Layered Animation using Displacement Maps

Raymond Smith, Wei Sun, Adrian Hilton and John Illingworth

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

Abstract

This paper presents a layered animation framework which uses displacement maps for efficient representation and animation of highly detailed surfaces. The model consists of three layers: a skeleton; low-resolution control model; and a displacement map image. The novel aspects of this approach are an automatic closed-form solution for displacement map generation and animation of the layered displacement map model. This approach provides an efficient representation of complex geometry which allows realistic deformable animation with multiple levels-of-detail. The representation enables compression, efficient transmission and level-of-detail control for animated models.

1 Introduction

Animation of 3D models with highly detailed surfaces is a common requirement of computer generated imagery for film and broadcast production. Character models are often sculpted from clay to achieve the desired shape and fine surface detail. Laser-scanning devices are used to digitise the surface. Advances in scanning technology together with development of surface reconstruction algorithms [3, 7] have enabled automatic capture of complex 3D models with highly detailed surfaces. Currently, animation of captured models requires a skilled animator several weeks to manually restructure the model geometry and topology. In this paper we introduce a layered animation framework which address the following issues:

1. Rapid reconstruction from captured data.
2. Efficient representation of surface detail.
3. Model generation at multiple levels-of-detail.
4. Seamless deformable animation of the surface detail.

Displacement maps are used to represent the high-resolution surface detail in an efficient image-based form. A new approach is introduced for automatically mapping between a captured high-resolution mesh model and a low-resolution control model. This mapping is used to generate a displacement map image representation of the high-resolution surface detail based on 2D texture coordinates for the low-resolution model. The displacement map represents the distance between the low and high resolution model surfaces as a scalar image. The low-resolution control model represents the object shape and topology and can be manipulated efficiently using an underlying skeleton. The resulting layered model provides an efficient representation. Real-time seamless deformable animation of the control model enables efficient seamless animation of the high-resolution surface detail.

Automatic mesh simplification algorithms have been widely developed for reducing the size of reconstructed 3D models from surface measurements [10, 8, 13, 16, 2]. Typically, tolerances are imposed to generate a representation which approximates the original surface within a specified geometric error. However, automatic simplification algorithms do not result in meshes which are directly suitable for efficient and realistic animation.

As the capture process is performed on a static object in a single pose there is no captured information on the underlying articulation structure. Therefore a manual process is required to position a subset of the mesh vertices to coincide with the required articulation structure. Perhaps the most closely related work to our approach is the work of Krishnamurphy and Levoy [14] where the boundaries of a low-resolution B-spline patch model are manually identified on a high-resolution captured model. Automatic fitting and mapping is then performed to obtain a B-spline representation of smooth surface regions and displacement map images to represent fine surface detail. The displacement maps provides an efficient representation of complex surfaces without loss of fine detail. However, the resulting model can not be directly animated. In this paper we present a general approach to obtaining a displacement map representation based on an low-resolution control model. Animation of the control model [12] enables seamless deformable animation of the high-resolution surface data.

Previous work has addressed automatic reconstruction of surface models with subdivision connectivity based on polygonal meshes [5] and higher order surface [9, 6] with piecewise smooth continuity. Subdivision models provide an efficient mechanism for generating models at multiple levels-of-detail. In this paper we use subdivision of the control model together with resampling of the displacement map image to generate models at multiple levels-of-detail. The combination of displacement maps with mesh subdivision provides a mechanism for representing highly complex surfaces. Adaptive mesh subdivision generates representations with variable level-of-detail or a specified geometric accuracy. Unlike smooth subdivision surface representations this approach enables accurate and efficient representation of surfaces with complex geometry including bump patterns and discontinuities.

2 Layered animation framework

In this paper we propose a layered animation framework using displacement maps to efficiently represent and animate high-resolution surface detail. The model consists of three layers: skeleton, control model and displacement map. Skeleton structures are widely used as the basis for manipulating animated models either interactively or from motion-capture data. The control model is a low-resolution polygon mesh, M^L, representing a coarse approximation of the character shape and topology. A control model can be derived either by simplification of a high-resolution model or from a library of generic objects models which have been structured for efficient animation such as those available from ViewPoint DataLabs [4]. The control model is animated based on the underlying skeleton structure and enables real-time visualisation. As in previous work [17] the vertices of the low-resolution model are mapped to the skeleton structure. Real-time seamless animation is achieved using a geometric transform [1, 12]. More realistic deformation corresponding to a particular surface type could also be implemented using a physics-based approach to deform the control model such as FFD [11] and DFFD [15].

In this work we introduce the use of a displacement map representation of the high-resolution surface detail. The displacement map is an image-based representation analogous to a texture map which encodes the distance along the surface normal between the low-resolution control model and a high-resolution polygonal mesh model, M^H. The high-resolution model is automatically reconstructed from surface measurements captured using a range-sensor system [7]. An automatic mapping technique is introduced in section 2.2 to establish a continuous mapping between the high-resolution captured surface model and the low-resolution control model. This mapping is used to generate an efficient displacement map image representation of the high-resolution surface detail.

Animation is achieved by manipulation of the skeleton based on either keyframe or motion capture data. Movement of the skeleton is used to seamlessly deform the low-resolution control model. Recursive subdivision of the animated low-resolution control model is used to efficiently reconstruct models at multiple levels-of-detail. This layered animation framework, combining displacement maps with mesh subdivision, enables efficient seamless animation of the high resolution surface detail.

2.1 Normal volume mapping

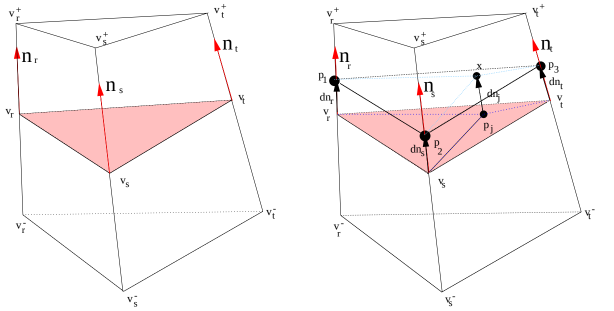
In previous work [17] we introduced the normal-volume mapping to automatically parameterise an arbitrary high-resolution mesh model, M^H, with respect to a low-resolution control model, M^L. It was shown that this parameterisation could be used to seamlessly animate the high-resolution model based on deformation of the underlying control model. In this section we summarise the key details of the normal-volume mapping which are used for displacement map generation.

For each triangle t_r in an arbitrary mesh M we define a *normal-volume*, $V^N(t_r)$, by displacing the triangle vertices, $\vec{v}_j$, along the vertex normals, $\vec{n}_j$, as illustrated in Figure 1(a). The union of the normal-volumes for all triangles in mesh M enclose a continuous volumetric envelope which can be used to define a continuous mapping between points in 3-space and the mesh surface.

