Jurnal Teknologi, 56 (Sains & Kej.), Sept. 2011: 65-82
© Penerbit UTM Press, Universiti Teknologi Malaysia

MESH SIMPLIFICATION – A NEW APPROACH

SITI KHADIJAH ALI[1*], RAHMITA WIRZA O. K. RAHMAT[2], FATIMAH KHALID[3] & NG SENG BENG [4]

Abstract. Mesh simplification has become an interesting research area over the past few decades. Some researchers have introduced various new methods while others have made some modifications or improvements to the existing method. However, the modified model tended to lose some of its original shape. Hence, to solve these arising problems, a new decimation algorithm is introduced, in which rational equation is used to select elements as candidates to be removed. The proposed algorithm also uses triangular mesh as the element. The results show that our proposed algorithm can preserve the shape of the mesh.

Keywords: Mesh simplification; rational equation; manifold; triangle contraction; mesh decimation

Abstrak. Pemudahan permukaan telah menjadi topik kajian yang menarik dalam beberapa dekad ini. Sesetengah penyelidik memperkenalkan kaedah yang baharu manakala penyelidik lain melakukan pengubahsuaian ataupun penambahbaikan terhadap kaedah yang sedia ada. Walau bagaimanapun, model yang telah dipermudahkan mempunyai kecenderungan untuk kehilangan sebahagian daripada bentuk asalnya. Dalam pada itu, untuk mengatasi segala masalah ini, satu algoritma baru diperkenalkan, di mana formula nisbah digunakan untuk mengenal pasti calon elemen yang layak untuk dibuang. Elemen yang digunakan dalam algoritma baru ini adalah segitiga. Keputusan kajian menunjukkan bahawa algoritma yang dicadangkan dapat mengekalkan bentuk permukaan model asal.

Kata kunci: Permudahan permukaan; persamaan rasional; berjenis; segitiga kontraksi; penipisan permukaan

1.0 INTRODUCTION

Nowadays, there are tools or devices such as range scanner and CAD tools that generate a millions of polygons (triangles/rectangles) when a

[1-4] Department of Multimedia, Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, 43400 Serdang , Selangor

* Corresponding author: sitikhadijahali.stud@gmail.com

model is constructed. Any model developed from such numbers of polygons does not fit in the rendering process. This is because the speed of the rendering process and memory requirements are proportional to the number of polygons [2],[3],[4]. In addition, a model consists of many triangles, in which could be problematic when transferring the model through a network [1]. Due to these problems, researchers found that there is a need in mesh simplification to overcome these problems.

In general, mesh simplification can be defined as a reduction of vertices or elements in an initial mesh without distorting the accuracy and the features of the mesh. Mesh simplification can be classified into two types: mesh decimation and mesh refinement. In mesh decimation, the points that are not important will be discarded, whereas in mesh refinement, the points are added until they meet a termination condition. Mesh decimation is applied in the finest resolution while mesh refinement is practical in a coarse model. However, mesh refinement would need more memory space due to the adding point's process that occurs in mesh refinement not in mesh decimation.

Surface or mesh decimation can be driven by two different purposes. Firstly, mesh simplification produces a more compact mesh, which is sufficiently similar in terms of visual appearance, while the second objective is, to produce a model, which satisfies a given accuracy [5]. According to Youngihn K *et al.* [12] many automatic algorithms have been developed to simplify complex models. Even though, they can produce plausible results in many cases, their simplifications do not preserve the visual appearance of the original model very well. Therefore, a new algorithm, which is based on mesh decimation, is proposed to simplify a complex model without sacrificing the accuracy.

The purpose of this work is to develop a new algorithm for triangular mesh decimation. The proposed algorithm can also identify elements that can be deleted to minimize polygons or triangles without sacrificing the system's accuracy, thus enhancing the method to match the remaining points. Nevertheless, the proposed algorithm is also able to display a good visual representation of the geometric model.

The paper is organized as follows. In section 2, the related works are presented. Then, in section 3, the proposed simplification algorithm is presented. In section 4, the results are shown. The last section, which is section 5, is the conclusion.

2.0 RELATED WORKS

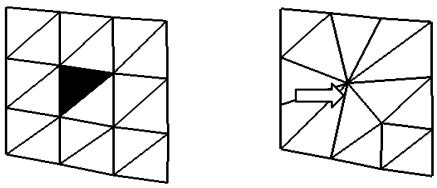
Geometry removal is one of the significant ways to simplify meshes and its operation is to iteratively remove or collapse the basic geometry unit, such as the vertex, the edge or a triangle face. A number of surface simplification algorithms in this area have been developed. Two classes of approach are most relevant to this work.

