Mesh Decimation for Displacement Mapping

Gordon Collins and Adrian Hilton

Centre for Vision, Speech and Signal Processing
University of Surrey, Guildford GU25XH, UK
g.collins,a.hilton@surrey.ac.uk
http://www.ee.surrey.ac.uk/Research/VSSP/3DVision

Abstract

We present a mesh decimation algorithm for triangular meshes. Unlike other decimation algorithms we are not concerned with geometric error but with the existence of a displacement mapping which can map between the original and decimated meshes. We use the implicit function theorem to derive a condition which ensures the existence of a displacement map. The algorithm is applied to some standard scanned models and reduction rates around 99% are seen.

1. Introduction

The high-resolution unstructured meshes which result from 3D scanned data contain highly detailed geometry which presents problems for animation and compression. The problems can be tackled by using a low resolution mesh and computing a mapping from this to the high-resolution mesh. At each point on the low-resolution mesh we calculate a distance along the normal to the high-resolution mesh. The resulting displacment map representation is compressed and can be regenerated at multiple levels of detail. Furthermore the low-resolution mesh can be easily animated and the high-resolution mesh will in turn animate in a suitable way. The success of this procedure relies on having a low-resolution mesh with a displacement map which captures *all* the high-resolution points. This paper is concerned with generating low-resolution meshes by decimating scanned high resolution meshes in such a way that all the detail can be represented by a displacement mapping.

Detail is lost by displacement maps when a fold on the high-resolution mesh is mapped to an unfolded low-resolution mesh. In this case the inverse map from the high-resolution mesh to the low-resolution mesh would be non-injective and therefore its inverse (the mapping from low-resolution to high-resolution mesh) would not exist. In other words there would be no scalar displacement map which represents all the points in the high-resolution mesh.

The problem can be avoided if a suitable low-resolution mesh is used. We generate low-resolution meshes so that the dot product of the normals of the high resolution mesh points $\mathbf{n}$ and the normals of the low-resolution mesh $\mathbf{N}$ is never zero ($\mathbf{n}.\mathbf{N} \neq 0$). Using the implicit function theorem we prove that, for the simple case of displacement mapping with face normals, this will guarantee the existence of a displacement mapping which describes a continuous differentiable surface. The idea is then extended to cope with interpolated face normals and triangulated surfaces.

The next section motivates the problem by discussing the use of displacement maps in mesh representation. Also we discuss mesh decimation and in particular the decimation algorithm of Lee et al. [5]. Section (3) describes the folding problem in detail. In section (4) the implicit function theorem is used to derive our condition for a displacement map representation in an idealised situation. Proceeding from this we outline our decimation algorithm in section (5) and then discuss the results of the applying this to some standard models in section (6).

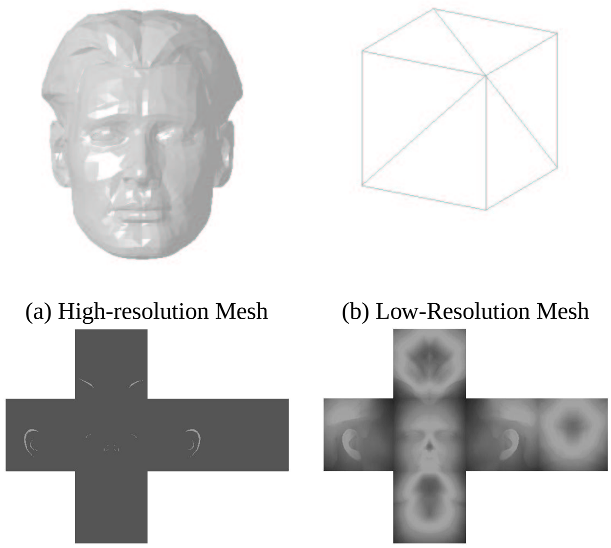
2. Background

2.1. Displacement Maps

In the general case we consider a local mapping given by $D : L_l \rightarrow H_l$ where $L_l \subset L$ is a subset of a low resolution surface in $\mathbb{R}^3$ and $H_l \subset H$ is a subset of a high resolution surface in $\mathbb{R}^3$. For now we consider these surfaces to be either meshes or continuosly differentiable surfaces. The mapping is described as,

$$\mathbf{x} = D(\mathbf{X}) = \mathbf{X} + d(\mathbf{X})\mathbf{N}(X) \quad (1)$$

where $\mathbf{X} \in L_l$ and $\mathbf{x} \in H_l$. For any point on H the mapping is inverted to give a distance d and a point of intersection with the low resolution mesh, $\mathbf{X}$. Once this mapping has been calculated, the high-resolution mesh can then be stored like a texture map (a displacement image) where each texture coordinate stores a distance instead of a colour. Figure (1d) shows the displacement image of a head model in figure (1a), which has been mapped onto a cube. The high-resolution mesh can then be reconstructed by subdividing the low-resolution mesh and rendering the model at a displacement from the low-resolution surface given by appropriate point in the displacement image (see figure (6)) The whole process is described in detail in our previous work [7, 9].



