Converting Triangulations To Grids:
(A Proposal)

Afra Zomorodian

June 14, 1998

Abstract

This paper outlines a simple and fast method for conversion of unstructured triangulations to connected grids. The conversion strategy is as follows: we begin by using mesh decimation by edge shrinks to define a triangulated base complex and parametrize the original mesh points on the faces of the base complex. We then convert base complex to a Steiner quadrangulation. Finally, we use the sampling to generate a hierarchy of coarse-to-fine grids. I will discuss previous work, the possible approaches to each step of the conversion, and alignment of grids to object features.

Overview

The goal of this project is to convert a triangulated 2-manifold to a number of connected grids describing the same surface. We seek to preserve both topology and geometry of the surface in this conversion.

I will first present an overall *scheme* for conversion. I call this a scheme as each step may be implemented via several approaches. Each of the steps will be discussed in more detail in the subsequent sections. Finally, I will discuss how the grids can be aligned with the features of the surface as well as how a user can provide feature constraints or "tags."

Note that grids may be used as bases for reconstruction of B-Splines and therefore this work may be compared with Krishnamurthy and Levoy's user intensive [8] and Eck and Hoppe's computationally-intensive [3] construction of splines. Both methods take about 5-6 hours of human or computer time. This method should take far less for meshes of comparable size (in the order of minutes.)

In the following, I refer to the input triangulation as M, the decimated triangulated base as M_0, the quadrangulation as Q_0, and the parametrization of mesh vertices as ϕ. The conversion scheme is as follows:

Conversion Scheme:

1. Decimate the M by topology preserving edge shrinks to form M_0. Use these transformations to build a parametrization of the vertices of M over the faces of M_0.
2. Convert M_0 to Q_0, a Steiner quadrangulation.
3. Construct a hierarchy of coarse-to-fine grids appropriate for multigrid methods by using ϕ to sample the M.

Intuition Behind Approach

The basic insight behind this approach is that the final product of any conversion process will have several grids, each of which may be viewed as a patch. This immediately implies that one approach to the conversion would be forming a small coarse of quadrangulation whose faces define the grids. To form such a quadrangulation, we may first seek a coarse triangulation with the same connectivity as the surface. We then convert this coarse triangulation to a quadrangulation.

Constructing the Base Triangulation M_0

Our goal in constructing the base triangulation is three-fold: we wish to capture the geometry and topology of the surface, while keeping the number of triangles low.

Eck et. al. [2] solve this problem by automatically growing Voronoi-like regions on M and use the Delauney triangulation of the resulting Voronoi sites as the base triangulation. Their base triangulation is used for remeshing M with subdivision connectivity so that the triangulation could be approximated by Wavelet coefficients. Eck and Hoppe [3] extend this work for automatic reconstruction of B-Splines. This approach to constructing M_0 is rather costly, however, as it involves solving very large systems of non-linear equations for parametrization of vertices of M (e.g. 600 minutes for a mesh of 17,000 triangles.)

Lee et. al. [9] use *mesh decimation* as a method for constructing M_0 in order to also remesh triangulations with subdivision connectivity. Their approach leads to an $\mathcal{O}(n \lg n)$ algorithm for M with n triangles and is relatively fast (e.g. 7 minutes for 100,000 triangles.)

Our approach would also use mesh decimation in creating M_0, although this approach is different from that of Lee et. al. in that we choose a different decimating transformation, as described below.

Decimating M

A mesh may be modified by global algorithms such as Turk's [13] or via incremental greedy algorithms. We choose the latter as there are several mesh decimation algorithms which have (sometimes provably) good results.

Any greedy incremental mesh decimation algorithm uses one or several local operations at each step. Each such operation a combinatorial and a numerical aspect. The combinatorial aspect of the operation may be viewed as a graph operation, where we look at the triangulation as an embedded graph. The numerical aspect gives us a method of preserving geometry by measuring the error caused by the operation. We may then use this error to prioritize the possible operations and stop the mesh decimation algorithm if the error reaches a threshold. It is important to note that in our case, the numerical heuristic we use will also imply our parametrization of the vertices of M on the faces of M_0. I will discuss each aspect in turn.

The Decimation Operation: The Combinatorial Aspect

There are a variety of mesh transformations one may use to modify or decimate a triangulation. Hoppe et. al. [7] use selective application of edge collapses, edge splits, and edge swaps to optimize meshes. In addition to those operations, one may use vertex removal and its inverse, vertex splits, to transform meshes.

There are currently two main approaches to incremental mesh decimation: Schroeder et. al. [12] use vertex removal and retriangulation. I believe this is the method implemented in the VTK graphics package. Lee et. al. [9] also choose vertex removal for their system, MAPS, stating that the main reason for doing so is that it guarantees a logarithmic bound on the number of levels of the constructed mesh hierarchy.

Hoppe, Garland and Heckbert, and Edelsbrunner, among others, advocate edge shrinks as the transformation of choice [6, 5, 4]. Edge shrinks are simpler, as there is no need for retriangulation. They are also easily invertible and can be made into continuous operations by linear interpolation.

