Generation of Partial Medial Axis for Disassembly Motion Planning

by

Yangsheng Xu, Raju Mattikalli, Pradeep Khosla

EDRC 24-47-91

GENERATION OF PARTIAL MEDIAL AXIS FOR DISASSEMBLY MOTION PLANNING[1]

Yang.heng Xu, Raju Mattikalli, Pradeep Khosla

**Engineering Design Research Center
and the Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213**

ABSTRACT

This paper presents an efficient approach for determining disassembly motion plans of a subassembly in the free space within its parent subassembly. We propose a two-step approach to generating motion plans. First, *all possible paths* within free space of the parent subassembly are generated using a partial medial axis. This is followed by a graph search for an optimal *global path*. Second, a *collision free motion* is planned using the global path and the *geometry of the moving subassembly*. In this paper, we focus on the first problem, i.e., generation of a partial medial axis for disassembly motion planning. We present a method to determine a partial medial axis (PMA) by generating Critical Points (CPs) and connecting them based on geometry of the parent subassembly. The PMA gives us all possible paths to disassemble subassembly out of its parent subassembly. The connectivity between segments of the PMA is represented in the form of a graph which is augmented with geometric information from the PMA. An optimal path is found by using graph search techniques on the augmented graph. A method called the *two-disks motion planning strategy* is proposed to plan the disassembly motion along the generated path. This method is discussed in detail in our companion paper [19].

[1]*This* **research** was **supported in part** by **the** Engineering **Design Research Center, an NSF Engineering Research** Center at Carnegie Mellon University

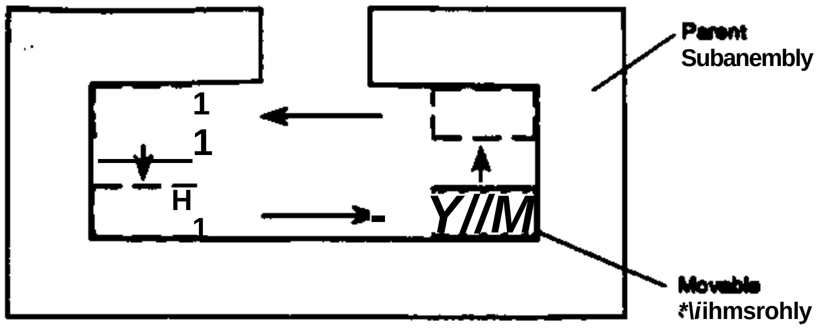


Figure 1 If there is a free space within the parent subassembly, the disassembly motion of the movable subassembly is iterative back and forth within the free space

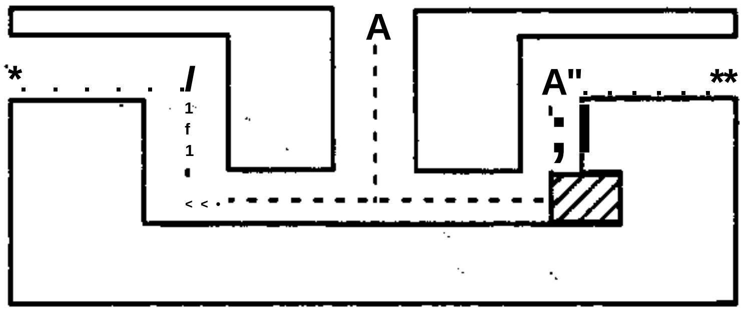


Figure 2 If there are multiple paths available for disassembly motion, an optimal path must be determined based on the degree of difficulties in each path

1 Introduction

Automatic generation of assembly motion for a Mechanical System/Assembly (MSA) is important **for assemblability evaluation** of a design at the design stage, and for automation of assembly **process [3, 9, 17]. Integrating** the design and assembly phases of the manufacturing cycle will lead to **a reduction of the** leading time and cost of products. Most approaches to automatic assembly **planning follow the** idea that assembly motion can be determined by reversing disassembly process **without consideration** of deformable parts during assembly [5, 12, 6, 13, 18, 11]. To disassemble a mechanical assembly, various approaches based on computer graphics [9, 3], geometric modeling [10], or knowledge-based [1, 16] have been proposed. The approach proposed in our early papers is based on the detailed geometry of parts and the topology of their connectivity. Using a geometric model of a MSA represented by the geometric modeler *Noodles* [4], we have developed a method, based on kinematic constraints, to automatically generate an assembly plan for a given 3-D MSA [10, 7, 11].

In generating a disassembly plan, the motion of a subassembly from its assembled position within its parent subassembly to a position outside must be determined. However, most existing methods cannot generate a disassembly motion in two cases: first, when there is a free space within the parent subassembly; and second, when there are multiple possible paths available for the disassembly motion. In the former case disassembly motion of the moving subassembly in the free space can be iterative, i.e., the moving subassembly would move back and forth within the parent subassembly and never come out, as shown in Figure 1. In the latter case depicted in Figure 2, the degree of difficulties for each path must be evaluated for determining an optimal path among all possible paths.

This paper presents an efficient approach for determining disassembly motion plans of a subassembly in the free space within its parent subassembly. We propose a two-step approach to generating motion plans. In the first step, *all possible paths* within free space of the parent subassembly are generated using a *Partial Medial Axis.* This is followed by a graph search for an optimal *global path.* In the second step, a *valid motion* is planned using the global path and the *geometry of the moving subassembly.* In this paper, we will focus on the first step. The second step is discussed in our companion paper [19].

We introduce terminology which will be used throughout this paper. Let an MSA be denoted by *A*, the moving subassembly be denoted by S_m and the stationary subassembly by S_s (S_s = *A* - S_m). The stationary subassembly is also called a parent subassembly of a moving subassembly. To simplify the problem, the parent subassembly and moving subassembly are considered to be polygons. We employ the *medial axis* (MA) transformation to represent the free space within the parent subassembly that is available to the moving subassembly for its motion out of the parent subassembly. The medial axis is the locus of points which are equidistant from two or more obstacle boundaries [8, 15]. A medial axis is a good abstraction of free space, since it reduces the dimensionality It can be used to easily generate alternative paths within the free space. However, computation of the complete MA for a realistic mechanical assembly is costly and unnecessary, and a partial MA is sufficient to generate a global path for disassembly motion.

Therefore, we first present a method to determine a partial medial axis (PMA) for a given subassembly that allows us to select a globally valid path for disassembly. The procedure for determining a PMA consists of two steps: (a) special points, called *Critical Points* (CPs) on the medial axis are identified, and (b) these points are connected using adjacency of CP's and the type of connecting segment between them. In this way, the PMA of the free space within the parent subassembly is generated. Consider, for example, the assembly in Figure 3 which shows a stationary subassembly and the PMA of the free space within it.

The generated PMA represents all possible paths for motion within the parent subassembly, i.e., stationary subassembly. The connectivity between segments of the PMA is represented in the form of a graph. This graph is augmented with geometric information from the PMA, and nodes corresponding to the initial and final positions are marked. An optimal path found using graph search techniques on the augmented graph gives us the required *global path* for disassembly. In this way, the moving subassembly can be disassembled from the stationary subassembly in an optimal path, and iterative motion within the stationary subassembly is avoided. Taking the same example of Figure 3, the global optimal path is determined by abstracting the PMA a* a path graph and then searching for the best path. This is depicted in Figure 4.

Based on this path information, we utilize a strategy called *two-disk motion strategy* to navigate the moving subassembly out of the stationary subassembly. The disassembly motion can be planned by creating two minimal overlapping disks that completely cover the given polygon, i.e., moving subassembly, and constraining the centers of the two disks to follow continuously the given path on the PMA. The success of the proposed strategy is guaranteed, as long as the smallest

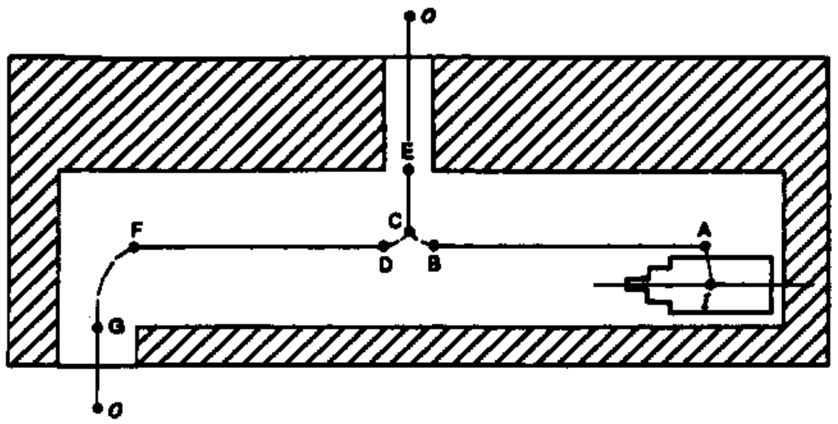


Figure 3 Generating of all possible path by the PMA which is determined by identifying the CPs within free space of the stationary subassembly and connecting the CPs in a special way

radius of the PMA along the path is no less than the radius of two circumscribing disks. The analysis and algorithms for generating the two minimal circumscribing disks for an arbitrary polygon are discussed in detail in a companion paper [19]. Using the same example as Figures 3 and 4, disassembly motion is planned as illustrated in Figure 5.

