# Articulated Body Deformation from Range Scan Data

Brett Allen Brian Curless Zoran Popović

University of Washington

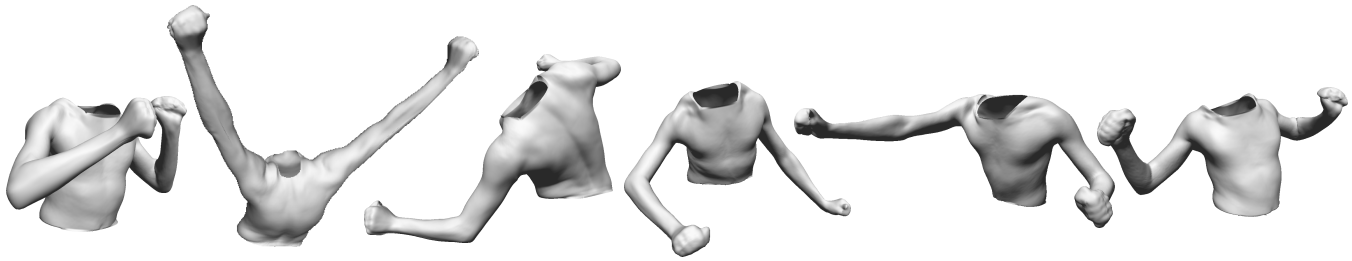

**Figure 1** Each of these 3D meshes are made from a skeletally driven subdivision surface. The displacements for the subdivision surface are interpolated from range-scan examples of the arm, shoulder, and torso in various poses. The joint angles for each pose are drawn from optical motion capture data.

## Abstract

This paper presents an example-based method for calculating skeleton-driven body deformations. Our example data consists of range scans of a human body in a variety of poses. Using markers captured during range scanning, we construct a kinematic skeleton and identify the pose of each scan. We then construct a mutually consistent parameterization of all the scans using a posable subdivision surface template. The detail deformations are represented as displacements from this surface, and holes are filled smoothly within the displacement maps. Finally, we combine the range scans using $k$-nearest neighbor interpolation in pose space. We demonstrate results for a human upper body with controllable pose, kinematics, and underlying surface shape.

**CR Categories:** I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Curve, surface, solid and object modeling; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation;

**Keywords:** animation, character animation, deformation, human body simulation, synthetic actor

## 1 Introduction

Creating realistic, virtual actors remains one of the grand challenges in computer graphics. Convincingly modeling human shape, motion, and appearance is difficult, because we are accustomed to seeing other humans and are quick to detect flaws. One possible avenue to realism is through direct observation and measurement of people. Motion capture, for instance, has become a standard method for obtaining detailed samples of skeletal motion which can themselves be edited plausibly, and image-based techniques show promise for accurately modeling the appearance of skin. In this paper, we explore a data-driven approach to modeling the shape of the human body in arbitrary poses.

email: {allen,curless,zoran}@cs.washington.edu

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ACM SIGGRAPH 2002, 21-26 July 2002, San Antonio, TX, USA
© 2002 ACM

Recent years have witnessed the evolution of numerous range scanning technologies, including whole-body scanners that can capture the static shape of a person quite accurately. Given such a static scan, an animator can warp the body into a different pose, but this approach ignores an important aspect of human movement: muscles, bones, and other anatomical structures continuously shift and change the shape of the body. Clearly, to create compelling animations by observation we need more than just a single scan. Scanning the subject in every pose needed for every frame of an animation is impractical; instead, we propose a system in which body parts are scanned in a set of key poses, and then animations are generated by smoothly interpolating among these poses using scattered data interpolation techniques.

The concept of interpolating sampled poses is not a new idea. What makes our approach unique is the use of real-world data to create a fully posable 3D model. In the process, we face several challenges. First, in order to establish a domain for interpolation, we must discover the pose of each scan. Second, interpolation techniques require a one-to-one correspondence between points on the scanned surfaces, but the scanned data consists of unstructured meshes with no such correspondence. This problem is particularly challenging because the scans are in different poses, so standard rigid-body registration techniques will not work. Third, range scans are frequently incomplete because of occlusions and grazing angle views. Thus, we are faced with the challenge of filling holes in the range data. Finally, due to the combinatorics of the problem, we cannot capture a human body in *every* possible pose. Thus, we must blend between independently posed scans.

In this paper, we present a general framework that addresses each of these problems. Using markers placed on the subject during range scanning, we reconstruct the pose of each scan. We then create a hole-filled, parameterized reconstruction at each pose using displacement-mapped subdivision surfaces. Lastly, we create shapes in new poses using scattered data interpolation and spatially varying surface blending.

On the way to achieving our goal we make contributions to the problems of fitting a skeleton to marker data, surface correlation for articulated objects, fair hole filling of surfaces, example-based interpolation with quaternion parameters, and blending range scans. Our primary contribution, however, is the process itself and the demonstration that we can derive realistic, posable human body deformations from range scan data.

### 1.1 Related work

The two main approaches to modeling body deformations are anatomical modeling and example-based approaches. The idea behind anatomical modeling is to use an accurate representation of the major bones, muscles, and other interior structures of the body. These structures are deformed as necessary when the body moves, and a skin simulation is wrapped around the underlying anatomy to obtain the final geometry. There is a large body of work on anatomically based approaches, including Wilhelms and Gelder [1997], Scheepers et al. [1997], Victor Ng-Thow-Hing [1999], and Aubel and Thalmann [2001].

