BUCKETING STRATEGIES FOR EFFICIENT TRIANGLE DETECTION IN CAM-SYSTEM BASED ON FACETED MODELS

Gandjar Kiswanto[1], Rahmat Widyanto[2], and Priadhana Edi Kresnha[2]

[1]Department of Mechanical Engineering, Universitas Indonesia, Kampus Baru – UI, Depok, 16424, Indonesia
[2]Faculty of Computer Science, Universitas Indonesia, Kampus Baru – UI, Depok, 16424, Indonesia

E-mail: gandjar_kiswanto@eng.ui.ac.id

Abstract

Triangle's checking in 3D faceted models is an essential step in gouging detection and elimination process. To check whether model surface intersects the tool, every triangle in the model has to be checked. Unfortunately, this process takes much time. Since the computer can't see and doesn't know which triangles located under the current tool position, then the triangles all over the surface model should be checked. To reduce the time, region checking at every position of tool has to be detected. Triangles that have to be checked are only in that region. The way of creating the region can be done by bucketing. Bucketing process will make some buckets as representation of regions and each bucket will be filled with triangles that lie in the corresponding region. There are several bucketing methods. This paper will explain all methods which have been implemented in the research.

Keywords: *bucketing, faceted models, gouging, triangle's checking*

Abstrak

Pengecekan segitiga dalam model 3 dimensi berfaset merupakan tahapan penting dalam deteksi *gouging* dan proses eliminasi. Untuk mengecek apakah permukaan model bersinggungan dengan alat, setiap segitiga pada model harus diperiksa. Akan tetapi, proses ini membutuhkan waktu yang lama. Segitiga-segitiga pada seluruh permukaan model harus diperiksa karena komputer tidak dapat melihat dan tidak mengetahui segitiga mana yang terletak di bawah posisi alat saat ini. Untuk mengurangi konsumsi waktu, pengecekan wilayah pada setiap posisi alat harus dapat dideteksi. Segitiga yang harus diperiksa adalah hanya yang berada pada wilayah tersebut. Cara yang dapat dilakukan untuk menetapkan wilayah adalah dengan menggunakan *bucketing*. Proses *bucketing* akan membuat beberapa *bucket* sebagai representasi dari beberapa wilayah dan setiap *bucket* akan diisi dengan segittiga yang terletak pada wilayah terkait. Terdapat beberapa metode *bucketing*. *Paper* ini akan menjelaskan semua metode yang telah diimplementasikan di dalam riset.

Kata Kunci: *bucketing, faceted models, gouging, pengecekan segitiga*

1. Introduction

Triangle's checking is one of the crucial processes in CAM system. The tool path in the CAM System has to be free gouging, which means there's no interference occurs between tool model and surface model. To fulfill this requirement, every side of surface model mustn't contact the tool to avoid gouging. Gouging means the tool cuts the material more than what has been specified.

The surface model, 3D faceted model, consists of many triangles which arrange the surface. To check whether there's interference between the surface and the tool, computer has to check all the triangles toward the tool. Since the tool path is arranged with several cc-points, then the computer has to check all triangles at every cc-point. This process takes pretty much time and inefficient. More over if the diameter of the tool and the average area of triangles are small, while the size of the model is big.

In fact, not all the triangles need to be checked at a cc-point. Only triangles which lie in some certain regions need to be noticed. For human, these regions can easily be seen and determined by showing the model and place the tool at one of the cc-point. Unfortunately, computer can't do the same as human. The computer can only calculate several numbers with incredible speed, but it can't see and estimate the triangles under the tool projection. Therefore all triangles have to be checked.

To reduce the number of triangles to be checked at every cc-point, an approach has to be implemented. The triangles have to be localized based on its position on the surface. Localization process can be done by bucketing. The model is divided into several buckets, and each bucket consists of triangles which lie in that region.

In [1-4] bucketing is explained. But the process isn't well described. This paper will explain bucketing process in the detail, and some approach to conduct the task.

2. Methodology

Let M be a surface model. M consists of triangles $T = \{t_1, t_2, \dots, t_n\}$, where n is the number of triangles composed the model. M is divided into several regions $R = \{r_1, r_2, \dots, r_m\}$, where m is the number of the regions. Every triangle t_i is included into one or more region based on its location toward the surface. All triangles have to be included at least into one region and there's no triangle without region.

