LAB REPORT: LAB 2

TNM079, MODELING AND ANIMATION

Alma Linder
almli825@student.liu.se

Wednesday 16th April, 2025

Abstract

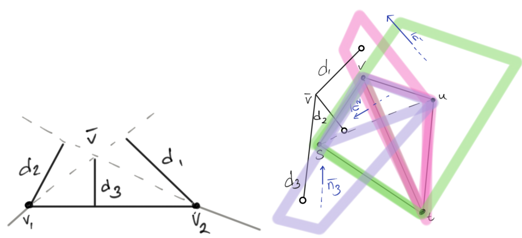
This work covers the efficient, general and quality-preserving `Quadric Decimation Technique` of Garland and Heckbert. Decimation save time and memory, and allows for dynamic management of these, e.g. real-time rendering. Apart from deriving and implementing the quadric distance-based cost function of collapsing edges, the work also includes a "LoD" heuristic cost function, to explore decimation and its applications further. Decimation with the `Quadric Decimation Technique` proves to preserve overall shapes longer, in comparison to a simpler decimation technique. Tweaking the simple decimation technique results in varying detail preservation for the same number of faces to decimate to, by only varying an artificial camera position.

This laboratory work aims for `grade 4`.

1 Background

A decimation technique regarding efficiency, quality and generality (connect separated regions in the mesh) is the `Quadric Decimation Technique` of Garland and Heckbert [1]. It removes vertices by merging them (edge collapse). A cost based on the squared distance for moving and merging each vertex is computed and tracks the decimation quality. Costs are stored in increasing order on a heap, which allows popping the cheapest vertex for each iteration of the algorithm.

Figures 1(a) and 1(b) illustrate the foundation of the `Quadric Decimation Technique`: Two vertices v_1 and v_2 in Figure 1(a) should be merged at the lowest possible cost location $\bar{v}$. The cost is given by the squared (to avoid negative distance values) distance to each of the 1-ring triangle face planes from $\bar{v}$ - sketched as d_1, d_2 and d_3 in Figure 1(b). The 1-ring neighborhood can be obtained for any vertex v on the original mesh, and in Figure 1(b) this is the green, pink and purple triangle.



(a) (b)

Figure 1: Sketches of two vertices v_1 and v_2 to be merged at a new, lowest cost location $\bar{v}$. The planes extended from the neighborhood faces of a vertex v are drawn in green, pink and purple, with normals in blue. d_1, d_2 and d_3 are their distances to $\bar{v}$.

Because the shortest distance from $\bar{v}$ to a plane can be computed via the dot product of the plane normal n, and the difference between $\bar{v}$ and any point v_0 on the plane, the error $\Delta(\bar{v})$ introduced by merging v_1 and v_2 at $\bar{v}$ is given by Equation 1, where f_i is the distance from $\bar{v}$ to each plane i in the neighborhood.

$$\Delta(\bar{v}) = \sum_i ((\bar{v} - v_0) \cdot n_i)^2 = \sum_i f_i(\bar{v})^2 \quad (1)$$

From the cartesian form of $f(\bar{v})$ (given by $ax + by + cz + d$, where $a^2 + b^2 + c^2 = 1$), Equation 1 can be rewritten to Equation 2 using the *associative law*, since there is one homogeneous coordinate representation $p = (a, b, c, d)$ for each plane i in the neighborhood, and $f(\bar{v}) = p_i^t$.

$$\Delta(\bar{v}) = \sum_i (p_i^t \cdot \bar{v})(p_i^t \cdot \bar{v}) = \bar{v}^t (\sum_i p_i p_i^t) \bar{v} \quad (2)$$

The cost $\Delta(\bar{v})$ becomes Equation 3, where Q is the contributed error to the cost of each triangle in the 1-ring of current vertex, v in Figure 1(b). To consider the cost of merging v_1 and v_2 their respective contributions Q_1 and Q_2 have to be summed, forming $\bar{Q}$.

$$\Delta(\bar{v}) = \bar{v}^t \bar{Q} \bar{v} = \bar{v}^t (\sum_i K_{pi}) \bar{v} \quad (3)$$

The smallest cost is sought for each vertex - i.e. a minimum is sought. Extreme points can be found with the derivative (since it is zero at such locations) and $\bar{Q}$ contains a summation of costs q_n - refer to Equation 4. Thus, the optimal position $\bar{v}$ is derived from the partial derivatives of $\bar{v}^t \bar{v} \bar{Q}$. Expanding Equation 3, applying partial derivatives and dividing by two yields the system displayed in Equation 5.

