Proceedings of DETC 2000:
2000 ASME Design Engineering Technical Conference
September 10-13, 2000, Baltimore, Maryland

DETC2000/CIE-14630

FINDING THE MAXIMAL CUTTER FOR 2-*D* MILLING OPERATIONS

Zhiyang Yao

Mechanical Engineering Department and Institute for Systems Research
University of Maryland
College Park,MD-20742
Email:yaodan@Glue.umd.edu

Satyandra K. Gupta*

Mechanical Engineering Department and Institute for Systems Research
University of Maryland
College Park,MD-20742
Email:skgupta@eng.umd.edu

Dana S. Nau

Computer Science Department and Institute for Systems Research
University of Maryland
College Park,MD-20742
Email:nau@cs.umd.edu

ABSTRACT

Finding the biggest cutter is expected to help in the selection of the right sets of tools and the right type of cutter trajectories, and thereby ensure high production rate and meet the required quality level. In this paper, we describe a new geometric algorithm to determine the biggest feasible cutter size for 2-*D* milling operations to be performed using a single cutter. Our algorithm works not only for the common closed pocket problem, but also for the general 2-*D* milling problems with open edges. In particular:

- We give a general definition of the problem as the task of covering a *target region* without interfering with an *obstruction region*. This definition encompasses the task of milling a general 2-*D* profile that includes both open and closed edges.
- We discuss three alternative definitions of what it means for a cutter to be feasible, and explain which of these definitions is most appropriate for the above problem.
- We present a geometric algorithm for finding the maximal cutter for 2-*D* milling operations, and we give an outline of a proof that our algorithm is correct.

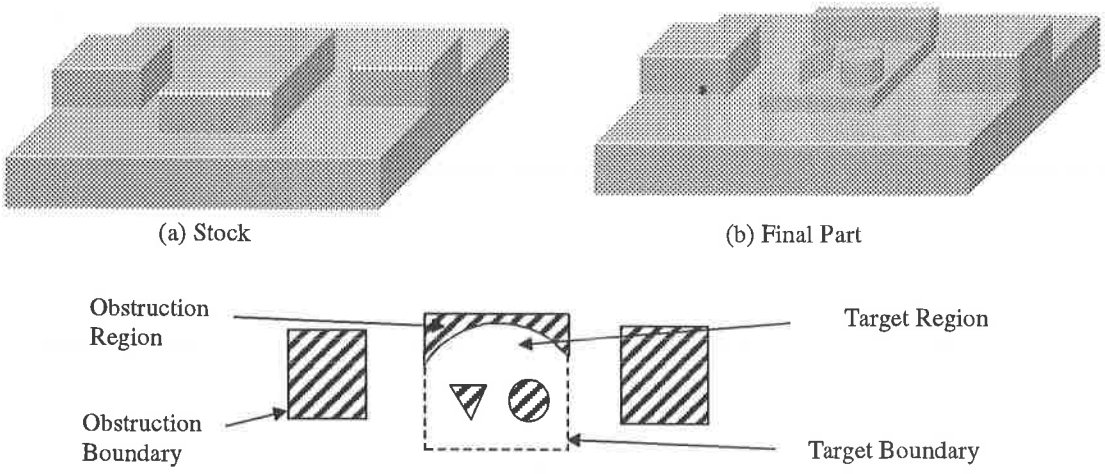
1. INTRODUCTION

NC machining is being used to create increasingly complex shapes. These complex shapes are used in a variety of defense, aerospace, and automotive applications to (1) provide performance improvements, and (2) create high performance tooling (e.g., molds for injection molding). The importance of the machining process is increasing due to latest advances in high speed machining that allows machining to create even more complex shapes. Complex machined parts require several rouging and finishing passes. Selection of the right sets of tools and the right type of cutter trajectories is extremely important in ensuring high production rate and meeting the required quality level. It is difficult for human planners to select the optimal or near optimal machining strategies due to complex interactions among tools size, part shapes, and tool trajectories.

Although many researchers have studied cutter selection problems for milling processes, there still exist significant problems to be solved. Below are two examples:

- Most existing algorithms only work on 2-*D* closed pockets (i.e., pockets that have no open edges), despite the fact that open edges are very important in general 2-*D* milling.
- Since there are several different definitions of what it means for a cutter to be feasible for a region, different algorithms that purport to find the largest cutter may in fact find cutters of different sizes.

* Corresponding author.



(a) Stock

(b) Final Part

(c) Target and Obstruction Regions

Figure 1: Notation and Nomenclature

In this paper, we present and implement an algorithm of finding maximal cutter for general 2-*D* milling operations to be performed using a single cutter.

- We formulate the general 2-*D* milling problem in terms of a *target region* and an *obstruction region*. This problem formulation encompasses the general problem of how to mill a 2-*D* region that has both open and closed edges (see Section 3 for the definitions).

- We analyze three different definitions of what it means for a cutter to be feasible, and explain why one of these definitions is more appropriate than the others.

- We define a "region covering" algorithm that finds the largest cutter that can cover the target region without interfering with the obstruction region, and we give an outline of the correctness proof for our algorithm.

In practice, quite often multiple cutters are used to machine a complex milling feature. We believe that finding the maximal cutter is a necessary step in the direction of finding the optimal set of cutters for machining a complex feature.

2. RELATED WORK

Because of the wide range of the complexity of products, requirement for machine accuracy, different machining stages, selecting optimal cutter size is an active area of research. However, for the task of 2-*D* milling, most research has focused on a restricted version of the problem in which the pockets have only closed edges.

While studying how to machine prismatic parts, Chang *et al.* presented an algorithm to select cutters for roughing and finishing [3]. Their work encompasses almost all the features found on prismatic parts, such as slots, steps. Their algorithm is based on geometric constraints. The basic idea is trying to fit the possible large circle into contours to select possible large cutter to save processing time. They take both cutter change time and geometric constraints into consideration. They stated the problem as follows. If there exists a set of cutters, and after machining with these cutters, only finishing machining is needed for fillet radii corners, so the problem is to determine the cutter with the largest radius in this set. The main concern is to make sure that the material left behind by the cutter at each of the convex vertices can be removed by one pass along the boundary of the finishing cutter. A convex vertex is defined as a vertex at which the interior angle is less than 180°. For each convex vertex, the radius of the circle touching the edges forming the vertex as well as the fillet circle is found. Then they check to see if this circle intersects any of the edges of the bounding polygon or any of the islands. If an edge that violates the circle is found, the circle has to be modified or the radius has to be reduced to remove this violation. The aim is to make the circle tangential to this edge. After that, the reflex vertices have to be handled. Reflex vertices are those vertices at which the interior angles are greater than 180°. To solve the problem caused by edges, perpendiculars are drawn from the reflex vertex to each of the edges. First of all, they check to see if the perpendicular hits the edge or not. If not, no action is taken. Otherwise they have to check to see if that particular edge is causing a valid constraint or not. To resolve the constraint caused by other reflex vertices, the distances to all the valid reflex vertices is determined. The smallest of all these distances gives the smallest constriction within the pocket. After determining the cutter size, feasible cutter motion region can be identified and the cutter movement within the region can be optimized.

