Volume 15, (1996) number 4 pp. 191–203

SCROOGE: Perceptually-Driven Polygon Reduction

M. Reddy

Department of Computer Science, the University of Edinburgh, Edinburgh, UK

Abstract

Many real-time 3D graphics renderers represent each object as a collection of simple polygons. The complexity of this polygon structure is of practical relevance because it can manifestly affect the performance of the graphics system. It is therefore commonplace to find techniques to reduce the polygonal complexity of a model with the ultimate aim of improving the interactivity of the application. In the past, many of these schemes have not been concerned with the perceptual side-effects of this reduction and as a result a number of visual incongruities are often perceivable when these correspondingly-reduced representations are employed. As an attempt to circumvent these problems, this paper presents a methodology for reducing the polygonal complexity of a model, whilst retaining a degree of perceptual predictability. This allows the visual consequences of the degradation to be quantified and accurately modelled.

Keywords: polygon reduction, level of detail, visual perception, human visual system.

1. Introduction

It is a well known fact that the number of polygons used to represent a 3D object in a real-time graphics system is proportional to the time required to process and display that object.[1] This is not a simple relationship because there are a number of interacting factors involved (including the shading model which is used, the number of lights in the scene, the use of texture mapping, the size and nature of the polygons, the ratio of polygons which are visible etc.). However, in general, we can state that a model with a large number of polygons will be rendered slower than a model with fewer polygons. In a time-critical graphics system it therefore becomes necessary to find the most optimal polygonal representation of an object so that we can minimise any latencies involved in displaying that object.

In addition to this requirement, a common technique which is used to modulate the frame rate of a simulation on-line is level of detail (LOD). This entails the use of a number of representations for a single object, each varying in polygonal complexity. The computer system can then decide which particular model to display at any instant based upon a number of criteria, such as: how overloaded the graphics system is,[2, 3] the distance of the object from the viewpoint,[4, 5] the size of the object on screen,[6] the velocity of the object across the screen[7, 8] or the displacement of the object from a focus point.[7, 8, 9]

In order to implement this, we require some mechanism to take an original polygon description of an object and create another such description which retains the general shape of the original model, but contains fewer polygons. A number of techniques to perform this have been proposed, and will be discussed presently. However, most of these offer no immediate provision to accurately control the perceptual effect of the degradation. That is, the extent of the degeneration cannot be implicitly restricted to a certain visual threshold: the principal aim is solely to reduce the polygon count. As a result, we have the common situation where a reduced LOD model is used in a graphics system, but the selection between different levels of detail incurs a noticeable flicker (the so called 'popping effect'). This is due primarily to the fact that the simplified models have been arbitrarily reduced and so there is little basis to formally define the LOD modulation thresholds. Consequently this is normally performed on a trial and error basis,[5] and some degree of hysteresis may also be incorporated to compensate for the visual incongruities.[10]

In response to this problem, this paper presents the concepts and design decisions behind the SCROOGE system, developed at the University of Edinburgh. SCROOGE attempts to incorporate a method of perceptual predictability into a generic polygon reduction algorithm in an effort to eliminate the undesirable visual artifacts mentioned above.

2. Related Research

There are a number of approaches which can be taken to reduce the polygon count of an object. Most of these can be categorised as either *local* or *global* techniques (although some algorithms do not fit neatly or exclusively into this classification). Local techniques operate on individual primitives such as vertices, adjacent edge segments or some polygon characteristic.[10] Global techniques attempt to optimise the polygon mesh based upon more general, high-level features of the model. The following paragraphs present a cross-section of contemporary polygon reduction schemes and offer some comment on their utility. Of these, Sections 2.1–2.5 can be considered local techniques; whereas Sections 2.6–2.9 offer more global solutions.

2.1. Vertex Collapsing

In this approach, the model is segmented into a number of sub-volumes. Then all vertices which exist within each sub-volume are collapsed to a single averaged position.[11] This technique is used by the ModelGen™ and MultiGen™ packages[12] from MultiGen Inc., as well as the Optimize tool supplied with the WebFORCE™ package from Silicon Graphics Inc. (SGI). The technique is relatively simple to implement but the results can be very coarse and often drastically degrade the form of the model. One refinement which can be used to reduce this effect is to introduce a tolerance factor. This essentially limits the collapsing of vertices to within a certain distance of the averaged vertex.[12] Also, vertices which lie on the convex hull of the object can be preserved in an attempt to retain the general form of the object.

2.2. Vertex Removal

Another vertex-based technique is to remove selected vertices from the mesh. The decimation algorithm reported by Schroeder, Zarge & Lorensen works on this principle.[13] In their system, every vertex is analysed for possible removal (based upon a distance criterion). If the vertex is to be removed then it, and every polygon using it, are deleted from the object description. A local re-triangulation scheme is then used to patch up any resulting holes. This process can be repeated until a specified percentage reduction has been achieved. The decimation algorithm was illustrated on a number of terrain and volume data: these showed far more visually acceptable results than the technique of vertex collapsing described above.

2.3. Edge/Polygon Curvature

By comparing the normals of adjacent polygons, or measuring the angle between connected edges, we gain a measure for the curvature of the mesh at that point. Several techniques have taken advantage of this information in order to reduce detail around areas of low curvature, whilst retaining detail around high curvature regions. This approach proceeds under the conjecture that regions of high curvature relate to visually important features within an object: regions which contribute strongly to the shape of an object. This is a valid assumption for most cases.

Hamann used this notion in his data reduction scheme.[14] This worked by iteratively removing a triangle from a mesh, based upon the curvature values at its vertices, and then re-triangulating the local changes. The principal advantage of curvature based reductions is that they remove nearly planar surfaces which do not affect the overall form of an object and hence these techniques provide good shape constancy between degraded models.

2.4. Polygon Area

The dimensions of a polygon can also be used as a reduction criterion. For example, polygons which are below a threshold size can be removed from the model; or a group of polygons which are below a threshold area can be merged into a single description. Kemeny reports that such a technique was used in the GENIE system to generate different levels of detail for their driving simulator.[4] This system worked by projecting the model onto a 2D grid and reducing the representation so that only one polygon occupied each grid square (or 'rexel'). If the size of each rexel were made to represent a single screen pixel, then this would provide a means of removing all sub-pixel detail. However, because the reduction operates on a single 2D projection of the 3D model, this will only be valid for the sampled orientation—i.e. it is not a viewpoint invariant method.