To obtain a continuous mapping we project points in 3-space along the corresponding interpolated triangle normal. A point $\vec{p}_j$ on the surface of triangle $t_j = (\vec{v}_r, \vec{v}_s, \vec{v}_t)$ and its unit normal $\vec{n}_j$ can be defined by bi-linear interpolation using barycentric coordinates as:

$$\begin{aligned} \vec{p}_j &= \alpha\vec{v}_r + \beta\vec{v}_s + (1-\alpha-\beta)\vec{v}_t \\ \vec{n}_j &= \alpha\vec{n}_r + \beta\vec{n}_s + (1-\alpha-\beta)\vec{n}_t \end{aligned} \qquad (1)$$

where for a point inside the triangle the barycentric coordinates α, β and $(1-\alpha-\beta)$ are all positive scalar variables. Bilinear interpolation of the normal gives a continuous variation in the triangle normal, $\vec{n}_j$, across the planar surface and between adjacent triangles. The resulting normal field is continuous such that for every point $\vec{x}$ inside the normal-volume there is a corresponding normal $\vec{n}_j$ which passes through that points. Figure 1(b) illustrates the normal passing through a point $\vec{x}$. Thus any point $\vec{x}$ inside the normal volume $V^N(t_r)$ can be expressed as:



(a)Triangle normal volume (b) Point mapping

Figure 1. Normal volume mapping.

$$\vec{x} = \alpha(\vec{v}_r + d\vec{n}_r) + \beta(\vec{v}_s + d\vec{n}_s) + (1 - \alpha - \beta)(\vec{v}_t + d\vec{n}_t) \quad (2)$$

where d is the Euclidean distance of the points $\vec{x}$ along the normal $\vec{n}_j$ from $\vec{p}$. The above equation can be viewed geometrically as an offset surface at a distance d which defines a plane passing through point $\vec{x}$ as illustrated in Figure 1(b). As shown in previous work [17] for an arbitrary point $\vec{x}$ we can obtain the parameters (α, β, d) for triangle t_r by solving for the plane in the normal-volume which passes through the point. If α, β and $(1 - \alpha - \beta)$, are in the range $[0, 1]$ then the point, x, maps to a point on the triangle surface.

Parameterisation of a high-resolution model, M^H, with respect to a low-resolution model, M^L, is achieved by evaluation of the normal-volume mapping to the high-resolution model vertices. Each model vertex $\vec{v}_i^H$ has an associated set of four mapping parameters $(r_i, \alpha_i^H, \beta_i^H, d_i^H)$ where r_i is the low-resolution triangle index, (α_i^H, β_i^H) are the barycentric coordinates of the corresponding point on the low-resolution model triangle and d_i^H is the distance along the interpolated triangle normal. Having computed this mapping the model can be exactly reproduced and animated from the low-resolution model, M^L, together with the high-resolution vertex parameters and mesh topology.

2.2 Displacement map generation

In this section we introduce a process for computing a displacement map representation of the high-resolution surface model. The displacement map is an efficient image based representation of the high-resolution surface detail. An approximation of the high-resolution model can be rapidly reconstructed from the displacement map image together with the low-resolution control model. Animation of the low-resolution control model can be used to achieve efficient animation of the high-resolution surface detail.

Figure 2 illustrates the process of displacement map generation for a single triangle in the low-resolution control model, M^L. Initially a mapping is established between the high-resolution mesh, M^H, and the control model, M^L, in 3D space using the triangle normal-volume as shown in Figure 2(a). The high-resolution model is then mapped to a 2D image plane using a set of texture coordinates for each triangle in the low-resolution model, Figure 2(b). The distance between the high and low resolution model surfaces along the interpolated surface normal is then sampled on a regular grid to obtain the displacement map image, Figure 2(c).

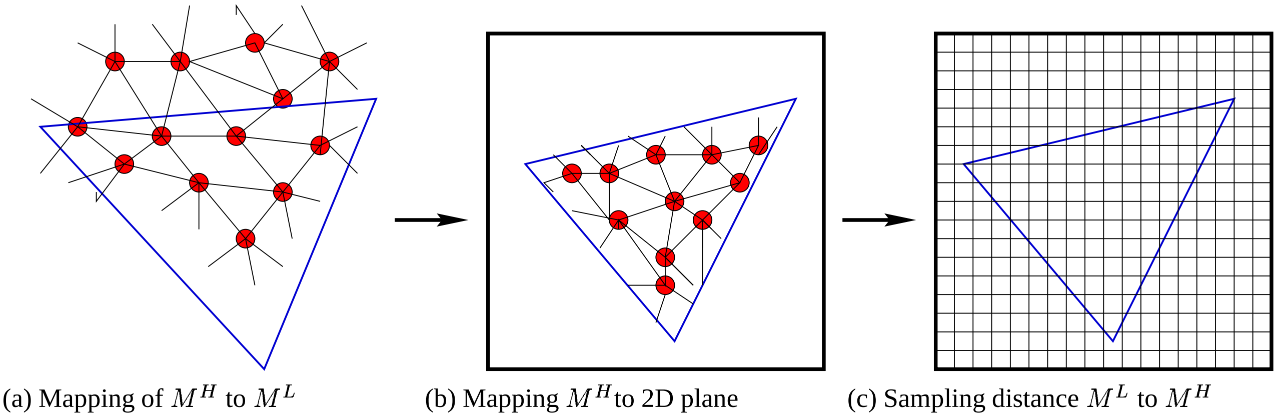
The normal-volume mapping defines a parameterisation for all triangle vertices $\vec{v}_i^H$ in a high-resolution model, M^H, in terms of the nearest triangle t_r^L on a low-resolution control model M^L. The mapping of a vertex, $\vec{v}_i^H$, to a point on triangle t_r^L is defined by equation 2 as a point on the triangle surface in barycentric coordinates $p_i^H = (\alpha_i^H, \beta_i^H)$ and the distance d_i^H along the interpolated triangle normal. Thus, for every vertex $\vec{v}_i^H$ in the high-resolution model M^H we obtain a mapping specified by four parameters $(r_i, \alpha_i^H, \beta_i^H, d_i^H)$ where r is the low-resolution triangle index.

Let us define a set of texture coordinates, $\{\vec{u}_i^L = (u_i^L, v_i^L)\}_{i=0}^{N_{tc}}$, for the low-resolution model M^L which uniquely map each triangle to a two-dimensional texture map plane, T. For each triangle, $t_r^L = (\vec{v}_o^L, \vec{v}_p^L, \vec{v}_q^L)$, there is a continuous mapping to a corresponding triangle in the texture image domain $t_r^{LT} = (\vec{u}_o^L, \vec{u}_p^L, \vec{u}_q^L)$. We can obtain a mapping of the high-resolution model to the texture map plane, T, by combining the low-resolution model texture mapping with the normal-volume mapping of the high-resolution model M^H onto the low-resolution model M^L. Given the normal-volume mapping $(r_i, \alpha_i^H, \beta_i^H, d_i^H)$ for vertex $\vec{v}_i^H$ in the high-resolution model onto triangle t_r^L in the low-resolution model the mapping to texture coordinates is:

$$\vec{u}_i = \alpha_i^H \vec{u}_o^L + \beta_i^H \vec{u}_p^L + (1 - \alpha_i^H - \beta_i^H)\vec{u}_q^L \quad (3)$$

This enables us to map any triangle $t_n^H = (\vec{v}_i^H, \vec{v}_j^H, \vec{v}_k^H)$ in the high-resolution model into a triangle the $t_n^{HT} = (\vec{u}_i^H, \vec{u}_j^H, \vec{u}_k^H)$ in the texture image plane, T. Thus any point on the high-resolution model surface $\vec{x}$ can be mapped to a point in the texture image plane $\vec{u}(\vec{x})$. It should be noted that this mapping is not injective, multiple points on the high-resolution model may map to the same point on the low-resolution model due to overfolding of the surface. This many-to-one projection is a known limitation of displacement maps. In practice this problem can be avoided by either adding additional triangles to the low-resolution model to ensure a one-to-one mapping for the entire surface or by approximating the surface geometry as discussed in section 3.2.