Some researchers have also made their efforts to select a set of cutters for pocket machining. Veeramani and Gau have

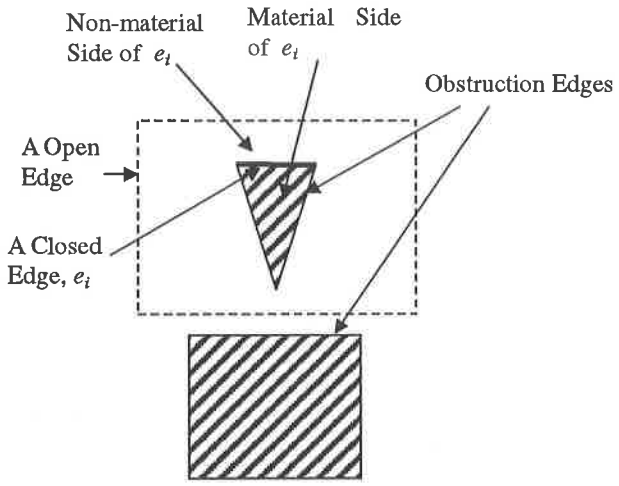


Figure 2: Closed and Open Edges

developed a two-phases method to select a set of cutters [4]. In the first phase, a concept called the Voronoi mountain is employed in order to calculate the material volume that can be removed by a specific cutting-tool size, the material volume remained to be machined subsequently and the cutter-paths for each cutting-tool. In the second phase, a dynamic programming approach is applied for optimal selection of cutting-tool sizes on the basis of the processing time. This algorithm considers geometric constrains as well as total processing time. It is possible to save processing and machining cost compared to using a single cutter to machine the entire pocket. However, such method can only apply to a CNC machining center with high speed of automatic tool change mechanisms.

Yang and Han presented a systematic tool-path generation methodology in which they incorporate interference detection and optimal tool selection for machining free-form surfaces on 3-axis CNC machines using ball-end cutters [8]. To find the optimal tools, a comparison of all possible combination of tools are performed. The optimal tool selection can be one cutter of no more than 3 cutters. This optimal tool selection method is designed aimed for any type of parametric surfaces to be machined. There are a few problems with this algorithm. First, because the algorithms are grid-line based, if very fine resolution of grid is imposed, high computational power is demanded to implement the algorithm. Second, if the number of available tools is large, the comparison of all possible combination of tools could be time consuming.

Sarma *et al.* are developing an alternative feature-free approach to automatic CAD/CAM integration for 3-axis machining [7]. The algorithm is based on Voronoi diagram. The objective is to select tools for global roughing and generate tool paths directly from the shape of the workpiece. First, slices are generated as sequences of closed contours. Then a Voronoi diagram is employed to generate the path of the centerline of the tool and calculate the accessibility region on each slice. The criterion to select a cutter in this algorithm is to select the cutter that can sweep much of the region of the slice instead of selecting a largest possible cutter. Because large tools have less reachable region than small tools so they incur the penalty of tool changes, so the optimal tool sequence should be selected based on the total time which depends on the region that each tool can access in each slice. To calculate the accessible region, the overall geometry of the tool assembly including the tool holder and the spindle is considered in this algorithm.

From the above, we can see the progress of the research on the cutter selection problem. Different machining stages, rough cut, finish cut and semi-rough cut have all been studied to some extent. In terms of the complexity of the products, research has advanced from pocket machining, given feature-based surfaces to feature-free sculptured surfaces. In the direction of finding the optimal set of cutters for a complex feature, finding of the maximal cutter is a very helpful step.

3. DEFINITIONS

3.1 Problem Formulation

The most common milling problem is the problem of cutting a given 2-*D* region at some constant depth using one or more milling tools. In addition to the region to be machined (which we call the *target region T*), there is also an *obstruction region O*, a region which the tool should not cut during machining. An example is shown in Figure 1. The target region and the obstruction region may each consist of a number of non-

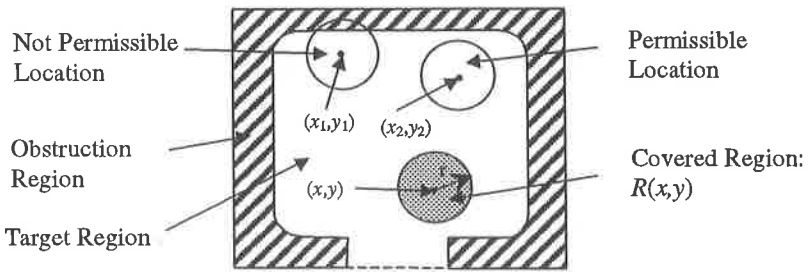


Figure 3: A covered region, and locations that are permissible and not permissible.

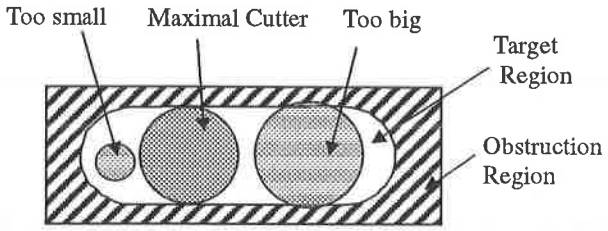


Figure 4: Maximal Tool

adjacent sub-regions:

$$T = T_1 \cup \ldots \cup T_j;$$
$$O = O_1 \cup \ldots \cup O_k.$$

We assume that the boundary of each sub-region consists of only of line segments and segments of circles.

As shown in Figure 1(c), the *target boundary* B_T is the boundary of the target region, and the *obstruction boundary* B_O is the boundary of the obstruction region. The edges on the obstruction boundary are called *obstruction edges*. We call an edge of the target boundary a *closed edge* if it is coexists with an obstruction edge; otherwise we call it an *open edge*. (Note that 2-*D* closed pockets do not have any open edges [1, 3-4].) Figure 2 shows examples of open and closed edges. Each closed edge or obstruction edge separates the *material* (i.e. the obstruction region) from part of the target region. The side on which the material lies is called the *material side* and the other side is called the *non-material side*. We will use the dashed line to represent the open edges, the shaded regions as the obstruction region.

Let C be a circular cutter of radius r, and (x, y) be a point. Then the *region covered* by C at the point (x, y) is the following set:

$$R(x,y) = \{ \text{ all points } (u,v) \text{ such that } \sqrt{(u-x)^2 + (v-y)^2} \le r \ \}$$

An example is shown in Figure 3.

A point (x, y) is a *permissible location* for C if the interior of $R(x, y)$ does not intersect with the obstruction region, or equivalently, if $O \cap^* R(x, y) = \varnothing$. An example is shown in Figure 3.

A set of points S is *coverable* using a cutting tool C if for every point p in S, there is a permissible location of C that covers p.

A cutting tool C_m is *maximal* for a target region T if C_m is the largest tool that is feasible for T (refer to Figure 4 for a graphical illustration). The size of C_m will depend on how we define *feasibility*. There are several alternative definitions for feasibility, as described below:

Most existing algorithms for cutter-tool-size selection are just used to find the "maximal cutter" without clearly stating what it means for this cutter to be the maximal feasible cutter. However, whether or not a cutter is feasible for a target region can be defined based on a number of criteria; and based on those different criteria, different maximal cutters can be obtained. In order to make our approach clearer, we describe the following three alternative definitions of cutter feasibility.