Although our approach for path planning using the reduced dimensionality provided by MA can be applied to £D objects in general, in this paper, we present a treatment for 2-D polygonal objects. The proposed method of obtaining the PMA using critical points is applicable for polygonal object only. It must also be noted that this method can also be used to generate the complete MA for a given polygonal region.

The rest of paper is organized as follows. In section 2, the critical points (CPs) on MA are classified. In section 3, an algorithm to generate the critical points is presented. In section 4, the generated CPs are connected using adjacent pair of CPs and the type of connecting segment, resulting in the PMA path. A system to generate the PMA for an arbitrary polygon is developed, and various cases are studied. Based on the PMA, a graph approach for determining the optimal path is discussed in section 5. A strategy of disassembly motion planning is briefly discussed in section 6. Finally we summarize this paper in section 7.

2 Definition of Critical Points

In order to obtain a partial medial axis within the free space of a stationary subassembly , we identify *critical points* and the connect then. In what follows, we define critical points for space that can be represented by 2-D polygons. An assembly shown in Figure 6(a) can be abstracted to a stationary subassembly S_s and a moving subassembly S_s in Figure 6(b). Some terminology may be introduced as follows. The sides of the polygon are referred to as *edges*. Tow adjacent edges meet at a *corner*. Corners are of two kinds, based on the outer angle between their adjacent edges: *concave* corners are those with an obtuse outer angle, and *convex* corners are those with an acute outer angle. We call the edges and corners *boundary elements*. For the polygon in Figure 6, the corners *Cu* ^2, C_3, C$_5$> C_6 are concave corners, and C_4 is a convex corner. Some edges are open to air, but for convenience, we treat these edges the same as others.

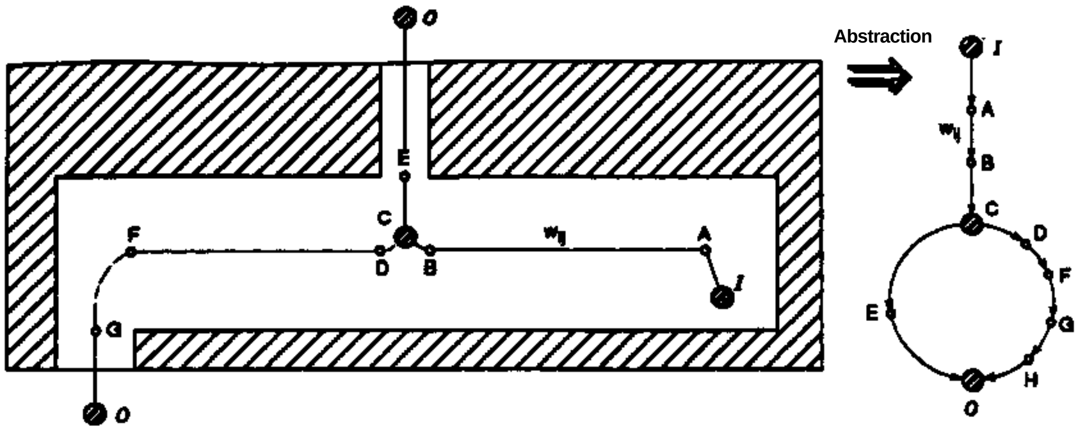


Figure 4 Determination of a global optimal path by generating a path graph abstracted from the PMA and searching for the best path

We first consider what a medial axis looks like for a given polygon, e.g., that shown in Figure 7. It has been known that the medial axis is the locus of points equidistant from at least two edges of the polygon, and moreover, is formed by either a straight-line segment or parabolic-arc segment for a polygon. Therefore, if we can identify the type of segments and the points where these segments meet, the medial axis can be determined.

A *Critical Point* (CP) is defined as a point on the MA at which the MA segments meet, i.e., a point where a straight-line segment (or a parabolic arc segment) meets another segment, or the MA bifurcates into two or more segments. If these two segments are not parallel, a transition may occurs on the point of CP, i.e., the MA may changes its direction or form on the CP.

Each point of the MA is generated from at least two boundary elements. There are two kinds of boundary elements, edge and corner. The MA segments that lie outside the polygon or join from the concave corners are generated from the concave corners. If we exclude these segments, (as we will see this is the difference between the PMA and complete MA), the boundary elements that we are interested in are *edges* and *convex corners.* They are abstracted as *lines* and *points* throughout the rest of paper.

Therefore, the MA segment that we are interested in is determined by two boundary elements, line or point. Possible combinations of the line and point are *(line to line), (line to point),* and *(point to point).* The (line, line) and (point, point) combinations generate a straight-line segment, and a (line, point) combination generates a parabolic arc segment. For a single segment, the corresponding boundary elements do not change. When two or more segments intersect, the CP occurs. Since this point of CP belongs to both segments, it must have the boundary elements of both segments. Therefore, the point of CP is of at least *three* boundary elements, because each segment has two boundary elements.

Let us look for points that are equidistant from three or more boundary elements. It must be noted that when we attempt to generate a point that is equidistant from three or more boundary elements, we must ensure that there is no other boundary element that is closer to the point. If any other boundary elements were closer, then this points would not be a point on the MA. Possible combinations for three boundary elements a*e (line, line, line), (line, line, point), (line, point, point), and (point, point, point). The following paragraphs describe methods used to generate

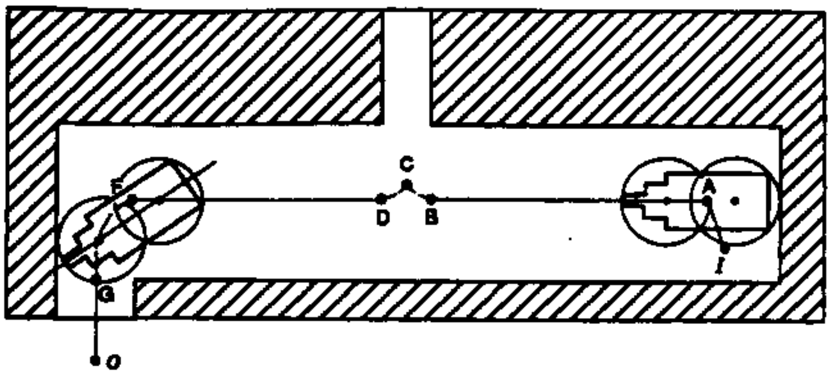


Figure 5 Motion planning on the global path by determining two minimal overlapping disks to cover the given movable subassembly and constraining two centers of disks to follow the path

points that axe equidistant from three boundary elements of the four form as above.

We consider the case (line, line, line) at first. As illustrated in Figure 8, the three edges (solid line in Figure 8) can be extended (dashed line) to form three *lines.* It is known that for three lines that are not all parallel, the point of intersection of two angular bisectors is equidistant from the three lines. Thus if we find such a point and ensure that no other boundary elements lie closer to it than the three lines, we can be sure that it is a CP. The distance from the point to closest boundary elements is the radius function of the medial axis, (see Figure 8).

The condition to ensure no other boundary elements are closer to the potential CP than the three lines, i.e., no boundary elements are within the circle with the potential CP as center and the distance from the point to closest boundary elements as radius, is referred to as *boundary condition.* The boundary condition will be examined for the rest of cases. For simplicity, we will give the radius *R* for each case, and will assume this condition has been checked.

The second case (line, line, point) is shown in Figure 9. If there exists a point on the angular bisector such that it is equidistant to the line and to the point, the point is a CP (shown in Figure 9). The radius function 72 is the distance between the point and corresponding edges, or the convex corner.

As a special case of the case (line, line, point), the given point may lie on one of the given lines, see Figure 10. In this case, the point CP that is equidistant to the line and to the point can be obtained directly by drawing a perpendicular line from the given point, and the intersection point of the angular bisector and the perpendicular line will be a CP.

The third case (line, point, point) is shown in Figure 11. A point on the perpendicular bisector of the line joining the two points that is equidistant from the line and the point is a CP. The radius function *Tl* is the distance between the point and the line or the two points.

The forth case (point, point, point) is shown in Figure 12. For the given three points, the intersection point of the perpendicular bisectors of segments connecting the three points is a CP. *TZ* is the distance between the point of intersecting and the given points.