Buckets can be illustrated as a set of triangles. Triangle t_i is an element of bucket b_j if the position of the triangle t_i is in the bucket $b_j (t_i \in b_j)$. A triangle t_i is included into bucket b_j if there's some part of t_i or the entire triangles t_i lies on bucket b_j. Triangle t_i can be included into more than one bucket if its location is between more than one bucket, $t_i \in b_j \wedge t_i \in b_{j+1} \wedge \dots \wedge t_i \in b_m$. This means that every bucket possibly intersect with other bucket(s). Figure 1 shows the intersection of buckets.

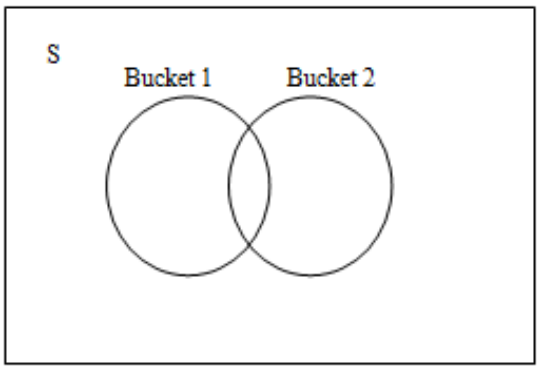

Fig. 1. Bucket 1 intersects bucket 2. There are triangles which included into both of bucket 1 and bucket 2.

Since all the triangles has to be included into at least one bucket, then the complement of union of all buckets is an empty set, $\overline{(\cup_{i=1}^{m} b_i)} = \{\ \}$ where m is the number of buckets.

Bucketing is conducted by projecting the surface model into 2D bucket coordinate. The area of the surface model projection is in accordance with the area of bounding box (figure 2). The projection is then divided into several rectangles denote the buckets, where each bucket has an information about triangles which belong to that bucket (figure 3). The number of bucket is determined based on bucket size (width and height) which is input by the user, and relative to the xy area of model's bounding box. The bigger size of the bucket, the more triangles belong to that bucket.

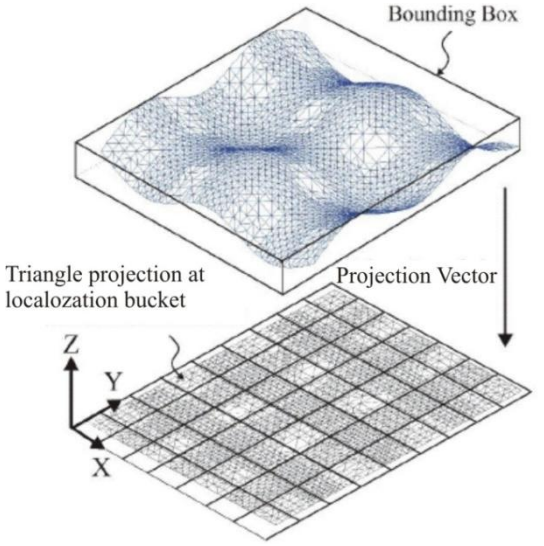

Fig.2. Faceted model 3D is projected and localized in xy plane [3].

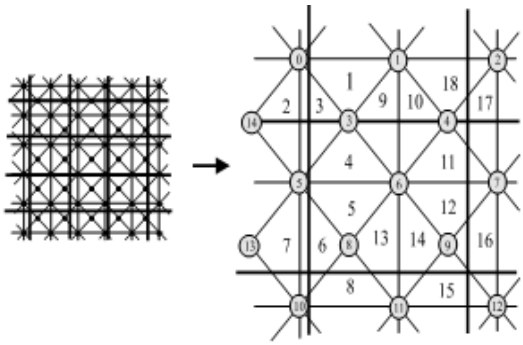

Fig. 3. Triangles in the bucket [3].