- *Alternative 1. Feasibility Definition Based on Tool's Ability to Cover the Target region*: A cutting tool C is *feasible* for a target region T if every point in T is coverable by C. In Figure 5, C_1 is an example of the maximal cutter based on this definition.

- *Alternative 2. Feasibility Definition Based on Existence of Continuous Tool Path between Every Pair of Points within the Target Region*: A cutting Tool C is *feasible* for a target region T if for every pair of points (u, v) and (u', v') in T, there is a continuous tool path H such that every point (x, y) in H is a permissible location for C and the covered region

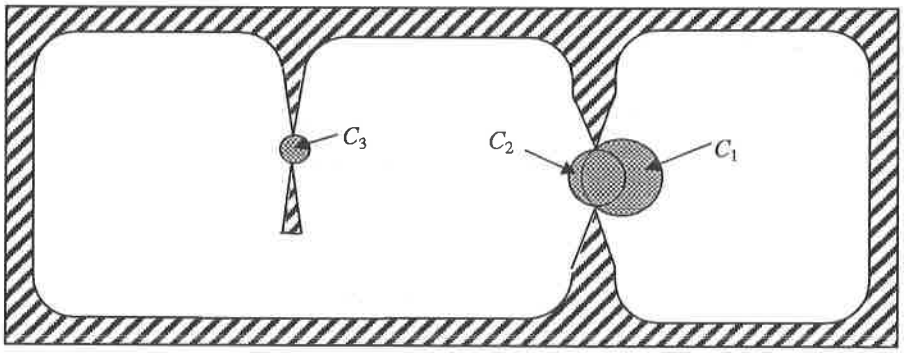
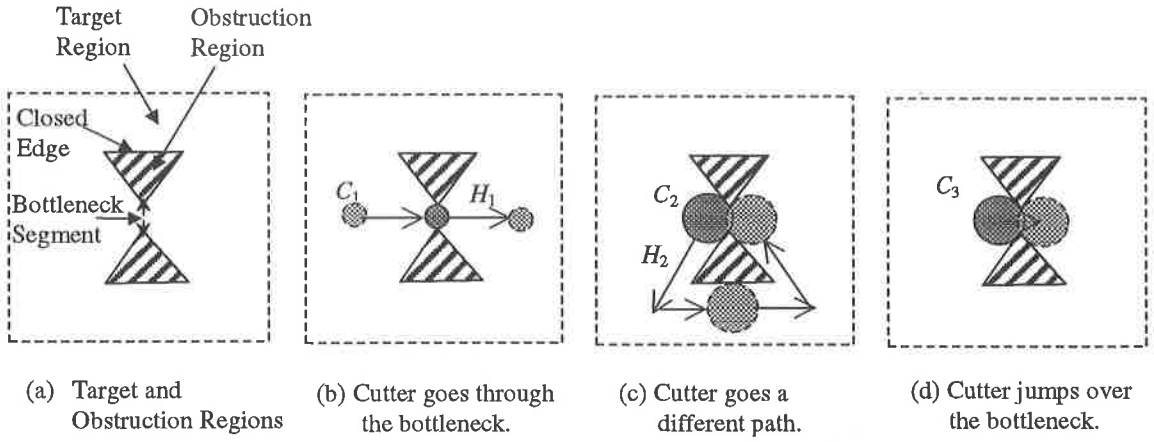


Figure 5: Maximal Cutters Found by Different Alternatives of Cutting Feasibility



(a) Target and Obstruction Regions

(b) Cutter goes through the bottleneck.

(c) Cutter goes a different path.

(d) Cutter jumps over the bottleneck.

Figure 6: Different Cutter Path with Different Maximal Cutter

of H contains (u, v) and (u', v'). In Figure 5, C_2 is an example of the maximal cutter based on this definition.

- *Alternative 3. Feasibility Definition Based on Passing Through All Bottleneck Segments*: A cutting tool C is *feasible* for a target region T if C can pass through all possible *bottleneck segments* formed by the Voronoi diagram of T. [1] In Figure 5, C_3 is an example of the maximal cutter based on this definition.

Most algorithms in the existing literature are based on Alternative 3. It is easy to think of taking the minimum of all possible "bottleneck segments" generated by the Voronoi diagram of target region T (for example, see Figure 6(a)), and then using this minimum distance as the diameter of the cutter as shown in Figure 6(b) [1]. However, in many milling operations we can get the required geometric form without forcing the cutter to go through *every* bottleneck. We can allow the cutter to take a different path (Figure 6(c)), leading to the definition of feasibility in Alternative 2. In addition, we can jump over the bottleneck by lifting the cutter up and placing it across the bottleneck (Figure 6(d)), leading to the definition of feasibility in Alternative 1. For simple cases (such as the situation shown in Figure 4), all three alternatives would give the same answer. However, in more complicated situations (such as the situation shown in Figure 5), Alternative 1 will give the largest cutter size and Alternative 3 will give the smallest cutter size.

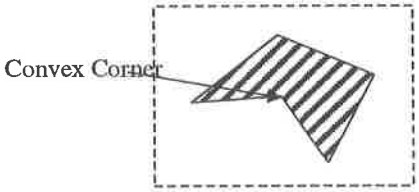


Figure 7: An Example of Un-solvable Problem

The main goal of finding the maximal cutter is to save the manufacturing time and therefore reduce manufacturing cost. Generally, the time spent in cutting by using a small cutter is much longer than using a bigger cutter along with some lifting operations or an alternative path. Since Alternative 1 gives us bigger cutters than the other alternatives, Alternative 1 is the definition of cutter feasibility that we think is preferable in actual practice—and thus it is the one that we use in this paper.

With the above definition in mind, we define the *cutter selection problem* as following: given a target region T and an obstruction region O, find the largest cutter C that can cover T.

Note that if any two closed edges meet at a convex corner (see Figure 7 for an example), then there is no cutter that can cover T. In this case, we say that the cutter selection problem is *unsolvable*. In all other cases, it is *solvable*.

3.2 Additional Definitions

Intuitively, the *edge region* $E_i(C)$ for a closed edge e_i, is the region formed by sweeping the cutter C along the no-material side of this edge. Mathematically speaking, a point p is in $E_i(C)$ if p is contained in some circle C of radius r that is tangent to e_i on the non-material side of e_i. For an obstruction region, O_j, the edge region E_j for this obstruction region is the regular union of all E_is of the closed edges on the boundary of this obstruction region. We use $E(C)$ to denote the regular union of all edge regions for a given problem. Figure 8(c) shows an example of an edge region.

The *maximal swept cutter* for a closed edge e_i is the largest cutting tool C_i such that $E_i(C)$ does not intersect the interior of O. The maximal swept cutter for *all* of the closed edges is the largest cutting tool C such that for every i, $E_i(C)$ does not intersect the interior of O. An example of a maximal swept cutter is shown in Figure 8(b).