(a) High-resolution Mesh (b) Low-Resolution Mesh

(c) Unrepresentable Folds (d) Final displacement Map

Figure 1: *Displacement map generation for head model with simple cube low-resolution mesh*

There have been three major research efforts [4, 9, 5] which can be characterised by their different ways of calculating the low-resolution surface normal $\mathbf{N}$ and in the representation of the low-resolution surface L.

The simplest solution is to take the normal to be the face normal of each low-resolution surface polygon and so to be constant across the face. The obvious disadvantage of this is that the normals will not be continuous across edges and this greatly restricts the number of high-resolution points which can be mapped. Furthermore when such a model is animated, high-resolution points near low-resolution mesh edges will move disproportionately to the movement of the low-resolution mesh surface. However, we employ the face normal scheme in section (5) for ease of explanation.

All published schemes take the normal to be a function of $\mathbf{X}$ and to be continuous across the surface. Krishnamurthy and Levoy [4] ensured a continuous normal by calculating displacements off a B-Spline surface. Lee et al. [5] base their approach on subdivision surfaces and take the normals of the limit function of their surface.

In Hilton et al. [7, 9] we have used a polygon representation and compute a continuous normal across the surface by interpolating the vertex normals $\mathbf{N}_i$ so that

$$\mathbf{N}(\alpha, \beta, \gamma) = \alpha\mathbf{N}_1 + \beta\mathbf{N}_2 + \gamma\mathbf{N}_3, \quad (2)$$

where (α, β, γ) are the barycentric coordinates of $\mathbf{X}$ with respect to the triangle on which it lies. The vertex normals define a “normal volume” which encloses the high-resolution points which will be mapped onto that triangle.

2.2. Low-Resolution Mesh Generation

Krishnamurthy and Levoy [4] construct their low-resolution mesh by manually defining spline patches on the surface of the high-resolution model. Hilton et al. [8, 9] use a similar approach of drawing polygons on the surface of the low-resolution mesh. They have also used models derived from a generic model database. Although constructed models are more easily animated they may take considerable manual effort to construct. Furthermore there is no control over loss of detail since there is no reason why thse meshes will support a scalar displacement mapping which represents the desired high-resolution surface.

Mesh simplification is now a mature field of study in computer graphics (see Heckbert and Garland [2] for a comprehensive review). Almost all simplification schemes are concerned with simplifying a mesh in order to minimise a measure of geometric error. The only scheme that is concerned with the existence of a displacement mapping representation is that of Lee et al. [5].

Lee et al. use an edge collapsing algorithm which, although it prioritises the edges to be collapsed by a measure of geometric error, will not allow a collapse unless the resulting mesh has similarily aligned normals. Using an efficient parameterisation algorithm (MAPS [6]), they keep track of all the points that would map to each triangle as the decimation proceeds. For each collapse the resulting 1-ring neighbourhood is calculated and for each triangle in this neighbourhood the vertex normals are calculated. The collapse is allowed if the Gauss map of these vertex normals encloses the Gauss map of all the normals of the high-resolution points which map to that triangle. Lee et al. provide no analysis of whether this method ensures a displacement mapping representation. Furthermore the authors have had to relax their condition in order to progress the decimation far enough and so produce a sufficiently simple low-resolution mesh.

3. The Problem

Detail is lost on displacement mapping when a fold in the high-resolution mesh is mapped onto an unfolded low-resolution mesh. Figure (2) shows that when this happens the inverse D^{-1} of the displacement map (1) is noninjective. Therefore there will be no scalar displacement map D which describes the surface H. If the mapping is calculated in the normal way the fold will be lost. Figure (1 c)) shows the folds of a head model when it is mapped onto an arbitrary low-resolution mesh - a cube. Information is lost around the ears and mouth as it is here where the model folds with respect to the cube.

We note that, locally, the mapping is invertible since for each point $\mathbf{x}_i$ in Figure (2) there is a solution to the displacement mapping equation (1). However, since there is no scalar valued *function* which describes the entire curve in Figure (2) this surface cannot be represented by a scalar displacement map. We say that the surface H is represented by a scalar displacement map D if all points on H can be written as equation (1).