Given the mapping of the high resolution model, M^H, to the texture plane, T, we can obtain a sample of the high-


(a) Mapping of M^H to M^L

(b) Mapping $M^H$to 2D plane

(c) Sampling distance M^L to M^H

Figure 2. Displacement map generation for high-resolution model, M^H.

resolution model surface, $\vec{x}(\vec{u})$, for any point $\vec{u}$ in the texture plane which is inside the region to which the high-resolution model maps. For a point $\vec{u}$ we can find the triangle $t_n^{HT} = (\vec{u}_i^H, \vec{u}_j^H, \vec{u}_k^H)$ from the high-resolution model which it is inside such that:

$$\vec{u} = \gamma \vec{u}_i^{HT} + \nu \vec{u}_j^{HT} + (1 - \gamma - \nu)\vec{u}_k^{HT} \qquad (4)$$

where $\gamma, \nu, (1 - \gamma - \nu) \in [0, 1]$ are barycentric coordinates. From the normal-volume mapping we know the distance d_i^H of each vertex $\vec{v}_i^H$ along the interpolated low-resolution triangle normal. The distance for any point inside the triangle is:

$$d^H(\vec{u}) = \gamma d_i^H + \nu d_j^H + (1 - \gamma - \nu) d_k^H \qquad (5)$$

The mapping defined above enables the distance between the low and high resolution models to be sampled at any point in the image plane. This mapping is used to generate a displacement map image D^H where the distance is sampled for a set of discrete points $\vec{u}(n, m)$ in the image plane. Generation of the displacement map image is discussed further in section 3.2.

2.3 Multiple Level-of-Detail Reconstruction

In this section we present a subdivision scheme for the low-resolution model which enables us to efficiently reconstruct high-resolution models at multiple levels-of-detail (LOD). An approximation of the high-resolution model can be reconstructed from the low-resolution control model M^L together with the set of vertex texture coordinates U^L and the displacement map image D^H. The location of points $\vec{x}$ on the high-resolution model can be reconstructed by sampling the displacement image D^H for any point on the low-resolution model M^L using equation 5. Figure 3 illustrates the reconstruction of a high-resolution model for a single triangle in the low-resolution control model. Initially the control model, M^L, is subdivided according to the required level-of-detail to generate a model, M^{LS} as shown in Figure 3(a). The displacement map image is then sampled at the vertices of the subdivided mesh, M^{LS}, to obtain a distance value. Finally, a new 3D mesh model, M^{HS}, is reconstructed by displacing the vertices of the subdivided mesh, M^{HS}, along the interpolated control model triangle normal by the sampled distance, as shown in Figure 3(c).

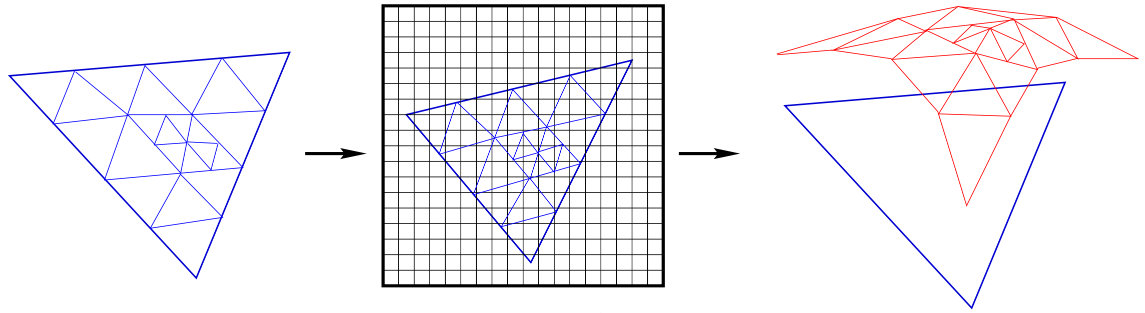
Given a low-resolution triangular mesh model we use uniform quarternery subdivision to generate a mesh to any specified level of detail. A recursive subdivision scheme is used to subdivide each triangle in the mesh into four subtriangles. New vertices are placed at the midpoints of the triangle edges. An arbitrary triangle t_r in the given mesh is split into four triangles after one level subdivision. Carrying on the process each of the four new triangles is further subdivided into four new triangles.

Figure 4(a) shows a low-resolution cube model subdivided at three different resolution levels ($N = 1, 3, 6$). Figure 4(b) shows the reconstruction of a head model at multiple levels-of-detail. The new high-resolution mesh models are generated by subdivision of the low-resolution cube and resampling of the displacement map image, shown in Figure 6(g).

The accuracy of the reconstructed model depends on the sampling resolution of the displacement image. If this resolution corresponds to the smallest triangle edge size, Δl, on the original high-resolution model then the maximum error will be $\frac{\sqrt{3}}{2}\Delta l$. Results indicate that with this sampling resolution there is no visible loss of accuracy in the reconstructed models.

2.4 Animation

The layered model structure enables us to efficiently animate the high-resolution surface based on manipulation of the underlying skeleton joint positions or angles. The model



(a) Subdivision of mesh M^L to obtain M^{LS}

(b) Resampling of displacement map for M^{LS}

(c) Generation of new model M^{HS}

Figure 3. Subdivision regeneration of mesh with variable level-of-detail

is animated in the following stages:

1. Manipulate the skeleton (manual or motion capture).
2. Seamless deformation of low-resolution model, M^L, by computation of new vertex positions.
3. Subdivision of low-resolution model to obtain, M^{LS}.
4. Sampling of displacement map image for each vertex in, M^{LS}.
5. Generation of a new model, M^{HS}, by displacement of vertices in M^{LS}.

