LAB REPORT: LAB 2

TNM079, MODELING AND ANIMATION

Algot Sandahl
algsa119@student.liu.se

Tuesday 4th May, 2021

Abstract

This lab report describes the tasks and results from lab 2 in the course TNM079, Modeling and Animation at Linköping University. The topic of the lab is mesh decimation using quadric error metrics. The error quadrics were also visualized, and an alteration to the cost heurisic, to decimate more heavily at the back of the models, was tested. The quadric error metric turned out to produce good results, and the alteration successfully decimated the back of the model more heavily.

1 Background

Rendering models using varying amounts of details depending on the visibility of the model is an essential technique in computer graphics. It allows the detail of the models to vary with the distance to the camera so that computer resources are distributed to provide the best image quality possible. The different models can of course be manually created, but it is also useful to automatically generate the less detailed models from an original "full detail model." The aim of this lab is to implement a quadrics based cost heuristic to determine how to decimate models.

The algorithm used to decide which edges to collapse can be briefly explained as iteratively collapsing the edge that has the least cost associated with it, until the number of remaining faces reaches a target.

The quadric error metric used in this lab is based on a method presented by Garland and Heckbert [1], and it builds on the fact that each of the adjacent faces of a vertex, $\mathbf{v}$, unambiguously defines a plane. The intersection of all of these planes is the point $\mathbf{v}$. By letting the distance from a new position, $\bar{\mathbf{v}}$, to each of these planes, $\mathbf{p}$, be $d_{\mathbf{p},\bar{\mathbf{v}}}$, the error metric of moving $\mathbf{v}$ to a new position $\bar{\mathbf{v}}$ can be defined using the squared distance:

$$\Delta(\bar{\mathbf{v}}) = \sum_{\mathbf{p}\in\text{planes}(\mathbf{v})} d^2_{\mathbf{p},\bar{\mathbf{v}}}. \tag{1}$$

To find the distances, $d_{\mathbf{p},\bar{\mathbf{v}}}$, the normal form of a plane is used

$$ax + by + cz + d = 0 \tag{2}$$

where a, b, and c are given by the face normal, $\hat{\mathbf{n}} = (a, b, c)$, which is known. The only unknown is d, which is easily obtained by

$$d = -(ax_0 + by_0 + cz_0) = -\mathbf{v}_0 \cdot \hat{\mathbf{n}} \tag{3}$$

where $\mathbf{v}_0 = (x_0, y_0, z_0)$ is a point on the plane (for example one of the triangle vertices). Using this information, (1) can be formulated as

$$\Delta(\bar{\mathbf{v}}) = \sum_{\mathbf{p}\in\text{planes}(\mathbf{v})} (\mathbf{p}^T\bar{\mathbf{v}})^2 \tag{4}$$

where $\mathbf{p} = (a, b, c, d)$ for each of the planes, and $\bar{\mathbf{v}}$ is given in homogeneous coordinates. Since $\bar{\mathbf{v}}$ does not depend on $\mathbf{p}$, it can be factored out of the sum:

$$\Delta(\bar{\mathbf{v}}) = \sum_{\mathbf{p}\in\text{planes}(\mathbf{v})} (\bar{\mathbf{v}}^T\mathbf{p})(\mathbf{p}^T\bar{\mathbf{v}}) \tag{5}$$

$$= \sum_{\mathbf{p}\in\text{planes}(\mathbf{v})} \bar{\mathbf{v}}^T(\mathbf{p}\mathbf{p}^T)\bar{\mathbf{v}} \tag{6}$$

$$= \bar{\mathbf{v}}^T \left(\sum_{\mathbf{p}\in\text{planes}(\mathbf{v})} \mathbf{K_p} \right) \bar{\mathbf{v}}. \tag{7}$$

where the fundamental error quadric for the plane $\mathbf{p}$ is

$$\mathbf{K_p} = \mathbf{p}\mathbf{p}^T = \begin{bmatrix} a^2 & ab & ac & ad \\ ab & b^2 & bc & bd \\ ac & bc & c^2 & cd \\ ad & bd & cd & d^2 \end{bmatrix}. \tag{8}$$