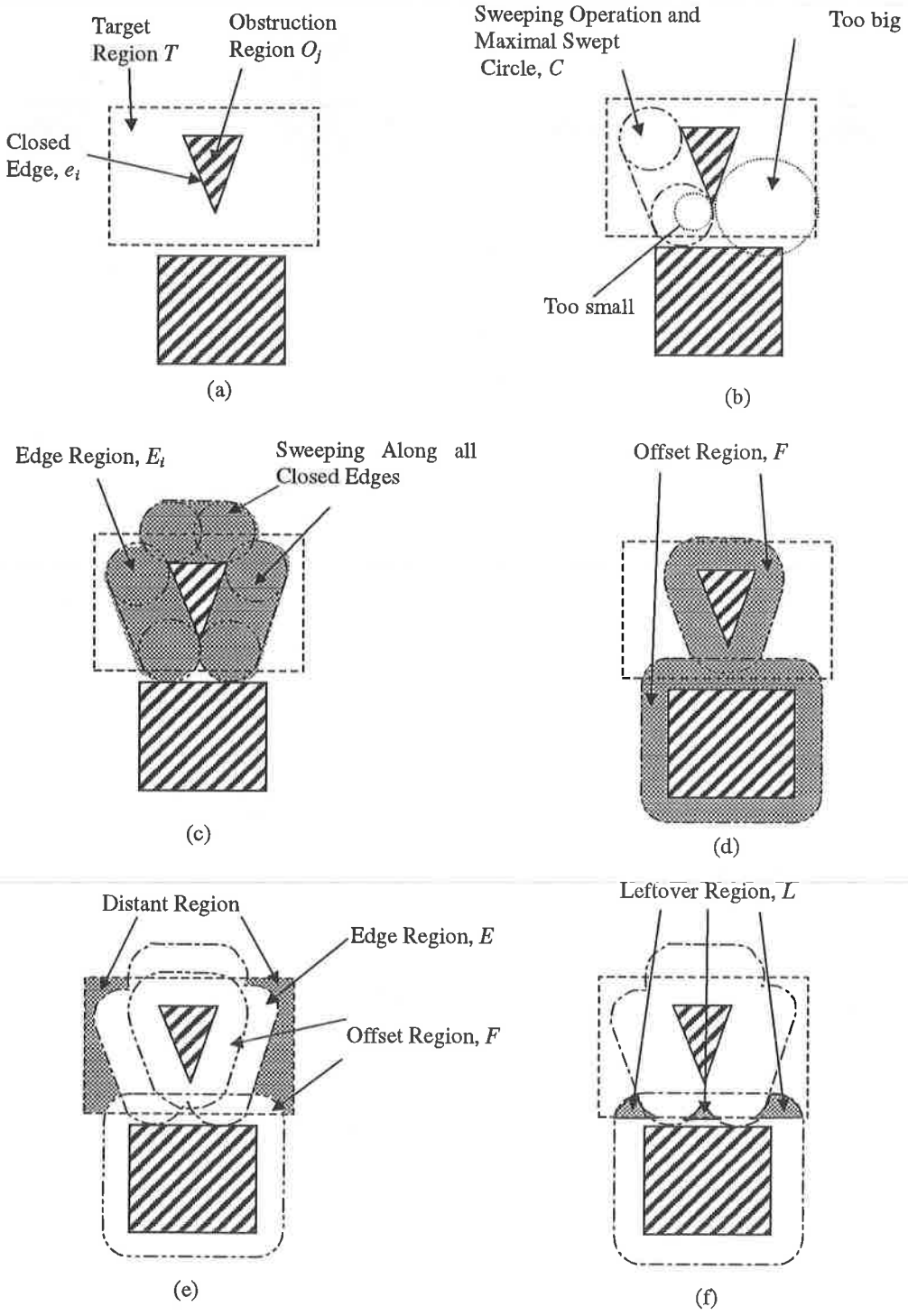


Figure 8: Examples of *E, F, D, L*, etc.

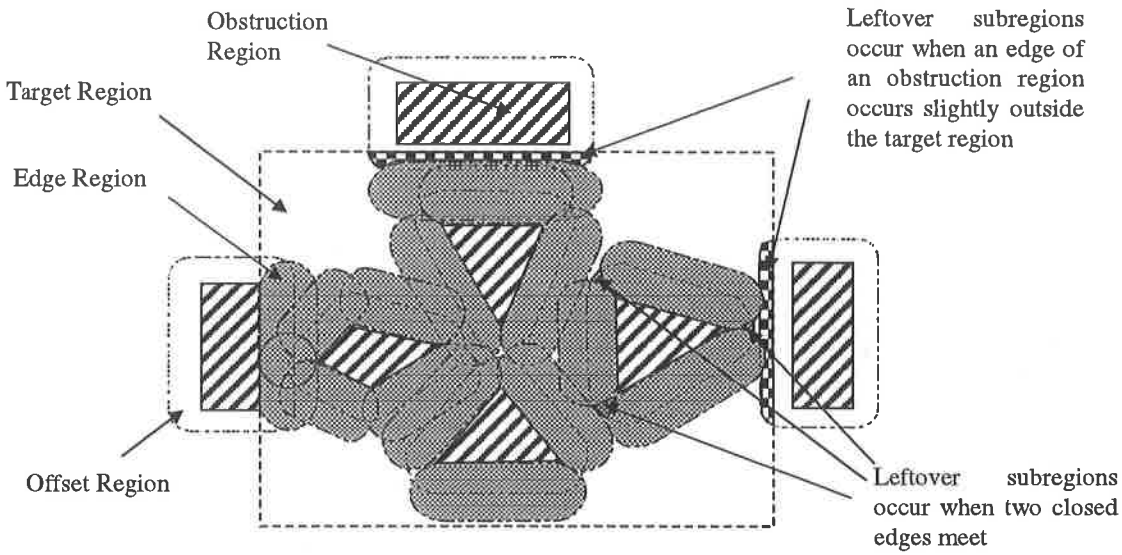


Figure 9: Some Examples of Leftover Region

Intuitively, the *offset region* $F_j(C)$ is the region formed by offsetting the obstruction subregion O_j using the radius r of the cutter C. Mathematically speaking, a point p is in $F_j(C)$, if $p \notin O_j$ and the minimal distance from p to any point in O_j is less or equal to r. The regular union of all $F_j(C)s$ is called the offset region $F(C)$ for all obstruction regions. Figure 8(d) shows an example of an offset region.

Intuitively, the *distant region* consists of all points in the target region that are "far away" from the obstruction region. Mathematically, the distant region is the following set of points:

$$D(C) = T -^* (E(C) \cup^* F(C)).$$

Figure 8(e) shows an example of a distant region.

The following proposition follows immediately from the above definitions:

Lemma 1 (*Property of Distant Region*): $D(C)$ is coverable by the maximal swept cutter C.