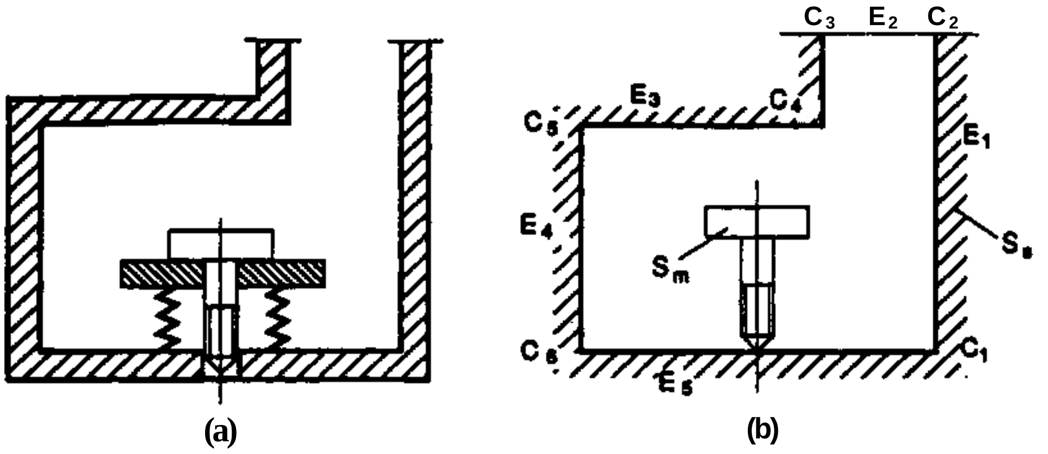


Figure 6(a) An example of mechanical assemble
(b) Definition of corrners and edges

3 Generation of Critical Points

In the previous section we defined the critical point and classified it into 4 cases. In this section, we will discuss the computational complexity in generating all critical points (CPs).

We notice that all possible CPs can be determined from the defined 4 cases, i.e., (line, line, line), (line, line, point), (line, point, point), (point, point, point), where the *line* is abstracted from edge, and *point* is abstracted from convex corner. For a polygon with n corners (edges) in which *m* corners are convex, the number of combination of case (line, line, line) is $\binom{n}{3}$. By the same reason, the number of combinations is $m\binom{n}{2}$ for the case (line, line, point), and $n\binom{m}{2}$ for the case (line, point, point), and $\binom{m}{3}$ for the case (point, point, point). Thus, totally we must examine N cases,

$$N = \binom{n}{3} + m\binom{n}{2} + n\binom{m}{2} + \binom{m}{3}$$

where,

$$\binom{n}{a} = \frac{n!}{(n-a)!a!}$$

For a polygon with n edges and m convex corners, the total number of combinations of 3 boundary elements is $\binom{n+m}{3}$. If P_1 is the computation time required to determine a critical point, and P_2 is the computation time to the distance of a point from an edge, the total computation time would be $(P_1 + P_2 n)\binom{n+m}{3}$. Therefore, roughly speaking, the computation is of $O(n^4)$.

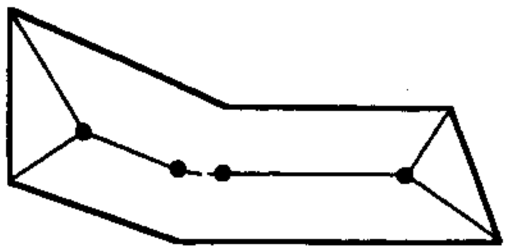


Figure 7 The medial axis of a polygon

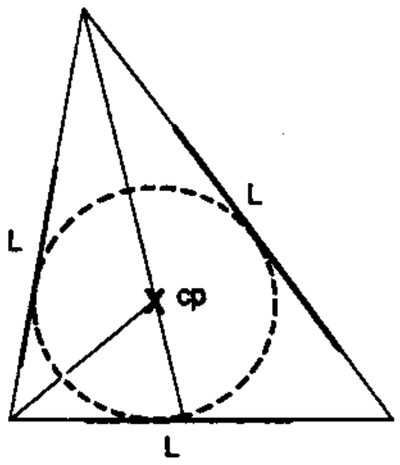


Figure 8 Critical Point: the case (line, line, line)

The computations can be reduced if the relative location of the edge segments and the convex **corners is examined,** before computing the CP. In this way, we may omit the test for the cases that are **impossible for the CP.**

To **analyze the case** (line, line, line), consider three arbitrarily lines as shown in Figure 13. The lines **divides the** plane into 7 regions. Let us examine possibility of the existence of a CP in each region. It is evident, CPs do not exist in the regions marked "x". This is because that the circle that produces the CP must make contact three lines, while any circle within these regions cannot make contact all third line. Therefore, there are only four feasible regions, i.e., (I), (II), (III), (IV), that could contain the CP.

Moreover, for each of these four regions, whether a CP exists depends upon the relative location of the edge *segment* with respect to the line. For example, for the region (I), Figure 13 shows one (and the only one) feasible case where three edge segments lie towards the region (I), i.e., in the configuration shown in Figure 13, because only in this case, a circle exists such that it makes contact with *three* edge segments. Generally speaking, for these four regions, there are only four

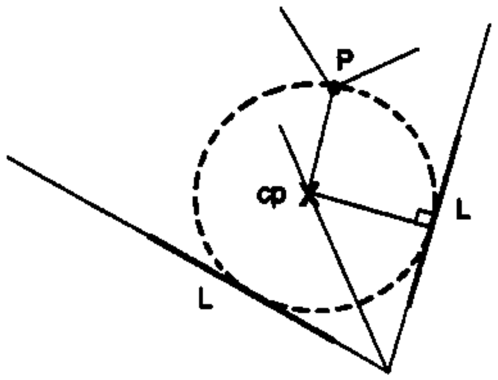


Figure 9 Critical Point: the case (line, line, point)

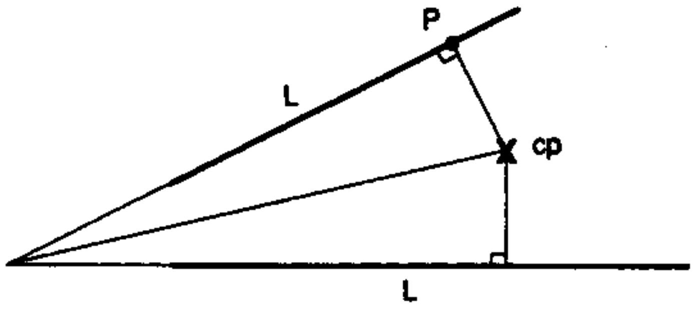


Figure 10 A special case of the case (line, line, point) when the point lies on one of two lines

feasible **cases that a potential CP** exists according to the relative location of the edge segment with **respect to the corresponding line.**

Therefore, we examine the relative location of the segments with respect to the lines for each **set (line, line, line). If** it is **one** of **these** four feasible cases, i.e., three edges are on the same sides of **one region,** then a CP may occur. For any other case, there is no possibility of generating a CP, and thus no further reasoning is needed.

In the same manner, we examine the case (line, line, point). Possible locations of a point with respect to two lines is shown in Figure 14, marked as points A, B, C, D, and E. When the point lies outside the acute angled sector formed by the two lines, marked as A and B, ther is no possibility of generating the CP, because it is impossible to create a corresponding circle touched to both lines and the point. Therefore, a potential CP may only exist when the point lies within the acute angled sector, marked as C, or on the lines, marked as D and E.

For the case (line, point, point), locations of the two points with respect to the line is shown in Figure 15. If the two points lie on different sides of the line, marked as points A and B, no circle

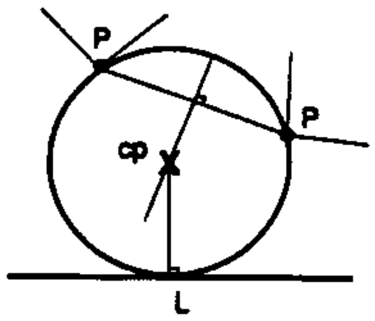


Figure 11 Critical Point: the case (line, point, point)

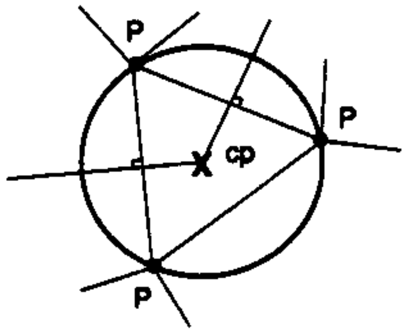


Figure 12 Critical Point: the case (point, point, point)

can **be created with contact to the line and** *both* points. The CP can potentially be generated when the **point are on the same** side of the line, marked as B and C.

For the **case (point,** point, point), if these three points are collinear, we omit the set, because a circle **cannot be made to pass** the three collinear points. All non-collinear (point, point, point) sets have to be tested, to discuss it.

From the discussion above it is easy to find a way of identifying the CPs for a given polygon. The procedure to compute the CPs for a polygon can be summaried as follows.