The number of buckets in the coordinate x and y can be calculated by the following equation.

$$n\_x = x\_max/bucket\_width$$
$$n\_y = y\_max/bucket\_height \quad (1)$$

Where *n_x* is number of bucket in *x* coordinate, *n_y* is number of bucket in *y* coordinate, *x_max* is the length of bounding box in *x* coordinate, and *y_max* is the length of bounding box in *y* coordinate. To check a region, one has to detect

the position of the bucket in that region. Since bucket is a 2D vector, the position of the bucket means the index of the bucket itself. It can be obtained by calculating *xy* location of the point in that region.

$$i = x / bucket\_width$$
$$j = y / bucket\_height \quad (2)$$

Where *i* is *x* index of the bucket, *j* is *y* index of the bucket and *xy* is the location of the point which region is checked.

By implementing bucketing, the process of triangle checking in the series of gouging detection and elimination process can be fastened and done efficiently, since not all triangles are checked. Only triangles in the buckets that are intersected with tool projection are considered (figure 4).

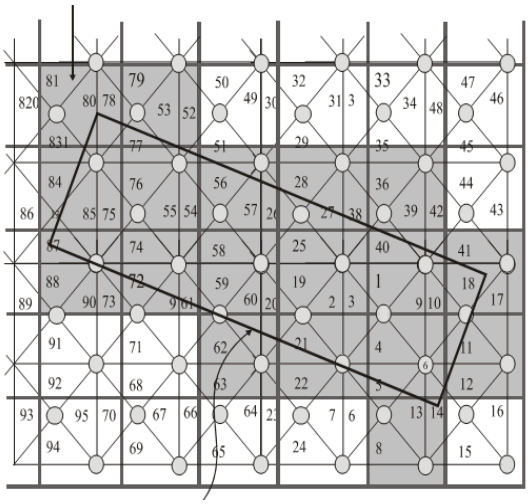

Fig. 4. Buckets which interference with tool projection (in gray color) [5].

3. Result and Analysis

Every triangle must be checked and included into one or more buckets. The problem is how to detect which buckets that the triangle belongs. This can be illustrated in figure 5. There are several methods that have been studied to implement bucketing process. They are polygon scanning method, point spreading method, and whole rectangle area method. Each method is explained in the following subsection.

In Polygon scanning method, there are 2 steps that have to be accomplished. First is detecting the side of the triangle edge in bucket coordinate, the second is getting all buckets which lie in the triangle.

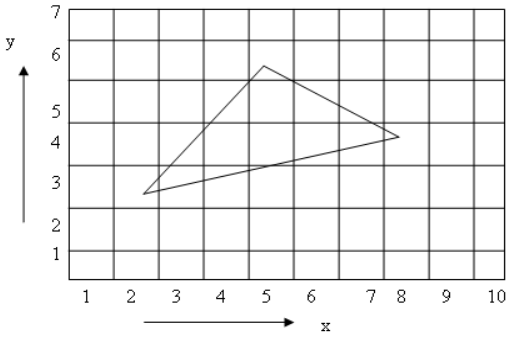

Fig. 5. Ilustration of a triangle belongs to several buckets.

There are several ways to detect triangle edges in bucket coordinate. Figure 5 shows a triangle in bucket coordinate. To get bucket that is located at triangle side, one has to obtain the bucket where the vertices of triangle lie. In figure 5, triangle vertices located at bucket (2,3), (5,6), and (8,4). Next the buckets along the obtained buckets are searched, which are buckets between bucket (2,3) and (5,6), buckets between (5,6) and (8,4), and buckets between (8,4) and (2,3). The following line formula can be used to get those buckets

$$y = mx + c \quad (3)$$

By incrementing x from x_1 to x_2, (x,y) indexes bucket coordinate between (x_1,y_1) and (x_2,y_2) can be obtained. Since the index coordinate can only handle non-floating point number, then the form of coordinate obtained has to be (x,round(y)) where round(y) is calculated by

$$Round\ (y) = \lfloor y + 0{,}5 \rfloor \quad (4)$$

However, equation 3 and 4 use many floating point operations, while the values needed are only natural number, i.e. integer. Another method to avoid floating point operations is midpoint algorithm which is proposed by Jack E. Bresenham (figure 6) [6].

After obtaining all buckets lying on triangle's edges, next is searching for buckets that located inside the triangle. This is conducted by getting the maximum and minimum value of x and y bucket index of triangle vertices. In figure 5, using triangle vertices which located in bucket (2,3), (5,6), and (8,4), those maximum and minimum xy can be found, which are 2 (minimum x), 8 (maximum x), 3 (minimum y), 6 (maximum y).