The first approach is the vertex decimation approach [1],[2],[4],[8]. This approach works by deleting a vertex and its surrounding regions. Then, the resulting hole is re-triangulated [1],[2],[4]. Different researches use the same approach but with different ways to select the suitable vertex to be removed. For example, Schroeder [4] used the distance from a vertex to the average plane, whereas Bostjan Pivec [1] used the average distance from a vertex to the average plane.

The second approach is the edge collapse approach [6],[7],[5]. The idea of this approach is collapsing an edge into a new vertex. Then, automatically, the two adjacent triangles are deleted. Many previous researchers used this approach but they used different methods to select the suitable edge to be collapsed. Some of them used the least curvature to identify a suitable edge [7], whereas others used the length of the edge multiplied by a curvature term [6]. This approach was chosen in this proposed algorithm. However, in our proposed algorithm, we collapsed the triangle mesh instead of collapsing the edges.

3.0 RESEARCH METHODOLOGY

This work applies an iterative simplification approach based on triangles collapsed as depicted in Figure 1. There are many ways to simplify the thousands of meshes such as vertex decimation, edge collapsed, triangle decimation, appearance-preserving simplification, and vertex clustering and shape approximation.



(a) Original Mesh (b) Simplified Mesh

Figure 1 Concept of triangle decimation

In the triangle collapsed approach, the triangle with the least value of error metric is selected first and collapsed. The decision to select the triangle is a key point in this algorithm. Rational equation involving the height from the center of gravity of the big triangle to the area of the big triangle is used to determine the triangle that can be a candidate to be eliminated. The flow of the work is shown in Figure 2.

The work starts by taking triangular mesh as an input. Then, neighbor for each mesh and vertices is determined and big triangle is formed then, the center of gravity (COG) for each big triangle is calculated. Next, the area and the distance between COG to big triangle is computed. After that, the ratio is calculated. Then, the meshes are arranged according to the ratio's values in ascending order. Planar test is applied onto selected mesh. If the planar test is pass, then the selected mesh is collapsed. The details on how to collapse the triangle mesh is in section 3.4. Then, boundary test and self-intersection preservation test are applied. If the test is pass. Then the selected mesh is confirmed as 'accepted' and the vertices and meshes positions are updated. The explanation for planar test, boundary test and self-intersection preservation test are in section 3.5, 3.6 and 3.7.