In the Viper system, Holloway attempted to improve system performance by simply terminating the display of an object if the graphics system became overloaded.[15] He notes that to reduce the visual consequences of this action, the polygons within each object should be sorted in order of size so that the larger polygons are always displayed first; leaving the smaller polygons to be successively removed as the system becomes overloaded.

2.5. Adaptive subdivision

Thus far, most of the reduction schemes could be described as top-down, in that they take the original polygon description and attempt to remove detail from it. By comparison, adaptive subdivision is a bottom-up approach. It begins with a crude approximation of the object and then recursively refines this by subdividing the model where it varies most from the original mesh.[16] Schmitt *et al.* used bicubic Bernstein-Bézier surface patches for the approximation and developed a metric to model the 'closeness' of the approximation to the original data. DeHaemer & Zyda used a similar technique, but they employed simpler polygon approximations because their graphics workstation did not support the rendering of filled and shaded bicubic surfaces.[17] DeHaemer & Zyda comment that this technique was particularly suited to their task of simplifying complex 3D range-data obtained through the laser scanning of real-world objects.

2.6. Polygon Re-tiling

The re-tiling technique formulated by Turk optimises a polygon mesh by introducing new vertices to the mesh and then discarding the old vertices to form a new representation.[18] This involves creating an initial triangulation of the surface with a user-defined number of vertices. These new points are pseudo-randomly positioned in the planes of the existing polygons and then successively repelled by their neighbours in order to create a uniform distribution. Once this relaxation process has converged, the old vertices are removed one by one and the surface is locally re-tiled at each step in order to retain the topography of the original surface. Turk notes that this technique works best for curved surfaces (such as the iso-surfaces from medical data or molecular graphics) and that it is less suited to angular entities such as buildings, furniture or machine parts.

2.7. Mesh Optimisation

Hoppe *et al.* present a triangular mesh simplification process[19] which was based upon their surface reconstruction work.[20] This technique introduced the concept of an energy function to model the opposing factors of polygon reduction and similarity to the original topography. The energy function was used to provide a measure of the deviance between the original mesh and the simplified version. This was then minimised to find an optimal distribution of vertices for any particular instantiation of the energy function. Hoppe *et al.* record that their mesh optimisation technique successfully distributed vertices in relation to surface curvature (i.e. areas of high curvature were densely coded with vertices: whereas relatively flat regions contained fewer vertices), thus providing a high degree of shape constancy between model approximations.

2.8. Object Replacement

Sewell suggests an intriguing approach to approximating certain classes of objects.[21] He developed a mechanism which computes a simple replacement primitive for a complex grouping. For example, his initial system analysed a model and attempted to approximate parts of it with spheres, boxes or ellipsoids (this technique has perceptual grounding in the concept of *geons*, proposed by Biederman.[22] This theory suggests that human object recognition is based upon identifying a small number of primitive shapes within an object). Sewell comments that this approach can produce substantial complexity reductions because it can decompose clusters of objects which most other techniques would treat as separate entities. However, he notes that the technique can generate various visual artifacts as a result of this.

2.9. Wavelet Encoding

Wavelets are a means of hierarchically decomposing a function so that it can be described as a coarse general form, augmented by a series of details at different scales.[23] Their applicability to polygon reduction is illustrated by Stollnitz, DeRose & Salesin who lucidly explain how a coarse approximation to an original object can be formed by omitting a number of these small detail terms (wavelet coefficients) when rebuilding the model.[24] Eck *et al.* note that this system offers potential in a number of application areas; including object compression, LOD and multi-resolution editing.[25] Indeed, wavelet-based solutions are becoming increasingly popular and offer a reasonable mechanism to limit the effect of a degradation.

3. Developing A Perceptual Approach

The visual acuity of the human vision system is principally dependent upon three factors: size, orientation and contrast.[26, 27] The relationship and sensitivity to these three phenomena has been extensively investigated by vision scientists[28] and can be accurately modelled through a number of functions know as Contrast Sensitivity Functions (CSFs).[29] These curves record the ability of an observer to resolve a series of alternating light and dark bars based upon their size and contrast. (N.B. the size of a feature is measured in terms of *spatial frequency*, which is inversely proportional to size). Figure 1 below presents a typical CSF for a standard observer. This graph illustrates that the human visual system has a peak visual acuity (occurring at around 2 c/deg) and also that it has a finite bandwidth (which is significant between 1–10 c/deg approximately).[30] This evidence is presented to enforce the size-sensitive nature of the visual system. Of course, when viewing a computer monitor screen, the observer's effective visual acuity will be influenced by the resolution of the display device too.

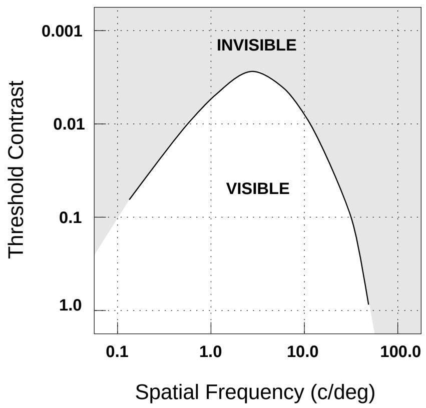

Figure 1: *An example Contrast Sensitivity Function for static detail. This represents the size-sensitivity of the human visual system in relation to the contrast of the stimulus.*

3.1. Applying the Perceptual Principles

Based upon these studies, we can formally state that in order to restrict the perceived effect of a reduction, we must incorporate some method to limit the spatial extent of the reduction (in an orientation-independent manner) and some mechanism to model the degeneration of luminance information.