Seamless deformation of the low-resolution control mesh, M^L, is implemented using a geometrically based approach. Initially the skeleton model is manually located inside the low-resolution control model. The mapping between the vertices of the control model and the skeleton is automatically established using a point-to-line mapping algorithm [17]. This mapping registers each control model vertex, $\vec{v}_i^L$, in a local-coordinate system with respect to a skeleton segment, $(\vec{o}_j, \vec{o}_k)$, where $\vec{o}_j$ is the position of the j^{th} skeleton joint. The mapping is defined in terms of three parameters $(\alpha_i^L, d_i^L, \vec{n}_i^L)$ where its position is given by:

$$\vec{v}_i^L = \alpha\vec{o}_j + (1-\alpha)\vec{o}_k + d(\alpha\vec{n}_j + (1-\alpha)\vec{n}_k) \qquad (6)$$

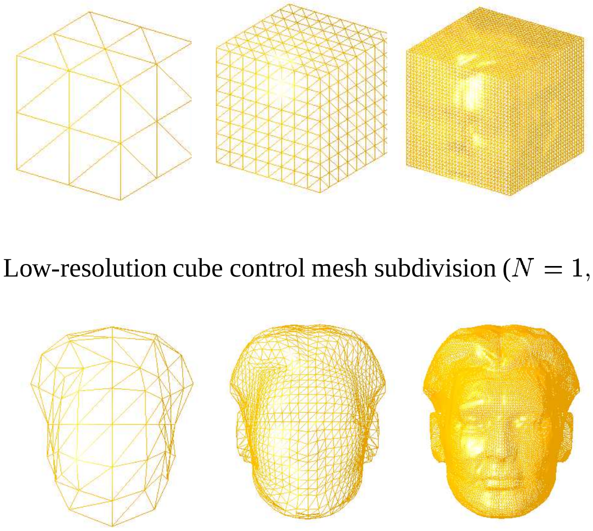
where α is a scalar representing the distance along the segment and d represents the distance from the segment along an interpolated normal. Vector $\vec{n}_j$ is a unit vector defined by the intersection of the plane defined by: $\vec{n}_i^L \cdot (\vec{o}_j - \vec{o}_k) = 0$ and the joint plane. Where the joint plane for the k^{th} joint is defined automatically by the position of its adjacent joints j and l such that: $(\vec{v}_{kl} \times \vec{v}_{kj}) \cdot (\frac{\vec{v}_{kl} + \vec{v_{kj}}}{2}) = 0$ with unit vector $\vec{v_{jk}} = (\vec{o}_k - \vec{o_j})/|(\vec{o}_k - \vec{o}_j)|$. As the joint planes are dependent on the joint positions animation of the joint positions will result in seamless deformation of the vertex positions as defined by equation 6, for further details see [17]. To avoid creases in the mesh for acute joint angles we introduce a scaling factor to ensure that the distance of the vertex $\vec{v}_i^L$ from the segment $(\vec{o}_j, \vec{o}_k)$ is maintained at a constant distance d. This scaling introduces rounding for vertices near the joint resulting in a smooth deformation of the mesh around the joint.

Animation of a new high-resolution model M^{HS} can be efficiently implemented from the animation of the low-resolution model, M^L. Having, animated the low-resolution model, M^L to obtain $M^{L'}$ the new high-resolution model $M^{HS'}$ is reconstructed according to the algorithm introduced in the previous section. Initially the low-resolution model, $M^{L'}$ is subdivided to obtain $M^{LS'}$ based on the animated low-resolution vertex position. The displacement image is then resampled based on the texture coordinates for the low-resolution model. The new high-resolution model, $M^{HS'}$, is then generated by displacing the vertices of the mesh, $M^{LS'}$, along the normal. The normals for the animated low-resolution model $M^{L'}$ are recomputed resulting in deformation of the normal-volume for the low-resolution mesh. The high-resolution vertex positions for the animated high-resolution model are then recomputed using equation 2 based on the modified low-resolution vertex positions and normals.

The cost of animating the high-resolution model is therefore the combined cost of seamless deformation of the low-resolution model (which is real-time) and the cost of generation of the new mesh. The resulting animation of the high-resolution model gives a smooth seamless deformation based on the deformation of the underlying low-resolution model.

3 Implementation

In this section we present details of the implementation of the algorithms used for mesh subdivision and displacement map generation. The implementation of these algorithms is critical to the computational efficiency of generat-



(a) Low-resolution cube control mesh subdivision ($N = 1, 3, 6$)

(b) Reconstructed head models at three LOD ($N = 2, 3, 6$)

Figure 4. Reconstructed subdivision head model at three levels-of-detail

ing the layered model and reconstruction of new models at multiple levels-of-detail. The resulting layered model enables real-time generation and animation of high-resolution surface models.

3.1 Mesh subdivision

Subdivision of the low-resolution triangle mesh is used to generate new high-resolution surface models at multiple levels-of-detail as presented in section 2.2. The reconstruction time for generating new models depends on the time taken to subdivide the low-resolution model. Therefore, a highly efficient scheme has been developed for generating the set of vertices and the triangle topology for a new mesh, M^{LS}, at a specified subdivision level, N.

A scheme has been developed for recursively generating a set of vertex indices for subdividing a triangle t_r^L to a given subdivision level N. The indices are numbered such that the set of vertex positions and more importantly the set of triangles for the subdivision mesh can be generated automatically. Subdivision triangles are defined with vertices in counter-clockwise order. Figure 5 shows the vertex index scheme for subdivision of a triangle in the low-resolution model with $N = 2$. The new triangles are divided into two categories shown as shadowed and unshadowed.

The vertex connections I_n^N for the n^{th} shadowed triangle are defined as:

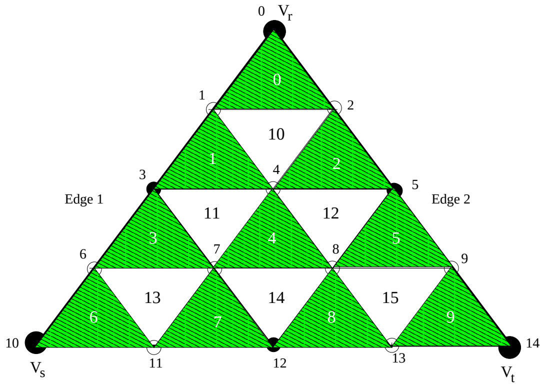


Figure 5. Subdivision indices

$$I_n^N = I_{p+k}^N = [\frac{(i+1)i}{2}+k, \frac{(i+1)i}{2}+k+1, \frac{i(i-1)}{2}+k] \qquad (7)$$

where $p = \frac{i(i-1)}{2}$, $i = 1, ..., 2^N$, $0 \leq k < i$. As shown in Figure 5, $I_0^2 = [1, 2, 0]$, $I_1^2 = [3, 4, 1]$ and $I_2^2 = [4, 5, 2]$.

Unshadowed triangle vertex connections I_n^N, $(2^{N-1}+1)(2^N-1)+2 \leq n < 4^N+1$, are defined as:

$$I_n^N = I_{p+k}^N = [\frac{(i+1)i}{2}+k, \frac{(i+1)(i+2)}{2}+k+1, \frac{i(i+1)}{2}+k+1] \qquad (8)$$

where $p = (2^{N-1}+1)(2^N-1)+\frac{i(i-1)}{2}$, $i = 1, ..., 2^N+1$, $0 \leq k < i$. As shown in Figure 5, $I_{10}^2 = [1, 4, 2]$, $I_{11}^2 = [3, 7, 4]$ and $I_{12}^2 = [4, 8, 5]$.