The primary strength of anatomical approaches is the ability to simulate dynamics and complex collisions. The main drawback is their computational expense, since one must perform a physical simulation to generate every frame, while taking care to conserve muscle volumes, and stretch the skin correctly.

An alternative paradigm is the example-based approach, where an artist generates a model of some body part in several different poses with the same underlying mesh structure. These poses are correlated to various degrees of freedom, such as joint angles. An animator can then supply new values for the degrees of freedom and the examples are interpolated appropriately. Example-based approaches are much faster computationally, and creating examples is often easier than creating a detailed and accurate anatomical model.

Lewis et al. [2000] and Sloan et al. [2001] describe similar techniques for applying example-based approaches to meshes. Both techniques use radial basis functions to supply the interpolation weights for each example, and, for shape interpolation, both require hand-sculpted meshes that ensure a one-to-one vertex correspondence exists between each pair of examples. This paper will also use an example-based approach, but the key difference is that we will start with uncorrelated range-scan data. In fact, with our method, even the poses of the examples will be derived from the data.

Other example-based approaches use scanned or photographed data. In the domain of facial animation, example-based techniques have been developed by Pighin et al. [1998], Guenter et al. [1998], and Blanz and Vetter [1999]. One of the few attempts to create articulated deformations from scanned examples is the work of Talbot [1998], who created a partial arm model with one degree of freedom. Our work takes a broader scope and can be applied to complex articulated figures.

### 1.2 Problem formulation

We formulate the problem of creating a posable human body as a scattered data interpolation problem in which shape examples are blended linearly to create new shapes. Thus, we must define a domain over which samples are taken and represent the samples in a form suitable for blending. The domain consists of all of the knobs that an animator will be able to tweak, such as controls for joint angles, muscle loads, body types, and so on. In our example upper-body model, the domain will consist entirely of joint angles, but in principle any kind of parameters could be used. Throughout this paper we will refer to a vector in the joint space as $\mathbf{q}$.

Having established the interpolation domain, we next need to select and enforce a representation suitable for blending between the example shapes. For unstructured range scans, this amounts to constructing a correspondence between surface points on different scans, i.e. , a mutually consistent parameterization. To this end, we will employ *displaced subdivision surfaces*, as introduced by Lee et al. [2000]. Displaced subdivision surfaces consist of a template subdivision surface, $T$, and a displacement map $d$ that describes the final surface $S$ by displacing the template along the normal, $\hat{\mathbf{n}}$, to the template surface. This representation is a kind of layered model [Chadwick et al. 1989], where the the local detail deformations are separated from the large scale (mostly affine) transformations applied to each body part. We will drive the underlying template surface using the pose, $\mathbf{q}$, resulting in a pose-varying surface:

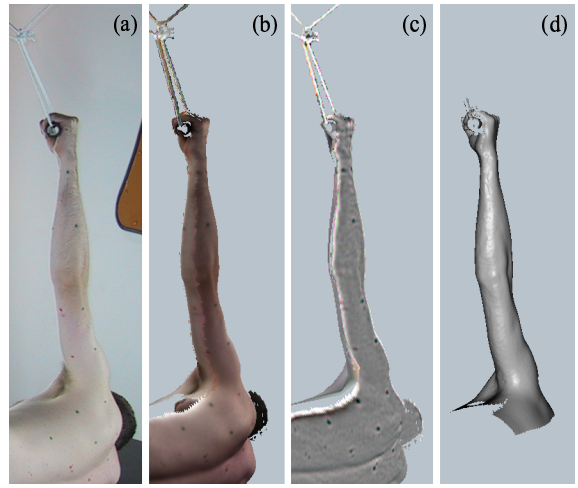


**Figure 2** (a) Photograph of the subject in the scanner. The left arm is about to be scanned. The ropes help the subject remain motionless during the scan. (b) The scanned surface with color data, rendered emissively. Note that surfaces parallel to the scanner's rays, such as the side of the torso, are not captured. The four meshes that were captured simultaneously have been registered. (c) Scanned surfaces after applying dot-enhancing filter to the color data. (d) Combined and clipped arm scan, rendered with Gouraud shading.

$$S(\mathbf{u}, \mathbf{q}) = T(\mathbf{u}, \mathbf{q}) + d(\mathbf{u}, \mathbf{q})\hat{\mathbf{n}}(\mathbf{u}, \mathbf{q}) \tag{1}$$

Notice that $d$ is also a function of the pose, $\mathbf{q}$. Unlike standard displaced subdivision surfaces, our displacements are based on multiple example shapes, allowing scattered data interpolation techniques to be applied. In particular, the interpolated displacements are a weighted sum of the example displacements:

$$d(\mathbf{u}, \mathbf{q}) = \sum_{i=1}^{n} w_i(\mathbf{u}, \mathbf{q}) d_i(\mathbf{u}) \tag{2}$$

where $n$ is the number of examples, $d_i(\mathbf{u})$ is the displacement map for the $i^{th}$ example, $w_i(\mathbf{u}, \mathbf{q})$ is the scattered data interpolation weighting function for the $i^{th}$ example, and $d(\mathbf{u}, \mathbf{q})$ is the interpolated displacement map for pose $\mathbf{q}$.

