Automatic Generation of Subdivision Surface Head Models from Point Cloud Data

Won-Ki Jeong Kolja Kähler Jörg Haber Hans-Peter Seidel

Max-Planck-Institut für Informatik, Stuhlsatzenhausweg 85, 66123 Saarbrücken, Germany
{jeong,kaehler,haberj,hpseidel}@mpi-sb.mpg.de

Abstract

An automatic procedure is presented to generate a multiresolution head model from sampled surface data. A generic control mesh serves as the starting point for a fitting algorithm that approximates the points in an unstructured set of surface samples, e.g. a point cloud obtained directly from range scans of an individual. A hierarchical representation of the model is generated by repeated refinement using subdivision rules and measuring displacements to the input data. Key features of our method are the fully automated construction process, the ability to deal with noisy and incomplete input data, and no requirement for further processing of the scan data after registering the range images into a single point cloud.

Key words: range scans, subdivision surface, facial animation, point cloud fitting

1 Introduction

In the task of modeling human heads from real individuals for facial animation we are usually confronted with two conflicting goals: one is the requirement for accurate reproduction of facial features, the other is the demand for an efficient representation which can be animated easily and quickly. Additional difficulties are brought in by limitations of current range scanning technology: the data is often noisy and has "holes" due to shadowing effects or bad reflection properties of the scanned surface. Some data, like the part of the lips on the inside of the mouth, cannot be captured at all. To create a triangle mesh that is suitable for real-time animation, extensive manual postprocessing of the scanned geometry is often necessary, followed by mesh simplification to reduce the complexity of the mesh. Unfortunately, mesh decimation techniques cannot exert enough control over the connectivity of the mesh to obtain an optimal mesh for animation: the distribution of vertices and alignment of edges should correspond to the basic symmetry of the face and the potential deformations of the mesh, which are not easily derived from the static shape. With low-polygon models, a single misplaced edge can destroy the visual impression of a smooth surface. Resorting to higher resolution meshes is nonetheless undesirable due to the additional computational load for the animation system.

We chose to use a subdivision surface representation in our facial animation environment. Subdivision surfaces have become increasingly popular due to their ability to bridge the gap between polygon meshes and higher-order surfaces. Since they are constructed by repeated refinement of triangle meshes up to an arbitrarily close approximation to their smooth limit surface, they also provide an effective means to control accuracy and efficiency in a systematic manner. By constructing the surface from a control mesh of known topology, we can also avoid the instabilities that are incurred by animating an irregular triangle mesh.

In this paper, we present our approach for generation of multiresolution head model geometry from sampled surface data. The method takes as input an unstructured point cloud that is obtained from range scans of an individual. A fully automatic procedure is used to fit a hand-designed generic control mesh to this point cloud, taking special care to match facial features such as ears and mouth. A hierarchical structure of displaced subdivision surfaces is then constructed, which approximates the input geometry with increasing precision, up to the sampling resolution of the input data. Our method generates useful models from noisy and incomplete input data, with no requirement for further processing of the scan data except for registration into a single point cloud. Figure 1 shows the main stages of our method.

Since we use an interpolating subdivision scheme, displacing a vertex of the subdivision control mesh corresponds directly to the local change of the surface. In our physics-based facial animation system, we interpret the vertices and edges of the control mesh as point masses and springs which are reacting to forces applied by virtual muscles. Animating this very coarse spring mesh is computationally efficient, while the level of detail for rendering the geometry is controlled separately. For a given refinement level, the support of a vertex in the control mesh is known, and thus we can achieve efficient animation by only locally updating the mesh structure.

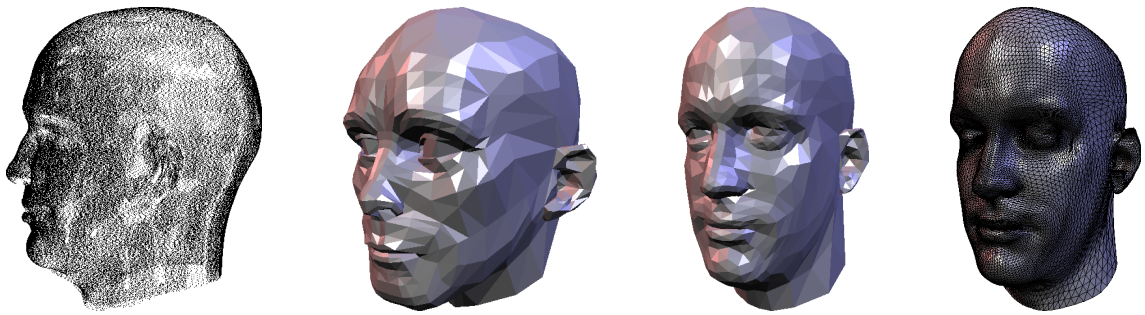

Figure 1: The main stages of head model construction. From left to right: a) a dense set of surface samples obtained directly from a range scanner (thinned in this image for visualization); b) generic mesh; c) deformed mesh used as subdivision control mesh; d) subdivision surface fitted to range data.