$$\bar{Q} = \begin{pmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{21} & q_{22} & q_{23} & q_{24} \\ q_{31} & q_{32} & q_{33} & q_{34} \\ q_{41} & q_{42} & q_{43} & q_{44} \end{pmatrix} \quad (4)$$

$$\begin{cases} q_{11}x + q_{12}y + q_{13}z + q_{14} = 0 \\ q_{12}x + q_{22}y + q_{23}z + q_{24} = 0 \\ q_{13}x + q_{23}y + q_{33}z + q_{34} = 0 \end{cases} \quad (5)$$

Rewriting Equation 5 to matrix form, and adding a one for cartesian coordinates gives Equation 6, in which $\bar{v}$ is solved for.

$$\begin{pmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{12} & q_{22} & q_{23} & q_{24} \\ q_{13} & q_{23} & q_{33} & q_{34} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$
$$\Rightarrow \bar{v} = inv(\hat{Q})\bar{v} \quad (6)$$

Equation 6 requires $\hat{Q}$ to be invertible (non-zero determinant). If not, position $\bar{v}$ is found on the segment $v_1 v_2$ by computing the cost for the midpoint and the endpoints and then choose which of the three cases that yields the lowest cost.

The decimation mesh data structure follows the half-edge structure, but it additionally has a heap of costs, and keeps addresses of what half-edges to collapse. Thus, what vertices to access for each iteration that the cost should be computed for was accessed indirectly from these edges. Face normals in the 1-ring were retrieved from the vertex index accessed from edges to collapse, and then plane parameters a, b and c were extracted from the normal vector on corresponding dimension. The distance K_p matrix using plane parameters in Equation 3 was defined in its 4×4 matrix form.

Finally, the structure for $\hat{Q}$ in Equation 6 was created from $\bar{Q}$, and the cases for $\hat{Q}$ being invertible or not tested and costs computed accordingly.

There are alternative ways to design a cost heuristic. To simulate a simple, static LoD functionality, an artificial camera at $[0, 0, -50]$ was instantiated. The midpoint of the segment $v_1 v_2$ was used to approximate the position $\bar{v}$ for the collapse, as it represents an average between the two endpoints.

To preserve detail in relation to camera distance from $\bar{v}$ (distance d in Equation 7), the camera position was multiplied by a factor $f = 0.01$ in those dimensions it had a non-zero value. The distance between the midpoint and an endpoint was then divided by the camera position under the influence of its factor f. This cost function is shown in Equation 7.

$$\Delta(\bar{v}) = \frac{||\bar{v} - v_1||}{fd} \quad (7)$$

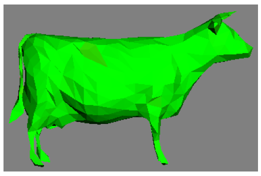
The camera position $[0, 0, -50]$ was chosen because distances are positive. If the camera was left at position $[0, 0, 0]$, a distance of -100 would be indistinguishable from 100, making the LoD results ambiguous. At -50, the distance from the camera to $\bar{v}$ is consistently positive, resolving the ambiguity and ensuring that distances are always on a positive scale. The factor $f = 0.01$ was applied because it ensures that the largest response occurs for objects closest to the camera: When the scaled camera position is small, d becomes large, and consequently, the ratio in Equation 7 becomes small, giving higher cost to closer objects and lower cost to objects further away.

2 Results

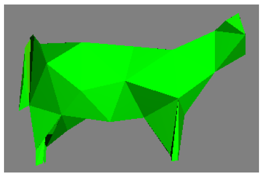
Figure 2 compares the `simple decimation` (cost computed as the distance between the midpoint of the segment $v1v2$ and the first point $v1$ on the segment) to that of the `quadric decimation`. When keeping around a sixth (1000 faces) of all triangles, the `simple decimation technique` manages to maintain most shape and detail - as is visible in Figure 2(a). However, the `quadric decimation` already here begins to show its superiority. Details such as eye socket, hooves and udder remain more intact - see Figure 2(c) - as evidently more triangles are kept in these areas (higher quadric cost) than in the coarser areas like the belly or flank. The `simple decimation technique` in Figure 2(a) on the other hand seems to produce a more constant triangle density over the entire mesh.

As the triangle count is decreased even more, the quadric distance-based cost heuristics really demonstrates its abilities to preserve structure and generality better, significantly outperforming the `simple decimation algorithm`. In Figure 2(d) for a 100 triangles, the overall cow shape remains, also around smaller sections like the head, tail, horn and legs, meanwhile the head suffers greatly for the same amount of triangles in Figure 2(b).

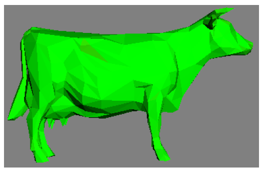
These results can undoubtedly be traced back to the fact that the `quadric decimation technique` considers a neighborhood on the surface for the cost of collapsing each edge. Thus, a means of monitoring the surface globally is obtained as an awareness is invoked. The `simple decimation technique` only says that shorter edges should be decimated first, as they produce smaller distances. However, small triangles can still bear a lot of importance as they build fine-detail regions.