Recent theories of visual perception now favour the view that the vision system analyses size and orientation information on a local basis.[31] This gives us perceptual evidence to favour a locally biased approach to polygon reduction, rather than a global one. However, although detail should be varied at a local level, the extent of a locality can vary (effectively, become more global) depending upon the perceptual threshold being applied. This suggests that a hybrid local/global scheme may be the best approach in this instance: i.e. a method which operates at a local level, over a range of different scales. (Again, this exhibits obvious conceptual parallels with the currently accepted 'multichannel' model of visual perception; which postulates the existence of a network of receptive fields in the eye and brain that are selectively sensitive to detail over a range of scales.[32])

From the literature review section above, we can observe that there are a number of possible ways to implement polygon reduction based upon local features such as vertices, edges or polygons. Arguably the best of these methods for our application is one which utilises curvature information, because this provides good control over how faithfully the approximation retains the shape of the original model. This is important not only from a perceptual standpoint, but also because most graphics renderers use polygon or vertex normal information to calculate the colour of a polygon. As a result, if we overtly degrade the curvature of a surface then the colour across the surface may be noticeably affected also.[21] We therefore wish to control the degeneration of curvature information as much as possible.

As the first statement in Section 3 implies, our perception of visual detail is based not only upon spatial criteria, but also on colour information.[33] Most polygon reduction schemes do not consider this characteristic of a model, and tend to be concerned solely with optimising the geometric description. However, to be complete, we must take into consideration the fact that polygons can be displayed with a variety of surface properties such as colour and texture.

Based upon the above discussion, the initial goals of the SCROOGE system can be summarised as follows:

1. The system should incorporate some mechanism to limit the visual consequences of the reduction.
2. The system should maintain, as much as possible, the form (or shape) of the original model.
3. The system should handle coloured and textured polygons in a principled manner.

4. Algorithm Overview

The remainder of this paper will describe the implementation and operation of the SCROOGE system. Briefly, this works by segmenting the object into a number of sub-volumes. For each sub-volume, all of the polygons which are wholly within that subset are considered for degradation. The degradation scheme proceeds by trying to collapse certain vertices so that

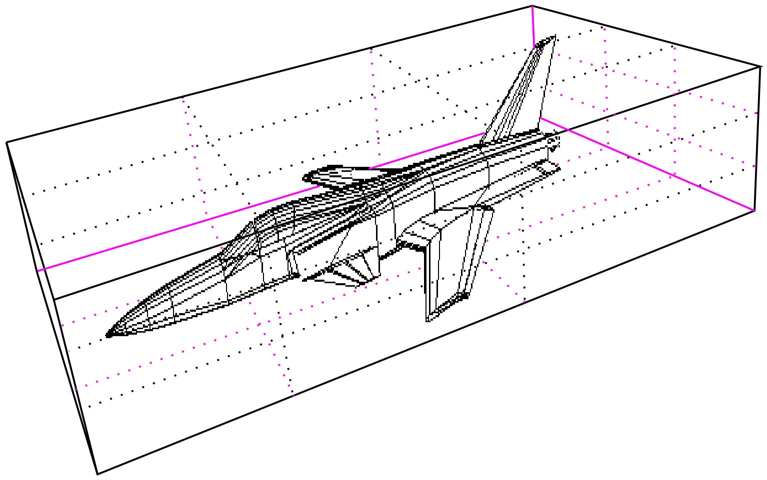

Figure 2: *An example model showing a sample sub-volume segmentation. The contents of each sub-volume is considered independently for possible degradation.*

they become co-linear with two neighbouring vertices. Then a reconstruction phase attempts to rebuild the model in an optimal fashion, removing these co-linear vertices where necessary. The top-level algorithm for the system can be encapsulated as:

1. Normalise the geometrical data.

2. Segment model into sub-volumes.

3. For all polygons which are wholly within each sub-volume:

3.1 Degrade all edges below threshold

3.2 Reconstruct all polygons.

4. Optimise the geometrical data.

Step 1, the Normalisation process, is a preprocessing stage which attempts to provide the geometric information in a standardised and consistent form. This includes removing multiply-defined vertices, edges and polygons, sub-polygons and other undesirable features. The operations which constitute steps 2 and 3 form the basis of the polygon reduction algorithm proper, and will be discussed in further detail promptly. Step 4 involves a final pass over the geometric data to remove any unused vertices etc. from the polygon mesh.

5. Sub-Volume Restriction

The mechanism which was used to restrict the extent of the reduction algorithm was to segment the model into a number of sub-volumes. This was done by finding the bounding box of the object and then choosing a granularity with which to divide this volume into smaller rectangular volumes. Figure 2 illustrates this concept by presenting a model with an example subdivision grid overlayed. (Note that the sub-volume need not be rectangular in shape, e.g. a spherical volume could easily be used instead. This may be more appropriate for certain applications.)

For each individual sub-volume, all polygons which are *wholly within* that volume are located and passed onto the next stage for decomposition. This ensures that only polygons whose extents are definitely smaller than the threshold volume are considered for reduction (i.e. polygons which constitute below-threshold detail). This task was made efficient through the use of a 3D data structure to rapidly locate all polygons within a desired volume (a hash table of 2D linked lists was employed; although other data structures are feasible, such as an octree). By altering the size of the grid we gain the ability to vary the scale of reduction which is performed: a fine grid mesh will remove only very small features; whereas a coarse grid will remove larger features.

The overall effect of this mechanism is to ensure that *no changes are made to the model above a certain threshold size.* Therefore, if we choose a grid volume which projects to the size of a single pixel for a particular viewing distance, then we gain the ability to automatically remove any detail which will not be displayed on the output device.

One limitation of this approach is evident: any polygons which lie on the boundary of a sub-volume will never be considered for reduction (because they are not wholly within any sub-volume). The solution to this problem is to overlap successive sub-volumes. E.g. if the dimensions of the sub-volume are w, h and d respectively, then possible positional offsets for a sub-volume can be given by: $(w \times l/2, h \times m/2, d \times n/2)$, where $l, m, n \in [0, 1, 2 \ldots]$.