Each new vertex $\vec{v}_n^N$ in triangle t_i^L is expressed in barycentric coordinates α_n, β_n, $(1-\alpha_n-\beta_n)$ in terms of the three triangle vertices $\vec{v}_r^L$, $\vec{v}_s^L$ and $\vec{v}_t^L$. New vertices v_p^N on the triangle edge $(\vec{v}_r^L, \vec{v}_s^L)$ are given by:

$$\vec{v}_p^N = \alpha_p \vec{v}_r^L + \beta_p \vec{v}_s^L \qquad (9)$$

where $p = \frac{i(i+1)}{2}$, $1 \leq i < 2^N+2$. $\alpha_p = 1 - \frac{i}{2^N}$, $\beta_p = \frac{i}{2^N}$.

Similarly new vertices $\vec{v}_q^N$ on the triangle edge $(\vec{v}_r^L, \vec{v}_t^L)$ are given by:

$$\vec{v}_q^N = \alpha_q \vec{v}_r^L + \beta_q \vec{v}_t^L \qquad (10)$$

where $q = \frac{(i+1)(i+2)}{2} - 1$, $1 \leq i < 2^N+2$. $\alpha_q = 1 - \frac{i}{2^N}$, $\beta_q = 0$.

Based on the vertices on the edge vertices given by equations 9 and 10 the rest of the new vertices in t_r^L can be computed as:

$$\vec{v}_n^N = \vec{v}_{p+k}^N = (1-\frac{k}{i})\vec{v}_p^N + \frac{k}{i}\vec{v}_q^N \qquad (11)$$

where $1 \leq k < i$, $\alpha_n = 1 - \alpha_p$, $\beta_n = \alpha_p(1-\frac{k}{i})$.

The above scheme defines the subdivision for each individual triangle in the low-resolution model, M^L. In practice adjacent triangles will share vertices in the subdivision mesh. To avoid redundancy edge-sharing information is used in the algorithm to avoid repetition of vertices. When a triangle is subdivided, the new vertices on its edges will be shared automatically by its neighbour triangles.

The output of the subdivision algorithm is a new mesh M^{LS} composed of a set of vertices and triangles, together with the barycentric coordinates of each new vertex with respect to a low-resolution triangle. The barycentric coordinates of each new vertex give its relative position inside a triangle of the original mesh. As the barycentric coordinates do not change under affine transformations they can be used to map the vertices of the subdivided model into the texture image plane. Resampling of the displacement image for each vertex on the subdivision model enables generation of a new high-resolution model. The barycentric coordinates also remain constant when the low-resolution control mesh is deformed during animation and can therefore be used to reconstruct and animate new high-resolution models.

This subdivision scheme gives a closed form algebraic solution for generating a new mesh M^{LS} at subdivision level N in real-time. Generation of the new high-resolution model M^{HS} is based on a simple lookup in the displacement image using barycentric coordinates. Therefore, the new model M^{HS} can be generated in close to real-time.

3.2 Displacement map generation

This section explains the practical details of displacement map generation. There are a number of issues which must be considered in implementing the displacement map generation. The principal consideration is to ensure a continuous mapping (without gaps) of the high-resolution model surface to the displacement image. Secondly, implementation should consider the computational efficiency of the displacement map generation. Finally, implementation should deal with the possibility of many-to-one mapping between the high-resolution model surface and displacement map. Figure 6 illustrates the displacement map generation for a high-resolution 3D head model on to a simple cube model. Figure 6(a-c) shows the 3D model and model mapping, (d) shows the mapping of the control model to the texture image plane, (e-f) show images of intermediate stages and (g) shows the resulting displacement map image. The scalar displacement maps values are shown in pseudocolour from red to blue with increasing magnitude of the displacement.

Two basic stategies to the displacement map generation could be adopted:

- **Forward mapping** takes each triangle, t_i^H, on the high-resolution model and maps it via the low-resolution control model triangle, t_r^L, into the texture map plane, t_i^{HT}. For each displacement image pixel, $\vec{u}(n,m)$, we then sample the distance between the low and high resolution model along the normal using equation 5.
- **Backward mapping** takes each point in the displacement map image, $\vec{u}(n,m)$, and find the triangles on the low, t_r^{LT}, and high ,t_i^{HT}, resolution models which the 2D point is inside. The distance between the low and high resolution models is then computed from equation 5.

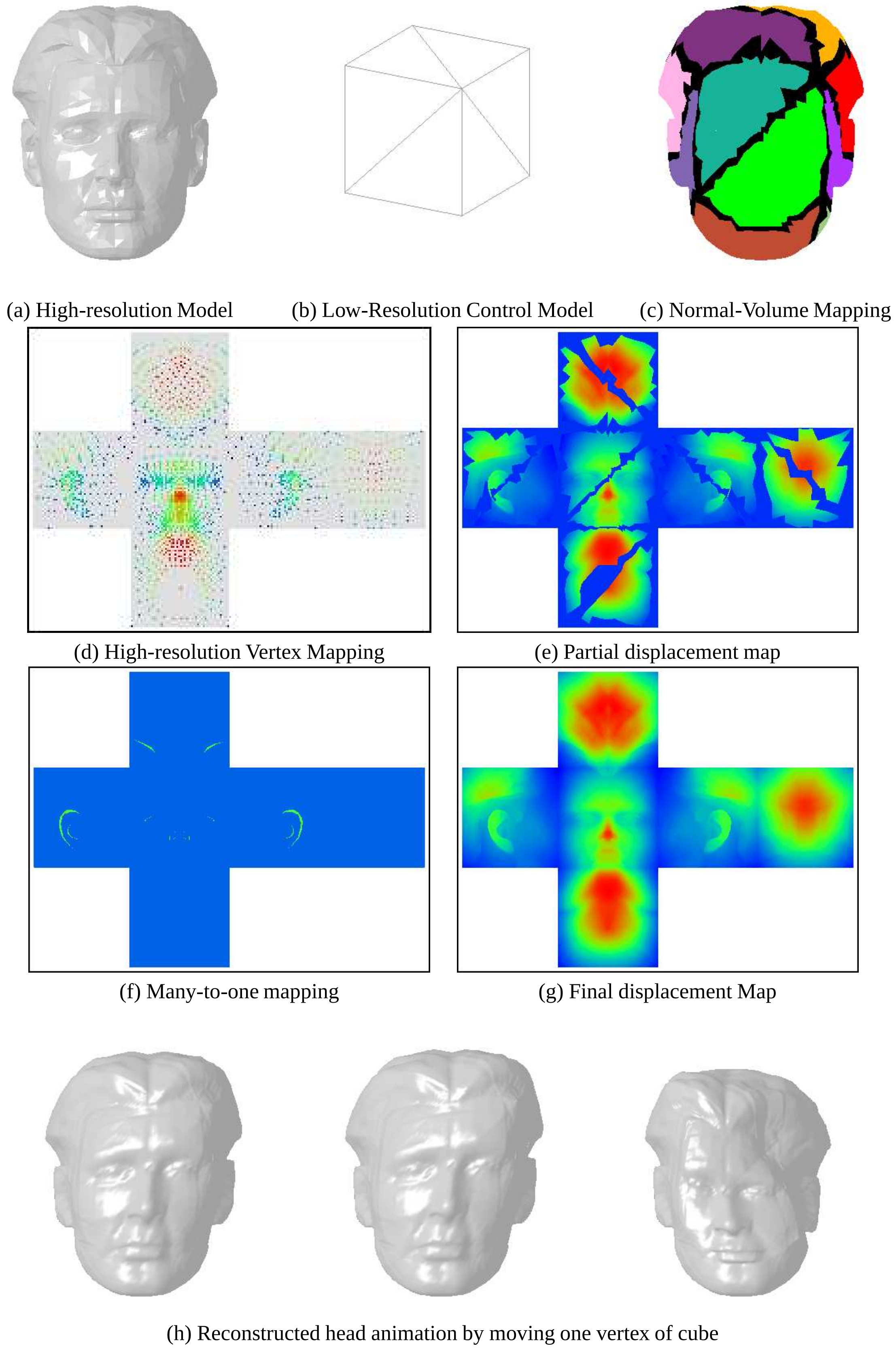
Backward mapping requires a search to find the corresponding triangles for each point. This search even in the 2D texture plane is potentially computationally expensive for high-resolution models with a large number of triangles. A second problem with the backward mapping is that due to the many-to-one mapping from the high-resolution model to the texture domain we must search for the set of corresponding triangles which map each point in the texture image. Therefore, in this work we have used forward mapping to implement the displacement map generation.

