Structure Aware Mesh Decimation

Vishal Vijayvargiya, Chandan Mishra

18 Apr 2016

Abstract

This report presents an approach to mesh simplification which uses structure information of object to preserve key features like symmetry and perpendicularity while decimation. We first gather data about symmetrical parts of object mesh, build correspondence among vertices of symmetrical parts, and feed this information to decimation framework. Near planar triangular faces are grouped together under a planar proxy which is then used in multifarious structure preserving rules. An experiment is performed comparing proposed method with that of multiple choice scheme for decimation and results show better performance.

1. Introduction

Structure aware mesh decimation is quite useful in design and analysis of man-made objects specially in CAD/CAM applications. With recent advancement in 3D-modelling, most of the man-made objects such as buildings, tables, chairs etc. are designed using large and complex mesh polygons. And from analysis point of view, we require extreme simplification of these objects with structure preservation. Furthermore, existence of inherent symmetry and orthogonality (boundary, corners) are key features which we need to preserve while simplification as humans readily notice departures from these properties.

Popular simplification tools such as Qslim [1] and multiple choice scheme [2] works well in most of polygon mesh, however after very low level of details they deform structure itself. Because these methods simply work on local mesh information such as nearby face/vertex quadrics. So the goal of our project is to investigate ways in which structure of mesh can be maintained up to very extreme simplification. To support this goal, we tried an approach to include global information in our quadric error computation and defined rules to preserve boundary and corners. We performed a pipelined approach that consists of segmentation of symmetric parts followed by establishing vertex to vertex correspondence between detected parts and feeding this information to our decimation framework which work on local as well as global quadric error.

The remainder of this report is structured as follows. Related work is mentioned in Section 2 followed by giving overview of our approach in Section 3. Section 4 will provide a closer look of algorithms and Section 5 will contain experiment and results. Section 6 will list out conclusion and possible improvements.

2. Related Work

A number of approaches have been proposed by researchers around the world to come up with an improved decimation process. Qslim [1] algorithm uses greedy approach to mesh simplification and a novel way to calculate the cost of edge collapse by quadric error metric. Jianhua and Leif [2] used probabilistic optimization technique to perform decimation. Multiple choice decimation technique developed by them provided considerable speed-up compared to many traditional decimation schemes. Manish, David and Pierre [3] developed an algorithm that generates from an input tolerance volume a surface triangle mesh guaranteed to be within the tolerance, intersection free and topologically correct.

They used a pliant meshing algorithm to capture the topology and discover the anisotropy in the input tolerance volume in order to generate a concise output.

Survey paper from Niloy, Michael, Richard Zhang, Daniel and Martin [4] mentioned key concepts and methodological approaches towards efficient structure-aware processing. Aleksey, Joshua, and Thomas [5] investigated a framework that used symmetrization and symmetric remeshing to enhance the symmetries of a mesh, to decompose a mesh into its symmetric parts and asymmetric residuals, and to establish correspondences between symmetric mesh features. This information then could be very useful for mesh simplification process. Concept of planar proxies for shape preservation was used by David, Florent and Pierre [6], they presented various algorithms that use planar proxies.

3. Overview

The approach we use can be divided in three major components: segmentation, correspondence establishment and decimation. First two components are used as preprocessing steps which will extract important information to be used in decimation process.