In the remainder of the paper, we describe the steps taken to construct an example-based posable human body:

1. Capture a set of example scans with markers (Section 2).
2. Using the markers, solve for the global kinematics of the body, $\mathbf{k}$, and the local pose of each scan, $\mathbf{q}_i$ (Section 3).
3. Create a template surface, $T(\mathbf{u}, \mathbf{q})$, based on the kinematics of the body, parameterize and resample the examples into displacement maps $d_i(\mathbf{u})$, and fill in any missing values (Section 4).
4. Compute the interpolation weights, $w_i(\mathbf{u}, \mathbf{q})$ (Section 5).

We demonstrate results using an upper body model in Section 6 and discuss conclusions and future work in Section 7.

## 2 Data acquisition

This section explains how we acquired our example data set. The overall idea is to sample the body's shape in a variety of poses covering the full range of motion for each joint. At the same time, we capture the location of markers on the body that we will use to determine the pose of each scan.

| **Left arm data set** (36 scans) | |
|---|---|
| Elbow bend | 0°, 60°, 90°, 130° |
| Elbow twist | 0°, 60°, 130° |
| Wrist flexion | −45°, 0°, 30° |
| **Left shoulder data set** (33 scans) | |
| Shoulder and clavicle | flexion, neutral, extension |
| | abduction, neutral, adduction |
| | medial rotation, neutral, lateral rotation |
| | shoulder girdle elevation (shrug), depression, protraction (forwards), retraction (backwards) |
| **Torso data set** (27 scans) | |
| Waist and abdomen | pronation, neutral, supination (twist) |
| | left and right lateral flexion, neutral |
| | left and right rotation, neutral |

**Table 1** We captured three data sets, each of which covered the range of motion of a group of joints, shown in the left column. The joint angles that we sampled are described in the right column. For an explanation of the terminology, the reader may refer to any reference on biomechanics or kinesiology, such as Gowitzke and Milner [1988].

### 2.1 Range scanning

We acquired our surface data using a Cyberware WB4 whole-body range scanner. This scanner captures range scans and color data from four directions simultaneously and has a sampling pitch of 5 mm horizontally and 2 mm vertically. Figure 2(a) shows the subject in the scanner. Overhead ropes helped the subject remain motionless during the seventeen seconds of scanning time. The scanned surface with color data is shown in Figure 2(b). The same mesh after merging the four scans [Curless and Levoy 1996] and clipping out the arm is shown in Figure 2(d).

### 2.2 Pose coverage

To create an upper body model, we needed to sample all poses of the wrist, elbow, shoulder, and torso. Due to the combinatorial nature of the problem, we split the upper body into three data sets captured separately: the arm, the shoulder, and the torso. We can split the body up because the joints on each part have little influence on the shape of distant body parts. At the interface between adjacent body parts, we must overlap the capture regions and blend them at a later stage. We also save work by capturing only the left arm and left shoulder and later mirroring the data to the right side.

Table 1 gives a summary of all captured poses. In the interest of taking as few scans as possible we made our sampling space fairly sparse. We sampled at least three angles for each degree of freedom, giving us a neutral middle value and the two extremes that generally have the most dramatic shape changes.

### 2.3 Markers

To enable precise determination of each scan, we placed colored markers on the subject using costume make-up. The markers are picked up by the scanner's color video cameras and mapped onto each range image. We used eight different marker colors to aid the identification process. Note that reflectance discontinuities when picked up by a range sensor can lead to geometric errors [Curless and Levoy 1995]. Our range data does not suffer from these artifacts, because the whole-body scanner uses an infrared laser and does not distinguish between skin and make-up colors.

For the arm and shoulder data sets, we used forty-eight markers, and for the torso data set we used ninety markers. Our goal was to have at least three markers visible per body part (the minimum number of markers capable of establishing a coordinate frame), and since the markers were often occluded or hard to identify we placed roughly four times that many.

Estimating poses from the marker imagery requires two preprocessing steps: locating the 3D coordinates of each marker and labeling and identifying the markers across all scans. To automate the process of locating the markers, we applied a broad Laplacian convolution filter to the color data of each range scan. This filter makes the dots stand out from the skin, as shown in Figure 2(c), so they can easily be identified by searching for extreme color values. Our marker-finding algorithm groups neighboring pixels of similar color and rejects clusters that are too large, too small, or too near the edge. By referring back to the range values, we find the 3D location of each pixel in the cluster and take the centroid.

The second step of marker-finding is to label the markers. Each marker that was placed on the subject is assigned a numerical index. We then determine the index of each marker located in the marker-finding step. We applied the graph-matching technique of Gold and Rangarajan [1996] based on matching geometric relationships and marker color; unfortunately we found this approach unsuitable due to the large number of missing markers. Consequently we label the markers manually after running our automatic location-finding technique. We hope to automate this step in future work.