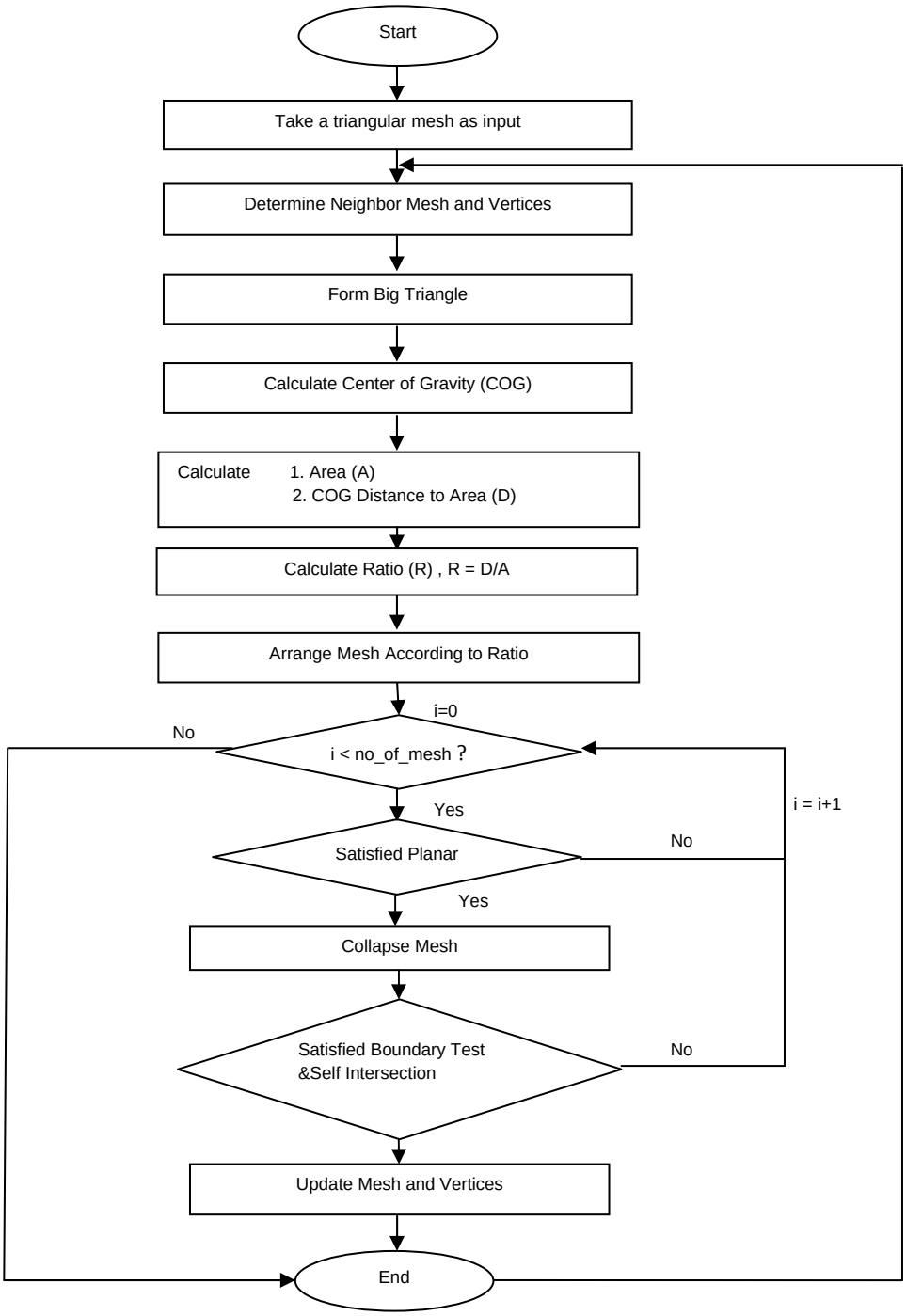


Figure 2 Flowchart

A new term is proposed in this work: "big triangle". Big triangle is a triangle that consists of complete neighbors as in Figure 3.

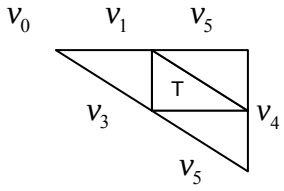


Figure 3 Big triangle

The Center of Gravity (COG) for each big triangle plays an important role in this work because it will determine the center for all vertices in the big triangle as in Figure 4. Let v_0, v_1, v_2, v_3, v_4 and v_5 as the vertices on the big triangle. For each v_i where $i = 0,1,2,3,4,5$ there is x_j, y_j, z_j where $j = 0,1,2,3,4$ and $i = j$. The calculations for COG are as below:

$$cog(x) = \frac{x_0 + x_1 + x_2 + x_3 + x_4 + x_5}{6},\ cog(y) = \frac{y_0 + y_1 + y_2 + y_3 + y_4 + y_5}{6},\ cog(z) = \frac{z_0 + z_1 + z_2 + z_3 + z_4 + z_5}{6} \quad (1)$$

Therefore, the center of gravity for each big triangle is $COG = (cog(x), cog(y), cog(z))$.

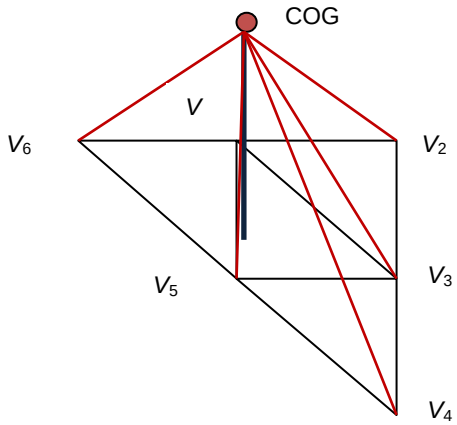


Figure 4 Center of gravity (COG) (Top view)

There are three steps in order to get the value for rational equation. These steps will be explained in section 3.1, 3.2 and 3.3.

3.1 Calculation of Area for Big Triangle

The triangles occurring in the 3D face models from the database are not right-angle triangles. Due to this reason, the basic area formula cannot be used in this experiment. As a solution, sine rule and cosine rules are applied. Figure 5, shows an example of not right-angle triangle. The variables a, b and c show the length of each edge of the triangle while the variables α, β and γ show the angles of the triangle. The values for a, b and c are known. As the values for a, b and c are known, it is easy to apply cosine rule.