6. Edge Degradation

The actual degradation scheme which was employed is based around a routine that takes two connected edges, defined by their vertices v_1, v_2 and v_3, and collapses these to a single edge. This is done by altering the value of vertex v_2 to its perpendicular projection onto the directed line segment $v_1 \rightarrow v_3$. The details of this procedure are illustrated in Figure 3 and described in the associated text.

Note that the actual edge definitions (and hence the polygon descriptions) are not modified at this stage: we only change the value of selected vertices (the geometry), not the structure of the model (the topology). The topology rationalisation is performed in the subsequent reconstruction stage.

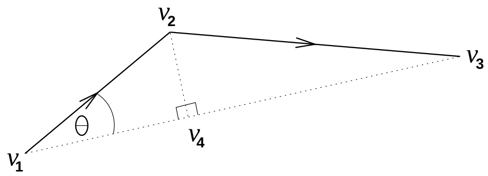

Figure 3: *Collapsing an intermediate vertex, v_2, onto the line segment, $v_1 \rightarrow v_3$.*

Given the three vertices, v_1, v_2 and v_3, we want to find the position of vertex, v_4, which is the perpendicular projection of v_2 onto the line segment $v_1 \rightarrow v_3$. We shall begin by making the following edge vector, $\overline{e}_n$, and length, l_n, definitions:

$$\begin{aligned} \overline{e}_1 &= v_2 - v_1, & l_1 &= |\overline{e}_1|, \\ \overline{e}_2 &= v_3 - v_1, & l_2 &= |\overline{e}_2|, \\ \overline{e}_3 &= v_4 - v_1, & l_3 &= |\overline{e}_3|. \end{aligned}$$

Given the above, it can be trivially shown that the location of v_4 may be efficiently expressed as follows:

$$v_4 = r\overline{e}_2 + v_1,$$

where,

$$r = l_3/l_2 = (\overline{e}_1 \cdot \overline{e}_2)/l_2^2.$$

6.1. Enforcing Sub-Volume Restriction

The sub-volume restriction phase provides us with the assurance that the available polygon subset is wholly contained within the current sub-volume; and so the effects of the degeneration can be constrained to a predefined spatial boundary. However, it is likely that a polygon may contain a vertex which is common to a polygon not wholly within the current sub-volume (as illustrated in Figure 4). In this situation, if we alter the position of such a vertex, then we must also alter the representation of a polygon which is outwith the sub-volume restriction. This side-effect cannot be tolerated if we require that no visual changes are to be made to the model above the sub-division threshold.

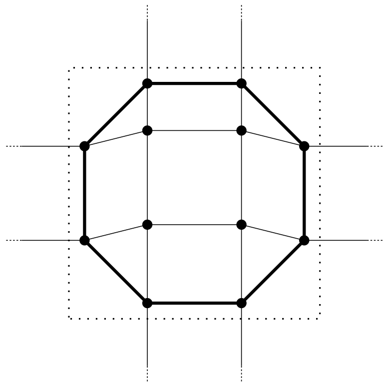

Figure 4: *A simple polygon mesh showing the extent of a sub-volume restriction (the dotted square). The bold demarcation illustrates the boundary edges within the sub-volume. Vertices on this boundary should not be considered for degradation.*

To enforce this requirement, a proviso must be added that no degeneration is attempted on a vertex which is shared by a polygon existing outwith the current sub-volume (e.g. a vertex which lies on the boundary edge of the sub-volume).

6.2. Incorporating Scale into the Model

We now have a means of replacing two edges by a single edge; but we do not wish to degrade all edges in a model in this way because some edges will be more visually important than others. We therefore need some criterion—or criteria—for choosing which edges to degrade. As mentioned previously, a good measure would be to utilise the curvature between two edges as a selection criterion, i.e. the angle between the two edges. This would have the effect of degrading fine variations in contour, but retaining the general form of the object.

However, the size of the feature being degraded is

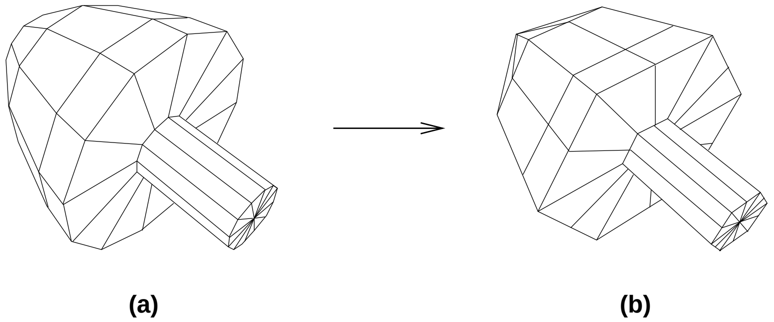

Figure 5: *(a) An original model (the hub of an aircraft's propeller), and (b) the same model after it has undergone the edge degradation phase.*

also of relevance here. For example, a highly angular feature which is very small can be removed without affecting the overall form of the object (e.g. the buttons on a television set), but if we collapse a very large angular feature, then the shape of the object may be severely distorted. We therefore require a scheme which will degrade edges below a certain threshold curvature, where the value of this angular threshold is attenuated with the scale of the particular edge pair (i.e. the size of the edge pair relative to the dimensions of the sub-volume).

To encapsulate this, we can introduce a function, $l(\theta)$, which will return the threshold length of an edge based upon θ, the angle of deviance between edges $v_1 \rightarrow v_2$ and $v_2 \rightarrow v_3$. We can then compare this value with the actual length between the two edges, $|v_1 \rightarrow v_3|$, in order to decide whether vertex v_2 should be considered for degradation.

In the implementation, a simple linear relationship was chosen between the scale and curvature of an edge. Using the general equation for a straight line, $y = mx + c$, we can subsequently define $l(\theta)$ as follows:

$$l(\theta) = m\theta + c,$$

where,

$$\begin{aligned} m &= l_{max}/(\theta_{max} - \theta_{min}), \\ c &= -m\theta_{max}. \end{aligned}$$

The variable, l_{max}, is the length of the largest possible edge within the sub-volume (e.g. the distance between two diametrically opposed corners of the sub-volume); and θ_{min} is the curvature threshold for any edge of length, l_{max}. Conversely, θ_{max} is the curvature threshold for the smallest possible edge in the sub-volume, notional taken as length 0. We therefore require suitable instantiations for θ_{min} and θ_{max}. In the former case, the value of $5°$ was chosen (Ware and Knight cite the resolvable orientation difference of the human visual system as being $5°$[30]). The value of θ_{max} is more arbitrary. Varying this parameter can affect the degree of degradation which is performed within a sub-volume. In this respect, a value could be adopted to match a particular class of model: e.g. a small value could be used for smooth, curved objects; whereas a larger value could be used for more angular objects. Figure 6 illustrates this relationship where $\theta_{max} = 90°$.