2 Previous Work

The literature on surface representations used in modeling and animation is vast. Since we are interested in constructing multiresolution models for real-time facial animation, we focus here on triangle mesh-based structures, because of their conceptual simplicity and the availability of efficient graphics hardware for rendering.

Adaptive refinement of arbitrary triangle meshes is an active topic in multiresolution editing [26, 11]. While these methods provide powerful tools for mesh deformations, the computational complexity is still considerably too high for real-time applications.

In physics-based animation, the vertices of a deformable mesh surface are often interpreted as nodes of a spring mesh [13, 22]. Adaptively refining such a mass-spring system is non-trivial [7]. An efficient method to smooth polygonal geometry proposed by VOLINO *et al.* [23] can be applied to the deformed geometry that results from a mass-spring simulation to improve visual quality. The technique uses on-the-fly refinement of polygons, where additional vertex positions are computed from the vertex normals of the original mesh. The resulting surface has smoother appearance, but does not contain additional geometric detail.

Subdivision surfaces have been successfully used in computer animation [3]. A wide range of subdivision schemes exists today, basically structured by the type of the control mesh (quadrilateral or triangular), the order of continuity they can achieve (usually C^1 or C^2) and whether they approximate or interpolate the control mesh after refinement [24]. While most subdivision schemes apply the subdivision operator uniformly to the surface, some recent results demonstrate adaptive refinement [26, 10]. The algorithms become considerably more complex in these cases.

PIGHIN *et al.* [18] have used an approach based on radial basis functions to match a generic head mesh to several photographs of a head simultaneously. Their approach doesn't need additional hardware besides a low-cost still camera, but requires the manual specification of facial features in all of the views. Animation is limited, because each expression needs to be captured in advance. An optimization process proposed by BLANZ *et al.* [2] generates close approximations to even only a single photograph. This technique draws from a large database of several hundred scanned faces. The resulting model has the same resolution as the scanned faces and cannot be readily animated.

In the context of medical imaging, SZELISKI *et al.* [21] minimize the distance between two surfaces obtained from volume scans of human heads by applying local and global deformations in a hierarchical manner. The deformations are modelled by a combination of global polynomial deformations and local free-form deformations [20]. The method does not require specification of corresponding features on the geometries.

The goal in the method presented by LEE *et al.* [13] is the automated construction of animatable head models from range scans. They adapt a generic face mesh with embedded muscle vectors to range scans of human heads. This process is largely automated, but relies on the inherent automatic registration of the texture to the range data. The model created from the scan data is fully animatable. The generated mesh approximates the input geometry well on a rather coarse detail level.

MARSCHNER *et al.* [17] match a subdivision surface to geometry measured by a range scanner. They use Loop subdivision rules [16] and a fitting algorithm based on the work of HOPPE *et al.* [5]. A continuous optimization process alters vertex positions to minimize an energy functional. For rendering, the surface is subdivided to the desired level. Due to the approximating nature of Loop subdivision, the coarser levels of the subdivision hierarchy do not resemble the scanned geometry everywhere.

Motion of the face is specified by varying positions of sample points on the surface and computing the corresponding control vertex displacements.

3 Overview

In our approach, facial geometry is acquired from real humans using a range scanner. The individuals are captured with a closed mouth and neutral expression. No triangulation or further post-processing such as hole-filling is performed on the scan data. The result of the acquisition stage is an unstructured, dense point cloud, possibly with infrequent large holes due to missing or bad data.

Given the generic head model as shown in Figure 1 b), we deform this mesh to approximate the point cloud in a three-step procedure, which runs automatically without user intervention:

1. **Initial alignment:** This step automatically computes an affine transformation (rotation, translation and non-uniform scaling) that minimizes the difference between the silhouette of the generic head model and the outer hull of the point cloud. This procedure uses an iterative optimization technique, and we exploit graphics hardware for evaluating the current match to achieve fast convergence.

2. **Local fitting:** Due to individual differences in facial proportions, the global affine transformation does usually not result in good registration of the prominent features of the face. Thus, we apply another optimization to the regions containing the ears, nose, and mouth, which are marked in the prototype mesh. Here, a local rigid transformation is found that minimizes the distance from the point samples to the mesh surface. The transformation is applied directly to the mesh regions and blended into the surrounding parts of the mesh to achieve smooth transitions.

3. **Global fitting:** The deformed prototype mesh is now well-aligned to the facial features of the sample data. The final fitting step reduces the distance from the point cloud to the mesh by global energy minimization. The employed energy functional is essentially the same as used by HOPPE *et al.* [6] and MARSCHNER *et al.* [17]. In addition to minimization of the distance between point cloud and mesh, the function accounts for smoothness and constraints defined on the generic mesh.