In the next section we provide a condition which ensures the existence of a displacement map representation.

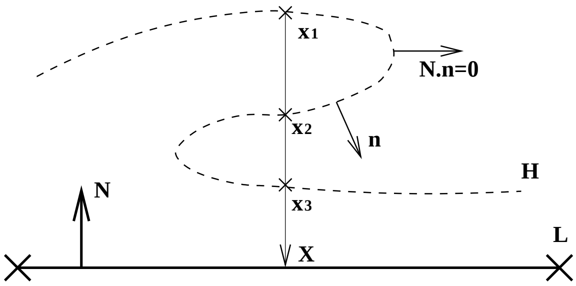


Figure 2: *Points x_1, x_2 and x_3 all map to the same point* $\mathbf{X}$*.*

4. Existence of Displacement Map Representation

We use the implicit function theorem to prove that in a restricted case if the normals of the high and low resolution surfaces are such that $\mathbf{n}.\mathbf{N} \neq 0$ then the high resolution surface can be represented by a displacement map from a mesh. We make two simplifications. Firstly, we consider the high-resolution surface, given by $F(\mathbf{x}) = 0$, to be continuously differentiable. Secondly, we consider mappings with a normal $\mathbf{N}$ which is constant across each face of the mesh. These assumptions are dropped in the next section when we consider mappings from meshes to meshes.

The implicit function theorem guarantees the existence of a function which "solves" the general algebraic equation $F(\mathbf{x}) = 0$ uniquely. The function must be away from any folds in the surface otherwise the resulting solution may be multi-valued in that neighbourhood. The theorem may be found in many advanced calculus textbooks (Fitzpatrick [1] for example). Here we state the theorem for a function of three variables.

The Implicit Function Theorem *Let U be an open interval in $\mathbb{R}^3$ and $F : U \to \mathbb{R}$ a continuously differentiable function. If, also there exists a point $(x_0, y_0, z_0) \in U$ such that,*

$$F(x_0, y_0, z_0) = 0 \text{ and } \frac{\partial F}{\partial z}(x_0, y_0, z_0) \neq 0$$

then there exists a neighbourhood U' of (x_0, y_0, z_0) and a continuously differentiable function $f : U' \to \mathbb{R}$ such that

$$z = f(x,y) \text{ and } F(x,y,f(x,y)) = 0 \; \forall (x,y,z) \in U'.$$

We now prove that given similarly oriented normals, there will be a displacement mapping from a plane onto a continuous surface.

Theorem *Let a continuously differentiable surface H be given by $F(x,y,z) = 0$ with normal $\nabla F(x,y,z)$ and let a plane P have a constant normal $\mathbf{N}$ such that,*

$$\mathbf{N}.\nabla F(x,y,z) \neq 0 \; \forall (x,y,z) \text{ on } P \tag{3}$$

then surface H is represented by a continuously differentiable displacement mapping D given by equation (1) with constant normal $\mathbf{N}$*.*

Proof If we consider the local coordinate frame of L with normal $\mathbf{N} = \mathbf{k}$ (see Figure (2), then

$$\mathbf{N}.\nabla F(x,y,z) = \mathbf{k}.\nabla F(x,y,z) = \frac{\partial F}{\partial z}(x,y,z) \neq 0 \; \forall (x,y,z) \text{ on } P.$$

The implicit function theorem tells us that, in the neighbourhood of any point on P, there is a continuously differentiable function,

$$z = f(x,y) \text{ such that } F(x,y,f(x,y)) = 0 \; \forall (x,y,z) \text{ on } H.$$

A point on H can therefore be written as,

$$x\mathbf{i} + y\mathbf{j} + z\mathbf{k} = x\mathbf{i} + y\mathbf{j} + f(x,y)\mathbf{k}.$$

Since, in local coordinates, any point $(x,y,0)$ is on P, we have represented any point on the surface with the displacement mapping D. □

If we now consider a triangle of a mesh M^L instead of a plane, then the above theorem holds for any point in H which has a mapping from some point on M^L (ie. D can be inverted locally). We therefore state the result for a mapping from a mesh to a surface.

Corollary *If for any point $\mathbf{x} \in H$ equation (1) can be solved for an $\mathbf{X} \in M^L$ and for all these points condition (3) holds, then H is represented by D.*

5. The Algorithm

In order to guarantee a valid displacement mapping from the low-resolution mesh M^L to the high resolution mesh M^H we proceed by decimating mesh M^H wherever the resulting mesh still allows a displacement map representation of M^H. We draw upon the analysis in the previous section in order to suggest a condition for the existence of a displacement map represenatation. In this case we must drop the assumptions of a continuously differentiable high-resolution surface in favour of a high-resolution mesh and and of a low-resolution surface with constant normals in favour of a continuous interpolated normal given by (2).