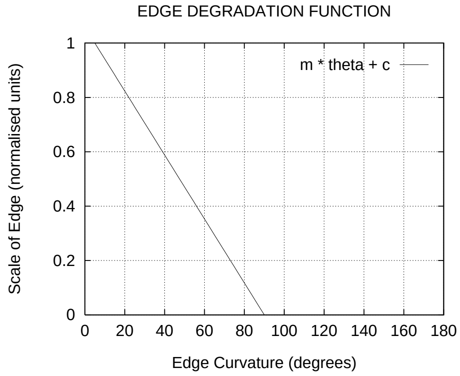

Figure 6: *The function, $l(\theta)$, which governs whether an edge pair should be degraded based upon its scale and curvature.*

Basically, this function dictates that every edge pair whose curvature is $< \theta_{min}$ will always be degraded, every edge pair whose curvature $> \theta_{max}$ will never be degraded, and the threshold for all intermediate angles is moderated linearly by the scale of the edge pair.

6.3. Degradation Scheme Overview

To summarise the above discussion, the implementation of the edge degradation stage can be described by the following steps:

1. Calculate the co-efficients for the degradation threshold function (based upon the dimensions of the current sub-volume).

2. Index all directed edges for every polygon in the current sub-volume.

3. For every directed edge:

   a. Locate a connected edge (if there are > 1 such edges, then the edge which produces the smallest angle between the first edge is chosen).

   b. For this pair of connected edges:

      i. Calculate the length of the edge between the two non-common vertices.

      ii. Calculate the magnitude of the angle between the two edges.

      iii. Calculate the threshold length for that angle.

      iv. If the length is less than the threshold, then degrade the common vertex so that it lies on the line between the two non-common vertices.

Figure 5 presents an example of how this algorithm can be used to degrade the curvature of a surface. In this example, the base of the hub section is modelled as a 12-sided plane. In the degraded section this has been reduced to a 6-sided plane.

7. Surface Reconstruction

The surface reconstruction phase takes all of the polygons in the current sub-volume and attempts to rebuild these into a more compact mesh. This involves merging any neighbouring polygons which are roughly coplanar (to within a desired threshold, e.g. 5°). This causes a reduction in complexity due to the efforts of the previous stage: the edge degradation phase alters the position of a number of vertices so that they become co-linear with two vertices on either side of it. This is done in an attempt to increase the frequency of planar polygons within the model and hence increase the number of polygons which are merged by the surface reconstruction phase. Figures 5 and 7 attempt to illustrate the two principal stages of this concept.

In practice, simply performing the surface reconstruction phase on a model can actually reduce the polygon count on its own. This is because, very often, 3D models are designed within a CAD package which is not concerned with the optimal polygonal representation of that object. It may therefore contain many surfaces which are planar and could be merged into a single description. However, the major factor for the complexity reduction in the reconstruction stage is due to effects of the edge degradation phase.

7.1. Applying Polygon Constraints

Each polygon which is formed by the surface reconstruction phase is then passed through a normalisation process to ensure that they are valid and optimal. This can include the following operations:

- Remove any co-linear vertices
- Reduce concave polygons → convex polygons
- Reduce non-planar polygons → planar polygons
- Triangulate all polygons

Each of these constraint operations can be employed independently by the system (through a number of user-selectable options). This allows the geometry of the resulting model to be customised to a particular graphics system. For example, if a graphics system requires that the model be described in triangles, then this can be output; or if arbitrary-sized convex polygons are supported, then these can be output too (the default operation is to remove all co-linear vertices and output convex, planar polygons).

(N.B. The facility to reduce non-planar polygons into planar ones is a remnant of an earlier incarnation of the SCROOGE system. The current manifestation of the system does not require this operation because polygons are merged into a group only if they are within a certain degree tolerance of the first polygon in the group.)

7.1.1. Removing Co-linear Vertices

As already noted, the edge degradation stage attempts to create vertices which are co-linear with two neighbouring vertices. These vertices can be selectively removed from the final mesh and thus reduce the complexity of individual polygons (i.e. the number of vertices in a polygon) as well as reducing the polygonal complexity (i.e. the number of polygons in the model). The user can however choose to leave these vertices in the polygon mesh. This may be desirous if, for example, the model is to be Gouraud shaded. In which case, removing the co-linear vertices may introduce slight tears in the shading between adjacent polygons (due to small rounding errors in the interpolation of vertices).

7.1.2. Reducing Concave Polygons

Many graphics renderers cannot properly handle arbitrary concave polygons. A facility is therefore provided to reduce these into a number of simpler convex polygons (a simple convex polygon can be defined as one in which all internal angles are $\leq 180°$). This reduction can be achieved by referring to the normal of each vertex: the normals for each vertex of a convex polygon will all be oriented in the same direction; but

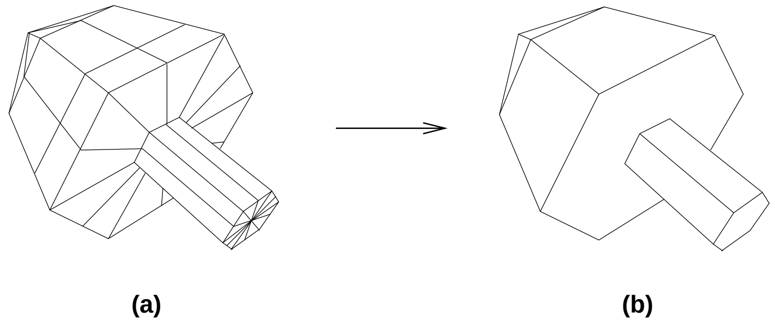

Figure 7: *(a) The aircraft hub model from Figure 5 after the edge degradation phase, and (b) the same model after it has undergone the surface reconstruction phase.*

if a vertex is introduced which makes the polygon concave, then this vertex will have an oppositely aligned normal.