Forward mapping gives a closed form solution to finding the set of points in the texture map for each triangle on the high-resolution model. This provides an efficient method for generating the displacement map image provided that the high-resolution triangle, t_i^H, only maps to a single triangle on the low-resolution model, t_r^L. This is the case for the majority of triangles where all three vertices of t_i^H map to a single triangle t_r^L. As both the low and high resolution triangles are by definition convex it is guaranteed that all points inside t_i^H map to points inside t_r^L.

However, if the vertices of t_i^H map to multiple triangles in the low-resolution model then we must consider the mapping of each part of the high-resolution triangle to the low-resolution model separately. In this case the mapping of the high-resolution triangle is only piecewise linear for each low-resolution triangle, this is illustrated in Figure 7(a). In addition, the high-resolution triangle may map to disconnected regions of the displacement map if the low-resolution triangles are not adjacent in the texture plane. In this section we introduce a closed form solution which enables efficient implementation of the mapping for high-resolution triangles which map to multiple low-resolution triangles.

3.2.1 Vertex Mapping

The first stage in generating a displacement map is to map the low-resolution control model into the 2D texture image plane. Each triangle in the control model will have texture coordinates for each vertex as discussed in section 2.1. This enables each low-resolution model triangle, t_r^L, to be



(a) High-resolution Model

(b) Low-Resolution Control Model

(c) Normal-Volume Mapping

(d) High-resolution Vertex Mapping

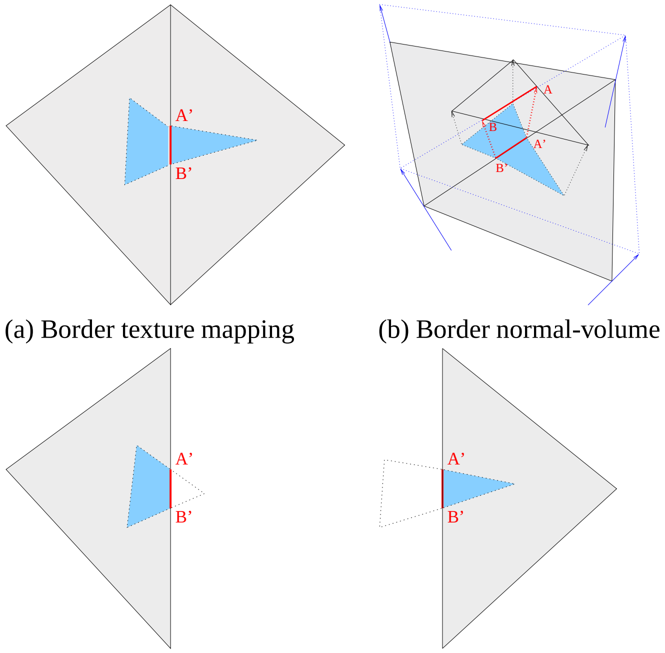
(e) Partial displacement map

(f) Many-to-one mapping

(g) Final displacement Map

(h) Reconstructed head animation by moving one vertex of cube

Figure 6. Displacement map generation for head model with simple cube control model



(a) Border texture mapping

(b) Border normal-volume

(c) Left-hand triangle

(d) Right-hand triangle

Figure 7. Normal-volume mapping of a high-resolution border triangle to multiple low-resolution triangles

uniquely mapped to non-overlapping triangles in the texture image plane, t_r^{LT}. The mapping presented in section 2.1 enables us to map each vertex from the high-resolution model $\vec{v}_i^H$ to a point in the texture image plane, $\vec{u}_i^H$. Mapping of vertices in the high-resolution model is illustrated in Figure 6(d). The distance for each vertex between the high-resolution and low-resolution surface is given by the normal-volume mapping distance parameter, d_i^H.

3.2.2 Filling Triangles

Having mapped the individual high-resolution vertices the next stage is to evaluate the distance for points in the displacement map which correspond to points on the high-resolution surface. The forward mapping of the high-resolution triangle vertices to the texture image plane defines texture coordinates for the high-resolution triangle vertices. As discussed previously there are two cases of triangle mapping to be considered:

1. Triangles t_i^H whose vertices map to a single low-resolution control triangle t_r^L.

2. Border triangles t_j^H whose vertices map to multiple low-resolution control triangles.

In the first case, we simply evaluate a displacement value using equation 5 from the previously computed high-resolution triangle vertex distances for each point $\vec{u}(n,m)$ which is inside the triangle, t_i^{HT}. Computation of the distance is a linear interpolation using barycentric coordinates and is therefore highly efficient. This case covers the majority of triangles in the high-resolution model. The resulting partial displacement map for a typical model is shown in Figure 6(e).

The remaining unmapped border triangles have vertices which map to more than one low-resolution triangle. The resulting mapping of the high-resolution triangle surface will intersect the set of edges between adjacent triangles to which the vertices map. The mapping across multiple triangles is illustrated in Figure 7(b). As the normal-volume mapping is in general different for adjacent triangles the continuous mapping of the surface of the high-resolution triangle to the low-resolution model is only piecewise linear for each triangle in the low-resolution model. This results in the high-resolution triangle t_i^H mapping to a non-triangular polygon p_i^{HT} in the displacement map image plane, as illustrated in Figure 7(a). Each intersection of the high-resolution triangle edge with a low-resolution triangle edge will result in one additional vertex in the polygon p_i^{HT}.

Generation of the displacement map for border triangles requires evaluation of the mapping for points where the triangle edge intersect the boundary. This can be achieved by evaluating the normal-volume mapping for each of the high-resolution triangle vertices in the plane of each low-resolution triangle which it intersects. The key difference here is that one or more of the triangle vertices may map outside the low-resolution triangle resulting in negative barycentric coordinates. Figure 7(c,d) shows the normal-volume mapping of the high-resolution border triangle to two adjacent low resolution triangles.