All three steps of this fitting procedure are explained in detail in Section 4. Provided the deformed generic mesh, we proceed with the construction of the subdivision hierarchy, using the interpolating Modified Butterfly scheme [25]. On the base level and on each level of refinement, the vertices are displaced along triangle normal direction to lie on the surface sampled by the point cloud. In this manner, a hierarchy of surfaces is generated with locally encoded details, similar to normal meshes [4]. The construction process is detailed in Section 5. For animation, we rebuild the surface in the changed areas, such that the local detail will follow the deformation properly, see Section 6.

4 Fitting the Control Mesh

For the following discussion we need some notation. We define a triangle mesh $\mathcal{M}$ as a tuple $(\mathcal{V}_\mathcal{M}, \mathcal{E}_\mathcal{M}, \mathcal{T}_\mathcal{M})$, denoting the vertices, edges and triangles of the mesh, respectively. We refer to the initial generic mesh as $\mathcal{G}$. This mesh will be deformed over the three fitting stages by updating its vertex positions, but leaving the connectivity unchanged. The sample data is given as a set of points $\mathcal{P}$. All points and mesh vertex positions are given as 3D coordinates and will be written in bold face: $\mathbf{p} \in \mathbb{R}^3$. For convenience, for a vertex $v \in \mathcal{V}_\mathcal{M}$, $\mathbf{v}$ will denote its position. We write $star(v)$ for the set of vertices directly connected to v via an edge, and $valence(v)$ for the number of these adjacent vertices.

4.1 Initial alignment

In the initial step of the subdivision surface fitting process, we approximately align the generic head model $\mathcal{G}$ to the point cloud using an affine transformation T. Throughout this section, $\mathcal{G}$ will mean the transformed version of the generic mesh using the current T, which is used – after convergence – as the input for the next step described in Section 4.2. The parameters for T are determined fully automatically by exploiting graphics hardware to perform a silhouette-based geometry fitting. Our approach thus extends the idea of the 2D silhouette-based texture mapping technique presented in [14, 15] to three-dimensional geometry.

To evaluate the current T, we render both $\mathcal{G}$ and $\mathcal{P}$ into a common frame buffer for one of the three canonical viewing directions along the coordinate axes. The frame buffer is initialized to black. Next, we render the point cloud with a white color, an identity modelview matrix, and no further lighting or depth test. We then set the OpenGL logical fragment operation to XOR and render the generic head model with the modelview matrix set to T and identical rendering parameters. Now the frame buffer contains those pixels in white that are covered by only one of the two geometric objects $\mathcal{G}$ and $\mathcal{P}$. The frame buffer image can thus be interpreted as the silhouette-based difference between $\mathcal{G}$ and $\mathcal{P}$ for the cho-

sen viewing direction[1]. Finally, the number of white pixels in the frame buffer is evaluated using the `glGetHistogram` function. It is a measure of how well the objects $\mathcal{G}$ and $\mathcal{P}$ are aligned in their image space projection. This process is repeated for all three different viewing directions in turn and their difference measures are summed up. The resulting *total difference* D is used to control an iterative optimization process to determine T.

We initialize the scaling and translation parameters of T such that the bounding boxes of the point cloud and the generic head model have the same size and center point. Since we know from our range scanner that the point cloud is roughly oriented in such a way that the face looks along the z-axis and the up-vector coincides with the y-axis, we set the initial rotation parameters of T accordingly. For the optimization process, we apply Powell's method [19, Sec. 10.5] to the set of parameters of T. The function to be minimized is given by the total difference D. Figure 2 shows three silhouette-based difference images (according to three orthogonal viewing directions) both after the initialization step and after convergence of the optimization process.

In addition, we have to take into account that the part of the point cloud $\mathcal{P}$ representing the neck of the head might be longer or shorter than the corresponding part of the generic head model $\mathcal{G}$. To avoid artificial "difference pixels" due to these non-corresponding parts of $\mathcal{P}$ and $\mathcal{G}$, we introduce a clipping plane in the world coordinate system (i.e. the location of the clipping plane is independent of T) to cut away those unwanted parts of $\mathcal{P}$ and $\mathcal{G}$. Also, the back of the head can often not be captured appropriately due to difficulties with structured light scanners in hair-covered head regions, so we add another clipping plane that removes this part in $\mathcal{P}$ and $\mathcal{G}$ as well. We provide a generic position and orientation for each of these planes based on the initial T, which may have to be modified by the user before the fitting process is started, according to the specifics of the range scan data.

Further optimization of the process can be achieved by initially rendering the point cloud once for each viewing direction and storing the resulting frame buffer images in three textures. During the optimization process, we then only need to render one textured quad for each viewing direction instead of rendering the full point cloud.

4.2 Adaptive Local Alignment

The initial alignment step gives an optimal fit for the generic mesh using an affine transform. Since individual faces are of different proportions, the main facial features like ears, nose and mouth are generally not registered well in the transformed generic mesh and the point cloud, which prohibits a direct execution of the global distance minimization procedure (section 4.3). Hence, we apply local deformations to the mesh in four areas of the generic model (one for each ear, the nose and the mouth).