- Case (line, line, line);
  - Group all edges as (line, line, line).
  - Extend segments as lines for each group.
  - Examine the relative location of the segments.
    If it is one of four cases that may create a CP, go to next step.
    If not, go to the second step.

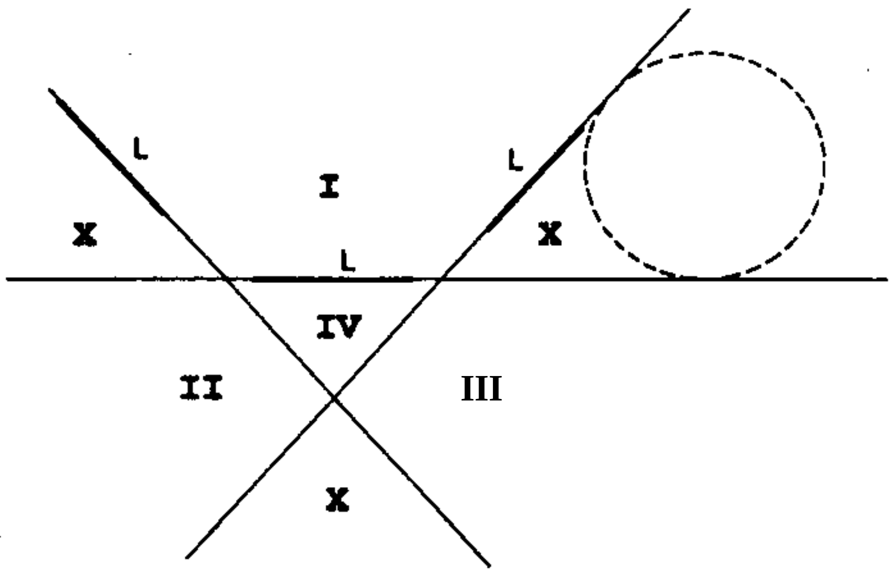


Figure 13 In the case (line, line, line), possible relative location of the edge segments with respect to the lines

- Identify the possible CP by drawing angular bisectors and determining the intersection points.
- Check the boundary condition, i.e., if a disk with this point as center and the distance between the point and lines as radius, is within the polygon.
  If not, go to the second step.
- The point is a CP and record it.

• Case (line, line, point);

- Group all edges and convex corner as set (line, line, point).
- Extend segments as lines for each set.
- Examine the relative location of the point.
  If it is one of two cases that a CP may exist, go to next step.
  If not, go to the second step.
- Identify the possible CP by drawing angular bisector and determining a point that is equidistant to the lines and the point.
- Check the boundary condition. It is valid, go to the next step.
  If not, go to the second step.
- The point is a CP and record it.

• Case (line, point, point);

- Group all edges and convex corners as set (line, point, point).
- Extend segments as lines for each set.

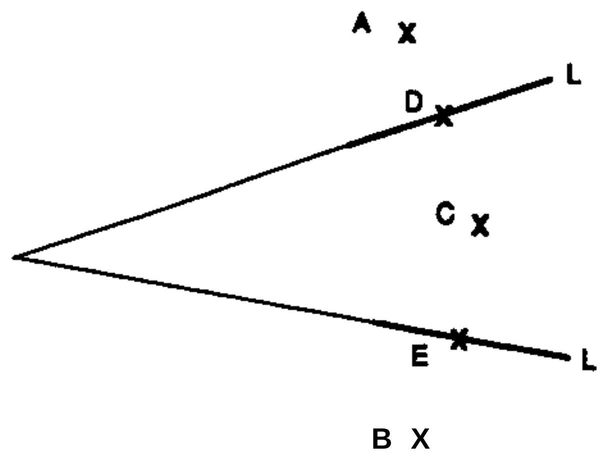


Figure 14 In the case (line, line, point), possible relative location of the points with respect to the lines

- Examine the relative location of the two points. If it is the case that a CP may exist, go to the next step.
  If not, go to the second step.
- Identify the possible CP by drawing connecting line between two points and its perpendicular bisectors and determining a point that is equidistant to the given points and the line,
- Check if the boundary condition is valid.
  If not, go to the second step.
- The point is a CP and record it.

• Case (point, point, point);

- Group all convex corner as set (point, point, point).
- Identify the possible CP by connecting three points and drawing their perpendicular bisectors, and determining the intersection points.
- Check if **the** boundary condition is valid.
  If not, stop.
- The point is a CP and record it.

4 Connecting the CPs to Form the PMA

After all possible critical points (CPs) within the polygon S_9 have been identified, they must be connected by line segments which are part of the medial axis. Once all such segments are generated, we obtain the desired partial medial axis. This problem of generating the PMA from CPs consists of (1) determining pairs of CPs that need to be connected, and (2) for each CP pair, determining the form of the line segment that connects them.

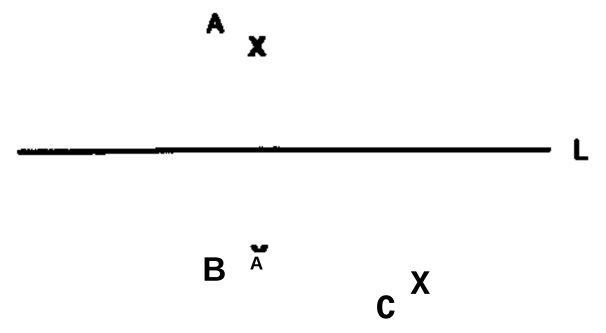


Figure 15 In the case (line, point, point), possible relative locations of the two points with respect to the lines

Let us consider the first problem, i.e., identifying pair of CPs that must be connected. As we know from the previous section, each CP is formed from three boundary elements of type, (line, line, line), or (line, line, point), or (line, point, point), or (point, point, point). It is also noted that the line segments of connecting these CPs are part of medial axis which is produced by at least *two* boundary elements. If two CPs must be connected, then the connecting line segment must have common boundary elements with both end points, i.e., a pair of the CPs. This implies that if two CPs have two common boundary elements, then there is guaranteed to be a segment between them. The expected rational connection of the CP can be shown in Figure 16. Thus, we may claim that *for any pair of CPs, if there exist at least two common boundary elements, this pair of CPs should be connected.*

The next problem is to determine the form of the line segment that connects the pair of CPs. It has been known that the medial axis is formed either by straight-line segments, or by parabolic arc segments. For the polygon case, the MA separates the loci of proximity of the boundary elements of the type (point, point), (line, line), and (point, line). Only a case (point, line) gives rise to arcs of parabola [14]. This means parabolic arcs can be produced only at critical points determined by the case (line, line, point) or (line, point, point). The case (line, line, line) or (point, point, point) would only result in a straight-line. A straight-line is determined by connecting two points on the medial axis, i.e., CPs, while a parabolic arc is determined by a line and a point belonging to the boundary elements. As stated in the procedure to generate CPs for the case (line, line, point) and (line, point, point), the linear equation of the straight-line and the coordinates of the point are known, thus the corresponding parabolic curve can be determined.

In summary, connection can be performed in the following steps.

Step 1: For each pair of critical points, check the associated boundary elements. If the result contains two or more elements, this pair is an adjacent pair, and must be connected.

Step 2: For each adjacent pair of critical points that are determined by the case (line, line, point) and case (line, point, point), use the line equation and the point coordinates, to create a parabolic arc connecting these two CPs.

Step 3: For each adjacent pair of critical points that are generated in cases (line, line, line) and (point, point, point), connect them by a straight-line.

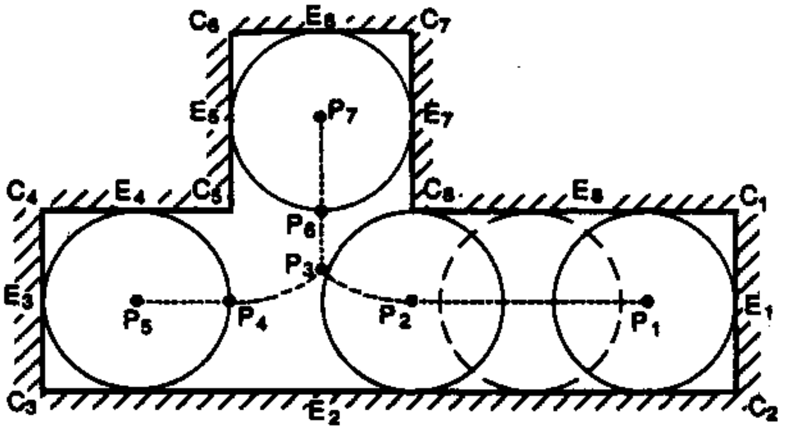


Figure 16 Illustration of connecting CPs

For the polygon shown in Figure 16, the CPs and their boundary elements can be listed as a table as follows.