The condition (3) guarantees a continuously differentiable injective displacement mapping from a mesh to a continuously differentiable surface H. In the case of high-resolution scanned data we desire a continuously differentiable displacement mapping which approximates a triangulated surface M^H. Since ∇F does not exist at vertices, we need to discretise condition (3) for triangular surfaces.

Condition (3) requires that the dot product of the normals of the high and low resolution mesh are never zero. In the discrete setting this will almost always be true. However we may assume that if this dot product changes sign then somewhere it has become 0. Furthermore, since the low resolution mesh M^L is generated from the high resolution M^H the dot product of their normals will initially be positive. Therefore, if this dot product ever becomes negative, we suppose that condition (3) has been violated.

For a high resolution vertex $\mathbf{v}$ with surrounding face normals $\mathbf{n}_i$ our discrete version of (3) is

$$\mathbf{N}(\mathbf{X}).\mathbf{n}_i > 0 \tag{4}$$

where $\mathbf{N}(\mathbf{X})$ is given by the interpolation (2). Note that we have generalised condition (3) further by using this interpolated normal instead of the constant normal.

Of course, now we have discretised our condition, the analysis of the last section does not prove the existence of a displacement map representation. For this to be proved we would need a discreet version of the implicit function theorem which holds for mappings along an interpolated normal. We are not aware of any such theorem and so we use the analysis in the previous section to suggest that this discretisation of condition (3) holds.

We employ a non-hierarchical decimation algorithm based on edge collapses as described in [3]. As the decimation progresses we ensure that each high-resolution vertex can be described by a displacment mapping to a triangle to which it is associated.

First we associate each high-resolution vertex with a triangle which it is on (and therefore it is representable by a displacment mapping (1) with $d = 0$) We then prioritise the edge collapses in order of edge length. We now begin the decimation loop by performing an edge collapse and testing whether the resulting mesh still supports a displacement mapping of the high-resolution mesh. All the vertices currently associated with triangles in the candidate edge's neighbourhood (its 2-ring) are tested on the collapsed neigbourhood of the edge so that.

- There is a unique mapping (1) to the neighbourhood of the edge for all the associated vertices.
- All the associated vertices satisfy condition (4).

The first condition guarantees an injective mapping D and the second that the surface M^H can be represented as a displacement map. These two conditions are the equivalent of the hypothesis in corollary 2. If any vertex fails these tests, then the collapse is not performed and the edge is prioritised last. To ensure mesh quality, an edge is collapsed only if its valence (the combined valences of its vertices minus one) is less than 12. If the collapse is performed then all the associated vertices are re-associated with the new triangles to which they now map. The process continues until all the edges remaining fail the tests above.

a) Original Mesh	b) Constant Normal	c) Interpolated Normal

Figure 3: *Monster head and decimations for injective displacement mapping*

6. Results

Meshes were generated by applying the algorithm described above to several standard examples and the results are shown in Figures (3), (4), (5) and (6). Figure (3 b)) shows a low-resolution mesh which allows displacement mapping with a constant normal which is discontinuous at mesh edges. This mesh for which the analysis of section (4) is valid shows the limitation of this approach since the mesh could only be reduced by 10%.

All these low-resolution meshes were tested for loss of detail by two methods. First an image of the folds, such as that shown in Figure (1 c), was computed. Each high-resolution point was mapped via the displacement mapping equation (1) into texture space. A fold is shown in white when many high-resolution points map to the same low-resolution texture coordinate. In Figure (1 c) it is clear that folds are seen around the ears and eyes. For the meshes generated here only a few pixels are shown as white in the image. These points may be due to rounding errors at the border of two high-resolution triangles or because of the quantisation error involved in sampling the image at each pixel. Secondly the model can be reconstructed from the displacement image and tested against the original. Work on this is still in progress although preliminary results suggest that even finely detailed folds are reconstructed (see figure (6).

	Monster	Horse	Bunny	Venus
High-Res Faces	30,086	96,966	69,451	100,004
Low-Res Faces	612	502	388	100
% Decimated	98.2	99.2	99.4	99.9

The table shows the amount of decimation possible for each model. Low-Resolution mesh sizes are, on the whole, smaller than those quoted in Lee et al. [6]. Simplification times are an order of magnitude larger than those of Lee et al. since they do not test each high-resolution point for each collapse.

