24-681 COMPUTER-AIDED DESIGN INTERIM PROJECT REPORT

Team Name: Mumbai Indians

Members: Ninad Kamat, Akash Sambrekar, Rahul Patil

MOTIVATION

Very often data received from STL files contains several details and features which requires a high number of elements for define them accurately. However, in practice, we may not always require such detailed features. It becomes essential to define ways in which a polygonal mesh can be approximated into mesh that can be represented using lesser elements while retaining several details and features of the mesh.

AIM OF PROJECT

This project aims at implementing one such technique. Here, given k partitions of a polygonal mesh, we wish to obtain an approximate mesh that accurately represents the mesh with much lesser polygons as compared to the one we started with.

ERROR METRIC

The error metric chosen for implementation is called the $\mathcal{L}^{2,1}$ metric which is defined as follows:

For a triangle T_i of area $|T_i|$, of normal n_i, and of associated proxy $P_i = (X_i, N_i)$, the $\mathcal{L}^{2,1}$ error is computed as follows:

$$\mathcal{L}^{2,1} = \|n_i - N_i\|^2 |T_i|$$

Now for region R_i, the optimal proxy normal N_i is simply the sum of product of area of triangle and its normal. We normalize this vector to make it a unit vector.

ALGORITHM USED

The basic high-level algorithm for achieving Geometric Approximation of a Polygonal Surface Mesh is as follows:

1. Clustering Triangles in the mesh based on the error metric
2. Once an accurate clustering is achieved, we extract anchor vertices from the clusters that define the clustering.
3. Given the Anchor vertices, obtain a new polygon mesh from these vertices.

We shall now see each of these steps in detail.

MODIFIED LLOYD CLUSTERING

In this method, we follow the following steps to obtain clustering:

1. We start with k random triangles and assign its barycenter and normal as initial proxy.
2. Using the error metric, we calculate neighbor of these triangle that is most similar to the triangle and add it to a global priority queue of triangles sorted according to the error metric after assigning the label of the proxy.
3. In this way we pass through the all the triangles using a growing algorithm such that any triangle appears in the priority queue thrice (from three of its neighbors).
4. We then traverse through the queue. If a triangle is appearing for the first time, we assigned the proxy indicated by its label and proceed. If it already is assigned a proxy, we do nothing.
5. We then calculate the new proxy for the given set of triangles associated with a given label as their barycenter and normal.
6. We repeat step 2 to 5 until we reach a point where the proxy does not change within a specified tolerance.

EDGE EXTRACTION

Once the polygons are clustered into their respective proxies, we need to find anchor vertices from which new mesh will be extracted. We follow below steps to achieve this:

1. Find vertices which are common to 3 or more proxy planes. Designate such a vertex as an anchor vertex
2. Obtain the projection of this anchor vertex on its respective proxy planes. Average out this projection to find the final spatial position of the anchor vertex
3. Maintain a pointer to the original vertex from which the anchor vertex originated.
4. Apply chord-subdividing algorithm to estimate the boundary curvature between the two proxies

REMESHING

Now that we have the new edges joining the anchor vertices which define the boundary of the proxies, we can proceed to triangulate each proxy region. We implement Constrained Delaunay Triangulation using Djisktra's multi-source shortest path algorithm which is described as below:

1. The flooding is initialized from each anchor vertex and constrained such that all the vertex near the boundary are first captured and colored depending on the closest anchor vertex
2. The flooding then continues to penetrate to the interior region unless each vertex of the region is visited and colored according to the closest anchor vertex
3. Since each vertex is shared by three triangles, we assign three colors to that common vertex to indicate the anchor vertex responsible for covering that vertex during the flooding process.
4. We now find all triangles such that all its vertices are assigned 3 different colors.
5. Now the new triangle is obtained by joining the anchor vertices such that each of these triangles contain a triangle with all its vertices having three different colors.
6. We can choose to remove the redundant edges in each proxy so as to get well shaped quads. However, we may want to check for edges which might induce concavity upon their removal.

DATA STRCTURES REQUIRED

SHELL FOR GEOMETRIC ASSOCIATIVITY

The shell data structure stores the connectivity information for a given polygonal mesh. It employs Hash Tables to store the associativity of polygons with edges and vertices. This data structure is useful to obtain neighboring polygons required for clustering and also to obtain polygons containing a particular vertex that is necessary in identifying Anchor vertices for edge extraction.

The data structure only stores the Vertex positions in form of an array of objects of the Vec3 class. Further, it also stores an array of Polygon information such as containing vertices and normal. The data structure is very well protected and cannot be modified by anyone from outside the class. This is essential as this represents the data that we begin with. Hence, all the tables are kept protected and can be only accessed through member functions. The class is designed in a way that it can provide answers to queries that any outside program may have through its object. Since, the class encapsulates a lot of operations. It is made such that there exists only one such object in the project, and data transfer is achieved by passing pointer to this object. Similarly, vertex and polygon information is passed through pointer to avoid excess memory usage.

LLOYD CLUSTER INFORMATION

The Lloyd cluster stores the information regarding associativity of a proxy to any polygon and vice versa. It allows queries for array of polygons with a given proxy and the proxy of a given polygon. Similar to Shell data structure, we protect the data within the class and only allow access through member functions designed to answer specific queries. Associativity is stored in the form of Hash Tables.

WORK DISTRIBUTION

1. Data Structures and Clustering: Ninad Kamat
2. Anchor Vertex Extraction: Akash Sambrekar
3. Remeshing and Constrained Delaunay Triangulation: Rahul Patil

WORK ACCOMPLISHED SO FAR

- Shell Data structure functional for clustering
- Lloyd Cluster Data handling programmed
- Anchor vertex extraction program complete

FUTURE WORK

- Vertex to Polygon Associativity in Shell.
- Clustering in Lloyd Cluster.
- Anchor Vertex addition for interpolating boundary.
- Constrained Delaunay Triangulation of Anchor vertices.