[1]If the point cloud does not contain enough data points to result in a fully covered silhouette during rendering, we repeat this step with `glPointsize` set to a slightly higher value.

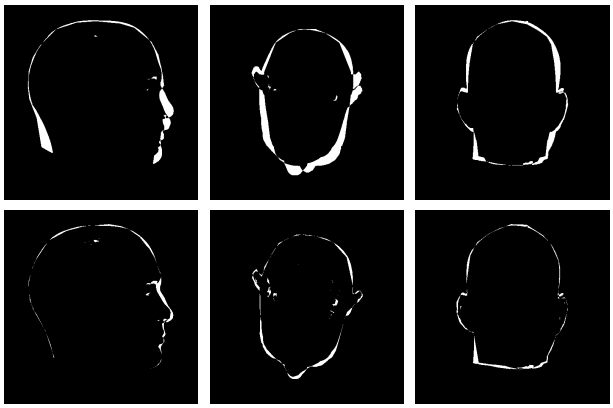

Figure 2: Silhouette-based geometry fitting: white pixels indicate the difference between the silhouettes of the generic head model and the point cloud. Top row: initial difference images for three orthogonal viewing directions. Bottom row: difference images after optimization process.

For this purpose, a set of bounding boxes $\mathcal{B}$ enclosing each feature is predefined for the generic mesh, as shown in Figure 4. The bounding boxes are chosen large enough to ensure that after the initial alignment the sample points belonging to the respective feature on the point cloud are also included in the box. In the following, we write $\mathcal{G}^b$ for the part of the generic mesh $\mathcal{G}$ that is enclosed in bounding box $b \in \mathcal{B}$. Similarly, $\mathcal{P}^b$ is the portion of the point cloud enclosed in box b.

For each bounding box, we now minimize the energy functional $E_{local} = E_{dist} + E_{stretch}$ by optimizing the position of the mesh vertices included in the box. E_{dist} is minimized as the distance from every point in the point cloud to the mesh becomes smaller:

$$E_{dist}(\mathcal{G}^b) = \sum_{\mathbf{p} \in \mathcal{P}^b} (\|\Pi(\mathcal{G}^b, \mathbf{p}) - \mathbf{p}\|^2),$$

where $\Pi(\mathcal{G}^b, \mathbf{p})$ is the projection of point $\mathbf{p}$ onto the nearest surface point on the mesh $\mathcal{G}^b$.

$E_{stretch}$ penalizes changes in length of edges in the generic mesh after transformation, and is defined as

$$E_{stretch}(\mathcal{G}^b) = \sum_{e \in \mathcal{E}_{\mathcal{G}^b}} \frac{1}{2} k |l_e - r_e|^2,$$

where k is a spring constant, l_e is the current length, and r_e the rest length of edge e. Including this term prohibits large changes in the position and orientation of the feature, and thus keeps the optimization from converging to a solution too far from the initial configuration.

To find an affine transformation that minimizes the non-linear function E_{local}, we employ Powell's algorithm, as in Section 4.1. Since the optimization is performed only locally over the mesh region $\mathcal{G}^b$, it can be carried out quickly.

After the energy minimization procedure has converged, we apply the transformation found for each bounding box to the contained geometry and perform a gradual blend with the surrounding area, similar to the method used in [1]. For blending, we use a set $L = \{\mathbf{l}_1, \ldots, \mathbf{l}_n\}$ of n landmarks defined on the undeformed generic mesh, which are contained in the above mentioned boxes. In practice, we use three landmarks for each ear, and four for the nose and the mouth, see Figure 4. The transformation found for each box is also applied to the contained landmarks, resulting in the transformed landmark set $\tilde{L} = \{\tilde{\mathbf{l}}_1, \ldots, \tilde{\mathbf{l}}_n\}$. The displacement vectors for all the landmarks are used to update the subset $\overline{\mathcal{V}^b} := \mathcal{V}_\mathcal{G} \setminus \{\mathcal{V}_{\mathcal{G}^b}\}_{b \in \mathcal{B}}$ of vertices of $\mathcal{V}_\mathcal{G}$ that are not contained in any box. The displacements are weighted by an exponential fall-off function according to the distance between landmark and mesh vertex:

$$\forall v \in \overline{\mathcal{V}^b} : \mathbf{v} \leftarrow \mathbf{v} + \textstyle\sum_{i=1}^{n} \exp(-\frac{1}{k}\|\mathbf{v} - \mathbf{l}_i\|)(\tilde{\mathbf{l}}_i - \mathbf{l}_i)$$

We initialize the constant value k to 1/30 of the diagonal length of the bounding box of the given point set. This parameter controls the size of the region influenced by the blending.