We choose edge shrinks as the decimation transformation because:

1. Edge shrinks that don't change the topology lead to *Irreducible triangulations* of the surface. Irreducible or minimal triangulations have been studied and their size is linear in the genus of the surface [10].

2. As already mentioned, edge shrinks are simple and require only the determination of the location of the new vertex.

3. The combinatorial and numerics of an edge shrink are well separable. In particular, one may experiment with different numerical heuristics for varying results. I discuss in the next section.

As M is a 2-manifold, we use the *Link Condition* to test whether an edge-shrink is *valid*, i.e. does not change the topology of the underlying space [4].

The Numerical Aspect of Edge Shrinks and Constructing ϕ

Recall that the numerics of our edge shrink operation is intricately connected with the parametrization of the vertices of M. We may choose computational intensive numerics for better quality grids or choose slower numerics for fast conversion.

We want each heuristic to give us a measure of geometric error introduced by the edge shrink. We use this error measure to prioritize all valid edge shrinks. We then apply each edge shrink in order and update the priority queue as needed. We may stop when the error reaches a threshold or increases dramatically.

Some of the possible error measures are:

- The quadric error metric [5]: Here, the error measures the distance of the new vertex from a set of planes.

- Harmonic Maps: Eck et. al. [2] use approximation to harmonic maps to generate their parametrization. To generate such approximations, one must solve a system of non-linear equations, with an equation per vertex. Their method is slow as each approximation involves hundreds of vertices. The star of an edge, however, is small (an average of 10 vertices) and therefore, our application will be much faster. There are two advantages to harmonic maps: they minimize distortion, and give an immediate method for parametrizing the points via barycentric coordinates following [9]. The measured distortion may be used as the error measure.

- Hoppe's Energy Method [6]: Hoppe defines an energy metric which he tries to minimize. I don't like this method because it seems rather ad-hoc. Moreover, he also tries to optimize scalar and discrete properties of triangulations (e.g. color) which are not our concern.

- Shrink To Middle: We may just shrink an edge to the middle of the edge. For the error measure, we can use the Quadrics measure mentioned above (here, however, we are restricting where the new vertex could be.)

- Shrink To Vertex: We may shrink the edge to one of the old vertices. Again, we use the Quadrics measure as the prioritizing error measure.

Except for the harmonic maps, the other measures require us to somehow parametrize the old vertices of an edge shrink. This could be done by projecting the vertices onto the new triangles and using barycentric coordinates.

Converting M_0 to Steiner Quadrangulation Q_0

Our basic approach to converting M_0 to Q_0 is by finding maximal matchings in the dual of the graph of our triangulation. The fastest algorithm uses an unweighted dual graph:

QUADRANGULATE:

1. Construct the dual graph Γ of M_0.
2. Construct a spanning tree of Γ using BFS [1].
3. Use the *Q-percolation* algorithm to construct quadrangulation. [11].

The Q-percolation algorithm constructs a matching in linear time, adding Steiner points when needed.

To get a better quality quadrangulation, we can assign weights to each edge of the dual graph. The possible weights could be measures of convexity, fatness, or the dihedral angle subtended by the two triangles. We could then use these weights in two ways:

1. Construct the minimum spanning tree using Kruskal's or Prim's algorithm. Then use the Q-percolation algorithm (forgetting the weights.)

2. Use a minimum weighted matching algorithm on the dual graph.

I have little knowledge of the weighted matching problem and therefore don't know if the second option would be feasible. Also, I'm not sure how we could deal with the non-matched edges in that case. I tend to think that the first option is probably feasible: minimum spanning tree algorithms are relatively fast. Also, as we want a quadrangulation, we really want as large of a matching as possible and there's no need to look for minimum weighted matchings.

We are able to monitor the quality of the base mesh by the error measure. As such, we might be able to get good results from QUADRANGULATE by lowering the threshold at which we stop the decimation process. However, if the results are still not good, we could implement the weighted options in order to see if better quadrangulations are possible.

Constructing A Hierarchy of Grids

We regularly (1:4) subdivide the base quadrangulation Q_0 and use our parametrization to obtain a finer quadrangulation. As in [9], we can use point location algorithms to find the triangle which contains our sample and use barycentric coordinates to define its location on the surface of our mesh. I am not very clear on the details (specific algorithms) of this step of the conversion.

Aligning Grids with Features

We wish to preserve features on our surface by aligning our grid boundaries to them. For our purposes, it's enough to define a feature line as discontinuity on the surface. Define a *sharp edge* to be an edge whose adjacent faces subtent a small dihedral angle. We may either define a parameter which defines "smallness" of the dihedral angle, or we may define "sharpness" as a continuous weight. The sharp edges together define a family of discontinuity curves on our surface.

We wish to preserve the discontinuity curves both topologically and geometrically. Hoppe, whose definition of sharpness differs from mine, gives sufficient conditions to preserve the curves topologically [6]. The key is to look at the local neighborhood of a sharp edge and only allow it to shrink if the operation does not modify the topology of the discontinuity curves which include the sharp edge.