CP	*Corresponding Case*	*Boundary Elements*
Px	(line, line, line)	*Ex, E-i, Es*
Pi	(line, line, point)	*Ei , Es, Cs*
Pz	(line, point, point)	*Ei , Cs, Cs*
PA	(line, line, point)	*Ei , E4, Cs*
Ps	(line, line, line)	*Ei , E3, E4*
Pe	(line, point, point)	*Es* (or *E*$_7$*)*, *C*$_5$, *Cs*
Pt	(line, line, line)	*Es, E$, Ef*

The generated PMA for the polygon is shown by a dashed line in Figure 16. Comparing the PMA in Figure 16 with the complete MA, we may find the the following two differences between them. First, the PMA is a part of the MA *within* the polygon, since we are not interested in the MA *outside* of it, while the complete MA includes the segments outside. Second, the PMA does not include the segment from the concave corner to the PMA. This is due to the fact that usually the initial position of the moving subassembly is not far away the closest CP and the segment from the concave corner to the closest CP is not needed. If needed, we may simply connect each concave corner to the closest CP by a straight-line to form a complete MA *within* the polygon. The proposed method is also feasible for generating the complete MA if we consider the above two cases. For our purpose of motion planning, this is unnecessary. Therefore, the generated PMA is actually a partial MA, excluding the meaningless line segments.

The last problem is to determine the minimum value of the radius function along each segment of the PMA. Each segment is generated from two boundary elements, edge(s) or convex corner(s). There is three possible combinations of edge corner pair, (edge, edge), (edge, corner), (corner, corner). They are shown in Figure 17 along with the generated PMA segment in each case.

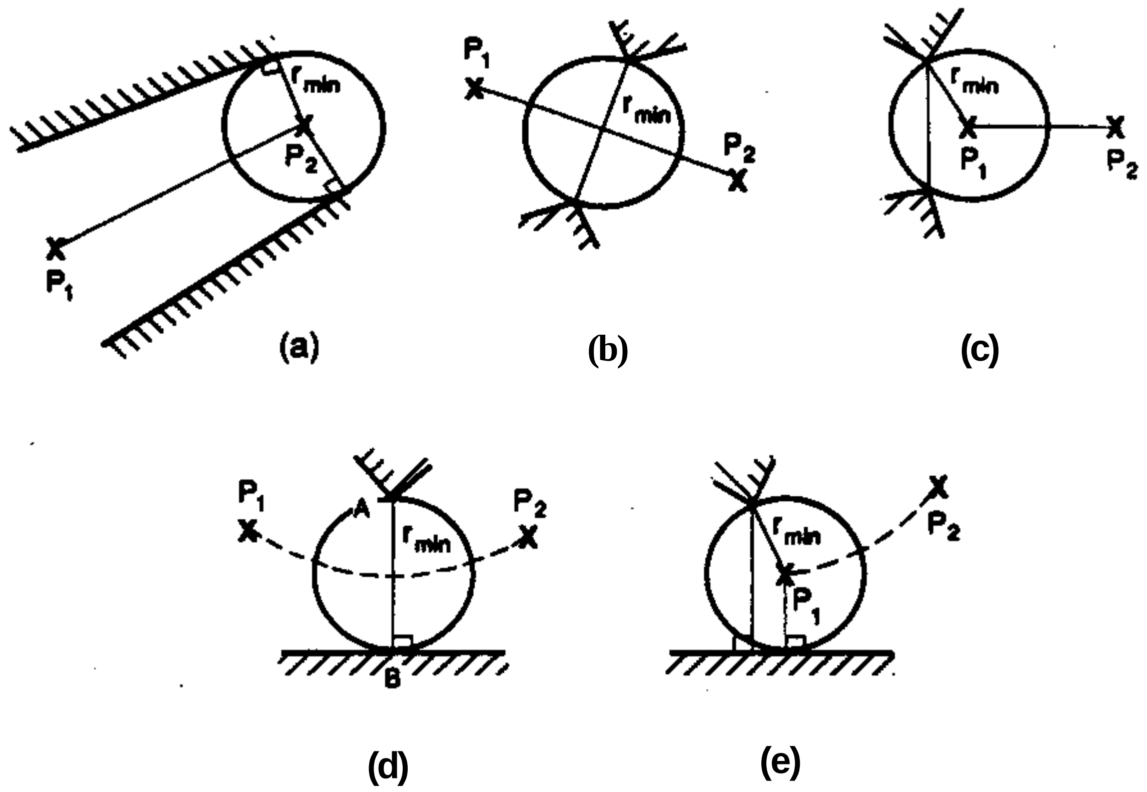


Figure 17 The minimum radius of the medial axis segment

Figure 17(a) shows the case (edge, edge); the corresponding medial axis is a straight-line. The radius function for this segment is monotonic; either increasing or decreasing from Pi to P_2« Thus the minimum radius along this segment is the smallest of the two values, i.e., r_{mtn} = min(iZ(Pi),i2(i$^>$2))- Figure 17(b) shows the case (corner, corner); the corresponding medial axis is also a straight-line. If the two CPs are on opposite sides of the line connecting the two corners, the minimum radius is half the length of the segment connecting the two corners. If the two CPs are on the same side of the connecting line, as shown in Figure 17 (c), the minimum radius is min(iZ(Pi),iZ(P2)). Figure 17(d) shows the case (edge, corner); the medial axis is a parabolic arc. Again when the two CPs are on the opposite sides of the perpendicular segment from the edge passing through the corner, the minimum radius is half the length of this segment. When the two CPs are on the same side of the line as illustrated in Figure 17 (e), r_{mtn} = *min(R(Px),R(P2))*.

The proposed algorithm to generate the PMA of an arbitrary polygon has been implemented, and some test cases are shown in Figure 18(a), (b), (c), (d), (e), (f). Consider the polygon in Figure 18(a) as an example, the computed results are listed below.

CP#	Coordinates	r_{min}
0	(0.500000,0.500000)	0.500000
1	(2.500000,0.500000)	0.500000
2	(3.750000,1.750000)	0.250000
3	(0.500000,1.000000)	0.500000
4	(1.000000,1.500000)	0.500000
5	(2.000000,1.500000)	0.500000
6	(2.500000,1.000000)	0.500000
7	(2.414214,1.414214)	0.585786
8	(2.500000,1.500000)	0.500000
9	(3.000000,1.750000)	0.250000
10	(0.585786,1.414214)	0.585786

The output of the segment equation and the minimum value of the radius function is listed below. The first column is segment equation, the second column is the two end critical points, and the third one is the minimal radius.

Segment equation	CP # on two ends	r_{min}
$0.00\,x^2 + 0.00\,y^2 + 4.00\,x + 0.00\,y + -2.00 = 0$	(0,3)	0.50
$0.00\,x^2 + 0.00\,y^2 + -3.00\,x + 0.00\,y + 7.50 = 0$	(1,6)	0.50
$0.00\,x^2 + 0.00\,y^2 + 0.00\,x + -8.00\,y + 14.00 = 0$	(2,9)	0.25
$-0.00\,x^2 + -4.00\,y^2 + 8.00\,x + 8.00\,y + -8.00 = 0$	(3,10)	0.50
$0.00\,x^2 + 0.00\,y^2 + 0.00\,x + -8.00\,y + 12.00 = 0$	(4,5)	0.50
$-16.00\,x^7 + -0.00\,y^2 + 32.00\,x + -32.00\,y + 32.00 = 0$	(4,10)	0.50
$-16.00\,x^2 + -0.00\,y^2 + 64.00\,x + -32.00\,y + -16.00 = 0$	(5,7)	0.50
$-0.00\,x^2 + -2.25\,y^2 + -4.50\,x + 4.50\,y + 9.00 = 0$	(6,7)	0.50
$0.00\,x^2 + 0.00\,y^2 + 6.00\,x + -6.00\,y + -6.00 = 0$	(7,8)	0.50
$-16.00\,x^2 + -0.00\,y^2 + 96.00\,x + -16.00\,y$ -1- $-116.00 = 0$	(8,9)	0.25

5 Finding an Optimal Path

The PMA of the free space within S_9 consists of critical points connected by line segments. A possible disassembly path is a connected set of critical points and line segments, starting from one that is closest to the initial position of the moving subassembly ($<S_m$) to **one that** lies **on the** outside **of the stationary subassembly (5«). In this section, we present a graph based approach to** find **an optimal path for disassembly.** This consists **of treating** the **PMA as a** weighted **graph and** using **graph search techniques to find an optimal disassembly path.**

5.1 Generating the Path Graph from the PMA

The medial axis contains spatial information in a simplified form. Extent information in one of the **dimensions (radius** function) is separated from that in an orthogonal direction (the medial axis). A reduction in dimensionality is achieved by considering the medial axis and the associated radius function in succession rather than concurrently. In particular, to plan the motion of the moving subassembly S_m within S_{SJ} the medial axis is first used to determine a path after which the radius **function is used to plan** detailed motions.