4.3 Global Control Mesh Fitting

Given the generic mesh $\mathcal{G}$ with locally aligned facial features, we can now perform straightforward global optimization by iteratively minimizing the distance from the points of the sample data set to the mesh surface. We perform least squares minimization of the energy functional

$$E_{global} = E_{dist} + \lambda E_{smooth} + \mu E_{binding} + \nu E_{constraint}.$$

Optimization stops, when the difference of the previous and current E_{global} drops below a user-specified error threshold. E_{dist} is just the same functional as used in Section 4.2, but this time applied to all vertices in $\mathcal{G}$. The user-specified weights λ, μ, and ν balance the additional terms against E_{dist}. To enforce local flatness of the mesh, E_{smooth} measures the deviation of mesh vertices to the centroid of their respective one-neighborhood:

$$E_{smooth}(\mathcal{G}) = \sum_{v \in \mathcal{V}_\mathcal{G}} \left\| \frac{\sum_{w \in star(v)} \mathbf{w}}{valence(v)} - \mathbf{v} \right\|^2,$$

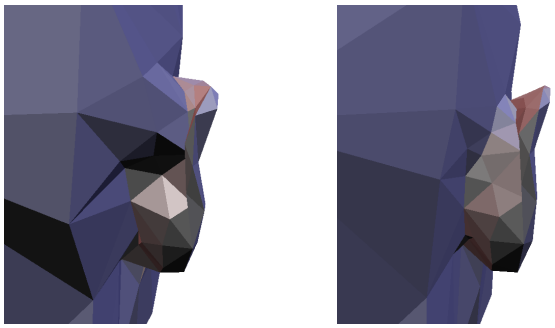

Figure 3: Improving definition of facial features in the fitted control mesh with imperfect scan data. Left: without using binding energy, the upper area behind the ear is flattened due to lack of data in the input sample set. Right: by use of binding edges the shape of the ear is improved.

Because the numerical value of E_{dist} increases with the local density of the points in the scan data set, E_{smooth} has only a comparatively small influence in the regions where data is present. In regions of the scan data set where there is no data, the smoothing term leads to a shrinking effect, since here the distance minimization does not act as a counter-force. We thus introduce additional constraint terms $E_{binding}$ and $E_{constraint}$ into the energy function.

$E_{binding}$ is used to minimize the length of a set of edges, which we call *binding edges*. This constraint helps to keep features in the face from flattening out due to lack of data, as is frequently the case behind the ears: our triangulation scanner cannot measure data in these regions due to shadowing between light source and projector. Since there are no data samples that the surface could approximate, the smoothing term eventually removes concavities. If we define binding edges in these regions, the flattening of that area is effectively prevented:

$$E_{binding}(\mathcal{G}) = \sum_{e \in \mathcal{E}^b} l_e^2,$$

where $\mathcal{E}^b$ is the set of binding edges and l_e is the length of edge e. These edges are for the most part defined by a subset of edges from $\mathcal{E}_\mathcal{G}$. Additionally, some vertices of $\mathcal{G}$ that are not actually connected by edges in $\mathcal{E}_\mathcal{G}$ are bound together in $\mathcal{E}^b$ (see Figure 4). These additional edges are used to keep corresponding vertices of the upper and lower lips together: unconstrained, the shrinking effect of the smoothing term leads to the introduction of a gap between the lips, which should remain closed as in the generic model $\mathcal{G}$. Figure 3 demonstrates the effect of the binding energy term.

Finally, we constrain vertices to their original positions by $E_{constraint}$. This constraint helps to keep the shape of

the inner part of the lips of the mesh model, which would otherwise be flattened onto the outside in the process of minimizing the distance to the point cloud (see Figure 4).

$$E_{constraint}(\mathcal{G}) = \sum_{v \in \mathcal{V}_{\mathcal{G}}^c} \|\mathbf{v} - \tilde{\mathbf{v}}\|^2,$$

where $\mathcal{V}_{\mathcal{G}}^c$ is the set of constrained vertices defined on $\mathcal{G}$, and $\tilde{\mathbf{v}}$ denotes the original position of vertex v before the optimization.

As described in [17], a sparse linear system can be built expressing the function E_{global}, which can be solved using the conjugate gradient method [19]. To be able to set up a linear system, E_{dist} has to be linearized, which can be done as shown in [5, 17].

5 Generation of the Multiresolution Model

We employ the Modified Butterfly subdivision scheme [25] to construct the subdivision hierarchy on top of the base mesh resulting from the fitting process described in the previous section. This scheme guarantees C^1 continuity everywhere and has the advantage of interpolating the vertices of the previous refinement levels, so that coarser levels of refinement serve as an approximation to the head geometry, which is not necessarily the case with approximating subdivision schemes [17].