Having evaluated the mapping for each vertex of the high-resolution triangle, t_i^H, with respect to a particular low-resolution triangle, t_r^L, we can map it to a triangle in the texture image plane, $t_i^{HT_r}$. Where one or more of the vertices of $t_i^{HT_r}$ is outside t_r^{LT}. The part of $t_i^{HT_r}$ which is inside t_r^{LT} gives the correct mapping of the portion of t_i^H which maps onto t_r^L through the normal-volume mapping. This is the correct mapping because the intersection of the mapping of t_i^H with the edge of t_r^L corresponds to the boundary surface between the normal-volume of t_r^L and the adjacent triangle. This boundary surface is continuous and intersects the high-resolution t_i^H along a line, as shown in Figure 7(b). Therefore projection of triangle t_i^H to the plane of t_r^L gives the correct points of intersection with the triangle edge.

Border triangles which overlap multiple low-resolution triangles can therefore be correctly mapped to the texture image in the following steps. For each low-resolution triangle t_r^L to which the high-resolution triangle t_i^H maps:

1. Map t_i^H onto the plane of t_r^L using the normal-volume.

2. Map t_i^H into the 2D texture image plane using the texture coordinates for t_r^{LT} to obtain $t_i^{HT_r}$.

3. For each point $\vec{u}(n,m)$ which is inside both the high-resolution triangle $t_i^{HT_r}$ and inside the low-resolution triangle t_r^{LT} evaluate the distance between the low and high resolution triangle from equation 5.

Repeating the above algorithm for each triangle to which the high-resolution triangle is mapped is implemented by steping between adjacent low-resolution triangles which the edges of the high-resolution triangle intersect. This ensures that the high-resolution triangle is mapped continuously into the texture image domain. Figure 6(g) shows the resulting pseudo-colour displacement map image with all triangles in the high-resolution model mapped.

3.2.3 Resolving many-to-one mapping

As discussed previously, overfolding of the surface may result in multiple parts of the high-resolution model being mapped to the same location on the low-resolution model or texture map plane. This is illustrated in Figure 6(f) where image points are coloured according to the number of high-resolution triangles which map to that point. The majority of the image is dark blue indicating that one triangle maps per point, green indicates two triangles and red three. It should be noted that in practice even in the case of a complex shape such as the head mapped to a simple cube control model only a few points in the displacement map image have correspond to more than one high-resolution triangle. Multiple triangle mapping occur around the ears, nose and hairline where the model overfolds relative to the cube.

In regions of the texture map plane where multiple maps occur we implement the displacement map generation by taking the largest distance to any point on the surface: $d^H(\vec{u}) = max\{d^H(\vec{u})\}$. This results in the reconstructed model surface being the convex hull of the original mesh. For meshes such as the head model this does not result in significant visual degradation of the reconstructed model as the unmodelled regions such as behind the ear are only visible from a small number of views.

However, for more complex models, such as a hand mapped to a cube, taking the largest distance would result in incorrect reconstruction of the fingers. In this situation it is necessary to have a more sophisticated control model which approximates the object shape. In the case of the hand this could be achieved by using a control model with fingers.

3.2.4 Image Sampling, quantisation and scaling

The mapping defined in section 2.1 enables us to sample the distance function $d(\vec{u})$ for points on a discrete grid, $\vec{u}(n,m) = (n\Delta u, m\Delta v)$, in the texture map plane to form a displacement map image $D^H = d^H(\vec{u}(n,m))$ for $n = 0, N-1$ and $m = 0, M-1$. To obtain an efficient displacement image representation we quantise the displacement $d(\vec{u})$ into 2^n levels. Distance values are scaled using the range of values in the entire image $[d_{min}, d_{max}]$ and quantised into discrete levels as:

$$d_q^H(n,m) = 2^n \frac{(d^H - d_{min}^H)}{(d_{max}^H - d_{min}^H)} \qquad (12)$$

The resulting quantised displacement image representation of the high-resolution model gives an efficient approximation which can be stored using standard image representations. Typically the displacement at each pixel can be represented by a single 8 or 16 bit number whereas each vertex in the original high-resolution model required three 32 bit floating point numbers. A quantitative comparison of representation costs is given in the results section. The results presented in this work use single byte values with $n = 8$ giving 256 distance levels. Images are stored in the standard grey scale image format with the range $[d_{min}, d_{max}]$ stored in the file header for reconstruction.

4 Results

This section presents initial results of for the generation of layered models using displacement maps. Layered displacement map models have been generated from high-resolution data of a human head and monster head. Results for these models are shown in Figures 6 and 8. The top-line of each figure shows the original high-resolution model, the low-resolution model and the normal volume mapping. Generation of models at multiple levels-of-detail is shown in Figure 8(d). Animation of the reconstructed models based on deformation of the low-resolution control model is shown in Figure 6(h) and 8(e). Results show that deformation of the control model enables efficient deformable animation of the high-resolution surface detail.

Table 1 shows the representation and computation costs for the layered animation model. The representation cost for the layered model is the combined size of the low-resolution control model and the displacement map image. Results show that the representation cost for the layered model is an order of magnitude smaller than for the original model. Results for the computation show the combined time for mesh subdivision and resampling the displacement map to generate the high-resolution model on a Intel PIII450MHz. Initial results indicate that even for the monster head model with subdivision level $N = 5$ the computation is performed

	Model Size		Generation Time	
	Original	Layered	$N = 3$	$N = 5$
Human Head	127Kb	11Kb	34ms	61ms
Monster Head	935Kb	34Kb	167ms	332ms

Table 1. Size and time costs

in 0.3 seconds for 60K vertices. This indicates that with the use of more efficient adaptive subdivision new models could be generated and animated in real-time using the layered displacement map model.

5 Conclusions

In this paper we have introduced a layered animation framework which uses a displacement map image to represent the high-resolution surface detail. Automatic generation of a displacement map image is achieved by mapping the high-resolution surface detail onto a low-resolution control model using the normal-volume mapping. The principal contributions of this approach are:

1. Automatic displacement map generation for a low-resolution model.
2. Seamless deformable animation of displacement.
3. Efficient representation of highly detailed surfaces.
4. Rapid generation of surface models at multiple levels-of-detail.
5. Efficient animation of high-resolution surface based on a low-resolution control model.

The results presented in this paper demonstrate the potential of animated displacement maps for representing complex surfaces such as those obtained using 3D surface measurement devices. To our knowledge, this is the first work to present both automatic generation and animation of displacement maps. Results demonstrate that the displacement map representation gives an order of magnitude reduction in model size and an order of magnitude increase in computational efficiency for animation.

Further work is required to address: adaptive subdivision; many-to-one mapping; and generation of models for a fixed geometric or appearance tolerance.

References

[1] C. Babski and D. Thalmann. A seamless shape for hanim compliant bodies. In *Proc. VRML'99*, pages 21—28, 1999.