To **determine a *global* path,** the connectivity between the various elements of the medial axis is **abstracted and represented** in the form of a graph, referred to as the *path graph.* By determining the ***best* path in this graph between** the initial and final nodes, an optimal disassembly path of S_m within *S,* **is** obtained.

The path graph is **generated** from the **PMA** in the following way. Each critical point is represented as **a** node, and each line segment as an edge in a graph. Additional nodes which represent the initial and final positions of S_m on the PMA **are** also created. The additional nodes will be referred to as N_i and *Nf* respectively.

Each line segment in the PMA is a portion of a disassembly path for the moving subassembly. A degree of difficulty *(dod)* is associated with each segment. This numeric value represents the difficulty that is encountered by the assembly facilities in moving S_m along that segment. Similarly a *dod* can be associated with each critical point. Criteria that are used to determine the dod include length of the segment, associated radius of the PMA, manipulation difficulty, and regrasping. Weight parameters are associated with each node and edge in the weighted path graph which correspond to the *dod* values. Finding the best path for disassembly is formulated as a *search* for the 'best' path from N_i to *Nf* in the path graph. The following section describes the search strategy that is employed.

5.2 Searching for the Best Path between Two Nodes in the Path Graph

5.2.1 Problem Statement

Given:

1. A **graph** $Q = (N,E)$; N s critical points in the PMA, E s connections between critical points, with each N and E having a weight parameter,
2. An initial node N_i, and
3. A final node N_f,

***To Find*:**
the *best* path from N_i to N_f; *best* being defined as one that minimizes the sum of the weights associated with each of the *Ns* and *Es* that make up the path.

5.2.2 Proposed Solution

Consider a node N_g of the path graph Q. We are concerned with only those paths in Q that originate at N_i. A ***path*** (V_g) to N_g is an ordered list of alternating *Ns* and *Es* that are connected, with *Ni* as the start **node and** N_g as the terminating node. The *path length* C_g of path V_g is defined as the sum of the weights of all the *Ns* and *Es* that constitute V_g. The graph is traversed so as to store at each N a V (and C corresponding to V) such that V represents the *current* best path to that N. If all edges of the graph have been traversed, then we will be assured that the V associated with *Nf* will be the required solution. This technique is similar to the one described in [2].

To systematically traverse all *Ns,* we use a breadth-first search strategy [2]. Before the search procedure is started, the following assignments are made. Let the graph consist of N nodes.

$\forall i, i = 1,\ldots,N$

$$C_i \leftarrow 0.$$

During the traversal of the graph, on arriving at a node (say N_g), we compute V_g and C_g. U C_g that is computed is less than the one that was stored at that N, then we store the newly computed values of V and £ at JV. If not, we ignore the newly computed values and continue the graph traversal. To traverse the whole graph, each E needs to be traversed only once. To prevent revisiting an E due to cycles in the graph, each traversed E is marked. Thus, the whole procedure is order n, where n is the total number of E% in the graph.

The path that is generated as a result of the search does not take into consideration the shape of S_m; it only reflects the *best path* for a point from $N\{$ to Nf taking into account the shape of the free space within S_9. Its primary function is to provide a *global* motion plan, not local detail, i.e., it makes decisions at any point along a disassembly path where there are alternate routes available. This path is the result of the second of the three steps in our top-down approach to generating a disassembly motion plan. The next step aims at adding details of the shape information of S_m to the global path by prescribing a motion for the moving subassembly (S_m) along the global path.

6 Description of Two-disk Motion Planning Strategy

The global path consists of a connected set of critical points and line segments of the partial medial axis (PMA) starting from a point on the PMA close to the initial position of the moving subassembly and terminating at a point on the outside of the stationary subassembly. This section describes in brief a method to plan the motion using the generated global path by taking into account the geometry of the moving subassembly S_m. A detail description can be found in [19].

If we determine a *single* enclosing circle for <S_m, i.e., a disk that fully covers 5_m, and constrain the center of t£e circle to follow the global path, we would obtain a feasible motion plan provided that the radius of the disk is less than the smallest radius function of the PMA along the path. For objects whose aspect ratio, that is a ratio of the greatest distance between any two convex corners to the smallest one, is close to 1, fitting a single disk to circumscribe the object is an acceptable strategy. However, for objects whose aspect ratio is greater than 1, the single disk approach is certainly not an ideal solution since the enclosing disk is far too conservative an approximation of an object. Naturally, if we attempt to circumscribe multiple overlapping disks over the <S_m, the radii of these disks are smaller than the single circumscribing disk. Surely, if an infinite number of disks are permitted, they would cover the object *snugly,* thus representing the exact shape of the object.

However, if multiple overlapping disks are used, the strategy of allowing the centers of the disks to follow the MA cannot be used. This is because it is impossible, in general, to ensure that all centers lie on the given MA at all locations. No more than *two* points on a rigid object can simultaneously lie on a given 2-D arbitrary path. In other words, it is impossible to make more than two points match a given path, in general, due to over-constraints. Therefore, maximally only *two* enclosing disks can be used for a given arbitrary polygon such that the centers of the disks follow the given path. This is the *two-disk motion planning approach* to navigate the given polygon out of the stationary subassembly (see Figure 5). To ensure collision-free motion, the radii of the disks must be no greater than the minimum value of the radius function along the given path.

The problem of generating two circumscribing disks that cover a given arbitrary polygon is discussed in detail in our companion paper [19]. One of the important findings is that the two minimal disks that cover a given arbitrary polygon are of equal diameter. This approach to planing the motion of a polygon in a constrained environment is feasible not only for the disassembly motion planning, but also for a general mobile robot motion planning. When the collision-free motion along the optimal path using the two-disk approach cannot be achieved, suboptimal paths can be tested to generate a valid motion. If no such path can be found, the given polygon cannot be navigated out of the $S_§$ using the proposed method. In the following, we will give some simulation results. The first example is to determine the two circumscribing disks for a rectangular in Figure 19. Definition of the unknown variable set is as follows:

$$P = [r, O_1(x_1, y_1), O_2(x_2, y_2), X(x_3, y_3), Y(x_4, y_4)]^T \tag{1}$$

The computed results using the developed simulation system are

$$\begin{bmatrix} r \\ x_1 \\ y_1 \\ x_2 \\ y_2 \\ x_3 \\ y_3 \\ x_4 \\ y_4 \end{bmatrix} = \begin{bmatrix} 1.8452 \\ 1.7761 \\ 1.4998 \\ 3.9255 \\ 1.5000 \\ 2.8510 \\ 3.0000 \\ 2.8509 \\ 0.0000 \end{bmatrix}$$

The history of the optimized radius r is

r=[2.5 3.0122 2.68 2.5781 2.4835 2.3915 2.3013 2.2163 2.1314
2.049 1.9696 1.8939 1.8403 1.8452 1.8452 1.8452]

The second example is a triangle shown in Figure 20, and the third example is from a realistic mechanical assembly shown in Figure 21.

7 Conclusions

We have studied the problem of determining a disassembly motion of a subassembly in the free space within a parent subassembly. We propose a two-step approach to generating motion plans. First, *all possible paths* within free space of the parent subassembly are generated using a partial medial axis. This is followed by a graph search for an optimal *global path.* Second, a *valid motion* is planned using the global path and the *geometry of the moving subassembly.*

We extensively discuss the first problem in this paper. We presented a method to determine a partial medial axis (PMA) by generating critical points and connecting them based on the geometry properties of the critical point. This results in all possible paths that is allowed from the geometry constraints from the initial position to the final position.

We represent the PMA in the form of graph using the critical point and boundary information. Based on this graph representation, standard search techniques on an augmented graph is utilized to find an optimal path of the disassembly motion.

Using the generated path, we briefly discuss the motion planning strategy to navigate the subassembly within the free space of the parent subassembly. We at first determine two minimal disks that fully cover the subassembly polygon, and then constraint the centers of two disks moving continuously along the path on PMA.

References

[1] K.H. Chang **and** W.G. **Wee.** A knowledge-based planning system for mechanical assembly using robots- *IEEE expert,* 3:18-30,1988.

[2] E. Charniak and D. McDermott. *Introduction to Artificial Intelligence.* Addison-Wesley Publishing Company, 1985.

[3] T.L. De Fazio and D.E. Whitney. Simplified generation of all mechanical assembly sequences. ***IEEE Journal of Robotics and Automation,*** **RA-3:640-658,1987.**

[4] L. Gursoz, Y. Choi, and F. Prinz. Vertex-based representation of non-manifold boundaries. **In *Geometric Modeling for Product Engineering,* pages 107-130, 1988.**

[5] R. Hoffman. Automated assembly in a csg domain. In *Proc. IEEE Conference on Robotics and Automation,* pages 210-215, 1989.