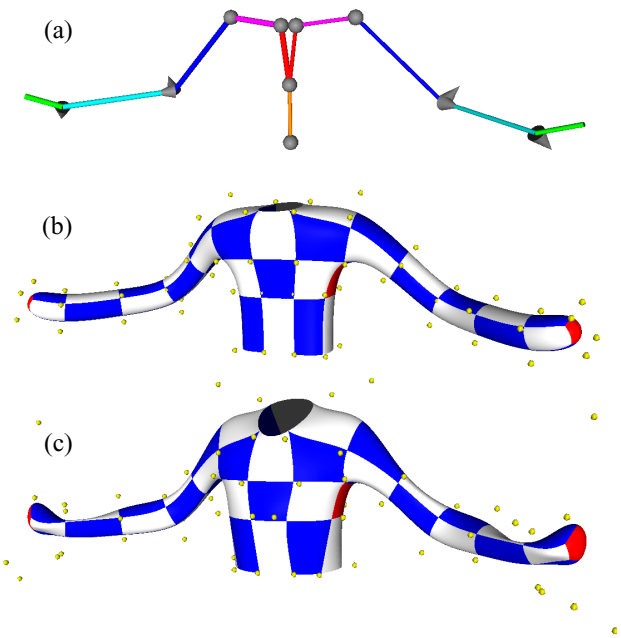


**Figure 3** (a) Our upper body skeleton after optimization. The large spheres are quaternion joints, and the cones are single-axis joints. (b) The control points for this skeleton, and the corresponding subdivision surface. The checkerboard pattern delineates the subdivision patches. (c) The control points and subdivision surface after refitting.

## 3 Determining kinematics and pose

We can think of each scan as an example of the body's shape in one particular pose. Therefore, we need to know the exact pose, $\mathbf{q}_i$, of each scan. We also need to know the kinematics, $\mathbf{k}$, of the body's skeleton, that is, the fixed transformations between each joint. This section describes our method for automatically determining the poses and kinematics of the scanned bodies.

### 3.1 Skeleton

We construct a skeleton containing the joints that the end user of our system will be able to animate. The goal is to have a skeleton that is a good approximation of true human kinematics, but not too complicated to solve for or animate. This tradeoff exposes some important design issues. For example, the human shoulder joint consists of four joints: one between the sternum and clavicle, one between

the clavicle and scapula, one between the scapula and humerus, and one between the scapula and the rib cage [Luttgens and Wells 1982]. However, the second and third joints are very close together, and the fourth joint has very little independent movement. Thus, we simplify the shoulder complex to two joints: a clavicle joint and a shoulder joint. The human spine is much more complicated, consisting of seventeen joints each with its own range of motion. We reduce the spine to just two joints, one at the waist and one at the abdomen. Another example is the elbow, which consists of two single-axis joints. Animators typically make the assumption that the axes of these joints are perpendicular and colocated. This is not in keeping with the actual bone structure of the human arm, and so our choice of skeleton allows the axes to have any relative orientation and an arbitrary translation between them. (We prefer a small translation, to prevent the bones from moving along the axes of rotation during the optimization.)

The skeleton hierarchy is rooted with a *base transformation* which moves from the origin of world coordinates to the coordinate frame of the hips. After the base transformation, each rotation joint in our skeleton is followed by a translation to the next joint. We will call these translation components the *bone translations*. Our upper-body skeleton (after optimization) is show in Figure 3(a).

### 3.2 Local marker positions

The *local marker positions*, $\mathbf{m}$, are a collection of 3D points describing the position of each marker within its joint's coordinate frame. We initially assign each marker to a joint coordinate frame based on its location on the body. For example, markers on the lower back are placed in the waist joint's coordinate frame, and markers on the upper back are placed in the abdomen joint's coordinate frame. The markers will be treated as if they moved rigidly with the skeleton. This assumption is not entirely accurate because of the body deformations that move the marker in non-rigid ways. However, we have obtained satisfactory results by using many markers and taking a least squares approach.

Even though the local marker positions will not be used at all in our deformation-building process, it is necessary to calculate them when solving for the poses and kinematics.

### 3.3 Optimization

A summary of all of the skeleton parameters is shown in Table 2. The goal of the optimization step is to determine the values of all of these parameters that best match the marker data.

Note that we can generate arbitrarily many versions of the same skeleton by applying a constant rotation to one joint in all frames, and then adjusting the bone translations and local marker positions to compensate. As a result, our skeleton parameterization is under-determined. For example, we could call the elbow angle of a straight arm $0^\circ$ or $180^\circ$ or any other angle, and all other arm poses will be measured relative to this. To eliminate this extra degree of freedom, we must lock all of the rotation joints in one of the scans to fixed values (such as zero) in order to provide a frame of reference to which the rotations will be compared. We call this special scan the *reference scan*.

Defining a reference scan offers an additional advantage: it provides an initial guess for the local marker positions, $\mathbf{m}$. Since the joint angles are pre-determined for the reference scan, we need only supply a rough approximation for the base transformation. The local marker positions for all markers visible in that scan can be easily computed and later refined.

We also lock any degrees of freedom that cannot be determined from the given marker data. For example, since we scanned only the left arm, we cannot solve for the joint angles in the right arm. In addition, we lock the torso joints for all of the arm and shoulder scans, and the arm angles in all of the torso scans.