Proof. Let p be any point in $D(C)$. Since the distance from p to any obstruction region is greater or equal to the radius of C, p is a permissible location for C. ◻

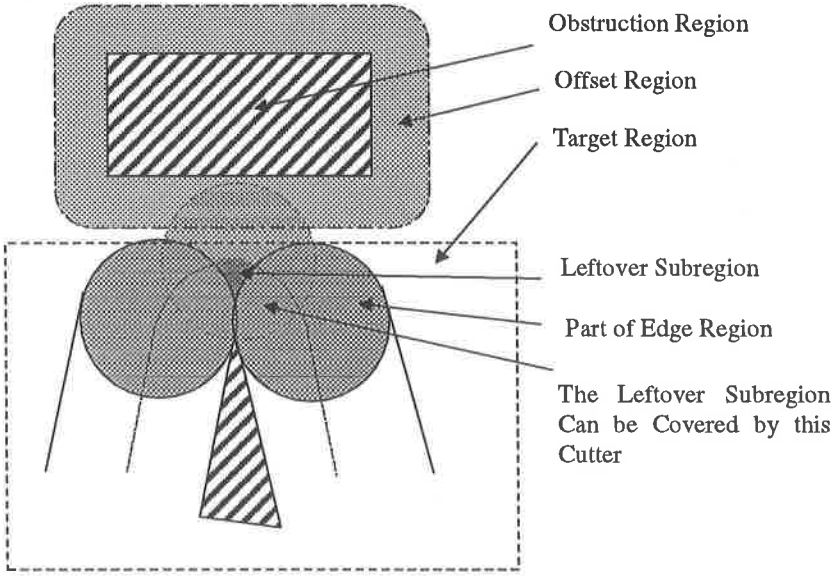


Figure 10: Some Leftover Subregion Can be Covered

The *leftover region* is the set of all points $L(C)$ such that

$$L(C) = T \cap^* (F(C) -^* E(C)).$$

This region may consist of a number of non-adjacent sub-regions:

$$L(C) = L_1(C) \cup \ldots \cup L_i(C).$$

Below we give several examples of what these leftover subregions can look like:

- One kind of leftover subregion can occur when two closed edges meet, as shown in Figure 9. However, this kind of region will not occur if the angle between the two closed edges is greater than 60 degrees.

- Another kind of leftover subregion can occur when an edge of an obstruction region occurs slightly outside the target region, as shown in Figure 9. However, this kind of region will not occur if the distance between the edge and the target region is greater than the radius of the cutting tool.

3.3 Some Properties of the Leftover Region

The following theorem is the basic property on which our algorithm is based:

Theorem 1: Let C be any cutter whose radius r is small enough that $E(C) \cap^* O = \varnothing$. Then for every point $p \in T -^* L(C)$, p is coverable by C.

Proof. Let p be any point in T –* $L(C)$. Note that

$$T -^* L(C) -^* D(C) -^* E(C)$$
$$= T -^* L(C) - (T -^* (E(C) \cup^* F(C)) -^* E(C)$$
$$= -(T \cap^* (F(C) -^* E(C)) + (T \cap^* (F(C) - E(C))$$
$$= \varnothing.$$

Thus, p must be in $D(C)$ or $E(C)$. If $p \in D(C)$, then from Lemma 1, p is coverable by C. If $p \in E(C)$, then it follows from the definition of $E(C)$ that p can be covered by C. □

If p is a point in the leftover region $L(C)$, then p may or may not be coverable by C. Below are some examples:

If L_i(C) is a sub-region of $L(C)$, then it is a subset of the offset region F_j(C). If L_j(C) does not intersect with any other offset region, then in most cases L_j(C) can be covered by the cutter C (the only exception is when the offset region F_i(C) is self-intersecting). See Figure 10 for an example.

If L_i(C) is a sub-region of $L(C)$, then it is a subset of the offset region F_j(C). If L_i(C) intersects some other offset region, then sometimes L_i(C) can be covered by the cutter C and sometimes it cannot. Figure 13 shows an example of the leftover region that cannot be covered.

4. ALGORITHM FOR FINDING THE MAXIMAL CUTTER

As described below, our main algorithm for the maximal cutter selection problem is called ***Find_Maximal_Cutting_Tool*** (***FMCT*** for short). For every closed edge a, this algorithm calls the algorithm ***Maximal_Swept_Cutter_For_Closed_Edge*** to find the maximal swept cutter C. Then we use the smallest of those maximal swept circles (denoted by C_S) to sweep along all closed edges in order to split the whole region into edge region, distant region and the leftover region. For the leftover region, the algorithm ***Maximal_Cutter_For_Leftover_Region*** is called to find the maximal cutter C_L that can cover it. The final result for the target region is the smaller one of C_S and C_L.

Procedure ***Find_Maximal_Cutting_Tool***(T, O)
//T is the target region, and O is the obstruction region.

1. Initialize d_S=∞, d_L=∞;
2. For each closed edge a and obstruction edge b:
   - d = ***Maximal_Swept_Cutter_For_Closed_Edge***(a,b) (see Section 5 for details);
   - $d_S = \min\{d, d_S\}$;

   //d_S is the diameter of the maximal swept circle C_S for all closed edges.
1. Let E be the edge region produced by using the cutter with diameter of d_S performing a sweeping operation along all closed edges.
2. Let F be the offset region of all obstruction regions by using r_S=d_S/2;
3. Let D be the distant region, D=T-*F;
4. Let L be the leftover region;
5. If $L \neq \varnothing$:
   d_L=***Maximal_Cutter_For_Leftover_Region***(T,O,L, d_S) (see Section 6 for details);
   //d_L is the diameter of the maximal circle C_L that can cover the leftover region.
6. Return $d = \min\{ d_S, d_L \}$;

5. FINDING THE MAXIMAL SWEPT CUTTER FOR A CLOSED EDGE

In the algorithm ***Find_Maximal_Cutting_Tool*** described in Section 4, the purpose of the subroutine ***Maximal_Swept_Cutter_For_Closed_Edge*** is to solve the following problem: given a closed edge a and an obstruction edge b, find the largest circle Y with diameter d that can be swept along a without intersecting b.

There are several possible cases, as shown in Figures 11 and 12. In these figures, the closed edge a, with end points p_1 and p_2, is the edge along which the circle Y (of diameter d) should be swept tangentially. The obstruction edge b', with end points q_1 and q_2, is the edge that Y cannot intersect during the sweeping operation. As defined in Section 3, each

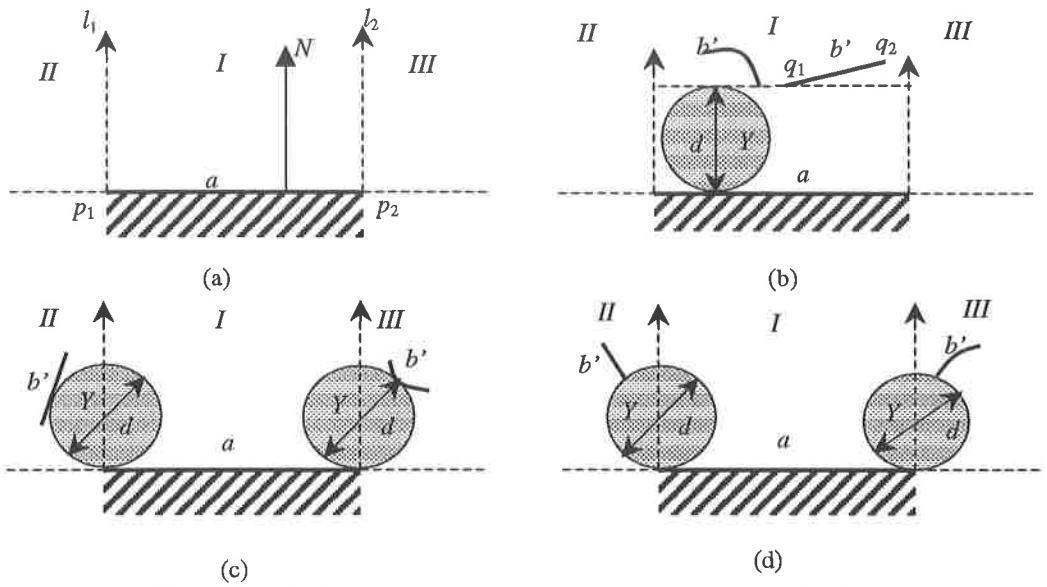


Figure 11: Finding the maximal cutter when the closed edge a is a line segment.

closed/obstruction edge has a material side and a non-material side. We define the positive normal direction N of a to be the direction vector that is perpendicular to a and points toward the non-material side of a. We say an obstruction edge b is *on the material side* of a closed edge a if and only if every point on b is in the material side of a. As shown in Figure 11, if a is a line segment, we draw two infinite rays, l_1 and l_2, that are perpendicular to a along N, pointing to the non-material side of a and starting from each end point of a. As shown in Figure 12, if a is an arc segment, we draw two infinite rays, l_1 and l_2, from each end point radially, pointing to the non- material side of a. In both cases, l_1 and l_2 will divide the whole non-material side of a into three sub-regions. We call those regions *I, II, III*, as shown in Figures 11 and 12.