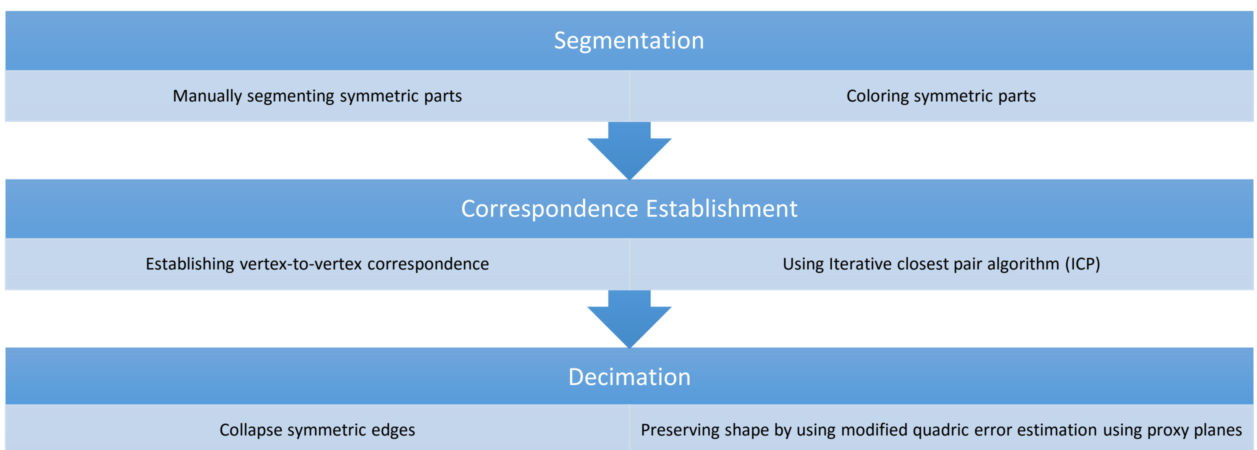

Fig 1: Pipeline showing major steps in our experiment

3.1 Segmentation

For simplicity and focus on other major part of our project, we chose to perform manual segmentation of symmetric parts. This can be easily performed using standard tools such as MeshLab by giving different color to different symmetric parts of model. Fig 2 gives an example how the table mesh looks in MeshLab.

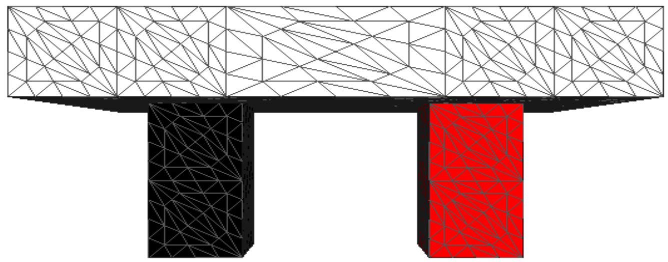

Fig 2: Segmented table with symmetric parts in black (source) and red(target).

3.2 Correspondence establishment

For vertex to vertex correspondence, we tried options like taking symmetry across common plane of reflection, however it requires manual input (identify and specify common plane of reflection) and also result was not much reliable because of issue highlighted in Fig 3. As you can see point P of source corresponds to bottom left of target however in actual source P should correspond to bottom right of target.

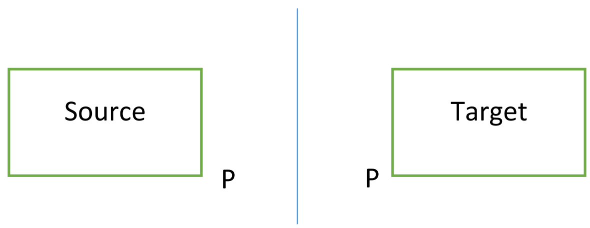

Fig 3: Plane of Reflection giving wrong correspondence

Finally, we choose customized Iterative closet pair algorithm (ICP) for segmented mesh to achieve correspondence between symmetric parts of model. Below is outline of algorithm.

<u>Mapping Algorithm based on Iterative Closet Pair:</u>

Input: Given a source mesh S and target mesh T
Output: A output file between S to T vertices correspondence

Pseudocode:

```
Initially assign error = infinity
While error > threshold && iterations > 10
    Compute initial alignment of source over target
    Apply alignment calculated in above step on source mesh
    Update error

End while

mapping = compute correspondence based on proximity of transformed source and
target

Return mapping
```

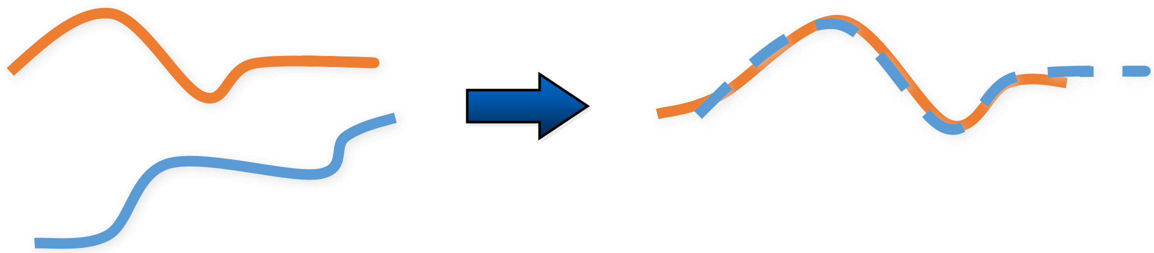

Fig 4: Iterative closet Pair overview

Figure 4 explains overview of aforementioned algorithm. The mapping generated will be used in decimation of edges between symmetric parts. Suppose edge E (V1->V2) is decimation candidate and reside in identified symmetric part, from mapping file fetch correspondence of V1 and V2 in other parts. Let it V1' and V2', so on decimation of V1->V2 from source also decimate V1'->V2' from target.