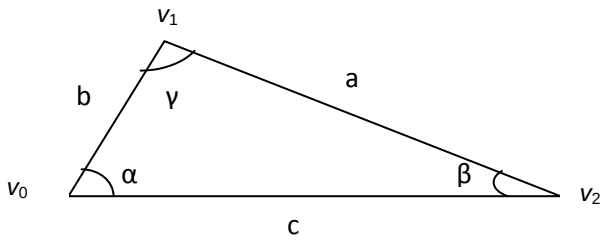


Figure 5 Example of not right-angle triangle

Cosine rule can be stated as follow:

$$a^2 = b^2 + c^2 - 2bc\ \cos\alpha \tag{2}$$

As show in Figure 5, there are three sides of lengths a, b and c and angles α, β and γ. The calculations of a, b and c are as follows:

$$|a| = |v_2 - v_1| = \left|\sqrt{(x_2, y_2, z_2) - (x_1, y_1, z_1)}\right| \tag{3}$$

$$|b| = |v_1 - v_0| = \left|\sqrt{(x_1, y_1, z_1) - (x_0, y_0, z_0)}\right| \tag{4}$$

$$|c| = |v_0 - v_2| = \left|\sqrt{(x_0, y_0, z_0) - (x_2, y_2, z_2)}\right| \tag{5}$$

Then, the values of a, b and c are substituted into equation (2). Equation (2) will become as follows:

$$\alpha = \cos^{-1}\left(\frac{b^2 + c^2 - a^2}{2bc}\right) \tag{6}$$

As mentioned above, most of the triangles in this work are not right-angle triangles. Due to this problem, sine rule is applied to find the area of the big triangle. The calculation is as below:

$$Area_of_big\ triangle = \frac{bc\sin(\alpha)}{2} \tag{7}$$

3.2 Calculation of Distance from COG to the Big Triangle

The formula of distance from point to plane is adopted in order to get the distance from the center of gravity to the big triangle (Figure 6). First, for each big triangle, there is a need to find the equation of a plane. The equation for a plane is $Ax + By + Cz + D = 0$ and $A^2 + B^2 + C^2 = 1$. The value of A, B, C and D can be calculated as follows:

$$\begin{aligned} A &= y_0(z_1 - z_2) + y_1(z_2 - z_0) + y_2(z_0 - z_1) \\ B &= z_0(x_1 - x_2) + z_1(x_2 - x_0) + z_2(x_0 - x_1) \\ C &= x_0(y_1 - y_2) + x_1(y_2 - y_0) + x_2(y_0 - y_1) \\ D &= -(x_0(y_1 z_2 - y_2 z_1) + x_1(y_2 z_0 - y_0 z_2) + x_2(y_0 z_1 - y_1 z_0)) \end{aligned} \tag{8}$$

Let (x_i, y_i, z_i) as the coordinate of COG. Then, the formula to calculate the distance is

$$\text{distance_COG_big_triangle} = \frac{|A * cog(x) + B * cog(y) + C * cog(z) + D|}{\sqrt{A^2 + B^2 + C^2}} \tag{9}$$

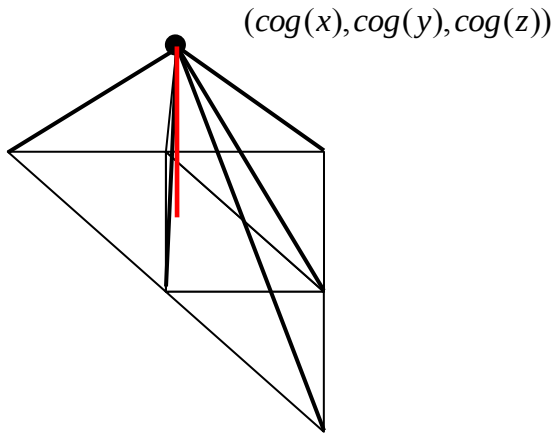


Figure 6 The red line shows the distance from COG to the big triangle

3.3 Calculation of Rational Equation

The final step is to calculate the rational equation. After getting the distance values from COG to the big triangle and area of big triangle, now the ratio between these two values can be calculated. The formula is as below:

$$Rational_equation = \frac{distance\ COG_bigtriangle}{Area\ of\ \ big\ \ triangle} \tag{10}$$

Then, all of the big triangles are ordered according to the computed ratios with the minimum-ratio triangle at the top.