We can now optimize over all remaining degrees of freedom. The objective function minimizes the sum of the squares of the distances between the calculated marker positions and the observed marker positions:

| Name | # global DOFs | # per-scan DOFs |
|---|---|---|
| Base translation | 0 | 3 |
| Base rotation | 0 | 4 |
| Waist rotation | 0 | 4 |
| Waist translation | 3 | 0 |
| Abdomen rotation | 0 | 4 |
| Left/right abdomen translation | 3 | 0 |
| Left/right clavicle rotation | 0 | 8 |
| Left/right clavicle translation | 3 | 0 |
| Left/right shoulder rotation | 0 | 8 |
| Left/right upper arm translation | 3 | 0 |
| Left/right elbow bend | 2 | 2 |
| Left/right elbow translation | 3 | 0 |
| Left/right elbow twist | 2 | 2 |
| Left/right lower arm translation | 3 | 0 |
| Left/right wrist bend | 2 | 2 |
| Left/right hand translation | 0 | 0 |
| Local marker positions | 411 | 0 |

**Table 2** Degrees of freedom (DOFs) of the skeleton. Global DOFs are constant across all scans; per-scan DOFs take on a different value for each scan. The left/right translations are mirror images of each other and thus share DOFs. The single-axis rotations in the arm have two global DOFs indicating the direction of the axis and two per-scan DOFs for the angles about that axis on the left and right arm. The hand translation has no DOFs because there are no joints below the hand in our model. We will call the per-scan DOFs $\mathbf{q}_i$, and the local marker positions $\mathbf{m}$; the remaining global DOFs comprise the kinematics, $\mathbf{k}$.

$$\arg\min_{\mathbf{m},\mathbf{q},\mathbf{k}} \sum_{i=1}^{p} \sum_{j=1}^{m} \left\| \mathbf{o}_{ij} - \mathbf{c}(\mathbf{m}_j; \mathbf{q}_i, \mathbf{k}) \right\|^2 \tag{3}$$

where $p$ is the number of poses, $m$ is the number of markers, $\mathbf{o}_{ij}$ is the observed location of marker $j$ in scan $i$, and $\mathbf{c}(\mathbf{m}_j; \mathbf{q}_i, \mathbf{k})$ is the calculated position of the same marker. In cases where a marker cannot be located in a pose due to scanning limitations, we omit the corresponding term from the summation.

This skeleton-finding problem is identical to the problem of fitting a skeleton to optical motion capture data. Silaghi et al. [1998] and Herda et al. [2001] have investigate this problem and describe a local (joint-by-joint) optimization technique for initializing the global optimization stated above. An initialization is necessary because the search space contains many local minima. However, we can avoid this extra step of running a local optimization using two improvements.

First, because we calculated initial values for the local marker positions using the reference scan, we can start our global solver with these positions locked. The solver usually reaches a bad local minimum because it moved the local marker positions to unreasonable locations and compensated with erroneous poses and bone lengths. By locking the local marker positions, we guide the solver toward finding reasonable poses first. After this optimization converges, we run it again with the local marker positions unlocked to get the best fit.

The second technique we use to aid convergence is scaling the degrees of freedom (DOFs). By scaling the DOFs, we ensure that all of their gradients have the same magnitude, improving solver performance [Gill et al. 1989]. First of all, we must account for the fact that our set of DOFs contains three kinds of values: radians, meters, and quaternions. We scale each of these so that their values range from -1 to 1. We then further scale each DOF according to how many joints are influenced by it. Thus, per-scan DOFs have a scaling factor equal to the number of transforms below that DOF,

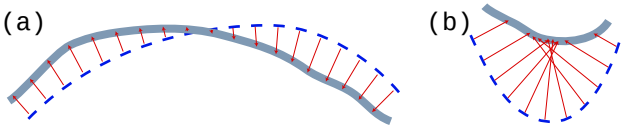


**Figure 4** (a) To construct a displaced subdivision surface, we cast rays (red arrows) perpendicular to the template subdivision surface (dashed blue line) to the nearest scanned surface (thick gray line). Because the direction of each ray is determined by the subdivision surface, we need only record the distance. (b) If the template surface is too curved and the scanned surface is too far away, then the rays can cross, causing the parameterization to fold over on itself. This can be avoided by ensuring that the template surface is close to the scanned surface.

and global DOFs are weighted by the number of transforms they influence multiplied by the number of scans.

We use L-BFGS-B, a quasi-Newtonian solver to optimize the goal function [Zhu et al. 1997]. We analytically compute the derivatives of Equation 3 relative to each degree of freedom. The running time for convergence is approximately forty minutes on a 1.5 GHz Pentium 4. This optimization only has to be run once because it incorporates all of the scanned poses for all body parts.

## 4 Determining deformations

At this point, we have determined the joint angles and the bone locations for each scan. The next step is to represent the deformations that each body part undergoes in each pose. The key issue here is one of correspondence: if we choose a vertex in one scan, where is the corresponding vertex in the other scans?

### 4.1 Parameterization

To overcome this difficulty, we devise a parameterization that is based on the skeleton, since each scan has the same skeleton in a known pose. To do this, we need to choose a parameterization that can move with the skeleton. One possibility is a cylindrical cross-section based parameterization as used by Shen et al. [1994]. This parameterization works well for cylinder-like body parts such as the arm, but it is inconvenient to use for branching body parts, such as the torso.

A more general parameterization can be derived from displaced subdivision surfaces, as described by Lee et al. [2000]. Essentially, one creates a subdivision surface that approximates the real surface, and records the distance to the real surface along the normal by raycasting, as shown in Figure 4(a). We employ a Catmull-Clark subdivision surface [Catmull and Clark 1978] starting from a quadrilateral control mesh. We could have used other displacement-based approaches, such as displaced B-spline surfaces [Krishnamurthy and Levoy 1996]; we chose a subdivision surface template because of its ease in handling irregular vertices, i.e., control vertices with valence other four, which appear near the red patches in Figure 3(c). The work of Praun et al. [2001] could also provide consistent parameterizations across poses, though this approach operates on hole-free meshes and would require substantial modification to interpolate articulated body structures.