After fitting, the surface of the deformed generic mesh approximates the point cloud in a least squares sense, i.e. the distance between the points and the surface is minimized. The control mesh vertices are *not* necessarily lying on the point cloud. Before refining the mesh using the Butterfly subdivision rules, we measure the distance from each vertex along normal direction to the nearest point on the point cloud and store this value as an additional displacement. The displacements are then applied to the vertices and the updated mesh is refined. On each level of the resulting hierarchy, we thus obtain a triangle mesh with vertices interpolating the original point cloud after the respective displacements have been applied. In areas of the point cloud with no data we cannot measure displacements, but the subdivision operator generates a smooth surface in these regions. This construction technique is similar to *normal meshes* [4], but we sample displacements to a set of sample points instead of to another mesh. By storing only one scalar displacement value per vertex introduced on each level, we achieve a storage-efficient hierarchical mesh structure [4, 12].

6 Animating the Surface

During animation, the control mesh vertices of the subdivision surface are displaced via simulated muscle contraction [9]. Since the subdivision hierarchy is built using an interpolating scheme, each refinement level including the control mesh can serve as an approximation to the limit surface for rendering. Depending on the available computation time per frame and quality requirements, an appropriate refinement level can be picked for display. Since the mesh topology is completely determined by the topology of the base mesh and the subdivision scheme, the area of change to the refined mesh induced by movement of a control mesh vertex is known a priori and is determined by the support of that vertex [8, 26]. In the Butterfly scheme, the support includes at most the three-neighborhood of a control mesh vertex. When a control mesh vertex is animated, we apply the subdivision rules and displacements only to this area. Thus, we can achieve fast updates of the refined geometry, without having to include the detailed geometry into the simulation task of the physics-based animation engine. Figure 5 shows snapshots from an animation of the head geometry.

7 Results

We have applied our surface generation method to scans of male and female individuals, see figures 1 and 6. With our structured light scanner, the scans show defects in large regions due to shadowing effects and bad reflective properties of the surface, especially on hair. Nonetheless, our algorithm generates subdivision models consistent with the range data, complementing it in a plausible manner in areas with no data. User intervention was not necessary, the only initial requirement being that both scan data and generic head model are looking roughly down the same axis in world coordinates.

The silhouette-based geometry fitting approach converges quite quickly in 8–14 iterations of Powell's optimization method. However, each iteration performs about 250–300 evaluations of the "function" to be minimized, i.e. counting the number of white pixels using `glGetHistogram` calls. For each of these calls, we have to read back and draw the framebuffer using `glCopyPixels` to retrieve the results of the histogram. Unfortunately, such framebuffer read-backs are still quite expensive on current PC graphics boards: using a 256×256 framebuffer, the optimization process takes about 6–10 min on a 1.7 GHz PC with a GeForce3 graphics board while completing within 90–150 sec on an `sgi` Octane with a 300 MHz R12k processor. Increasing the resolution of the framebuffer to 512×512 results in a slightly better alignment and a (relative) convergence speed-up of 1–2 iterations. Due to the larger amount of data that is read from and drawn into the framebuffer, the whole process takes about 20–30 min on the PC and 5–8 min on the Octane in this case.

In the current implementation, we have to reduce the input complexity for the local and global fitting steps to

handle large data sets. On the PC, with a dataset of 50k points subsampled from the initial 300k points, the local fitting process runs for approx. 5 minutes, performing about 35 iterations per bounding box. The global fitting process performs 100 iterations in approx. 40 minutes until convergence. The multiresolution mesh is then built using the complete set of point samples with no subsampling. Even though the processing takes up considerable time even on a fast PC, this does not impair usability to a large extent due to the full automation. Not counting times for the simulation and rendering parts in our facial animation system, updating the animated mesh on the second refinement level requires about 40–50ms on the PC, corresponding to a rate of 20–25 fps.

For the models shown in Figure 6, the following table shows the development of the mean and maximum distances from the point cloud to the current control mesh after each of the three steps of the algorithm. The last column shows the distance to the final subdivision surface with displacements applied. The values are normalized to "percent of the point cloud bounding box diameter". The maximum distance to the final surface reflects the "noisiness" of the input data: outliers in overlapping regions of the range scans are not interpolated.

	initial alignment	local fitting	global fitting	subdiv. surface
male / mean	1.51	1.42	0.11	0.04
male / max.	13.13	13.13	1.28	1.30
female / mean	1.09	0.91	0.12	0.08
female / max.	11.35	10.26	1.55	1.55

8 Future Work

Apart from geometric similarities, we would like to employ texture information to improve the accuracy of feature matching. Also, instead of simply generating a smooth surface in areas with no data, it would be interesting to generate artificial surface detail to make the surface match the surrounding areas. For the animation system, we are planning the automatic insertion of separate components representing eyes, teeth, and tongue.

References

[1] T. Akimoto, Y. Suenaga, and R. S. Wallace. Automatic creation of 3d facial models. *IEEE Computer Graphics & Applications*, 13(5):16–22, September 1993.

[2] V. Blanz and T. Vetter. A Morphable Model for the Synthesis of 3D Faces. In *Computer Graphics (SIGGRAPH '99 Conf. Proc.)*, pages 187–194, 1999.

