in [1].

2 Partition $bd(P_{out})$ into critical intervals using *Algorithm Partition-bd(P_{out})*.

3 For each critical interval $[E,F]$ (assume that F is clockwise of E) **do** {

3.1 Retrieve $G(E)$, $G(F)$, $Anti(E)$, $Anti(F)$ and identify $Int(E)$ and $Atgt(E)$.

3.2 If ($E_{\Phi} \neq E$) AND ($\|G(E)\| = \Phi$) **then** {

3.2.1 All points on $bd(P_{out})$ in the intervals $[E,F]$ and $[Anti(E),F_{\Phi-1}]$ are labelled with a Polygon Index = Φ.

3.2.2 Determine whether Case A or B is applicable and compute the outer and the inner envelopes of the region swept out by slack cone. All points in this region, including the ones on the envelopes are labelled with a Polygon Index = Φ. } **end if**

3.3 If ($E_{\Phi} = E$), (i.e. E is a tight point) **then** {

3.3.1 Retrieve the function $\delta E_{\Phi}(d)$. By examining its first two derivatives at $d = 0$, determine whether, for a point inside the interval, the Turn Index is Φ or $\Phi+1$.

3.3.2 If (Turn Index in (E,F) is $\Phi+1$) **then** {
All points in the line segment $[Anti(E),E]$ are given a Polygon Index = Φ. } **endif**

3.3.3 If the Turn Index in the interval (E,F) is Φ **then** {
All points on $bd(P_{out})$ in intervals $[E,F]$ and $[Anti(E),F_{\Phi-1}]$ are labelled with a Polygon Index = Φ.
Determine whether Case A or B is applicable and compute the outer and the inner envelopes of the region swept out by slack cone. All points in this region, including the ones on the envelopes are given a Polygon Index = Φ. } **end if** }
end if } **end for** ◇

As before, it can again be argued that this algorithm takes $O(n\Phi)$ time and produces as many regions in the annulus with the property that any two points within a region have the same Polygon Index. Note however, that in this case, these regions may not be disjoint, as it is possible that the same point may be within the slack cone of several points on $bd(P_{out})$. For any point $v \in [P_{out}\text{-}P_{in}]$, there exists a Minimal Nested Polygon passing through v iff $\|P(v)\| = \Phi$ and hence this scheme identifies all points in the annulus that can be assumed to lie on some Minimal Nested Polygon for P_{out} and P_{in}.

Recovery of Optimal Solutions: Finally, we address this issue by answering the following question: once the two algorithms are over, given a point $x \in [P_{out}\text{-}P_{in}]$, how do we determine if there exists a Minimal Nested Polygon passing through x, and if so, construct it. To do this it should first be checked to see whether x lies in any of the different regions produced by *Algorithm Partition-* $\|T(v)\|$ whose label is Φ - this can be performed in $O(n\Phi)$ time for the entire annulus. If x does belong to one such region, then by drawing $G(x)$, we can get the required P*. If not, it remains to be checked if there is a P* with an edge passing through x. To verify that, we then check if x belongs to any one of the $O(n\Phi)$ regions produced by *Algorithm -* $\|P(v)\|$. Suppose we find that x belongs to the region swept out by the slack cone in the interval $[E,F]$, as shown in figure 3. Then choose any point in $[E,F]$ that is visible to x - say y as shown in the figure. By drawing the $G(y)$ we can find a P* that passes through x. If x does not belong to the any of the $O(n\Phi)$ different regions produced by *Algorithm-* $\|P(v)\|$, then there is no P* passing through x. Just as before, this can also be shown to take $O(n\Phi)$ time.

As this is a first pass at the problem, we have not attempted to investigate the possibility of using advanced data structures to improve the time and space complexity of the two algorithms and the recovery procedures - future work may address this issue. Another strand of future research may be to investigate extensions of the algorithms to non-convex polygons.

Acknowledgement. The authors gratefully acknowledge the financial support provided by NSERC and the Morris Hite Center at UT-Dallas.

References

[1] A. Aggarwal, H. Booth, J. O'Rourke, S. Suri and C.K. Yap, "Finding Minimal Convex Nested Polygons", *Information and Computation*, Vol. 83, No. 1, (Oct. 1989), 98-110

[2] V. Chandru, S.K. Ghosh, A. Maheshwari, V.T. Rajan, S.Saluja, "*NC*-Algorithms for Minimum Link Path and Related Problems", *Journal of Algorithms*, Vol. 19, No. 2, (1995),173-205.

[3] S. K. Ghosh and A. Maheshwari, "Optimal Algorithm For Computing Minimal Nested Non-Convex Polygon", *Information Processing Letters*, 36 (1990), 277-280.

[4] S. K. Ghosh, "Computing The Visibility Polygon From A Convex Set and Related Problems", *Journal of Algorithms*, 12 (1991) 75-95.