Lee et al. [2000] use a simplified version of the target mesh to define the control points. In our case, we want the control points to depend on the skeleton. We define coordinate frames on the skeleton based on joint coordinate frames. We then place rings of control points into these frames and a form a quadrilateral control mesh that follows the skeleton. To ensure that we have smooth transitions near the joints, the control point coordinate frames may be combinations of adjacent joint coordinate frames. For example, the coordinate frame centered at the abdomen joint has a rotation halfway between the lower spine's orientation and the upper spine's orientation. The resulting surface appears in Figure 3(b).

### 4.2 Fair hole filling

One of the critical problems with range scan data is that the meshes are generally incomplete. To interpolate the examples, however, we need complete information. One might think that the problem could be avoided by basing the pose space interpolation at each vertex on just the examples that do not contain a hole at that point. This approach has two problems. First, surface discontinuities will arise at the hole boundaries, because the adjacent points will be based on data drawn from different meshes. Secondly, the presence of holes is strongly correlated with the pose of the body, and so entire groups of poses will not have any data for a particular area. Thus, the missing data could only be drawn from poses that are quite different from the ones with holes.

Instead, we fill holes directly in each scan. Hole-filling in 3D can be quite tricky; we simplify the problem by operating directly on the displacement maps. We can easily detect the presence of holes within our parameterization when a displacement ray does not hit the surface. Our idea is to fill the holes by smoothly interpolating displacement values from neighboring vertices. Using the displacement parameterization we have made our 3D hole-filling problem analogous to the 2D problem of image inpainting, by considering the displacement values to be a grayscale image (on an unusual manifold).

Observing that our displacement "images" are typically very smooth and continuous, we fill in the missing area by minimizing curvature using a discrete thin-plate objective function. Since the points near the missing data are typically unreliable, we also apply the objective function near the edges of the holes, but with an additional term to keep those points close to their original value. Stated mathematically, we compute:

$$\arg\min_{d(\mathbf{u}_j)} \sum_{j=1}^{n} \kappa_j \left[d(\mathbf{u}_j) - \hat{d}(\mathbf{u}_j)\right]^2 + (1-\kappa_j)\left[\nabla^2 d(\mathbf{u}_j)\right]^2 \qquad (4)$$

where

$$\begin{aligned} n &= \text{the number of points to be filled or faired} \\ \mathbf{u}_j &= j^{th} \text{ point in the parameterization } \mathbf{u} \\ d(\mathbf{u}_j) &= \text{the new displacement at } \mathbf{u}_j \\ \hat{d}(\mathbf{u}_j) &= \text{the original displacement at } \mathbf{u}_j \\ \kappa_j &= 0 \text{ inside the hole, ramping toward 1} \\ &\quad \text{within three pixels of the hole} \end{aligned}$$

The results of this algorithm as applied to one of the scans are shown in Figure 5. The results are generally satisfactory; the most noticeable artifact is the absence of range sensor noise in the filled region.

### 4.3 Refitting

A significant problem with displaced subdivision surfaces occurs when the template surface is too far from the scanned surface and the curvature is too high. In this situation adjacent rays will cross and part of the surface will be covered several times. This problem is illustrated in Figure 4(b). The solution is to ensure that the subdivision surface is close enough to the scanned surface.

Our initial mesh, shown in Figure 3(b), is reasonably close to the scanned mesh, but it still has some problem areas. To avoid requiring excessive hand-tweaking from the user, we seek an automatic refitting step. Ideally, we would like to optimize the template's control points so that the surface is as close as possible to the data surface at all points in all poses. We take a simpler approach that works reasonably well in practice. After calculating the hole-filled displaced subdivision surface, we move the control points so that the subdivision surface goes exactly through the scanned surface at the control points in a selected "average" pose. This step is done by

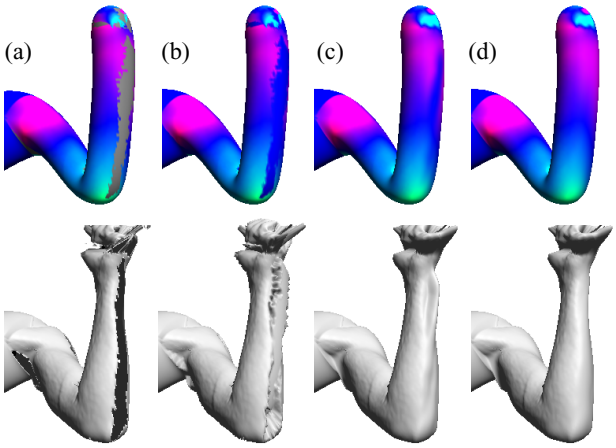


**Figure 5** Hole-filling an arm scan. On the top we show the displacement values on the template surface; blue indicates zero displacement, magenta a negative displacement, and cyan a positive displacement. The subdivision surface with the displacements applied is shown on the bottom. (a) Original surface after parameterization. There is a large hole along then forearm, and smaller holes in the underarm, shoulder, and hand. (b) We initialize the missing areas with a zero displacement. (c) After running one-quarter of the smoothing optimization. (d) After full optimization. The discontinuity between the shoulder and the torso is intentional.

solving the linear system $\mathbf{MA} = \mathbf{V}$, where $\mathbf{V}$ is the desired template surface locations at each control point, $\mathbf{M}$ is the limit mask matrix, and $\mathbf{A}$ is the new control point positions. Since there are only 72 control points, this is an easy calculation. The refitted surface is shown in Figure 3(c).