Garland and Heckbert approximates the error matrix of the contraction $(\mathbf{v}_1, \mathbf{v}_2) \to \bar{\mathbf{v}}$ as the symmetric matrix

$$\bar{\mathbf{Q}} = \mathbf{Q}_1 + \mathbf{Q}_2 = \begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{12} & q_{22} & q_{23} & q_{24} \\ q_{13} & q_{23} & q_{33} & q_{34} \\ q_{14} & q_{24} & q_{34} & q_{44} \end{bmatrix} \tag{9}$$

where $\mathbf{Q}_1$ and $\mathbf{Q}_2$ are the error matrices for $\mathbf{v}_1$ and $\mathbf{v}_2$. To find the optimal position for $\bar{\mathbf{v}}$, the extremum of $\Delta(\bar{\mathbf{v}})$ is found using

$$\frac{\partial\Delta(\bar{\mathbf{v}})}{\partial x} = \frac{\partial\Delta(\bar{\mathbf{v}})}{\partial y} = \frac{\partial\Delta(\bar{\mathbf{v}})}{\partial z} = 0 \tag{10}$$

which can be formulated in matrix form as

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

and if the matrix is not singular—i.e. has a non-zero determinant—$\bar{\mathbf{v}}$ can be found according to

$$\bar{\mathbf{v}} = \begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{12} & q_{22} & q_{23} & q_{24} \\ q_{13} & q_{23} & q_{33} & q_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}. \tag{12}$$

Since the calculations above are not always successful, alternatives are required. The alternative $\bar{\mathbf{v}}$ positions used in the lab are $\mathbf{v}_1$, $\mathbf{v}_2$, and $(\mathbf{v}_1 - \mathbf{v}_2)/2$. The decision is made by simply choosing the new position among the alternatives that results in the lowest error.

A simple modification to the error metric was also tested. The goal was to decimate the model more at the back, where it cannot be seen. To accomplish this goal, the angle, θ, between the surface normal, $\hat{\mathbf{n}}$, and direction, $\hat{\mathbf{v}}_c$, from the camera to the new position is used. To avoid explicitly calculating the angle, the cosine is used:

$$\cos\theta = \hat{\mathbf{n}} \cdot \hat{\mathbf{v}}_c. \tag{13}$$

The cosine is negative for faces that point towards the camera, and positive for faces that point away from the camera. This is used in the following way: If the cosine is smaller than a threshold, g, the cost given by the error quadric method is multiplied by a weight, w, resulting in a higher cost for faces that face the camera. The constant, g, is chosen such that the result looks good. It might seem natural to set $g = 0$, but this might be quite noticeable around the "edges" of the model—thus it might be better to select a threshold slightly above 0.

Of course, this heuristic would not be very useful if the model moves in such a way that the user will eventually see the back. It could, however, be useful when that is not the case, as the computer resources are put primarily towards triangles that can actually be seen.

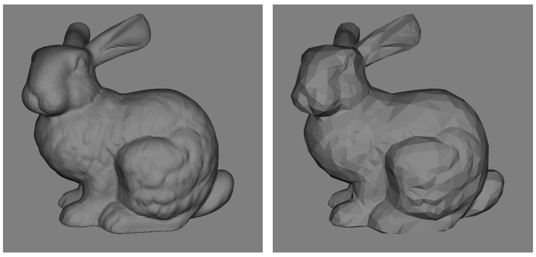
The quadric error metric can be visualized in the form of isosurfaces. Let ϵ be an error value, then the desired isosurface can be described by

$$\mathbf{x}^T\mathbf{Q}\mathbf{x} = \epsilon. \tag{14}$$

Garland [2] describes a convenient visualization method that is briefly described here. The Cholesky decomposition can be used to factor a symmetric positive definite matrix, $\mathbf{Q}$ as

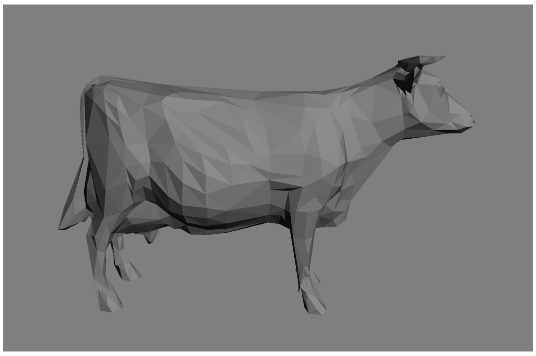
$$\mathbf{Q} = \mathbf{R}^T\mathbf{R} \tag{15}$$

where $\mathbf{R}$ is an upper triangular matrix. It can be shown that $\mathbf{Q}$ satisfies the requirements if and only if its isosurface is an ellipoid. If it is not, it is not visualized. The remaining

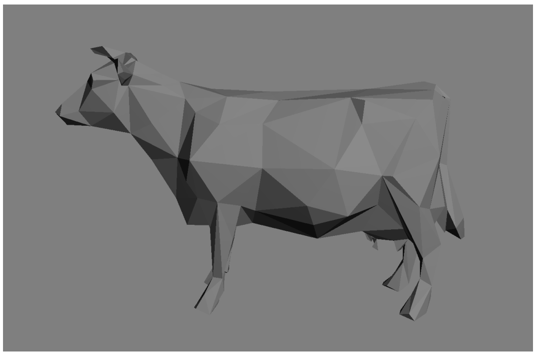


(a) Original (35,032 faces). *(b)* Decimated to 3000 faces.