[5] S. Suri and J. O'Rourke, "Finding Minimal Nested Polygons", *Tech. Report*, The John Hopkins University, (1985).

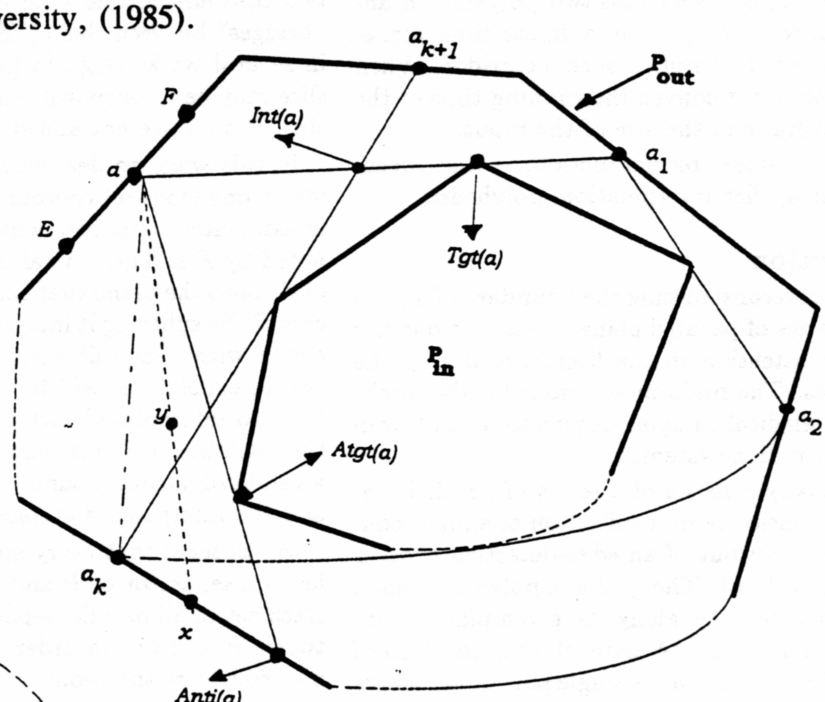

Figure 1

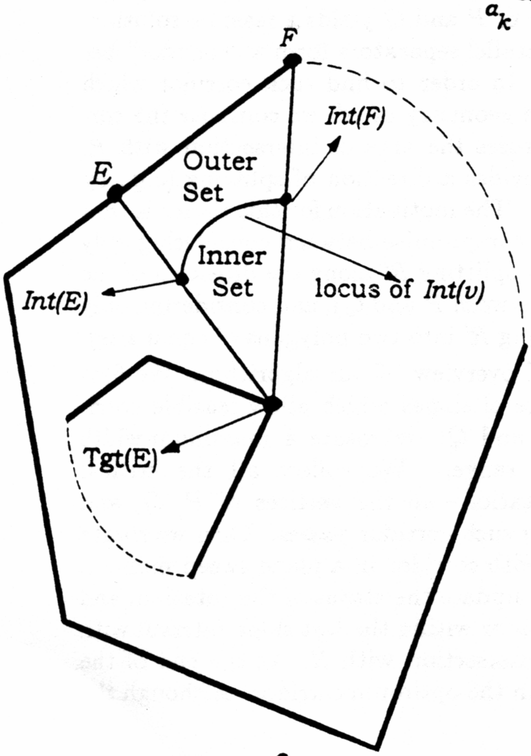

Figure 2

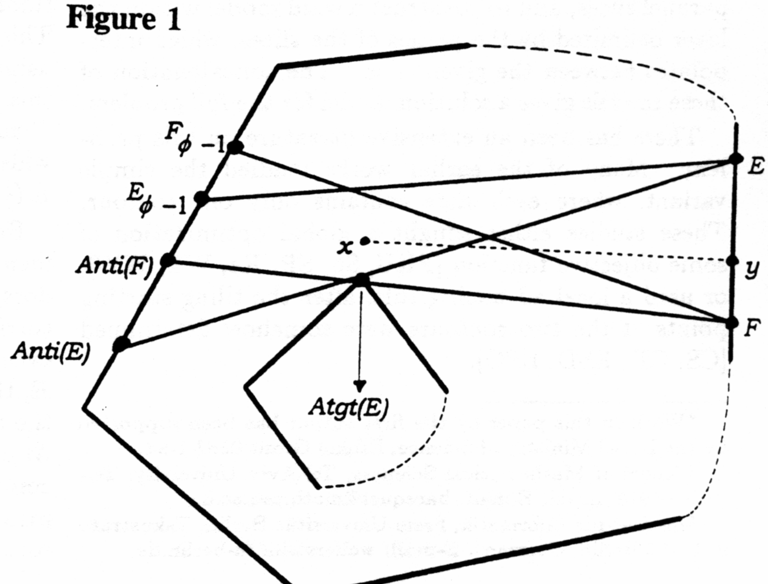

Figure 3