[3] T. DeRose, M. Kass, and T. Truong. Subdivision Surfaces in Character Animation. In *Computer Graphics (SIGGRAPH '98 Conf. Proc.)*, pages 85–94, 1998.

[4] I. Guskov, K. Vidimce, W. Sweldens, and P. Schröder. Normal Meshes. In *Computer Graphics (SIGGRAPH '00 Conf. Proc.)*, pages 95–102, 2000.

[5] H. Hoppe, T. DeRose, T. Duchamp, M. Halstead, H. Jin, J. McDonald, J. Schweitzer, and W. Stuetzle. Piecewise Smooth Surface Reconstruction. In *Computer Graphics (SIGGRAPH '94 Conf. Proc.)*, pages 295–302, 1994.

[6] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Mesh Optimization. In *Computer Graphics (SIGGRAPH '93 Conf. Proc.)*, pages 19–26, 1993.

[7] D. Hutchinson, M. Preston, and T. Hewitt. Adaptive Refinement for Mass-Spring Simulation. In *7th EG Workshop on Animation and Simulation*, pages 31–45, 1996.

[8] I. P. Ivrissimtzis and M.A. Sabin. On the support of recursive subdivision. *submitted for publication*, 2001.

[9] K. Kähler, J. Haber, and H.-P. Seidel. Geometry-based muscle modeling for facial animation. In *Graphics Interface2001 Conf. Proc.*, pages 37 – 46, 2001.

[10] L. Kobbelt. $\sqrt{3}$-Subdivision. In *Computer Graphics (SIGGRAPH '00 Conf. Proc.)*, pages 103–112, 2000.

[11] L. Kobbelt, S. Campagna, J. Vorsatz, and H.-P. Seidel. Interactive Multi-Resolution Modeling on Arbitrary Meshes. In *Computer Graphics (SIGGRAPH '98 Conf. Proc.)*, pages 105–114, 1998.

[12] A. Lee, H. Moreton, and H. Hoppe. Displaced Subdivision Surfaces. In *Computer Graphics (SIGGRAPH '00 Conf. Proc.)*, pages 85–94, 2000.

[13] Y. Lee, D. Terzopoulos, and K. Waters. Realistic Modeling for Facial Animations. In *Computer Graphics (SIGGRAPH '95 Conf. Proc.)*, pages 55–62, 1995.

[14] H. P. A. Lensch, W. Heidrich, and H.-P. Seidel. Automated Texture Registration and Stitching for Real World Models. In *Proc. Pacific Graphics 2000*, pages 317–326, 2000.

[15] H. P. A. Lensch, W. Heidrich, and H.-P. Seidel. A Silhouette-based Algorithm for Texture Registration and Stitching. *Graphical Models*, 2001. in print.

[16] C. T. Loop. Smooth Subdivision Surfaces Based on Triangles. Master's thesis, University of Utah, Department of Mathematics, 1987.

[17] S. Marschner, B. Guenter, and S. Raghupathy. Modeling and Rendering for Realistic Facial Animation. In *Rendering Techniques 2000 (Proc. 11th EG Workshop on Rendering)*, pages 231–242, 2000.

[18] F. Pighin, J. Hecker, D. Lischinski, R. Szeliski, and D. H. Salesin. Synthesizing Realistic Facial Expressions from Photographs. In *Computer Graphics (SIGGRAPH '98 Conf. Proc.)*, pages 75–84, 1998.

[19] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, Cambridge, MA, 2nd edition, 1992.

[20] T. W. Sederberg and S. R. Parry. Free-Form Deformation of Solid Geometric Models. In *Computer Graphics (SIGGRAPH '86 Conf. Proc.)*, pages 151–160, 1986.

[21] R. Szeliski and S. Lavallée. Matching 3-D Anatomical Surfaces with Non-Rigid Deformations using Octree-Splines. *International Journal of Computer Vision*, 18(2):171–186, 1996.

[22] A. Van Gelder. Approximate Simulation of Elastic Membranes by Triangulated Spring Meshes. *Journal of Graphics Tools*, 3(2):21–41, 1998.

[23] P. Volino and N. Magnenat-Thalmann. The SPHERIGON: A Simple Polygon Patch for Smoothing Quickly your Polygonal Meshes. In *Proc. Computer Animation '98*, pages 72–79, 1998.

[24] D. Zorin and P. Schröder. Subdivision for Modeling and Animation. *Computer Graphics (SIGGRAPH '00 Course Notes)*, 2000.

[25] D. Zorin, P. Schröder, and W. Sweldens. Interpolating subdivision for meshes with arbitrary topology. In *Computer Graphics (SIGGRAPH '96 Conf. Proc.)*, pages 189–192, 1996.

[26] D. Zorin, P. Schröder, and W. Sweldens. Interactive Multiresolution Mesh Editing. In *Computer Graphics (SIGGRAPH '97 Conf. Proc.)*, pages 259–268, 1997.