Figure 1: Decimation demonstration.



(a) Facing camera.



(b) Back of model.

Figure 2: A demonstration of the alternative cost heuristic. The model has been decimated to 1,000 faces using the threshold $g = 0.2$ and the weight $w = 10$. As seen, the faces that did not face the camera were much more heavily decimated.

theory assumes that the requirements are satisfied. Consider the surface given in (14)—by decomposing $\mathbf{Q}$, it can be rewritten as

$$\mathbf{x}^T\mathbf{Q}\mathbf{x} = \mathbf{x}^T(\mathbf{R}^T\mathbf{R})\mathbf{x} = (\mathbf{R}\mathbf{x})^T(\mathbf{R}\mathbf{x}) = \epsilon. \quad (16)$$

By letting $\mathbf{y} = \mathbf{R}\mathbf{x}$, it can be noted that $\mathbf{y}^T\mathbf{y} = \epsilon$ describes a sphere—$\mathbf{R}$ transforms the ellipsoid into a sphere of radius ϵ. Using this finding, the isosurface can be rendered by morphing a sphere with radius ϵ into the error quadric isosurface by applying the transform $\mathbf{R}^{-1}$.

2 Results

The results of the lab are presented in the following sections.

2.1 Decimation using the Quadric Error Metric

A simple demonstration of the decimation is shown in Figure 1. A comparison between different levels of decimation is shown in Figure 4 together with the same levels of decimation for the `SimpleDecimationMesh`. The `SimpleDecimationMesh` places the contracted vertex halfway between the two contracted vertices and calculates the cost as the distance between the new position and the uncontracted vertex.

2.2 Alternative cost heuristic

The effect of the alternative error metric is shown in Figure 2. Both the side facing the camera and the opposite side is shown to demonstrate the difference. It can also be compared with the decimations to 1,000 faces shown in Figure 4.

2.3 Quadric Visualization

The visualization of the quadrics calculated for the vertices in the cow model is shown in Figure 3.

3 Conclusion

The quadric error metric is a useful metric that preserves details reasonably well when used for decimation.

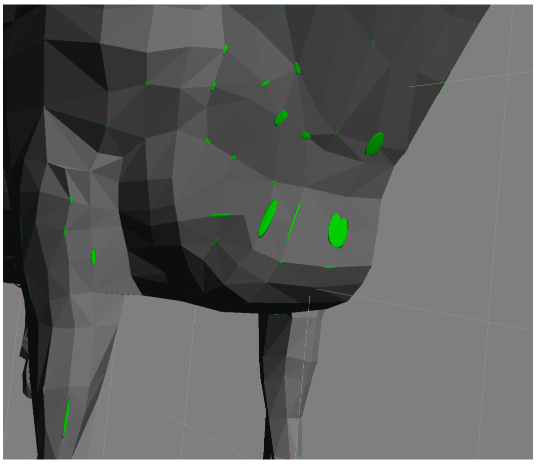


Figure 3: Visualization of the error quadric isosurfaces on the full cow model.

3.1 Decimation using the Quadric Error Metric

The quadric error metric provides a useful metric for how much a movement of a vertex changes the overall apperance of the model. As seen when compared to the `SimpleDecimationMesh`, the quadric error metric preserves the details better. This is attributed partly to the fact that the optimal position of the contracted vertex is calculated, but also to the fact that the cost heuristic makes it so that more faces remain in detailed regions of the model—this can be seen quite clearly in Figure 4. The `SimpleDecimationMesh` model gets decimated in a more uniform way. See for example the cow decimated to 250 faces—the head of the `SimpleDecimationMesh` cow is very deformed, whereas its general shape is preserved when using the quadric error metric in the `QuadricDecimationMesh`. It can be seen that the quadric error metric results in fewer triangles on the rest of the cow, where there are less details.

3.2 Alternative cost heuristic

As demonstrated, the alternative cost heuristic introduced efficiently decimates the back of the model while keeping the details on the front. When comparing Figure 2a with the cow decimated to 1,000 faces in Figure 4, it can be seen that the alternative cost heuristic provides more detail at the front of the model. In essence, the presented alternative cost heuristic has the desired effect but could be improved by more sophisticated techniques, such as also taking screen coverage into account.

3.3 Quadric Visualization

The visualization of the quadrics, provide some insight into the inner workings of the decimation method, on a per-vertex basis. It can be seen that on flatter surfaces, the points can be moved quite far along the surface without introducing a big error, whereas on sharp points, the quadrics can barely be seen because moving the corresponding vertices even a little bit would introduce a big error. It can also be seen how moving vertices along somewhat straight edges produces a fairly low error. These observations all coincide well with what intuitively would create the largest or smallest changes. For example, moving a vertex placed on a large flat surface along the surface produces very small changes, whereas moving a vertex at a pointy area would change the surface drastically.