We can also consider ways of preserving the discontinuity curves geometrically. Hoppe uses modifications to his energy calculation. We could also modify our error term for neighborhoods of sharp edges, although I'm unclear whether this is necessary.

Having preserved sharp edges in our mesh decimation step, we need to make sure they don't disappear in forming the quadrangulation. One approach would be to not allow the spanning tree of the dual graph from having edges corresponding to sharp edges. This could, however, lead to the spanning tree becoming a forest. We can run the *Q-percolation* algorithm on each tree and then subdivide each remaining triangle using additional Steiner points (this is problematic, however, as Steiner points are needed on the boundary of the triangle.)

Alternatively, we could use weights to try to avoid losing feature lines. This approach would tend to preserve features lines and not have as many problems as the approach discussed above.

User Intervention and Tagging

The user may tag edges and vertices as feature lines. It is simple to preserve such features lines during decimation as follows: to preserve edges, we call any tagged edge "sharp" as above. To preserve vertices, we make any edge which has tagged vertices sharp and shrink all edges of a tagged vertex to the vertex itself. We still have the additional difficulty of losing the features in the quadrangulation step.

Conclusion

The purpose of this paper is present a proposal for a project, as well as organize my thoughts and the many possibile approaches under one heading. I believe implementing this conversion scheme would give a useful and fast algorithm. I'm not sure whether this project could lead to a paper, however.

References

[1] Cormen, T. H., Leiserson, C. E., and Rivest, R. L. *Introduction to Algorithms.* The MIT Press, Cambridge, Massachusetts, 1994.

[2] Eck, M., DeRose, T., Duchamp, T., Hoppe, H., Lounsbery, M., and Stuetzle, W. Multiresolution analysis of arbitrary meshes. In *SIGGRAPH 95 Conference Proceedings* (Aug. 1995), R. Cook, Ed., Annual Conference Series, ACM SIGGRAPH, Addison Wesley, pp. 173–182.

[3] Eck, M., and Hoppe, H. Automatic reconstruction of B-Spline surfaces of arbitrary topological type. In *SIGGRAPH 96 Conference Proceedings* (Aug. 1996), H. Rushmeier, Ed., Annual Conference Series, ACM SIGGRAPH, Addison Wesley, pp. 325–334.

[4] Edelsbrunner, H., and Nekhayev, D. V. Topology and curvature preserving surface decimation. Raindrop Geomagic Inc., Report rgi-tech-97-010, 1997.

[5] Garland, M., and Heckbert, P. S. Surface simplification using quadric error metrics. In *SIGGRAPH 97 Conference Proceedings* (Aug. 1997), T. Whitted, Ed., Annual Conference Series, ACM SIGGRAPH, Addison Wesley, pp. 209–216. ISBN 0-89791-896-7.

[6] Hoppe, H. Progressive meshes. In *SIGGRAPH 96 Conference Proceedings* (Aug. 1996), H. Rushmeier, Ed., Annual Conference Series, ACM SIGGRAPH, Addison Wesley, pp. 99–108.

[7] Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., and Stuetzle, W. Mesh optimization. In *SIGGRAPH 93 Conference Proceedings* (Aug. 1993), J. T. Kajiya, Ed., vol. 27 of *Annual Conference Series*, ACM SIGGRAPH, Addison Wesley, pp. 19–26.

[8] Krishnamurthy, V., and Levoy, M. Fitting smooth surfaces to dense polygon meshes. In *SIGGRAPH 96 Conference Proceedings* (Aug. 1996), H. Rushmeier, Ed., Annual Conference Series, ACM SIGGRAPH, Addison Wesley, pp. 313–324.

[9] Lee, A. W. F., Sweldens, W., Schöder, P., Cowsar, L., and Dobkin, D. MAPS: Multiresolution adaptive parametrization of surfaces. In *SIGGRAPH 98 Conference Proceedings* (Aug. 1998), Annual Conference Series, ACM SIGGRAPH, Addison Wesley. To be published.

[10] Nakamoto, A., and Ota, K. Note on irreducible triangulations of surfaces. *J. Graph Theory 20*, 2 (1995), 227–233.

[11] Ramaswami, S., Ramos, P., and Toussaint, G. Converting triangulations to quadrangulations. *CGTA: Computational Geometry: Theory and Applications 9* (1998).

[12] Schroeder, W. J., Zarge, J. A., and Lorensen, W. E. Decimation of triangle meshes. In *SIGGRAPH 92 Conference Proceedings* (July 1992), E. E. Catmull, Ed., vol. 26 of *Annual Conference Series*, ACM SIGGRAPH, Addison Wesley, pp. 65–70.

[13] Turk, G. Re-tiling polygonal surfaces. In *SIGGRAPH 92 Conference Proceedings* (July 1992), E. E. Catmull, Ed., vol. 26 of *Annual Conference Series*, ACM SIGGRAPH, Addison Wesley, pp. 55–64.