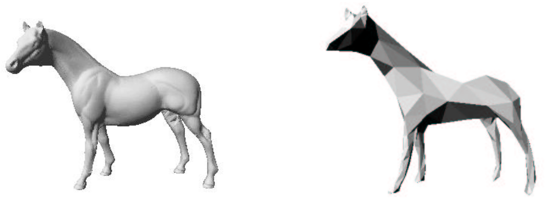


Figure 4: *Horse Model and Decimated Model*

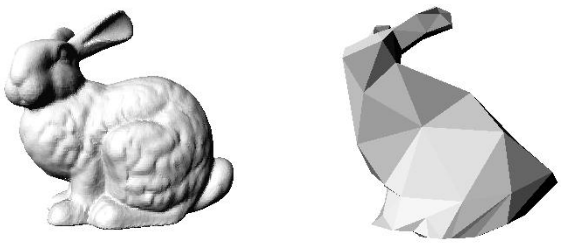


Figure 5: *Bunny Model and Decimated Model*

7. Summary and Future Work

We have given a mathematical formulation to the problem of detail loss in displacement mapping. We presented an algorithm which draws on this analysis to give low-resolution meshes which allow a scalar displacement map representation of the high-resolution mesh.

Further work would formulate theorem (1) for triangular surfaces and for continuous displacement maps. Also, in order to build low-resolution meshes for animation further constraints are needed on decimation to produce low-resolution meshes which bend in the right places. Lastly a

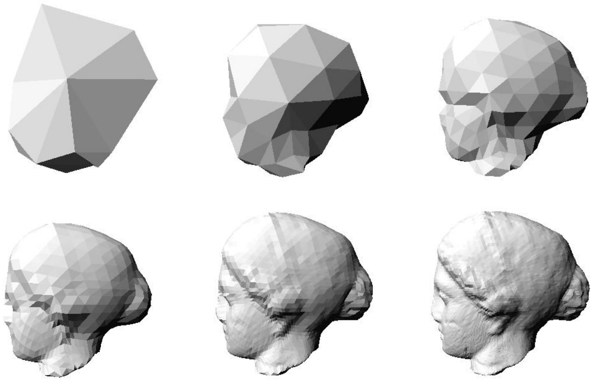


Figure 6: *Venus model subdivided 0,1,2,3,4 and 5 times.*

compression analysis at different levels of detail is needed to assess the compactness of the representation.

8. Acknowledgements

The authors would like to thank Stanford University for their bunny model and Cyberware for their Venus model. This research was funded by the EPSRC project GR/89518 “Building Realistic Models for Virtual Reality and Animation” and EU project IST-2000-28700 “MELIES”.

References

1. P. Fitzpatrick. *Advanced calculus :a course in mathematical analysis*. Boston, Mass.;London:PWS, 1996. 3

2. Paul S. Heckbert and Michael Garland. Survey of polygonal surface simplification algorithms. Technical report, CS Dept., Carnegie Mellon U., to appear. http://www.cs.cmu.edu/ ph">http://www.cs.cmu.edu/ ph. 2

3. Hugues Hoppe. Progressive meshes. *Proceedings of SIGGRAPH 96*, pages 99–108, August 1996. ISBN 0-201-94800-1. Held in New Orleans, Louisiana. 4

4. Venkat Krishnamurthy and Marc Levoy. Fitting smooth surfaces to dense polygon meshes. *Proceedings of SIGGRAPH 96*, pages 313–324, August 1996. ISBN 0-201-94800-1. Held in New Orleans, Louisiana. 2, 2, 2

5. Aaron Lee, Henry Moreton, and Hugues Hoppe. Displaced subdivision surfaces. *Proceedings of SIGGRAPH 2000*, pages 85–94, July 2000. ISBN 1-58113-208-5. 1, 2, 2, 2

6. Aaron W. F. Lee, Wim Sweldens, Peter Schröder, Lawrence Cowsar, and David Dobkin. Maps: Multiresolution adaptive parameterization of surfaces. *Proceed-*

ings of SIGGRAPH 98, pages 95–104, July 1998. ISBN 0-89791-999-8. Held in Orlando, Florida. 2, 4

7. R. Smith, W. Sun, A. Hilton, and J. Illingworth. Layered animation using displacement maps. In *IEEE International Conference on Computer Animation*, pages 146—154, May 2000. 2, 2

8. W. Sun, A. Hilton, R. Smith, and J. Illingworth. Building layered animation models from captured data. In *Eurographics Workshop on Computer Animation*, pages 145—154, September 1999. 2

9. W. Sun, A. Hilton, R. Smith, and J.Illingworth. Layered animation of captured data. *Visual Computer: International Journal of Computer Graphics*, 17(8):457—474, 2001. 2, 2, 2, 2