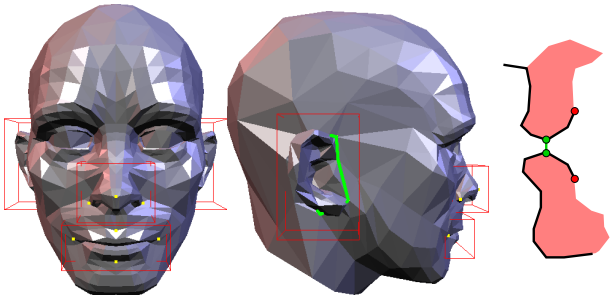

Figure 4: Extra information stored with the generic head model: Left and center: four bounding boxes (red) around ears, nose and mouth; edges to which binding energy constraint is applied (green lines); landmarks on the boxed features (yellow dots). Right: cross-section of the mouth region. The binding energy term is applied to virtual edges connecting upper and lower lip vertices (green line and dots). The inner part of the lips is kept in shape by constraining vertices to their positions (red dots).

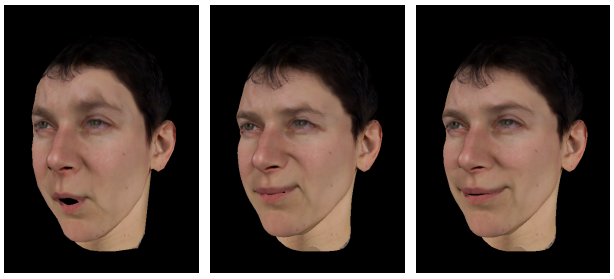

Figure 5: Displaced subdivision surface head model showing different expressions.

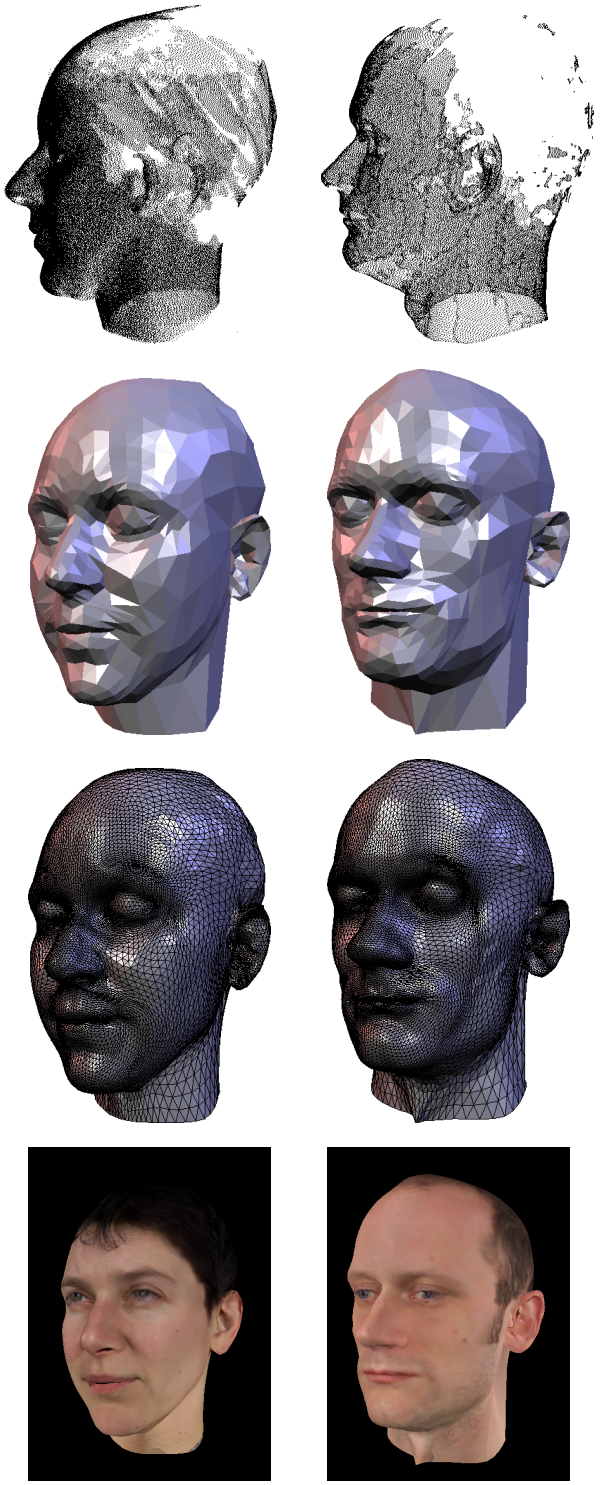

Figure 6: Two more point sample sets and the head models generated from them, shown at 2 levels of refinement of the base mesh. Left: female scanned with a bathing cap to enable our scanner to capture data on the back of the head. Right: scan of a male, exhibiting lack of sample data in the hair-covered region.