[6] Y.F. Huang and C.S.G. Lee. Precedence knowledge in feature mating operation assembly **planning. In *Proceedings of IEEE Conference on Robotics and Automation,* pages 216-221,** 1989.

[7] P.K. Khosla and R.S. Mattikalli. Determining the assembly sequence from a 3-d description **of an assembly. In *Proc. of 2nd International CIRP Conference on New Manufacturing Technology,*1989.**

[8] D.G. Kirkpatrick. Efficient computation of continuous skeletons. In *Proceedings of 20th IEEE* ***Annual Symp. on Foundations of Comput. Science,*** **pages 18-27, 1979.**

[9] H. Ko and K. Lee. Automatic assembling procedure from mating conditions. *Computer Aided Design,* 19:3-10, 1987.

[10] R.S. Mattikalli and P.K. Khosla. Determining the assembly sequence from a 3-d model. In ***Proc. of 3rd Anual Expert Systems Conference and Exposition,*** **1989.**

[11] R.S. Mattikalli, P.K. Khosla, and Y. Xu. Subassembly identification and motion generation for **assembly: A geometrical approach. In *Proceedings of IEEE Conference on System Engineering,*** pages 399-403,1990.

[12] L.S. Homem De Mello and A.C. Sanderson.· Automatic generation of mechanical assembly sequences. Technical Report CMU-RI-TR-88-19, The Robotics Institute, Carnegie-Mellon University, 1988-

[13] J.M. Miller and R.L. Hoffman. Automatic assembly planning with fasteners. In *Proceedings* ***of IEEE Conference on Robotics and Automation,*** **pages 69-74, 1989.**

[14] F.P. Preparataand M.I. Shamos. *Computational Geometry : an Introduction.* Springer-Verlag, 1985.

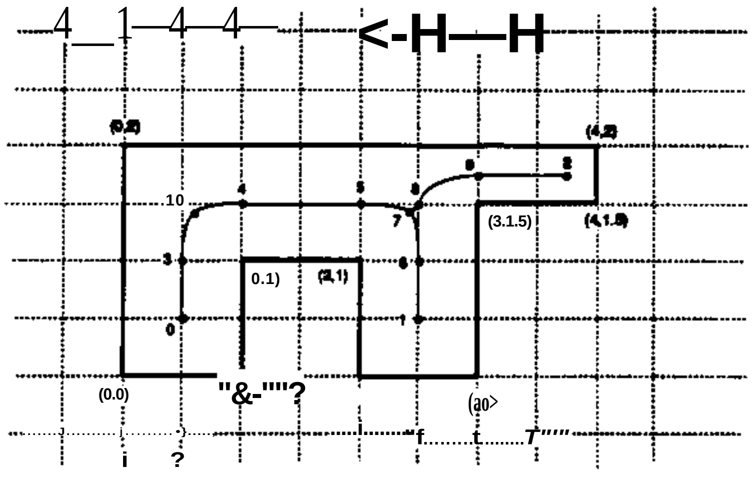
[15] R.O.Duda and P.E. Hart. *Pattern Classification and Scene Analysis.* Wiley-Interscience, New York, 1973.

[16] J.M. Rondeau and H.A. ElMaraghy. Development of a knowledge-based robot task planning **system for mechanical assembly. In *Proc. of ASME Conference in Flexible Assembly,* 1989.**

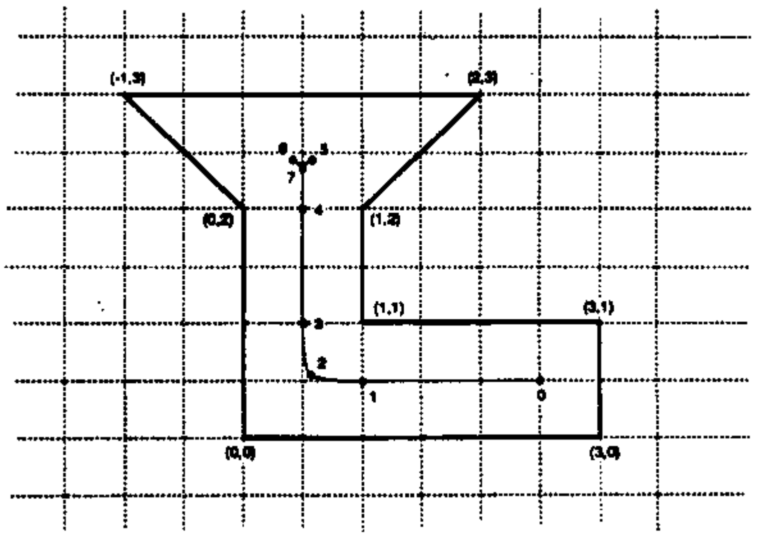
[17] **S.W. S«d» and S.N. Talukdar.** A disassembly planner for redesign. In *Proc. of the Winter Annual Meeting of ASME,* **pages 95-100,1987.**

[18] **J.D. Wolter. On the automatic planning** with **fasterners.** In *Proceedings of IEEE Conference **on Robotics and Automation,*** **pages 62-68,1989.**

[19] Y. Xu, R.S. Mattikaffi, **and** P.K. **Khosla.** Two-disk **motion** planning strategy, *(unpublished),* **1991.**