By combining minimum and maximum xy using Cartesian product, vertices of rectangle buckets area that the triangle lies in are known. Using the same example, the vertices of rectangle

are (2,3), (2,6), (8,6), and (8,3). The checking triangle is considered as polygon where the vertices of triangle (in bucket index coordinate) are the polygon vertices. The process of gaining the buckets inside the triangle is conducted by scanning the polygon from minimum xy until maximum xy (from bucket index (2,3) until bucket index (8,6)). Those buckets that located inside the polygon must include the triangle that is being checked.

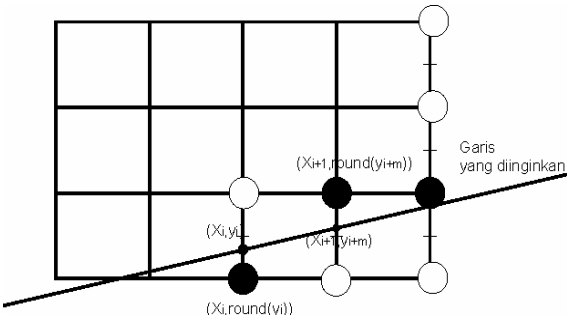

Fig. 6. Pixel grid of midpoint algorithm.

However, this method is ineffective since sometimes there are buckets which must contain a triangle that being checked, but using the method described they don't save it. If the size of triangle is smaller than bucket, this incident wouldn't emerge. But if the triangle's size is bigger than the bucket, this often happens. I.e. in figure 7 the triangle's vertices in coordinate bucket are (1,1), (3,8), (8,2), then using bresenham algorithm, the buckets along the position of (3,8) and (8,2) are searched. The buckets detected using the algorithm are {(3,8), (4,7), (5,6), (5,5), (6,4), (7,3), (8,2)}. Meanwhile the fact shows that the buckets which affiliates with the triangle are {(3,8),(4,7),(4,6),(5,6),(6,5),(6,4),(7,4),(7,3),(7,2), (8,2)}. There are some buckets that should be detected as intersected with triangle's edge, but in fact they aren't detected (index (4,6), (6,5), (7,4), and (7,2)). It appears to be line bresenham algorithm is not appropriate to be used for bucketing process.

Point Spreading Method uses points that are spread all over the bucket with a constant distance to check whether a triangle is included into the bucket or not (figure 8). The number of points is determined by the system. The more point spread, the more accurate the result will be, but the computational time will be longer. System developer has to decide how many points needed and how much is the distance between points. The step of getting all buckets detected inside the triangle is similar with previous method. By obtaining maximum and minimum xy values based on triangle vertices in bucket coordinate, the bucket is then scanned from maximum xy to minimum xy. At each scanned bucket, the points are spread to detect if the triangle belongs to the checked bucket. If one point is found to be inside the triangle, then the triangle is included into the bucket.

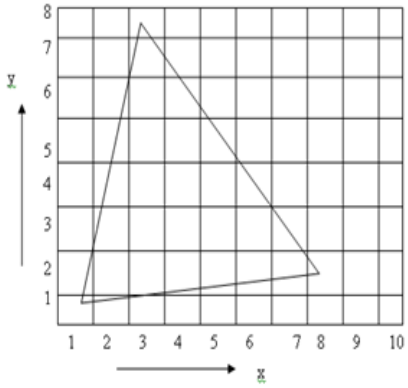

Fig. 7. Triangle size is bigger than bucket size. Line bresenham algorithm is not effective to handle this case.

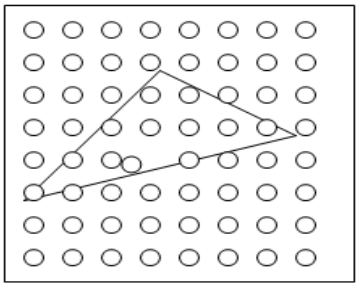

Fig. 8. Points spreading method. If one point is detected inside the triangle, then the triangle is included into the related bucket.