3.4 Decimation Framework:

Our Decimation framework is as follows

Input:

Triangulated mesh M with identified symmetric parts

Mapping file with vertex-to-vertex correspondence information

Output:

Simplified Triangulated mesh with shape preservation

The decimation approach we use is a hybrid approach which combines concept of planar proxies and structure preserving rules. Many input meshes exhibits near-planar parts that can be detected by common shape detection approaches. These near-planar parts are represented using planar proxies. More specifically, a planar proxy consists of a set of vertices and a plane $ax+by+cz+d = 0$ represented as a vector [a b c d], where n = [a b c] is the unit normal vector to the plane. We would use set of structure preserving rules which would preserve structure of proxies during decimation by not performing edge collapse operators that violate a set of structure-preserving rules.

4. Shape preservation algorithms

4.1 Detecting near planar parts

For detecting near planar parts in a mesh structure we make use of depth first search technique. Each triangular face is considered to be connected with the neighboring face if the deviation between the normal of two faces is within a specified tolerance. This neighborhood search terminates once all the triangular faces have been visited. The result of this process is a set of planar proxies where each mesh face will belong to a planar proxy. A face could also belong to multiple proxies but in our approach we restricted it to be only one. Vertex on the other hand could belong to multiple proxies. Planar proxies associated with the vertex are the proxies of faces which vertex is part of.

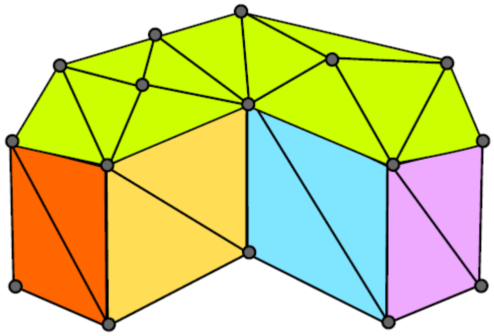

Fig 5: Planar proxies for a building

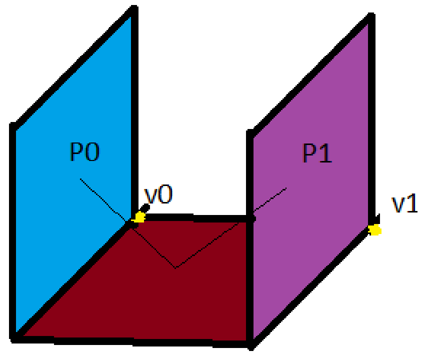

Fig 6: Connection between proxies

4.2 Error quadric

Quadric calculations for faces as mentioned in Qslim [1] uses only the plane of face. In the approach we follow we have introduced the calculations of planar proxy as well. We define a parameter μ which will decide the weightage to be given among proxy plane and face plane. The modified quadric calculation looks as follows:

$$Q = (1 - \mu)\ Q' + \mu Q''$$

Here Q' is the quadric associated with the supporting plane of face T and Q'' is the quadric associated with the plane of proxy to which T belongs.

4.3 Avoiding proxy merge

We could transform the set of proxies detected in earlier step into a graph. Each proxy would be represented by a point and this point is connected to another proxy point if there exists a vertex in the mesh with both of them in its proxy plane set. Keeping this in mind, figure 6 shows that collapsing an edge between v0 and v1 would introduce a new edge between P0 & P1 into the proxy graph. This collapse would distort the structure of shape and thus needs to be avoided. Hence while selecting an edge for decimation we make sure that the proxy set of one of the end vertex of edge is subset of the other end.

4.4 Proxy preservation

When performing a collapse operation, the resulting vertex receives the union of proxies of the two edge vertices. However, a proxy may degenerate into a single vertex or edge during decimation. We want to avoid such situations as a proxy represents an important feature of shape. Therefore, we maintain a check that at least 2 faces for each proxy plane should exist in mesh.

4.5 Corner preservation

As corners are very noticeable in an object, it is of utmost importance to maintain them throughout the decimation process. Simplest approach to detect corners is to look for vertex with 3 or more proxies. In most of the cases corners are the intersection of 3 or more proxy planes. We would make sure a high cost is associated with the collapse of edges with at least one of the vertex being a corner.

5. Experiment and results

We tested the mentioned approach on various mesh structures. For demo and proof of concepts purpose we use a table mesh and check the performance. We created three mesh structures of table one with 64 vertices and 124 faces, with 374 vertices and 744 faces, and then with1490 vertices and 2976 faces. For quadric calculations we set the parameter μ to 0.5. This value of μ would provide a balanced weightage among proxy planes and face planes. The results obtained were better than traditional multiple choice scheme. With original mesh having 374 vertices and 744 faces, and after reducing the face count to 160 by successive decimation, shape was kept intact in our approach while with multiple choice scheme the legs of table were distorted. Additionally, the approach also maintains mesh structure of leg as shown in figure 9.