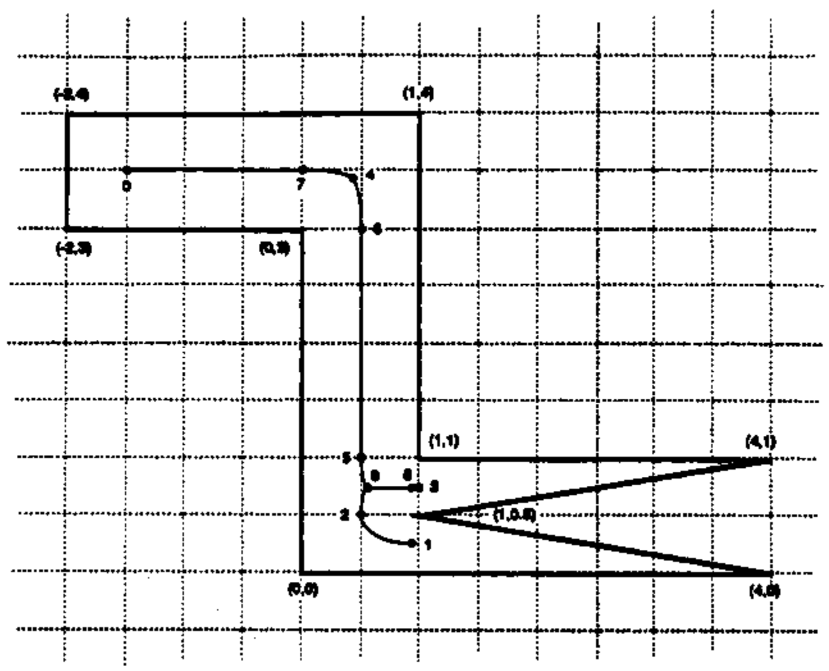


(a) Example 1

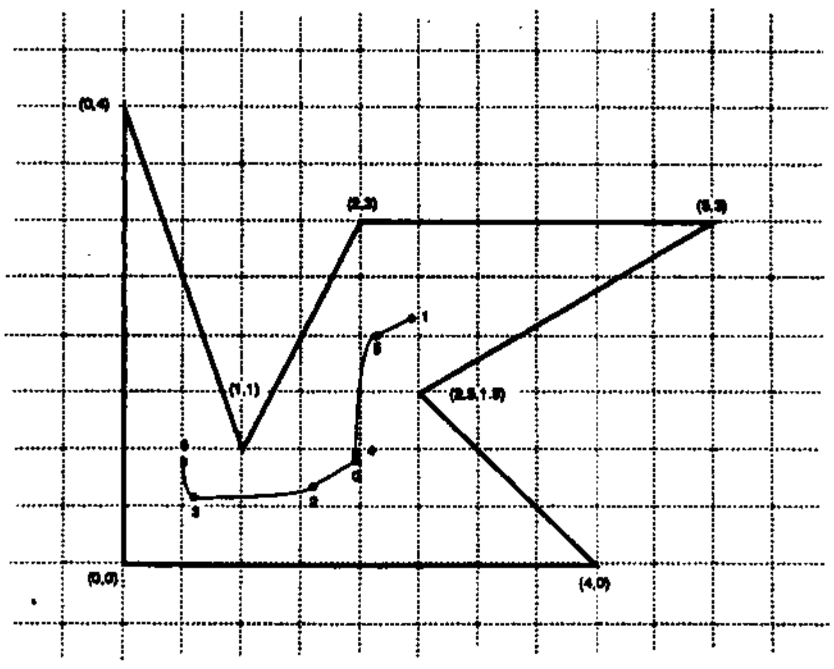


(b) Example 2

Figure 18 Generation of the partial medial axis by the proposed method

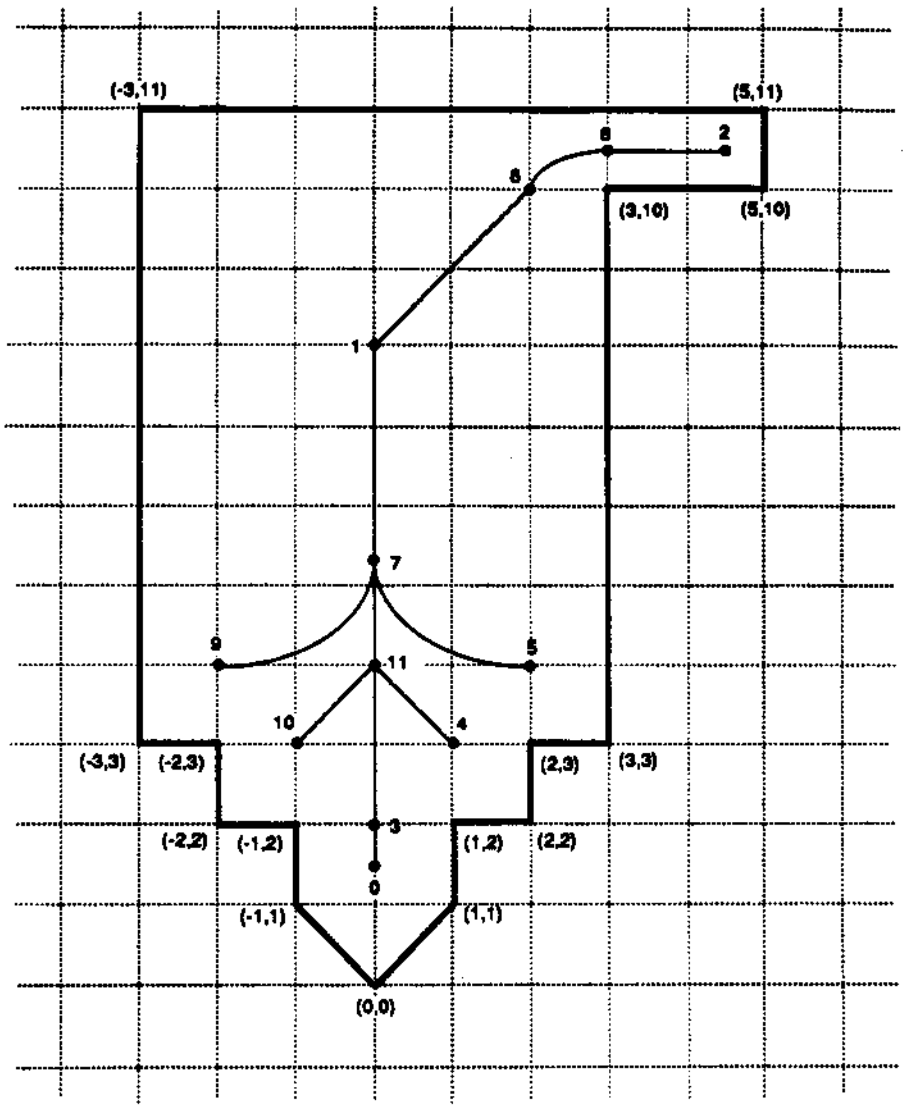


(c) Example 3



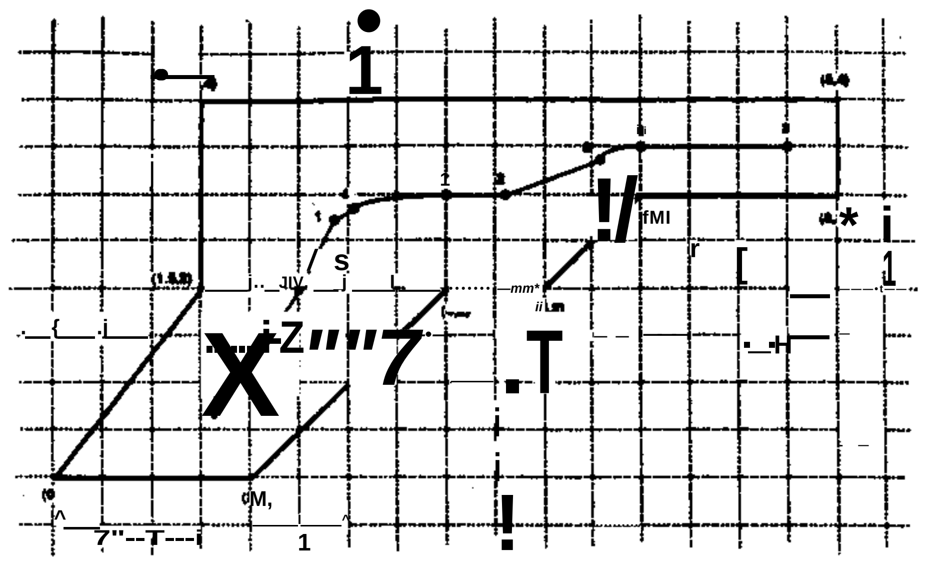
(d) Example 4

Figure 18 Generation of the partial medial axis by the proposed method



(e) Example 5

Figure 18 Generation of the partial medial axis by the proposed method



(0 Example 6

Rgur»18 Q«n«r«tion of th« partial medial axis by th« propOMd nwthod

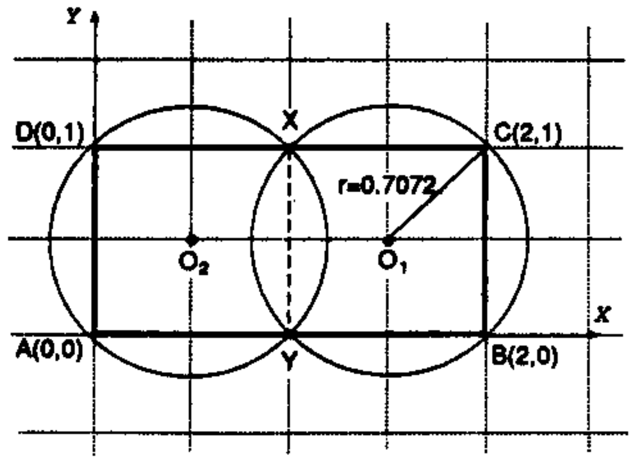


Figure 19 Two minimal overlapping disks for covering a rectangular

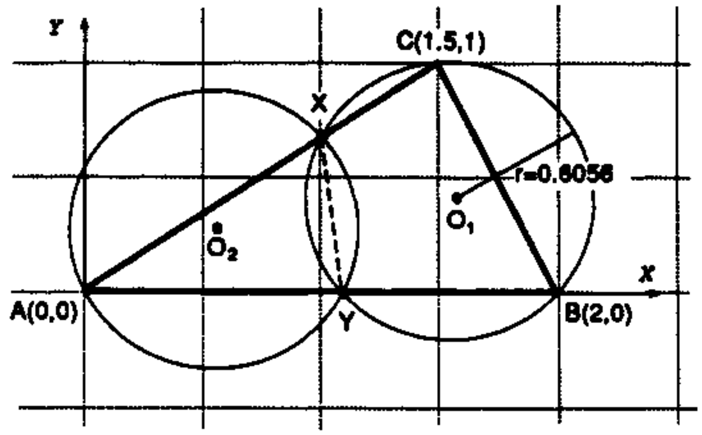


Figure 20 Two minimal **disks** for **covering a** triangle

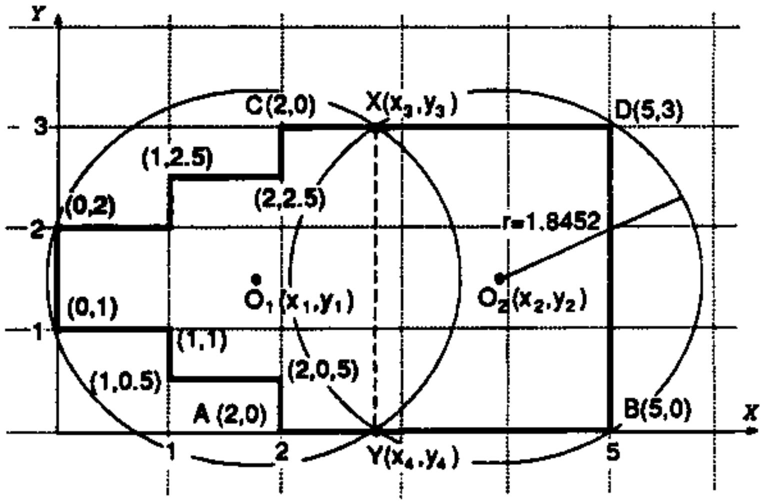


Figure 21 Two minimal disks for covering a pioygon