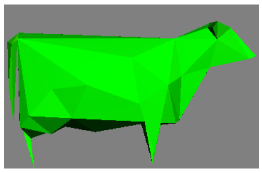
(a) `Simple Decimation`, 1000 faces.



(b) `Simple Decimation`, 100 faces.



(c) `Quadric Decimation`, 1000 faces.

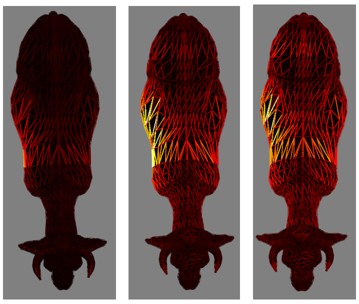


(d) `Quadric Decimation`, 100 faces.

Figure 2: Comparison of decimation results between the `simple decimation technique` and the `Quadric Decimation Technique` on 1000 faces and on 100 faces.

Figure 3 verifies the correctness of the LoD heuristics in that the left side of the cow - which is closer to the camera - indeed has higher costs and thus will be preserved longer, and that the higher costs influence larger sections of the cow as the camera closes up (moves to the right). To ensure not only long edges in the direction of the camera (i.e. z-direction) were preserved, a small offset in another direction ($x = 3$) was used as well.

Evidently, the amount of detail preserved differs more between the furthest camera position in 3(a) and the middle position 3(b), meanwhile the difference is much smaller between the middle and the closest distance 3(c) - only slightly more expensive to collapse around the lower back and pelvis when $z = 50$.

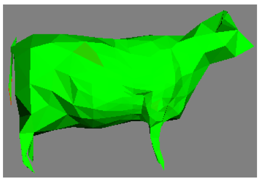


(a) Far, $z = 10$

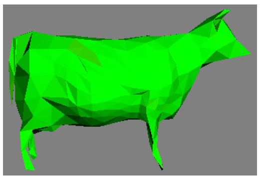
(b) Middle, $z = 25$

(c) Close, $z = 50$

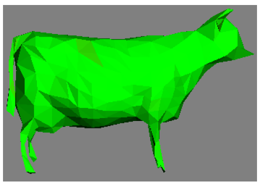
Figure 3: LoD costs mapped to the *"hot"* color map before decimation with constant camera position in $x = 3$ but with varying z-position. Brighter red/yellow color indicates higher cost.



(a) `LoD Decimation`, camera position in $\{3,0,10\}$



(b) `LoD Decimation`, camera position in $\{3,0,25\}$



(c) `LoD Decimation`, camera position in $\{3,0,50\}$

Figure 4: Decimation results of the LoD cost heuristics. Number of faces constantly 500 and varying artificial camera position back and forth (z-direction).

Decimating the cases in Figure 3 down to 500 faces yields the results of Figure 4, with the side of the cow closest to the camera in focus. Unsurprisingly, the globally coarsest shape is produced in Figure 4(a), when the camera is further away, as most of the cow is simplified. The most preserved area according to the cost map in Figure 3(a) should be around the upper belly and shoulder. Looking at the decimated result in 4(a) this appears to be the case.

The more similar cases middle and close, in Figures 4(b) and 4(c) respectively, interestingly differ mostly around the tail and muzzle - i.e. the ends furthest away from the camera - but in general look quite similar. Table 1 over the collapse costs at 500 faces explains why this phenomenon occurs. Minimum costs vary rather evenly between far and middle (difference of 0.026) and middle and close (difference of 0.024), meanwhile the maximum costs vary significantly (differences of 0.7 and 0.16 respectively).

Table 1: Collapse cost at 500 remaining faces for the LoD cases $z = 10$, $z = 25$ and $z = 50$.

	Min Cost	**Max Cost**
$z = 10$	0.093	1.01
$z = 25$	0.067	0.31
$z = 50$	0.043	0.15

This would suggest that there exists a threshold below which proximity to the camera no longer affects edge priority significantly. Thus, it stands to reason that the LoD heuristics is view-dependent only in theory, as it gives a spatial bias to an area but has no awareness of surface curvature or perception. One explanation could be that the range of values used to scale the camera's influence was too narrow to produce meaningful variation in the cost function, and so experimenting with other scaling factors f could be valuable. Furthermore, the LoD cost heuristics could be combined with the more globally aware `Quadric Decimation technique` to produce more evenly spaced cost differences and consequently yield perceptually improved decimated versions.

References

[1] Michael Garland and Paul S Heckbert. Surface simplification using quadric error metrics. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 209–216, 1997.