[2] P. Cignoni, C. Montani, and R. Scopigno. A comparison of mesh simplification algorthms. *Computer and Graphics*, 22(1), 1998.

[3] B. Curless and M. Levoy. A Volumetric Method for Building Complex Models from Range Images. In *ACM Computer Graphics Proceedings, SIGGRAPH, NewOrleans,USA*, pages 303–312, 1996.

[4] Viewpoint Datalabs. *Viewpoint Catalog*. (http://www.viewpoint.com), 1996.

[5] M. Eck, T. DeRose, T. Duchamp, H. Hoppe, M. Lounsbery, and W. Stuetzle. Multiresolution analysis of arbitary meshes. In *SIGGRAPH*, pages 173—182, 1995.

[6] M. Eck and H. Hoppe. Automatic reconstruction of b-spline surfaces of arbitrary topological type. In *Proc. ACM SIGGRAPH*, pages 325—334, 1996.

[7] A. Hilton, A.J. Stoddart, J. Illingworth, and T. Windeatt. Implicit surface based geometric fusion. *International Journal of Computer Vision and Image Understanding, Special Issue on CAD Based Vision*, 69(3):273–291, March 1998.

[8] H. Hoppe. Progressive meshes. In *Proc. ACM SIGGRAPH*, pages 99—108, 1996.

[9] H. Hoppe, T. DeRose, T. Duchamp, M. Halstead, J. Hin, H. McDonald, J. Schweitzer, and W. Stuetzle. Piecewise smooth surface reconstruction. In *Computer Graphics Proceedings, SIGGRAPH*, pages 295—302, 1994.

[10] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Mesh optimization. In *Computer Graphics Proceedings, SIGGRAPH*, pages 19—26, 1993.

[11] John E.Chadwick, David R.Haumann and Richard E.Parent. Layered Construction for Deformable Animated Characters. In *Proceedings of SIGGRAPH '89, Boston, 31 July-4 August, 1989*, pages 243–252, 1989.

[12] N. Kalra, N. Magnenat-Thalmann, L. Moccozet, G. Sannier, A. Aubel, and D. Thalmann. Real-time animation of virtual humans. *IEEE Computer Graphics and Applications*, 18(5):42—56, 1998.

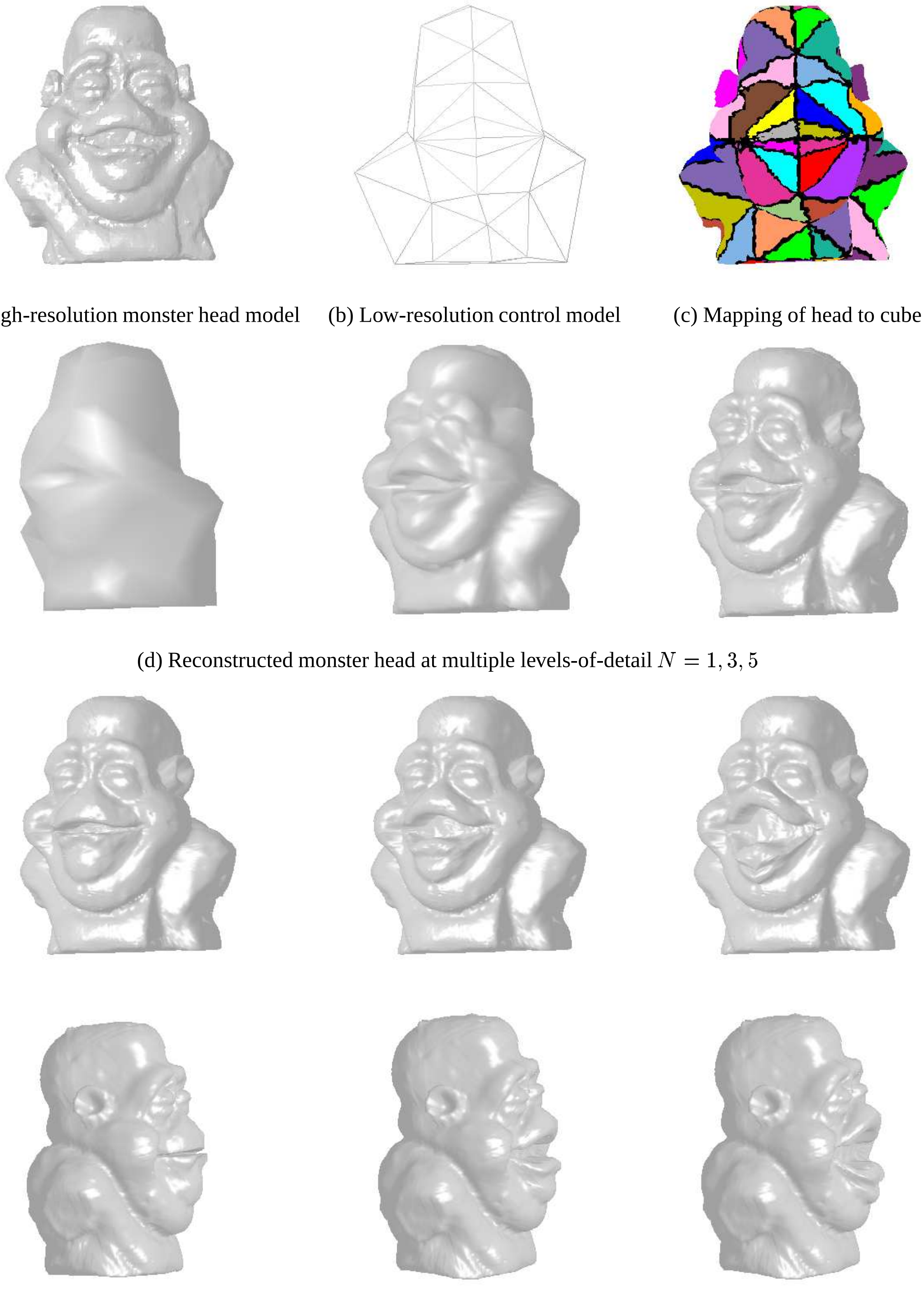
[13] A. Kalvin and R. Taylor. Superfaces: polygonal mesh simplification with bounded error. *IEEE Computer Graphics and Applications*, 16(3):64—77, 1996.

[14] V. Krishnamurthy and M. Levoy. Fitting smooth surfaces to dense polygon meshes. In *In ACM Computer Graphics Proc. SIGGRAPH, NewOrleans, USA*, 1996.

[15] L. Mocozzet and N. Mangnenat Thalmann. Dirichlet free-form deformations and their applications to hand simulation. In *IEEE Int.Conf. on Computer Animation*, 1997.

[16] W.J. Schroeder, J.A. Zarge, and W.E. Lorensen. Decimation of triangular meshes. In *SIGGRAPH 26(2)*, pages 65—70, 1992.

[17] 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.



(a) High-resolution monster head model (b) Low-resolution control model (c) Mapping of head to cube

(d) Reconstructed monster head at multiple levels-of-detail $N = 1, 3, 5$

(e) Monster head mouth animation using layered displacement map model $N = 5$

Figure 8. Layered displacement map model and animation for monster head