3.4 Triangle Collapse

One hole is formed after the triangle is removed. Due to this problem there is a need to fill the hole. There are few ways to refill the hole. One is by applying the triangle-collapsing process, which is used in this work. This process stretches the three edges that formed the deleted triangle to form a new vertex. There are three ways to determine the new point. The first is by selecting one of the two endpoints. The second is by choosing the optimal point. The third is by picking

the average of the three points that form the removal triangle. In this project, the third choice is chosen and the formula is as follows:

$$x_{np} = \frac{(x_0 + x_1 + x_2)}{3} \qquad y_{np} = \frac{(y_0 + y_1 + y_2)}{3} \qquad z_{np} = \frac{(z_0 + z_1 + z_2)}{3} \tag{11}$$

Therefore, the coordinate for the new vertex is: (x_{np}, y_{np}, z_{np}).

After the triangle is removed and collapsed, we apply a planar test to ensure that the new triangular mesh is located on the planar surface. The details for this test are explained in the next section.

3.5 Planar Test

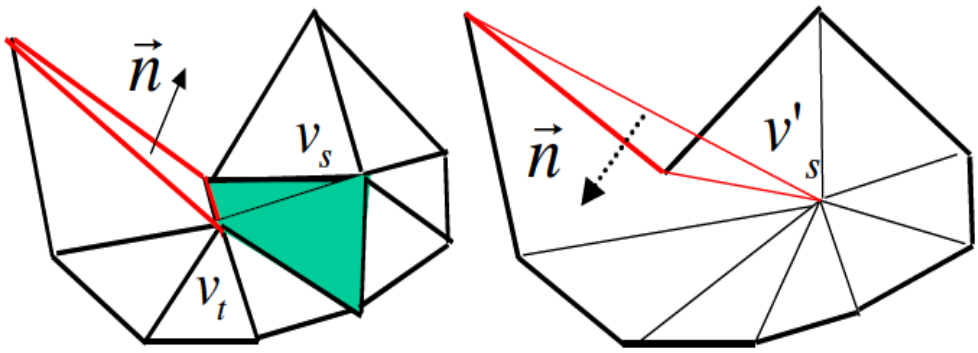
A planar test is used to ensure that the triangular mesh is located on the planar surface. In this final step, we will check the angle from the new triangular mesh to the triangles surrounding it. As the requirement of this test if the angle is greater than 165°, then the triangular mesh is considered to be located on the planar surface and can be deleted: if the angle is less than 165°, the triangle mesh will be kept and the next candidate will be check. Then, the boundary preservation test is applied to the selected mesh.

3.6 Boundary Preservation

In this work, only original model with open surface is considered and in order to maintain the shape of the original model, there is a need to preserve the boundaries of the object. For this case, the entire triangle that is a candidate to be eliminated must not be formed from the boundary vertices. The boundary vertices are detected by checking the numbers of the neighbors for each mesh. If the number for each mesh is less than 3, then this mesh is considered as located at the boundary whereas if the number for each mesh is 3, then the mesh is not at the boundary vertices. Then, the self-intersection preservation test is applied on the selected mesh.

3.7 Self-Intersection Preservation

Self-intersection occurs when there is an intersection involving one triangle with the interior of another triangle (Figure 7). This situation must be avoided. In this works, the same method as in [8] is applied to check the self-intersecting problem. In order to avoid this condition, the normal of each neighboring mesh is compared before and after the contraction. If the normal changes too much, then the mesh is not deleted.



(a) Before collapse (b) After collapse

Figure 7 Triangle mesh with self – intersecting[13]

Based on the discussion above, the algorithm removed four triangles mesh and two points at each step.

4.0 RESULTS AND DISCUSSION

The proposed algorithm was implemented on Visual C++ by using OpenGL library. We tested our algorithm onto bivariate function [10],[11]. At first, we fixed the x_a and y_b values in the range of 0.0 until 1.0 where a and b are in the range from 0.0 until 1.0. The range for x_a and y_b can be written as follows:

$$((x_a, y_b) = (0+a, 0+b) \mid a, b = 0.0, 0.1, 0.2, 0.3, \ldots\ldots 1.0)$$

For the z values, we applied the bivariate function and substituted the x and y values into the function. Then the graph or function can be defined as follows:

$$((x_a, y_b, f(x_{a,} y_b))^T \mid a, b = 0.1, 0.2, 0.3.....,1.0)$$

Three types of functions are chosen for the purpose of validation. These three functions are cosines, exponential and bump in which the sine and cosines functions are combined. These functions were chosen due to their capability to provide the sinusoidal exponential shape. This characteristic is important in order to ensure that our algorithm can preserve the shape of the original graphs functions.

