LAB REPORT: LAB 2

TNM079, MODELING AND ANIMATION

Viktor Sjögren
viksj950@student.liu.se

Sunday 25th April, 2021

Abstract

We discuss the quadric-based error metric to facilitate more elegant mesh decimation operations. The quadric error metric aims at making edge contractions select the new vertex in a position which is the least visually impactful for appearance of the model. This cost has been implemented and used in decimation's of different models. To see how much of the quadric-based cost affects the edge collapse operations, another less elegant cost function has been compared to which always chooses the shortest edge to collapse.

1 Background

In many applications of computer graphics, it is important to produce and render realistic models. The reason why a model can look realistic in the first place is because of the amount of detail and information that can be stored about the mesh. Generally, the more information that builds up the mesh, means that the model becomes more realistic and fancier in appearance.

However, one cannot simply overlook the cost of this. A fully detailed model may not cost an unreachable amount of computational power on its own, but most of the times much more then one model is displayed at once, meaning the cost increases rapidly for a whole scene. This means that different methods has to be put in action in order to effectively decrease the computational strain that comes with having highly detailed models. The tricky part is finding a balance between performance and visual fidelity, as there is rarely no increase in the other, without the counterpart decreasing.

The method discussed here stems from Garland and Heckbert's paper [1], in which they propose a quadric based mesh decimation.

The way one can reduce the information for models is by reducing the amount of vertices and edges that the mesh contains, in order to do this, edge contractions are performed. For the scope of the lab which this report builds upon, only manifold half edge mesh structures were considered when applying these operations.

The edge collapse operation is a method for removing vertices and corresponding edges to reform the mesh into a lower detailed version, but tries to keep the appearance as close as possible after the operation. In essence when edge collapse is performed, one of the two vertices that are involved is completely removed, while one remains. This means that for each edge collapse the neighborhood information for the remaining vertex has to be updated. In Figure 1, a visual description can be seen of a general edge contraction operation. In that particular case, $V1$ is removed and $V2$ remains, and all adjacent edges are updated to form a connection with $V2$.

In order to measure which parts vertices of a mesh that can be decimated, with the least visual impact, it is vital to effectively associate a cost for each vertex. This cost, as suggested in [1], is computed with regards to the squared distance of the new vertex position in rela-

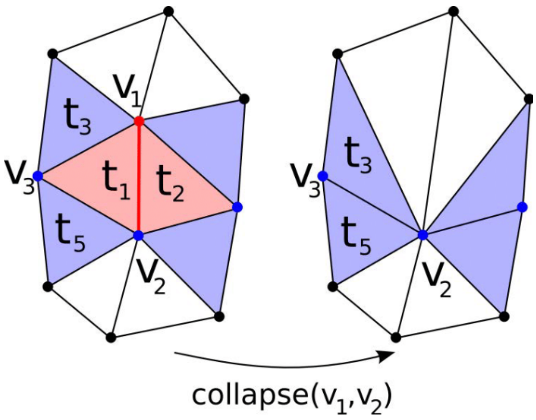


Figure 1: Visual representation of the edge contraction operation

tion to the original positions of the vertices, the error simply put. In essence, that means the algorithm puts a higher cost on edge contractions where the joint new vertex is placed relatively far away from the original positions of the vertices associated with the removed edge. While on the other hand, a new vertex position that has a relatively short distance to the old vertices would be seen as a lower cost.

The way the cost are calculated in the Lab code is by associating a symmetric 4x4 matrix to each vertex. This matrix is labeled as **Q** and for each vertex, $\mathbf{v} = [v_x\ v_y\ v_z\ 1]$ we also associate the error in the quadratic form, as presented by [1] as $\Delta(\mathbf{v}) = v^T Q v$. The new position of the remaining vertex $\tilde{v}$ after the edge collapse operation will be approximated as a vertex represented by $\tilde{Q}$. The computation of the matrix is determined by the additive rule

$$\tilde{Q} = Q_1 + Q_2 \tag{1}$$

where Q_1 and Q_2 are the associated matrices for the two original vertices.

$$\begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{12} & q_{22} & q_{23} & q_{24} \\ q_{13} & q_{23} & q_{33} & q_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tilde{v} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{2}$$

The new position $\tilde{v}$ can be chosen so that the error is minimized by solving equation 2 for $\tilde{v}$. If however the matrix is not invertible, the minimzed error cannot be obtained through 2, and we fall back on simpler schemes. In this Lab, either one of the positions of the old vertices is selected, just as in Figure 1, or the middle point between the two vertices is selected. The choice is based on whichever has the lowest cost.

The code was implemented in the class *QuadricDecimationMesh.cpp* and in the function *computeCollapse*.