7.1.3. Triangulating Polygons

A number of graphics systems require all polygons to be supplied as triangles. We should therefore be able to convert arbitrary polygon descriptions into collections of triangles. This process was made easier by first employing the above constraint routines so that we only have to deal with planar, convex polygons. Then a simple convex polygon triangulation scheme was performed on each polygon.

7.2. Rendering Considerations

Figure 7(b) conveniently highlights a couple of pertinent considerations for polygon reduction in general. As can be observed, the base of the hub section (where it contacts with the cylindrical section) has been merged into a single 6-sided polygon. Whereas beforehand, this was modelled with 12 individual polygons which extended internally up to the sides of the cylinder. I.e. the original grouping looked like a hexagon with a hexagonally-shaped hole in the centre, and we have replaced this with a single hexagonal polygon; effectively, ignoring the hole. This gives us a much better reduction ratio, but it also raises two potential problems:

1. It would be possible for the point of contact between the hub base and the cylinder base to incur two coplanar polygons which lie on top of each other. Some graphics renderers do not handle this situation well and the result can be an oscillatory flickering between polygons. This problem was circumvented by removing all sub-polygons from a model, i.e. polygons which lie inside another polygon and which are coplanar with it.
2. The hub base no longer shares any vertices with the cylinder section. This could theoretically introduce small tears into a model if the object were Gouraud shaded, for the reasons outlined above. If this is considered an important issue, then such a situation cannot be allowed to arise, i.e. if a grouping of planar polygons contains a hole, then the grouping must be segmented further. This process was deemed unnecessary in the SCROOGE system because this application is concerned primarily with the generation of models to be displayed at a distance such that these small artifacts will have no visible effect.

7.3. Colour Blending

Modern graphics applications do not usually display objects as simple wireframe models or bland single-coloured entities. Instead, an object will normally contain various polygons of different colours and may also included texture mapped polygons. How then should the surface reconstruction process cope with the merging of polygons with different surface properties?

If we consider the function of the human visual system once more: our vision system has an amazing ability to integrate over features which are below threshold. E.g. If we have a fine mesh of black and white squares and display this at a distance such that it is below the threshold of vision, then we can no longer resolve discrete black and white regions. Instead we perceive a single wash of colour which exhibits the average contrast of the two component colours.[33] This phenomenon is used to great effect in the newspaper industry where halftoned patterns can be used to portray a wide range of grey-scale levels with only a single, black pigment.

To apply this principle to our problem, it was decided that the colour of a new polygon (which is formed by merging several other polygons) should be based upon the average colour of all the component polygons. I.e. the RGB values for each polygon are averaged. This results in an averaging of luminance (achromatic) as well as colour (chromatic) information. However, the case is not quite that simple because the relative sizes of the different colour stimuli will affect the perceived result. E.g. if the black squares in our mesh were slightly larger than the white squares, then the below-threshold wash would appear darker than when the squares were equally sized (again, this principle is used to good effect in halftoned newspaper photographs). Therefore, the resulting colour of a polygon is found by an *area-weighted averaging* of the RGB colours for each component polygon.

The principles for dealing with texture mapped polygons are exactly the same. In fact, we can think of a single texture mapped polygon as fulfilling the same visual function as a collection of tiny, individually-coloured polygons. Stated this way, we can suggest that a good replacement for a texture mapped polygon would be to use an untextured polygon whose colour is found by averaging all of the pixels in the texture map image. This also has the desirable side-effect of replacing textured polygons with less complex primitives. Texture mapped polygons normally incur a greater computational overhead than simple flat or smooth shaded polygons. Therefore, removing these primitives should improve the rate at which the model is rendered: hence adding another facet to the degeneration algorithm.

8. Results

Figure 8 illustrates the effect of the polygon reduction algorithm on a model containing 1,050 triangles (where $\theta_{max} = 45°$). Figures 8(b)–(d) present the reduced versions using sub-volume sizes of one sixty-fourth, one eighth and all of the object's bounding volume, respectively (i.e. the widths of each sub-volume are one quarter, one half, and equal to the width of the bounding volume, respectively). This implies that the model in Figure 8(b) can be used when one sixty-fourth of the object's bounding volume projects to a single pixel on screen, and similarly for Figures 8(c) and (d).

Table 1 below reports the complexity of each model in terms of the number of polygons and vertices which it contains.

Model	Polygons	Vertices	Update Rate
Figure 8(a)	1050	524	5.9Hz
Figure 8(b)	747	496	8.1Hz
Figure 8(c)	300	302	17.2Hz
Figure 8(d)	186	242	30.2Hz

Table 1: *Comparison of Degraded Models.*

The Update Rate column provides average frame rates which were achieved when each model was rendered to the screen (using Open Inventor V2.0 on an SGI RealityStation). The absolute value of these figures is not important because these will vary across platforms and graphics packages, but their relative magnitudes provide some indication towards the performance benefits which can be reaped when employing each degraded model. Note also that the above table reports the number of (convex and planar) polygons in a model, whereas Figure 8 states the number of triangles for each model (after undergoing an implicit triangulation process by the Open Inventor package).

Polygon reduction algorithms are normally designed to operate off-line; where rapid execution time is not a stringent factor. However as interest in real-time manipulations of virtual environments increases, we should also consider the computational components of polygon reduction schemes. With regards to the costs of the current algorithm, execution times were collected for the above reductions using the standard UNIX time utility (on the aforementioned platform). Execution times of 7.89, 6.34 and 4.72 seconds were found for the generation of Figures 8(b), (c) and (d) respectively. This exhibits a smaller computational cost for larger reduction ratios. This somewhat paradoxical result can be attributed to the increased complexity caused by processing a larger number of independent sub-divisions of the object when a more fine degradation is required.