Lab Partner and Grade

The lab was done together with Viktor Sjögren. All lab tasks were finished and the report aims for grade 5.

References

[1] M. Garland and P. Heckbert, "Surface simplification using quadric error metrics," *Proceedings of the ACM SIGGRAPH Conference on Computer Graphics*, vol. 1997, 07 1997.

[2] M. Garland, "Quadric-based polygonal surface simplification," Ph.D. dissertation, School of Computer Science, Carnegie Mellon University, 1999.

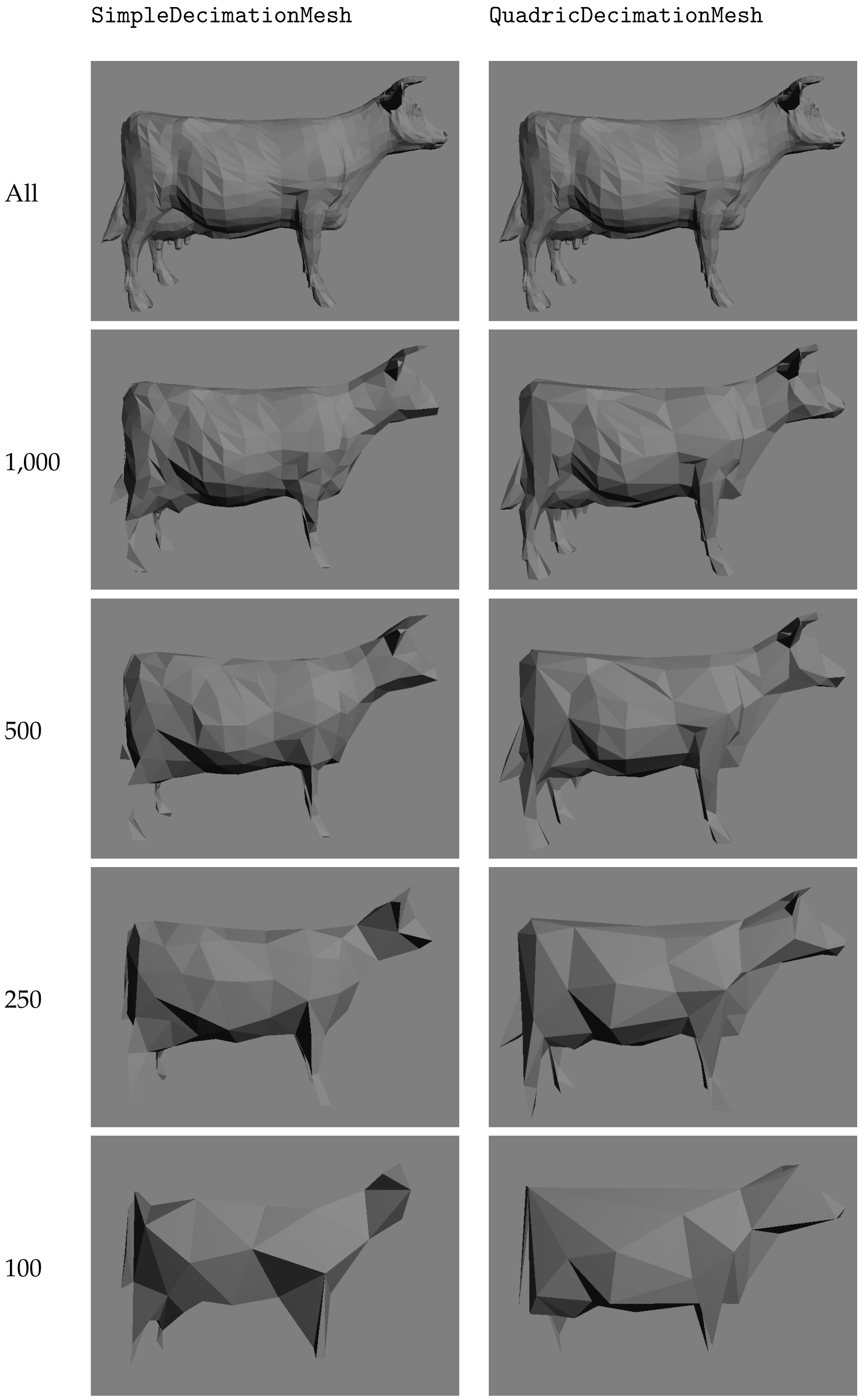


Figure 4: A comparison between `SimpleDecimationMesh` and `QuadricDecimationMesh` and different levels of decimation. The numbers to the left are the number of remaining faces. The original mesh consisted of 5,804 faces.