The weakness of this method is if the triangle checked is not inside the bucket, the computational time will reach its worse case which is $O(n^2)$ where n is the number of points in x element or y element. Moreover this method is not really accurate to predict a small triangle, smaller than the distance between points that makes the point can't predict the triangle. Another weakness is the maximum number of triangles inside the bucket that can be detected is the same as the number of the points, where each point detects one triangle. It the system spreads 625 points all over a bucket (25×25 points), then the maximum number of small triangles detected will be 625. If there's small triangle between points but it doesn't cover the point, then the triangle is not detected to be inside of the bucket being checked.

Whole Rectangle Area Method is similar with two previous methods, this method collecting the predicted buckets using minimum and

maximum xy values of triangle vertices in bucket coordinate. But instead of scanning the polygon, a triangle is included in all predicted buckets. This method is obviously faster then the first and the second method since it doesn't need to scan the polygon area of the rectangle.

Figure 9 shows the area (in thick line) where the triangle should be included in. The buckets which contain the triangle are {(3,2), (4,2), (5,2), (6,2), (3,3), … , (6,6)}. Although some bucket don't even touch the triangle like bucket (3,2), (6,4), etc, since it is in the predicting area, they have to insert the triangle into their group.

The weakness of this method is obvious. There are many triangles included into the buckets where they shouldn't belong to. Even though the process of bucketing is faster than the other method, there would be a lot of wasting time when searching a triangle in some area, since many triangles that are out of checking area is checked as well as they are inside that region. However, this method is the fastest and most effective among several methods explained, since it doesn't let any triangles that a bucket should have are missed.

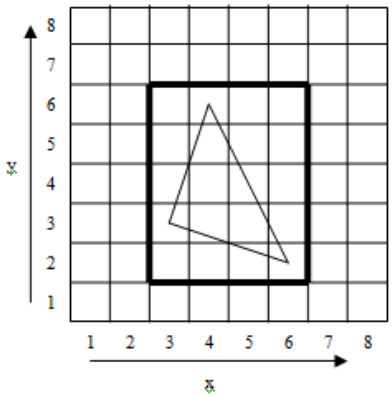

Fig. 9. Whole rectangle area method. Thick line is the region of buckets that are belonged the triangle.

4. Conclusion

Triangle checking is an important part of gouging detection and elimination in CAM system. To check if the tool model intersects the surface model at a cc-point, every triangle on the surface model should be checked. If one of the triangles intersects the tool, then the tool should be lifted or rotated to avoid gouging. But the process of checking all triangles takes much time. Instead of checking triangles all over the surface model, region checking is preferable. This makes the computational cost for checking greatly reduced. Only regions which located in the tool projection are checked. To provide this ability, bucketing is implemented. This is conducted by creating several buckets to localize triangles based on their location.

There are three methods that have been studied for this process. They are Polygon Scanning Method, Points Spreading Method, and Whole Rectangle Area Method. All methods have been implemented, and each method has its advantages and weaknesses. Based on the logic that every method has, Whole Rectangle Area Method is the most applicable for the CAM-system due to its speed and reliability in collecting triangles into buckets.

Reference

[1] G. Kiswanto, Pengembangan & Pembuatan Sistem CAM (Computer Aided Manufacturing) yang Handal Berbasis Model Faset 3D untuk Pemesinan Multi-axis dengan Optimasi Orientasi Pahat dan Segmentasi Area dan Arah Pemesinan, Laporan Kemajuan RUT XII Tahap II, Indonesia, 2005.

[2] G. Kiswanto & P.E. Kresnha, "Triangles Bucketing and Running Bucket Simulation for Possible Gouging Area Verification in CAM System Based on Faceted Model" *In Proceeding in National Conference on Technology In Simulation TechnoSim 2006*, 2006.

[3] P.E. Kresnha, Gouging Elimination for 5 Axis Machining Based on Faceted Model In the Developing CAM System, Practice Work report, Faculty of Computer Science, Universitas Indonesia, 2007.

[4] P.E. Kresnha, Comparison of Curvature Estimation Method in Determining Shape of the Plane, Final Work report, Faculty of Computer Science, Universitas Indonesia, 2007.

[5] B. Choi & R.B. Jerard, *Sculptured Surface Machining*, Kluwer Academic Publishers, Dordrecht, 1998.

[6] J.E. Bresenham, "Algorithm for Computer Control of a Digital Plotter," *IBM Systems Journal*, vol.4, pp. 25-30, 1965.