4.1 Cosine Function

For the cosine function, let $f(x_a, y_b) = \dfrac{1.25 + \cos(5.4y)}{6 + 6(3x-1)^2}$. In this function, in order to make sure that the fraction is 'not undefined' which means the denominator is not zero, the quadratic equation is applied and then added with any numbers, where for this experiment the number '6' is chosen. The original shape of this function is shown in Figure 8. In the Table 1, the results are shown.

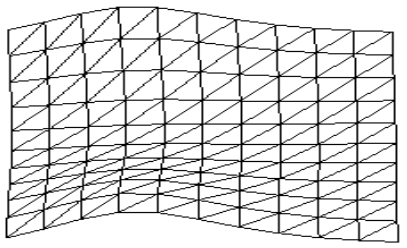


Figure 8 Cosine function

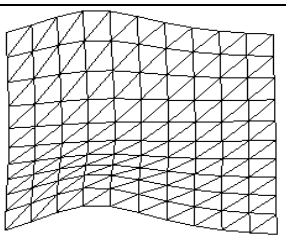
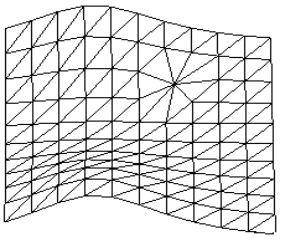
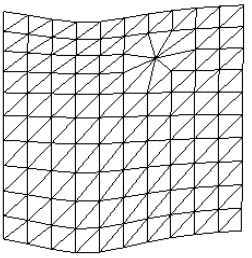
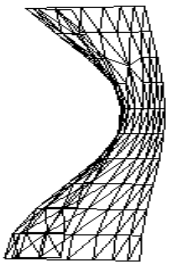
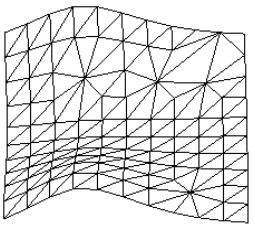
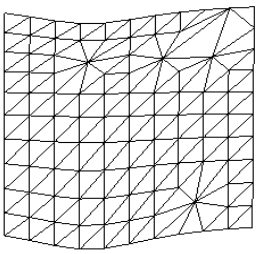
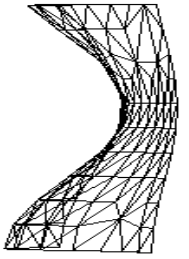
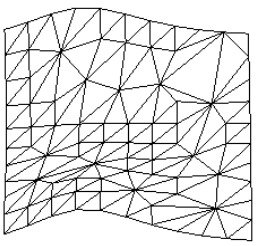
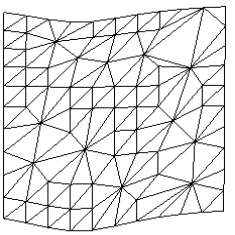
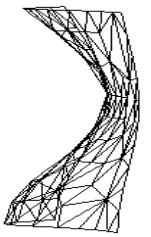
			
Triangles : 196 Vertices : 119			
Triangles : 180 Vertices : 111			
Triangles : 142 Vertices : 91			

Table 1 Result for cosine function

4.2 Exponential Function

The second function is the exponential function. For this function, let $f(x_{a,}y_b) = \exp\frac{((\frac{-81}{4})((x-0.5)^2+(y-0.5)^2))}{3}$. For this function, make sure that the fraction part is 'not undefined' which means the denominator is not zero. The original shape for this function is show in the Figure 9. The result for this function is shown in Table 2.