9. Conclusions

This paper has presented a mechanism for polygon reduction—driven by models of human visual perception—in an effort to produce a system which can offer some means of predicting the perceptual side-effects of a reduction. The algorithm is simple and intuitive to use; requiring only one parameter to operate: the granularity of the sub-volume grid (although, optionally, the user could be given control over the θ_{max} variable too).

The reduction scheme can offer appreciable reductions in the polygonal complexity of a model, although these do not necessarily exceed the performance of existing algorithms. However, this is to be expected due to the additional constraint of retaining a degree of perceptual predictability. A number of techniques potentially provide good reduction ratios (such as

Sewell's object replacement approach), but the resulting meshes are often notably deformed. Therefore, if these models where to be used seemlessly in a distance-based LOD scheme, then they could only be employed when the object is very small on the output device, perhaps only a few pixels in size. In order to extract the most performance benefit out of the graphics system, it would be desirable to have models which could be switched in at far larger screen-size thresholds, with no perceivable visual artifacts. The algorithm presented in this paper is therefore suggested as one possible mechanism to achieve this goal.

The principal contribution of this paper is the notion of restricting the spatial extent of a degeneration. The actual reduction scheme proper is a more secondary concern; and indeed other reduction schemes could feasibly be used instead of the edge degradation approach chosen here. Further work can therefore be identified; including an investigation into the benefits and effects of using other degradation schemes in conjunction with the sub-volume restriction paradigm.

References

1. M. F. Deering, "Data Complexity for Virtual Reality: Where do all the Triangles Go?", in *Proceedings of the VRST'94 conference*, pp. 357–363, (1994).
2. B. J. Schachter, "Computer Image Generation for Flight Simulation", *IEEE Computer Graphics and Applications*, **1**, pp. 29–68 (1981).
3. M. M. Wloka, "Thesis Proposal: Time-Critical Graphics", Technical Report CS-93-50, Dept. of Computer Science, Brown University, Providence, RI, (November 1993).
4. A. Kemeny, "A Cooperative Driving Simulator", in *Proceedings of the International Training Equipment Conference and Exhibition*, (London, UK), pp. 67–71, (4–6 May 1993).
5. C. A. Chrislip and J. James Frederick Ehlert, "Level of Detail Models for Dismounted Infantry in NPSNET-IV.8.1", Master's thesis, Naval Postgraduate School, Monterey, CA, (September 1995).
6. J. Wernecke, *The Inventor Mentor: Programming Object-Oriented 3D Graphics with Open Inventor(TM), Release 2.* Reading, MA: Addison-Wesley, (1993). ISBN 0-201-62495-8.
7. L. E. Hitchner and M. W. McGreevy, "Methods for user-based reduction of model complexity for Virtual Planetary Exploration", in *Proceedings of the SPIE - The International Society for Optical Engineering*, vol. 1913, pp. 622–36, (1993).
8. T. A. Funkhouser and C. H. Séquin, "Adaptive Display Algorithm for Interactive Frame Rates During Visualization of Complex Virtual Environments", in *Computer Graphics (SIGGRAPH'93 Proceedings)*, pp. 247–254, (August 1993).
9. M. Reddy, "Musings on Volumetric Level of Detail for Virtual Environments", *Virtual Reality: Research, Development and Application*, **1**(1), pp. 49–56 (1995).
10. P. Astheimer and M.-L. Pöche, "Level-Of-Detail Generation and its Application in Virtual Reality", in *Proceedings of the VRST'94 conference*, (Singapore), pp. 299–309, (August 23–26 1994).
11. J. R. Rossignac and P. Borrel, "Multi-Resolution 3D Approximations for Rendering Complex Scenes", Technical Report RC 17697 (#77951), IBM Research Division, T. J. Watson Research Centre, Yorktown Heights, NY, (1992).
12. MultiGen Inc., San Jose, CA, *ModelGen2 Modeler's Guide*, (September 1994). Revision 14.1.
13. W. J. Schroeder, J. A. Zarge, and W. E. Lorensen, "Decimation of Triangle Meshes", in *Computer Graphics (SIGGRAPH'92 Proceedings)*, vol. 26, pp. 65–70, (July 1992).
14. B. Hamann, "A Data Reduction Scheme for Triangulated Surfaces", *Computer Aided Geometric Design*, **11**(2), pp. 197–214 (1994).
15. R. L. Holloway, "Viper: A Quasi-Real-Time Virtual-Worlds Application", UNC Technical Report No. TR-92-004, Dept. of Computer Science, University of North Carolina, Chapel Hill, NC, (December 1991).
16. F. J. M. Schmitt, B. A. Barsky, and W.-H. Du, "An Adaptive Subdivision Method for Surface-Fitting from Sample Data", *Computer Graphics*, **20**(4), pp. 179–188 (1986).
17. M. J. DeHaemer, Jr. and M. J. Zyda, "Simplification of Objects Rendered by Polygonal Approximations", *Computer & Graphics*, **15**(2), pp. 175–184 (1991). Pergamon Press, UK.
18. G. Turk, "Re-tiling Polygonal Surfaces", in *Computer Graphics (SIGGRAPH '92 Proceedings)* (E. E. Catmull, ed.), vol. 26, pp. 55–64, (July 1992).
19. H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, "Mesh Optimization", in *Computer Graphics (SIGGRAPH'93 Proceedings)*, pp. 19–25, (1993).

20. H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, "Surface Reconstruction from Unorganised Points", *Computer Graphics (SIGGRAPH'92 Proceedings)*, **26**(2), pp. 71–78 (1992).

21. J. Sewell, "Automatic Generation of Simple Replacements for Groups of Objects." The University of Cambridge Computer Laboratory, UK, (1995).

22. I. Biederman, "Recognition by Components: A Theory of Human Image Understanding", *Psychological Review*, **94**(2), pp. 115–147 (1987).

23. E. J. Stollnitz, T. D. DeRose, and D. H. Salesin, "Wavelets for Computer Graphics: A Primer, Part 1", *IEEE Computer Graphics and Applications*, **15**(3), pp. 76–84 (1995).