Procedure *Maximal_Swept_Cutter_For_Closed_Edge* (a, b)

1. Split b into segments such that each segment is completely contained in regions *I, II, III*, or the material side of a;
2. $d = \infty$;
3. For every segment b' of b, do the following:
   - If b' is on the material side of a then we don't need to consider b', therefore $d' = \infty$;
   - Else:
     - If a is a line segment, $d' =$ ***Maximal_Swept_Circle_For_Line***(a, b');
     - If a is an arc segment, $d' =$ ***Maximal_Swept_Circle_For_Arc***(a, b');
   - $d = \min\{d, d'\}$;

Procedure *Maximal_Swept_Circle_For_Line*(a, b')

1. If b' is in region *I*, return d = distance between a and b';
   //Because in this region, when the circle sweeps along a, the distance between a and b' will be the maximal diameter of the circle. An example is shown in Figure 11(b).
2. If b is in region *II* or *III*:

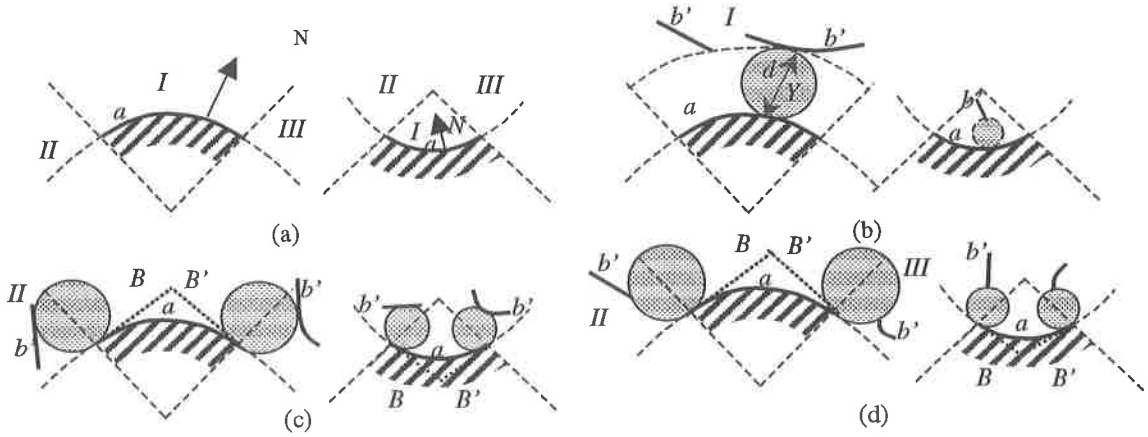


Figure 12: Finding the maximal cutter when the closed edge a is an arc segment.

- Try to find a circle Y that is tangent to both b' and a at one of the end points of a and the center of Y is located on l_1(for region *II*) or l_2(for region *III*), then return d ={the diameter of Y};

  //An example is shown in Figure 11(c).
- If such a circle does not exist, then construct two circles passing through two end points of b' such that centers of circles are located on l_1 or l_2, and circles are tangent to a at the one of its end points. Return the diameter of the smaller circle.

  //An example is shown in Figure 11(d).

Procedure ***Maximal_Swept_Circle_For_Arc***(a, b')

1. If b' is in region *I*, return d= the distance of a and b'};

   //Because in the region, when the circle sweeps along a, the distance between a and b' will be the maximal diameter of the circle. An example is shown in Figure 12(b).
2. If b' is in the region *II* or *III*, from each end point of a draw two rays tangent to a and pointing to the non-material side of a, called B, B':
   - Try to find a circle Y that is tangent to both b' and B (or B') at the start point of B (or B') and the center of Y is located on l_1 (for region *II*) or l_2 (for region *III*), then return d ={the diameter of Y};

     //An example is shown in Figure 12(c).
   - If such a circle does not exist, then construct two circles passing through two end points of b' such that circle centers are on l_1 or l_2, and they are tangent to B (or B') at the start point of B (or B'). Return the diameter of the smaller circle.

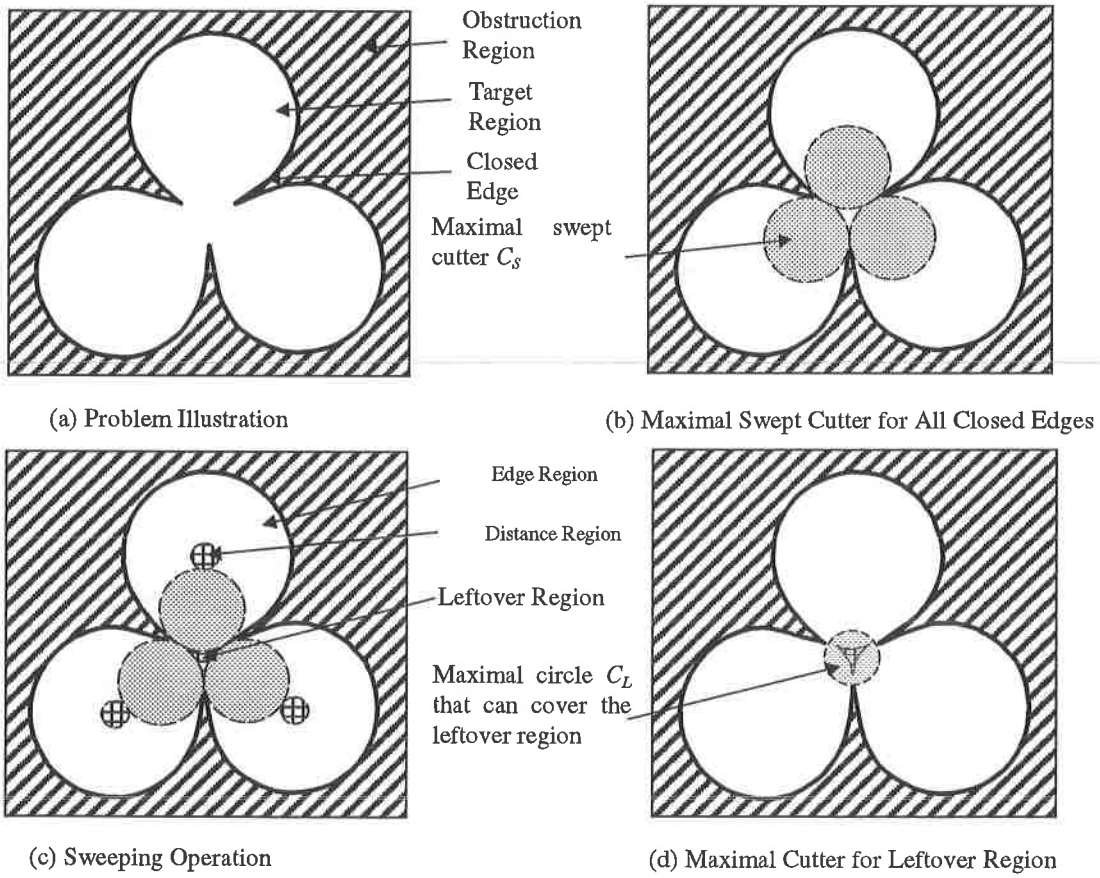
     //An example is shown in Figure 12(d).