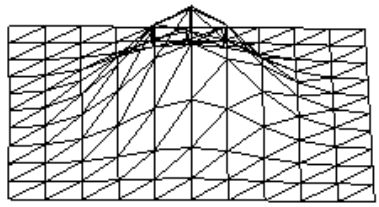
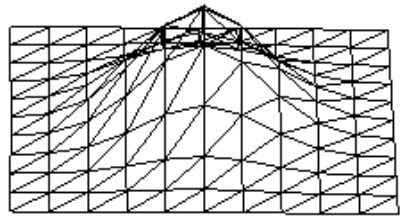
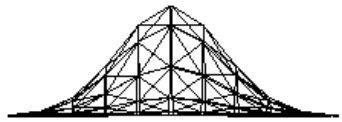
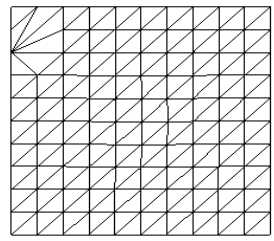


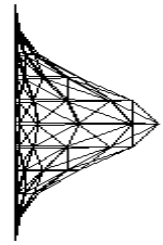
Figure 9 Exponential function



Triangles : 196
Vertices : 119







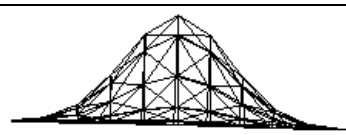
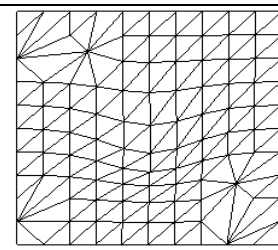
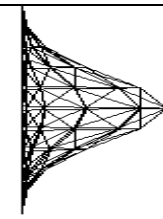
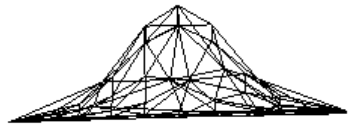
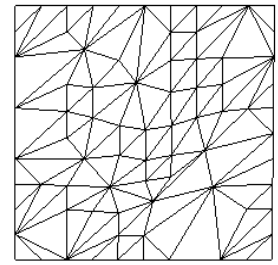
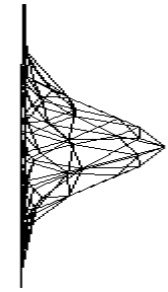
Triangles : 180 Vertices : 111			
Triangles : 142 Vertices : 91			

Table 2 Result for exponential function

4.3 Bump Function

The final function that tests the proposed algorithm is Bump. Bump is actually the combination between cosine and sine function. The original shape of Bump is shown in Figure 10. We let $f(x_a, y_b) = \frac{\sin(5x) * \cos(5y)}{5}$. The denominator is fixed to be 5 in order to avoid an undefined value. In Table 3, the result after the simplifying process is shown.

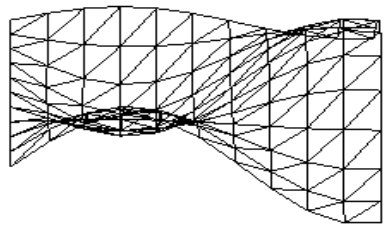


Figure 10 Bump function

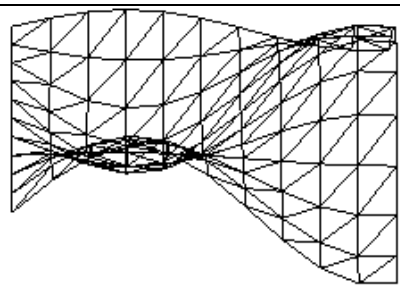
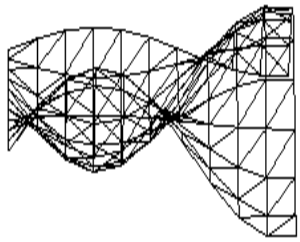
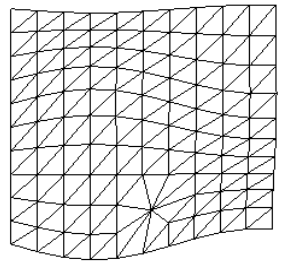
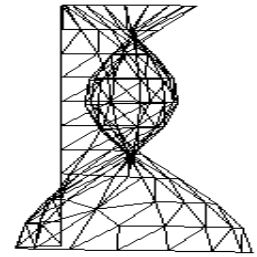
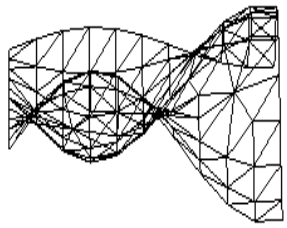
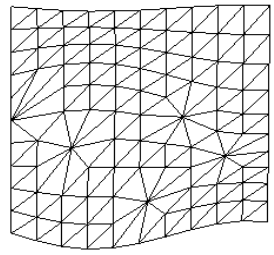
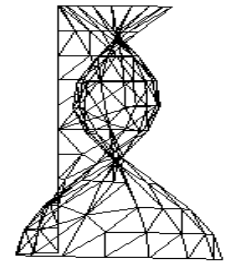
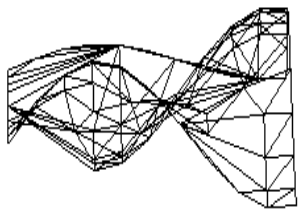
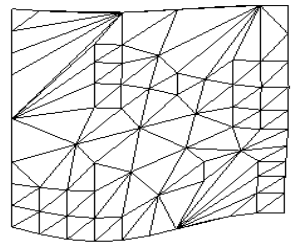
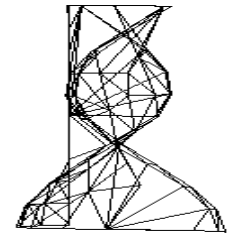
			
