LAB REPORT: LAB 2

TNM079, MODELING AND ANIMATION

Lucas Hägg
lucha567@student.liu.se

Thursday 9th May, 2024

Abstract

In computer graphics, using highly detailed 3D models makes everything look more realistic, but it also requires a lot of computing power to render. To solve this, a technique called quadric-based mesh decimation was used, which simplifies the complexity of these models while still keeping their overall look. This method particularly focuses on reducing the number of edges in a model, using something called quadric error metrics. It relies on the half-edge data structure to make these reductions efficiently by calculating errors from the surrounding faces of each edge. This way, the number of polygons in a model can be decreased while keeping its important shapes and details. The results show that this technique works well in keeping the quality high in models with reduced resolution, making it really useful for applications that need to render graphics in real-time.

1 Introduction

In computer graphics, reducing the complexity of 3D models, or mesh decimation, is essential for keeping up performance while still keeping good visual quality. This lab is all about learning how to simplify models using a technique based on Quadric Error Metrics (QEM), introduced by Garland and Heckbert. This method is chosen because it efficiently lowers the number of polygons in detailed models, which helps balance the need for computational speed with the desire to keep models looking good. The lab focuses on applying this method to see how well it can maintain the original shapes and details of the models during the simplification process.

2 Background

This section provides a description of the tasks that were implemented during the lab. The main focus was on implementing a mesh decimation technique using quadric error metrics, creating quadrics for vertices and faces, as well as computing optimal collapse conditions for edges.

2.1 Creating Quadrics for Vertices and Faces

First was the creating of quadrics for each face of the mesh. A quadric for a face is calculated using the formula that considers the outer product of the plane parameters which consist of the normal to the face and a point on the face. This results in a 4x4 matrix that quantifies the squared distance from a point to the plane of the face, storing geometric error information. Given a face with normal

$$\vec{n} = (n_x, n_y, n_z)$$

and a point on the face,

$$\vec{p} = (p_x, p_y, p_z)$$

the plane equation is

$$n_x x + n_y y + n_z z + d = 0$$

where

$$d = -\vec{n} \cdot \vec{p}$$

The quadric matrix Q is computed as:

$$Q = \begin{bmatrix} n_x^2 & n_x n_y & n_x n_z & n_x d \\ n_x n_y & n_y^2 & n_y n_z & n_y d \\ n_x n_z & n_y n_z & n_z^2 & n_z d \\ n_x d & n_y d & n_z d & d^2 \end{bmatrix}$$

Next, for each vertex, a cumulative quadric is computed by summing the quadrics of all faces adjacent to that vertex. This quadric represents the total error at a vertex and is used for determining how the mesh can be simplified with minimal change from the original.

2.2 Edge Collapse Computation

An edge collapse involves merging two vertices into one. The choice of which edge to collapse is guided by the quadric error metrics calculated earlier. The edge that minimizes the error when collapsed is chosen. This process involves calculating the optimal position for the new merged vertex using the combined quadrics of the vertices at either end of the edge. The new vertex position is computed such that it minimizes the geometric error in terms of the original mesh surface. This involves solving a system of linear equations derived from setting the gradient of the quadric error function to zero, outlined in Section 4.1 of Garland and Heckbert's paper [2].

3 Tasks

This chapter goes through the tasks implemented during the lab.

3.1 Create quadric for face

The **CreateQuadricForFace()** function calculates the quadric matrix for a given face in the mesh. It was implemented by:

1. Extract the normal and a point on the face from the face data structure.
2. Compute the plane equation parameters as described in chapter 2
3. Calculate the outer product of the plane parameter vector with itself to form the quadric matrix.

The equation for a face quadric follows:

$$Q = \text{outer}(\vec{plane}, \vec{plane})$$

3.2 Create quadric for face

The **CreateQuadricForVert()** function sum the quadrics of all faces adjacent to a vertex to compute the vertex's quadric. This cumulative quadric is used to evaluate how the simplification would affect the mesh geometry at that vertex. It was implemented by looping through all faces adjacent to the vertex, and summing up respective quadrics.

3.3 Compute Collapse

The **ComputeCollapse()** function calculates the optimal target position for an edge collapse and estimates the cost of this operation based on the vertex quadrics at the endpoints of the edge.

1. Compute the sum of the quadrics for the two vertices of the edge, $\mathbf{Q} = \mathbf{Q_1} + \mathbf{Q_2}$
2. Calculate cost of the collapse by solving the equation

$$\Delta(v) = v^{-T} Q v$$

3. Check if the resulting quadric matrix is invertible to find the optimal collapse position. If invertible, the position will be:

$$\vec{v}_{\text{opt}} = Q^{-1} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

This calculation solves the equation:

$$\begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{12} & q_{22} & q_{23} & q_{24} \\ q_{13} & q_{23} & q_{33} & q_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ v_z \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

4. If the Q matrix is not invertible, a position along the edge was chosen. Either at v1, v2 or in the middle, depending on the least cost.

   The algorithm loops through all possible edge collapses and then chooses the one with the least cost.

4 Results

This section show mesh decimation on a cow mesh model at different levels of detail. As the number of faces decreases, the model becomes less detailed while still keeping the overall shape.

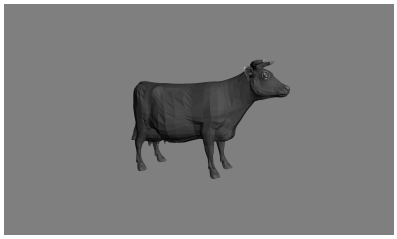

Figure 1: Original model with 5804 faces.

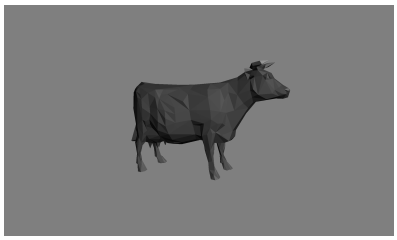

Figure 2: Reduced to 994 faces.

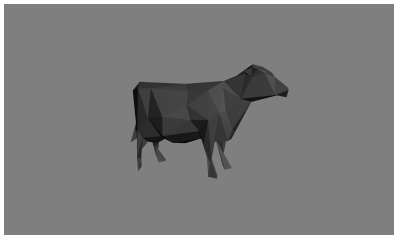

Figure 3: Further reduced to 248 faces.

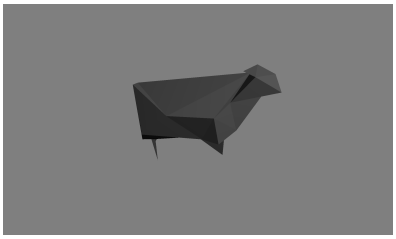

Figure 4: Simplified to 64 faces, showing the most aggressive reduction.

5 Conclusion

Quadric-based mesh decimation is effective in simplifying 3D mesh models. The half-edge data structure helps the decimation algorithm with neighborhood access and edge collapses. The results illustrate the ability to maintain good visual quality despite a significant reduction in faces.

6 Lab partner and grade

Lab partner was Kenton Larsson and the grade I am for is 3.

References

[1] Mark Eric Dieckmann. *MESH DATA STRUCTURES*. Linkoping university, 2023.

[2] Michael Garland and Paul S. Heckbert. *Surface simplification using quadric error metrics*. ACM Press/Addison-Wesley Publishing Co., 1997.