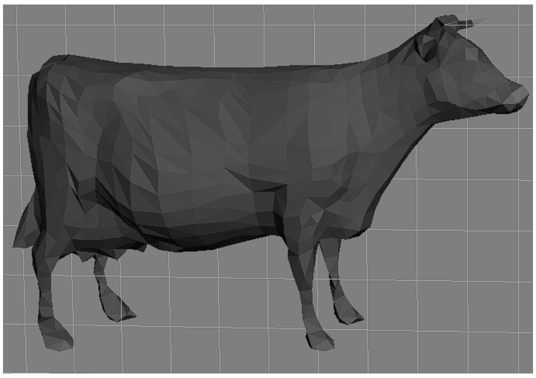
2 Results

To see the clear advantages of having error cost associated to each edge collapse, a comparison between the discussed quadric-based method and a simpler method that always performs an edge collapse where the shortest edge lies is going to be shown.

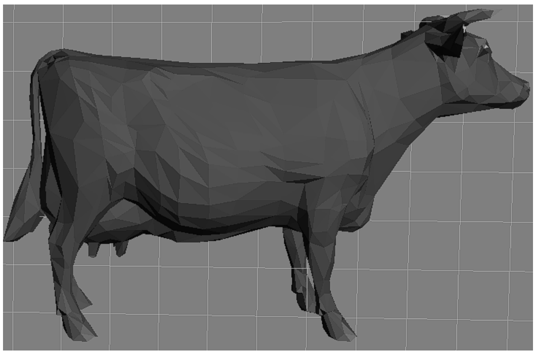
In Figure 2(a), the simple scheme of selecting the shortest edge to collapse is seen, while in Figure 2(b) the quadric-based method is seen. Notably, both contain the same amount of faces, yet differ much in visual appearance. The quadric-based decimation retains most of the face features of the cow, while the simple scheme loses much of the original appearance. Even more notably, the thinner and sharper areas, (look at the teats for example) is completely unrecognizable in 2(a) from the original model.

Reducing the face count even more proves the importance of an elegant cost measurement even better, in Figure 3(a) and 3(b) the appearance is much worse then original model, as expected when reducing at that amount. However, the overall shape and outline, and even some of the minor details such as the horns and teats of the cow is retained in the quadric-based decimation.

Interestingly, for some models, such as a sphere with less varying faces and edges, the difference between an elegant cost measurement and a simpler one is not nearly as important. Studying Figure 4(a) and 4(b) shows that essentially the same visual fidelity is retained once decimated.



(a)



(b)

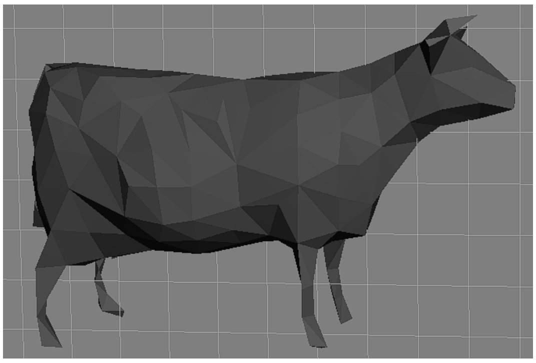
Figure 2: Resulting cow mesh, when decimating using two methods, left is the shortest edge method and right is the quadric-based. The model is down to 34.5% of its original face count. (5804 faces $\Longrightarrow$ 2000 faces)

3 Lab partner and grade

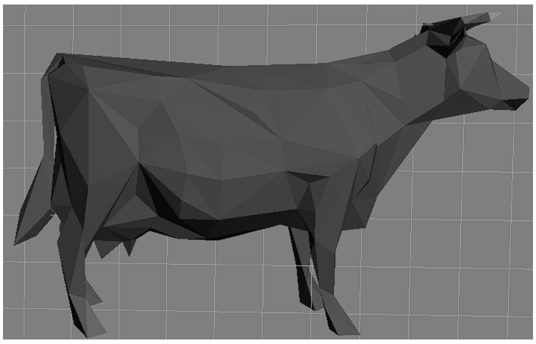
The lab was made in collaboration with Algot Sandahl. The report scope and content is aimed at grade **3**.

References

[1] Michael Garland and Paul S.Heckbert. *Surface sim-plification using quadric error metrics*. NewYork, NY, USA. ACM Press/Addison-WesleyPublishing Co., 1997.

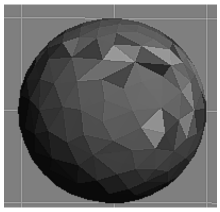


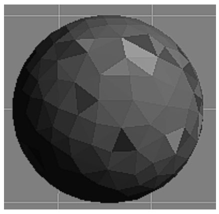
(a)



(b)

Figure 3: Resulting cow mesh, when decimating using two methods, left is the shortest edge method and right is the quadric-based. The model is down to 8.62% of its original face count. (5804 faces $\Longrightarrow$ 500 faces)





(a) (b)

Figure 4: Resulting sphere mesh, when decimating using two methods, left is the shortest edge method and right is the quadric-based. The model is down to 25% of its original face count. (1984 faces $\Longrightarrow$ 496 faces)