Triangles : 196 Vertices : 119			
Triangles : 180 Vertices : 111			
Triangles : 142 Vertices : 91			

Table 3 Results for bump

The results are presented in Tables 1, 2 and 3. The results from these three figures show that our algorithm can preserve the shape of the original graph's

functions. From these three figures, it is much easier to show that the proposed algorithm starts to reduce at the planar surface. It goes to cosine function and Bumps. This happened because a planarity test is used to ensure that the deleted triangular mesh is located on the planar surface.

5.0 CONCLUSION

A new algorithm for the simplification of triangular meshes is presented in this paper. The simplification of the mesh is based on a triangular-collapsed algorithm. Based on the results above, it is proven that the algorithm can preserve the shape of the original graphs functions.

In future, the algorithm will be apply onto human faces to test the accuracy of this work. Futhermore, we will find another method of collapsing the removed vertex in order to achieve optimal results.

REFERENCES

[1] Pivec, B., & V. Domiter. 2007. A General Simplification Algorithm. *International Journal of Computers*. 1(4): 307–311.

[2] Park, I. , S. Shirani & D. W. Capson. 2005. Area of Surface as a Basis for Vertex removal based Mesh Simplification. *IEEE International Conference on Multimedia and Expo*. 273-276.

[3] Hussain, S., H. Grahn & J. Persson. 2008. Feature-preserving Mesh Simplification: A Vertex Covers Approach. *Proc. of the IADIS International Conference on Computer Graphics and Visualization 2008 (CVG 2008)*. 270-275.

[4] Schroeder, W. J, J. A. Zarge & W E. Lorensen. 1992. Decimation of Triangle Meshes. *Proc. of the 19th Annual Conference on Computer Graphics and Interactive Techniques*. 26(2): 65-70.

[5] Cignoni, P., D. Costanza, C. Montani, C. Rocchini, & R. Scopigno. 2000. Simplification of Tetrahedral Meshes with Accurate Error Evaluation. *Proc. of the 11th IEEE Visualization Conference* 2000: 85-92.

[6] Xianfeng, W. & Y. Jianhui. 2003. A New 3-D Mesh Simplification Algorithm. *Colombian Journal of Computation*. 4: 65-73.

[7] G. Jianlin, Z. Meng Chu, & W. Haimin. 2000. Mesh Simplification with Average Planes for 3-D Image. *IEEE International Conference*. 2: 1412-1417.

[8] Garlang, M. & P. S. Heckbert. 1997. Surface Simplification using Quadric Error Metrics. Computer Graphics. *Proc. Annual Conference Series (Siggraph'97)*. 31(3): 209-216.

[9] Gieng, T. S, B. Hamann, K. I. Joy, G. L. Schussman & I. J. Trotts. 1998. Constructing Hierarchies for Triangle Meshes. *IEEE TRANSACTIONS ON VISUALIZATION AND COMPUTER GRAPHICS*. 4(2): 145-160.

[10] Hamann, B. 1994. A Reduction Scheme for Triangulated Surfaces. *Computer Aided Geometric Design*. 11(2): 197-214.

[11] K3dsurf.sourceforge.net/

[12] K. Youngihn & G. Micheal. 2003 . User-Guided Simplification. *Proceedings of the 2003 Symposium on Interactive 3D graphics SI3D 03* :

[13] J. Wang, L. Wang, J. Li, I. Hagiwara. 2011. A Feature Preserved Mesh Simplification Algorithm. *Journal of Engineering and Computer Innovations*. 2(6): 98-105.