6. FINDING THE MAXIMAL CUTTER FOR LEFTOVER REGION

If the leftover region in the Step 6 of ***Find_Maximal_Cutting_Tool*** described in Section 4 is not empty, then the maximal swept cutter C_S found by our algorithm may not be able to cover the leftover region. Therefore, we need to reduce the cutter diameter to make sure that the smaller cutter is found that can cover the leftover region. The purpose of the subroutine ***Maximal_Cutter_For_Leftover_Region*** is to find a maximal cutter that can cover the leftover region.

Procedure ***Maximal_Cutter_For_Leftover_Region***(T, O, L, d_S)

//T is the target region, O is the obstruction region, L is the leftover region and d_S is the diameter of the maximal swept



(a) Problem Illustration

(b) Maximal Swept Cutter for All Closed Edges

(c) Sweeping Operation

(d) Maximal Cutter for Leftover Region

Figure 13: An example of the operation of our algorithm.

cutter.

1. Let $d_L = d_S$.
2. Let U be a finite region that encloses obstruction and target regions.
   //In practice we compute U by computing bonding box of $T \cup^* O$.
3. Compute offset region F for the obstruction region O.
   //In practice we compute F by offsetting edges that define the boundary of O by a distance of $d_I/2$.
4. $K = O \cup^* F$.
   //K is a set of locations that are not permissible for cutter with diameter d_L.
5. $P = U -^* K$.
   //P is set of permissible locations for cutter with diameter d_L.
6. Compute the covered region S by placing cutter with diameter d_L in every point p in P. $S = \{p' \mid p \text{ in } P \text{ and } |p'-p| \leq d_I/2\}$
   //In practice this step is computed by offsetting edges that define the boundary of S by a distance of $d_I/2$.
7. If the left over region L is included in S, then return d_L.
8. Otherwise, set $d_L = d_L - d_i$ and goto Step 3.
   // d_i is a constant set by the user. It should correspond to the increment in tool sizes that are available in a shop floor.

7. DISSCUSSION OF CORRECTNESS OF OUR ALGORITHM

For a solvable problem, our algorithm exhibits the following two properties:

- *Property 1*: If the leftover region in Step 6 of ***FMCT*** is empty, then the diameter found by ***FMCT*** is exactly the theoretically maximal diameter.

  If there is no leftover region, then the result found by ***FMCT*** will be the maximal swept cutter for all closed edges. Therefore, as a direct consequence of Theorem 1, the result found by ***FMCT*** will be able to cover the target region. Therefore, the result found by ***FMCT*** is feasible. This result was found by taking the minimum of all the maximal swept cutters for various closed edges. Therefore, a bigger cutter would intersect with O during the sweep along some closed edge. Therefore no bigger cutter will be feasible. Since the result returned by ***FMCT*** is always feasible and maximal, ***FMCT*** is sound for every solvable problem. By the nature of ***FMCT*** algorithm, it always returns a result for a solvable problem, therefore ***FMCT*** is complete. These two characteristics make ***FMCT*** a correct algorithm that returns exactly the theoretically maximal diameter if the leftover region in Step 6 of ***FMCT*** is empty.

- *Property 2:* If the leftover region in Step 6 of ***FMCT*** is not empty, then (1) result returned by ***FMCT*** is feasible, and (2) the difference between the theoretically maximal diameter and result returned by ***FMCT*** is smaller than d_i.

  Let d be the results produced by ***FMCT***. From the nature of ***FMCT*** we know that $d \leq d_S$ (where d_S is the diameter of the maximal swept cutter). Therefore, as a direct consequence of Theorem 1, d can cover the region $T -^* L$.

  Procedure ***Maximal_Cutter_For_Leftover_Region*** only allows cutters that can cover L. Therefore d can cover the target region and is feasible.

If d is equal to d_S, then d is the exact solution and there is no difference between theoretical answer and the result found by ***FMCT***. If $d < d_S$, then $d + d_i$ cannot be a feasible solution. Otherwise, ***FMCT*** would have returned this solution. Therefore, in this case, theoretically maximal diameter $d^* < d + d_i$. Therefore the difference between the theoretically maximal diameter and result returned by ***FMCT*** is smaller than d_i.

8. IMPLEMENTATION AND EXAMPLES

We use the example shown in the Figure 13 to illustrate the operation of our algorithm. The target region and obstruction regions are shown in Figure 13(a). The details are as follows:

- First we need to find the maximal swept cutter C_S that can be swept along all closed edges (shown in Figure 13(b)).

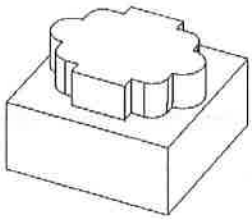
- Then we use C_S to perform sweeping operation along all closed edges to divide the whole region into the edge region and the distant region (shown in Figure 13(c)).

- We then get the leftover region. For the leftover region, we will find the maximal circle C_L that covers it and does not interfere with obstruction region, as shown in Figure 13(d).

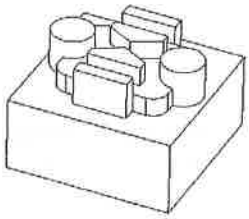
- The maximal cutter that covers the target region without interfering with the obstruction region will be the minimal one of C_S and C_L.

We have finished implementation of finding the maximal swept cutter. The core program work is done by using C++ on UNIX system. Meanwhile, we are creating the user's interface by JAVA applet. The user can input line and arc segments on the web page and thus generate the profile of a given 2-*D* milling problem. Then by executing the core program, the right size of the maximal cutter can be found automatically. We are also working on linking our core code with the ACIS toolkit such that we can show the 3-*D* version of the maximal cutter selection problem.

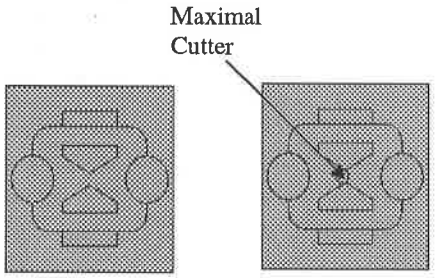
Some testing examples of our program are shown in Figure 14-17. In those figures, the maximal cutters found by using our algorithm are shown.