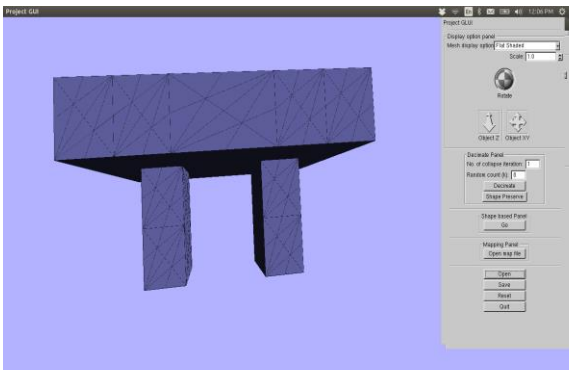

Fig 7: Original table

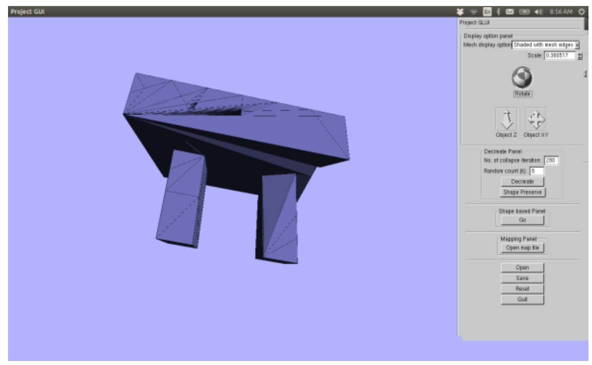

Fig 8: Result with multiple choice scheme

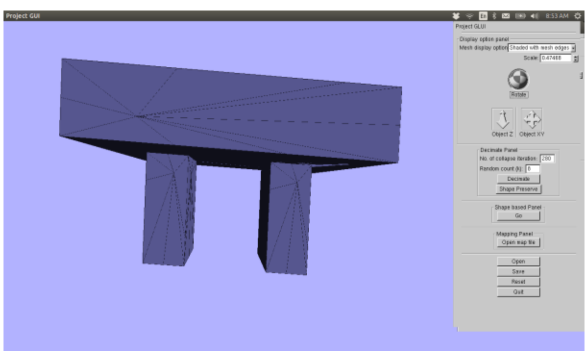

Fig 9: Result with the approach we used

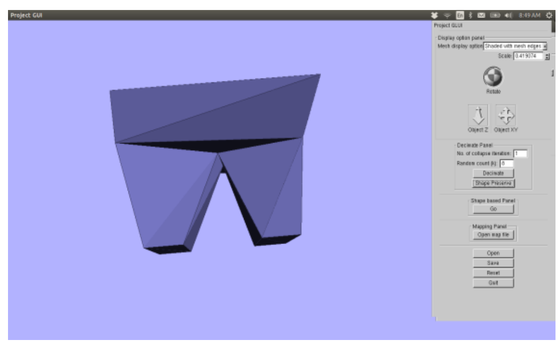

Fig 10: Mesh structure when extensive decimation

6. Conclusion and Future Work

We have demonstrated that the approach we used does preserve the shape of object and symmetry of mesh structure. It provides better result than MCS but doesn't provide consistent improved performance when compared with Qslim.

There is lot scope of improvement in the process we followed. Handling of mesh when number of vertex and faces are less need to be improved as it becomes difficult to maintain symmetry when face counts are very low. Coming up with a general cost function for edge selection which incorporate costs for planar parts and works for all type of meshes. Also we used planar proxies which won't work on non-planar parts, thus restricting the set of meshes the algorithm could work on.

7. References

[1] Michael Garland and Paul S. Heckbert: Surface simplification using quadric error metrics. SIGGRAPH '97.

[2] Jianhua Wu, Leif Kobbelt. Fast mesh decimation by Multiple-Choice Techniques.

[3] Manish Mandad, David Cohen-Steiner, Pierre Alliez: Isotopic approximation within a tolerance volume.

[4] Niloy J Mitra, Michael Wand, Richard Zhang, Daniel Cohen-Or, Martin Bokeloh: Structure-aware shape processing.

[5] Aleksey Golovinskiy, Joshua Podolak, and Thomas Funkhouser: Symmetry-aware mesh processing.

[6] David Salinas, Florent Lafarge, Pierre Alliez: Structure-Aware Mesh Decimation. Computer Graphics Forum, Wiley, 2015, pp.20. <hal-01111203>

[7] ICP: https://en.wikipedia.org/wiki/Iterative_closest_point