An additional motivation for having a template surface that is close to the scanned mesh is that it allows us to reject rays that intersect too far away. This problem occurs particularly in regions such as the elbow crease and underarm where cast rays pass through holes in the mesh and hit a surface much further away. Having a well-fit template surface allows us to easily reject these rays by treating large displacements as holes.

## 5 Interpolation and Reconstruction

Referring back to our formulation in Equations 1 and 2, we have now established a template surface, $T(\mathbf{u}, \mathbf{q})$, and a complete displacement map, $d_i(\mathbf{u}, \mathbf{q})$, for each example. All that remains is to specify the weighting function for each example, $w_i(\mathbf{u}, \mathbf{q})$. We split this into two functions: $w_i^p(\mathbf{q})$, which performs scattered data interpolation based on the pose, and $w_i^b(\mathbf{u})$, which blends the arm, shoulder, and torso data sets. These two functions will be multiplied to give: $w_i(\mathbf{u}, \mathbf{q}) = w_i^p(\mathbf{q}) w_i^b(\mathbf{u})$.

### 5.1 Pose-based weight calculation

Given a new point in the pose space we need to calculate a weight, $w_i^p(\mathbf{q})$, for each example. The interpolated displacements will be a linear combination of the examples, using these weights. These weights have three constraints:

1. At an example point, the weight for that example must be one, and all other weights must be zero.
2. The weights must always sum to one.
3. The weights must be continuous so that animation will be smooth.

We initially tried using cardinal radial basis functions, as described by Sloan et al. [2001]. This worked well for the our arm model because it consists only of single-axis rotations. However, when working with full quaternion rotations, naive application of radial basis function interpolation does not work well, because it treats the quaternion components (or Euler angles) as if they were independent linear dimensions. Another problem with cardinal radial basis functions is that they give negative weights. Although there is no problem with small negative weights, in some regions of joint space the magnitude of the weights becomes quite large, exaggerating the deformations to an unreasonable degree.

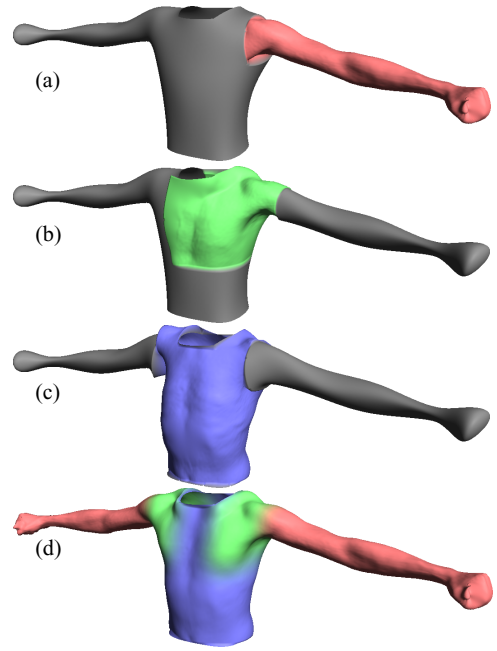


**Figure 6** Blending the three data sets. (a) A sample arm pose. (b) A sample shoulder pose. (c) A sample torso pose. (d) Blend of arm, shoulder, and torso, with a mirrored right shoulder and right arm. Color indicates the blending weight.

An alternative technique is $k$-nearest-neighbors interpolation. The idea is to choose the $k$ closest example points and assign each of them a weight based on their distance. All other points are assigned a weight of zero. The goal is to create a function of the distances that meets the three criteria listed above. Buhler et al. [2001] developed an interpolation function of this sort. Before normalization, it takes the form:

$$w_i^p(\mathbf{q}) = \frac{1}{D(\mathbf{q}, \mathbf{q}_i)} - \frac{1}{D(\mathbf{q}, \mathbf{q}_t)} \tag{5}$$

where $D(\mathbf{q}, \mathbf{q}_i)$ is the distance between the new points and example $i$, and $t$ is the index of the $k^{th}$ closest example. For our upper body model, we found that $k = 8$ gave satisfactory results.

We use a different distance function for each data set. For scans from the arm data set we use a distance function of $\sqrt{(\Delta\text{elbow angle})^2 + (\Delta\text{forearm twist})^2 + (\Delta\text{wrist angle})^2}$. In the shoulder and torso data sets, the pose space includes quaternions, so we define a more appropriate distance function: the great-arc length between the two quaternions on a four-dimensional sphere.

The weights must be normalized since they will not necessarily sum to one. If the desired pose is equal to an example pose, then that example has infinite weight and, after normalization, is the sole contributor to the reconstructed shape in that pose.

### 5.2 Combining Parts

Using the technique above for calculating $w_i^p$, we can interpolate the shape of each body part separately. The final step is to blend the body parts together using the spatially varying blending weight, $w_i^b$.