(a) Starting Stock

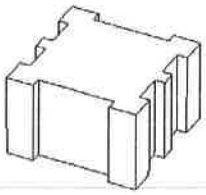


(b) Final Part

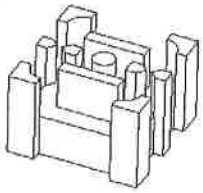


(c) Profile of Final Part (d) The Maximal Cutter

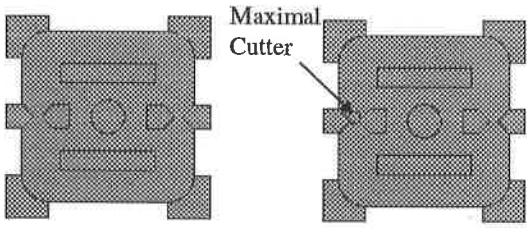
Figure 14: Example 1



(a) Starting Stock

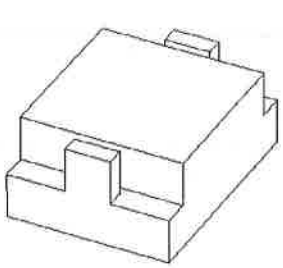


(b) Final Part

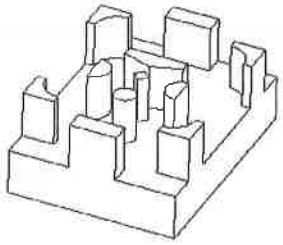


(c) Profile of Final Part (d) The Maximal Cutter

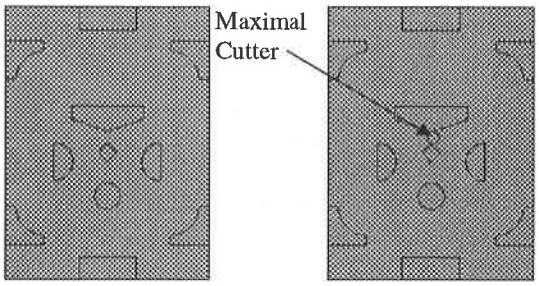
Figure 15: Example 2



(a) Starting Stock

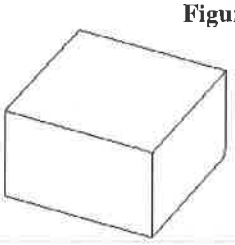


(b) Final Part

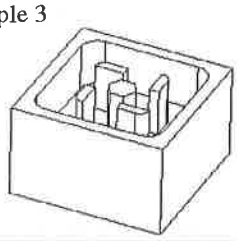


(c) Profile of Final Part (d) The Maximal Cutter

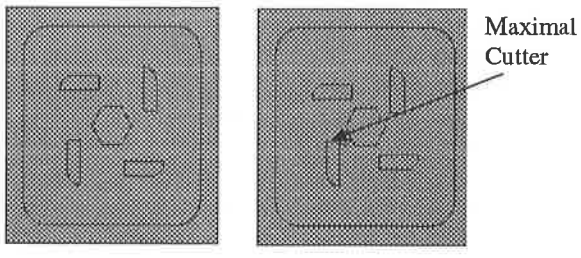
Figure 16: Example 3



(a) Starting Stock



(b) Final Part



(c) Profile of Final Part (d) The Maximal Cutter

Figure 17: Example 4

9. CONCLUSION AND DISCUSSION

In this paper, we have presented a geometric algorithm for finding the maximal cutter size for a 2-*D* milling process. Our algorithm has the following properties:

- It finds the largest cutter that can cover the region to be machined without interfering with the obstruction region.
- In addition to solving traditional pocket-milling problems, our algorithm can solve a wide variety of milling problems that involve open edges. Consideration of open edges is extremely important when near-net shape castings are used as starting stocks.
- Our algorithm uses a cutter feasibility definition based on cutter's ability to cover the target region. Therefore, it can find larger cutters than the ones found by algorithms that are based on alternative definitions of feasibility (e.g., either based on covering every bottleneck in the target region or existence of continuous path between every pair of points in the target region).

The maximal cutter that we have found is based on the geometric constrains. In actual machining, we will need to consider several other cutting constrains. Here are some examples:

- In facing operations, there is no benefit of using a bigger cutter than the three time of the width of the target region.
- There are several cutting parameters that may influence the selection of cutter size. We cannot use cutters that will conflict with the machining constraints. For example, we know that the material removal rate is proportional to the diameter of the cuter in milling operations. As a result, if we use a bigger cutter, then we can have higher material removal rate, thus we can save the cutting time. On the other hand, the maximum power of a machine is constant. The required cutting power is proportional to the metal removal rate. Therefore the maximal diameter is actually limited by the maximum machine power.

We are currently extending our algorithm to perform cutter selection optimization by considering multiple cutters.

ACKNOWLEDGMENTS

This research has been supported by the NSF grants DMI9896255 and DMI9713718. Opinions expressed in this paper are those of authors and do not necessarily reflect opinion of the National Science Foundation.

REFERENCES

1. S. Arya, S. W. Cheng and D. M. Mount. Approximation algorithm for multiple-tool milling. Proc. Of the 14th Annual ACM Symposium on Computational Geometry, 1998, pp. 297-306.
2. K. Lee, T. J. Kim and S. E. Hong. Generation of toolpath with selection of proper tools for rough cutting process. Computer-Aided Design, Nov. 1994, vol(26) 822-831.
3. Bala, M and Chang, T C, Automatic cutter selection and optimal cutter-path generation for prismatic parts. International Journal of Production Research, 1991, 29(11), 2163-2176.
4. D. Veeramani, and, Y. S. Gau. Selection of an optimal set of cutting-tool sizes for 2.5D pocket machining. Computer-Aided Design, 1997, 29(12), 869-877.
5. Y. S. Lee, T. C. Chang. Using virtual boundaries for the planning and machining of protrusion free-form features. Computers in Industry 25(1994) 173-187.
6. Y. S. Lee, B. K. Choi and T. C. Chang. Cut distribution and cutter selection for sculptured surface cavity machining. International Journal of Production Research, 1993, 30(6), 1447-1470.
7. B. Mahadevan, L. Putta and S. Sarma. A feature free approach to tool selection and path planning in 3-axis rough cutting. Proceedings of First International Conference on Responsive Manufacturing, Nottingham, September 1997, pp.47-60.
8. D. Yang and Z. L. Han. Interference detection and optimal tool selection in 3-axis NC machining of free-form surface. Computer-Aided Design, 1999, Vol.31, pp.303-315.
9. Y. S. Lee and T. C. Chang. Automatic cutter selection for 5-axis sculptured surface machining. International Journal of Production Research, 1996, 34(4), 977-998.
10. S. Marshall. and, G. J. Griffiths. A survey of cutter-path construction techniques for milling machines. International Journal of Production Research, 1994, 32(12), 2861-2877.
11. C. F. You and C. H. Chu, An automatic path generation method of NC rough cut machining from solid models. Computers in Industry 1995, 26(1), 161-173.
12. H. Li, Z. Dong and G. W. Vicker. Optimal toolpath pattern identification for single island, sculptured part rough machining using fuzzy pattern analysis. Computer Aided Design, 1994, 26(11), 787-795.
13. Y. S. Lee and T. C. Chang. Application of computational geometry in optimization 2.5D and 3D NC surface machining. Computers in Industry, 1995, 26(1), 41-59.
14. Z. Dong, H. Li and G. W. Vicker. Optimal rough machining of sculptured parts on a CNC milling machine. Transactions of ASME Journal of Engineering for Industry, 1993, 115(64), 424-431.