24. E. J. Stollnitz, T. D. DeRose, and D. H. Salesin, "Wavelets for Computer Graphics: A Primer, Part 2", *IEEE Computer Graphics and Applications*, **15**(4), pp. 75–85 (1995).

25. M. Eck, T. DeRose, T. Duchamp, H. Hoppe, M. Lounsbery, and W. Stuetzle, "Multiresolution Analysis of Arbitrary Meshes", Tech. Rep. 95-01-02, University of Washington, (1995).

26. F. W. Campbell and J. G. Robson, "Application of Fourier Analysis to the Visibility of Gratings", *Journal of Physiology*, **197**, pp. 551–566 (1968).

27. C. Blakemore and F. W. Campbell, "On the Existence of Neurones in the Human Visual System Selectively Sensitive to the Orientation and Size of Retinal Images", *Journal of Physiology*, **203**, pp. 237–260 (1969).

28. D. Lamming, "On the Limits of Visual Detection", in *Vision and Visual Dysfunction: Limits of Vision* (J. Cronly-Dillon, ed.), vol. 5, ch. 2, pp. 6–14, MacMillan Press, Ltd., (1991). ISBN 0-333-52713-5.

29. D. H. Kelly, "Motion and Vision. II. Stabilized Spatio-Temporal Threshold Surface", *Journal of the Optical Society of America*, **69**(10), pp. 1340–1349 (1979).

30. C. Ware and W. Knight, "Using Visual Texture for Information Display", *ACM Transactions on Graphics*, **14**(1), pp. 3–20 (1995).

31. J. G. Daugman, "Spatial Visual Channels in the Fourier Plane", *Vision Research*, **24**(9), pp. 91–910 (1984).

32. R. Sekuler and R. Blake, *Perception.* New York, NY: McGraw-Hill, Inc., third ed., (1994). ISBN 0-07-113683-5.

33. M. S. Livingstone, "Art, illusion and the visual system", *Scientific American*, **258**(1), pp. 68–75 (1988).

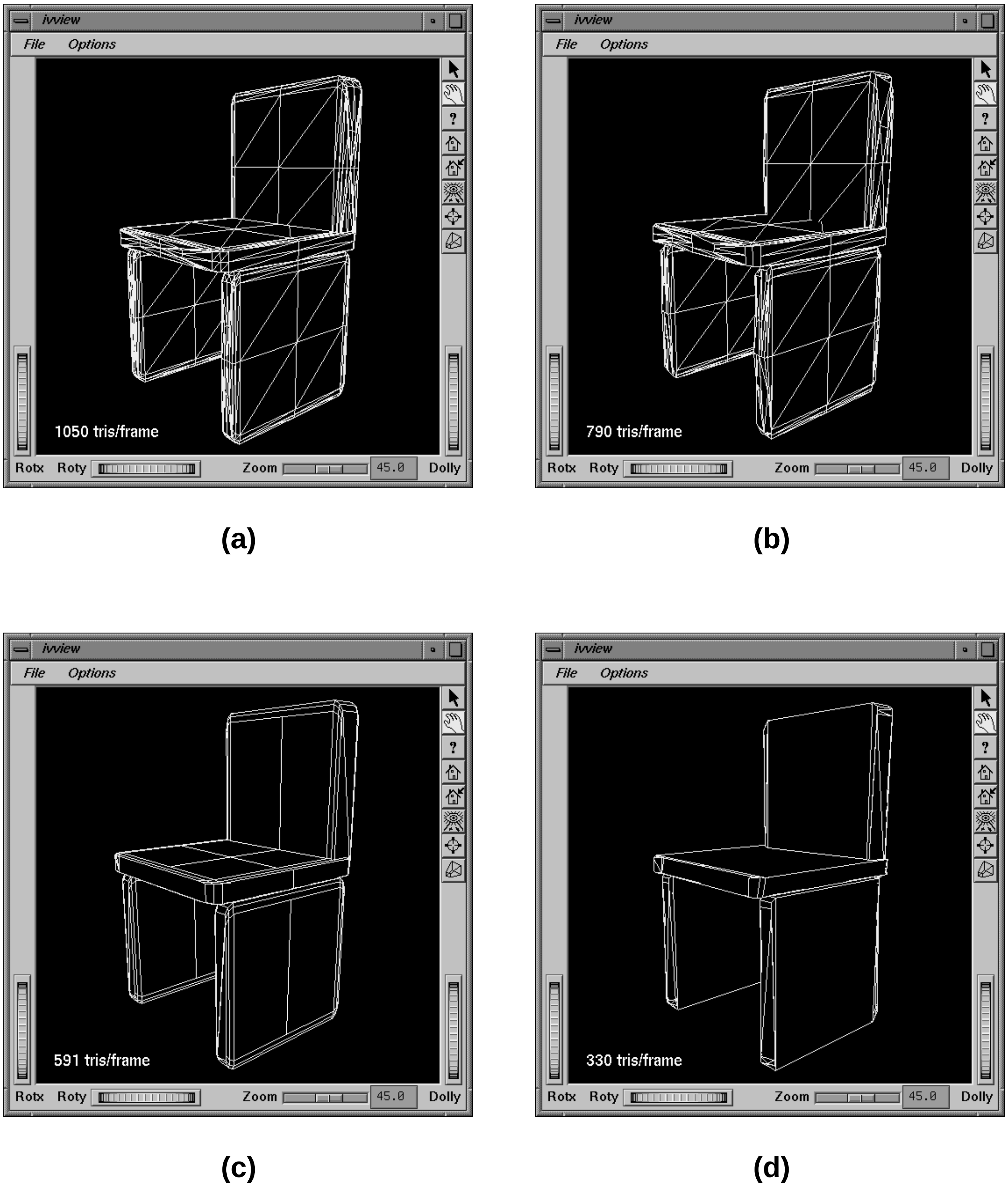

Figure 8: *Example reductions for a sample geometric model. (a) the original model, (b) reduced model using a sub-volume one sixty-fourth of the total volume, (c) reduced model using a sub-volume which is one eighth of the total volume, (d) reduced model where the sub-volume equals the entire volume.*