Subdivision patches which are only covered by one data set have

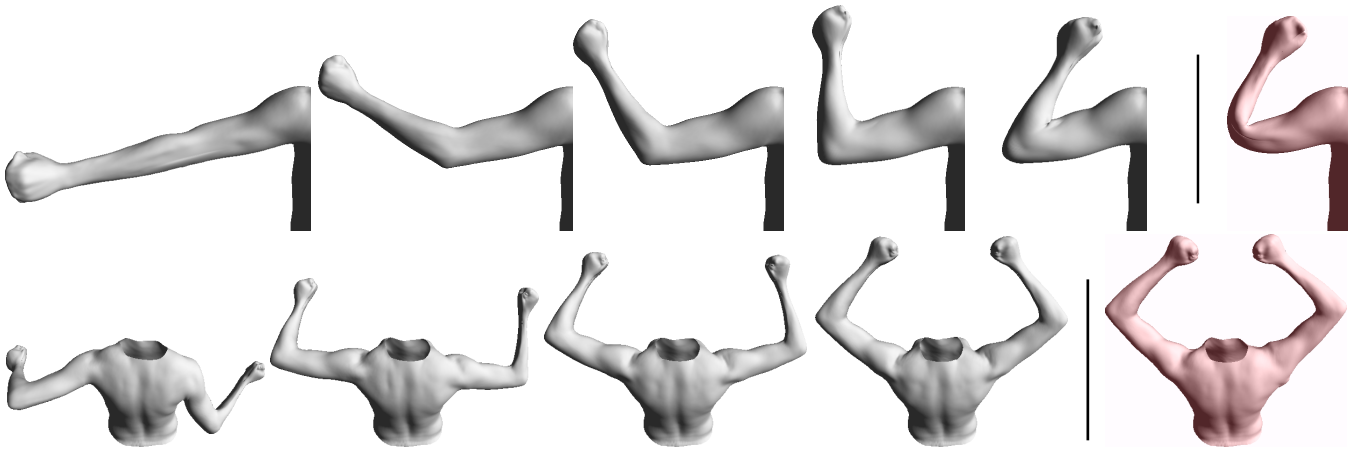

**Figure 7** An interpolation, in gray, between two poses with different elbow angles (above) and different shoulder and clavicle angles (below). The red models on the right were generated by applying the displacements from the left-most poses onto the subdivision surface from the right-most poses. The top red model shows that the protrusion of the elbow and the slight contraction of the biceps are determined by the displacements. In the bottom red model notice that the dimples at the top of the shoulder and at the scapulae, and the correction of the underarm are not visible if the displacements are not updated.

a blending weight $w_i^b(\mathbf{u})$ of 1 if $i$ is a member of the data set and 0 otherwise. For areas that are covered by more than one data set, we need to smoothly blend across the overlapping region. Therefore our blending function must take the value 0 at one boundary and 1 at the other boundary of the overlap region. A linear blending function based on Euclidean distance has this property. However, we also want our function to be C1 continuous at the edges. Therefore we need to use a higher order blending function; we chose to use one wave of the cosine function as follows:

$$w_i^b(\mathbf{u}) = 1 + \frac{1}{2}\cos\left[\left(\frac{b(\mathbf{u}_i)}{x} - 1.0\right)\pi\right] \tag{6}$$

where $b(\mathbf{u}_i)$ is the distance between $\mathbf{u}_i$ and the patch boundary, and $x$ is the width of the overlap region.

The blending weights for our upper body model and a sample blend are shown in Figure 6. We can construct a right arm and shoulder by mirroring all joint angles and deformation data through the sagittal plane, thereby avoiding the work of scanning both arms.

## 6 Results

We have tested our system for creating posable human shapes starting from the data set described in Section 2. Figure 7 shows two simple interpolations between novel poses. In each of these figures we also show the effect of moving the template surface but not adjusting the displacements in order to illustrate the difference between the deformation caused by the template surface and the deformation cause by the interpolated displacements. The most egregious error in the non-interpolated meshes is at the elbow, where the bones of the arm do not protrude. Other prominent artifacts include a lack of swelling of the biceps, and for the shoulder example, missing creases in the shoulder, and a protrusion in the right underarm. The interpolated meshes have none of these problems and are a more faithful portrayal of the subject's anatomy.

We can also control our model with motion capture data. Figure 1 demonstrates a variety of poses drawn from motion capture of a different individual, with the joint angles mapped onto the range scanned subject's skeleton. The accompanying video shows full animations generated from motion capture data. Although some of the poses in Figure 1 go beyond the range of pose space that we captured, the template surface enables extreme poses to look reasonable.

The biggest problems arise in the crease areas, such as the inside of the elbow and the underarm. Creases cause problems for three reasons. First, they cannot be accurately scanned because parts of the surface are completely occluded. Secondly, creases are by their very nature areas of high curvature, which, as shown in Figure 4(b), can be a problem for the displaced subdivision surface parameterization. Our refitting algorithm helps, but occasionally poorly parameterized areas remain. Finally, our approach does not perform actual collision detection. As a result, it cannot be expected to obtain correct results when the deformations are caused by collisions. Figure 7 shows evidence of these issues; in the top right gray pose, a small ridge near the elbow crease is caused by interpolating a creased and non-creased surface.

One strength of our approach is speed. Our upper body model has a control mesh with 65 faces, and each face is subdivided five times, giving a mesh with roughly sixty-six thousand vertices. Even with this dense mesh, we can generate and render novel poses at nearly interactive rates (3-5 frames per second); this rate can be increased by by sampling at a smaller subdivision level.

Although the model we have developed yields a fairly faithful reproduction of the posable shape of only a single individual, the framework does enable some editing operations to change the body appearance of that individual. For instance, changing bone lengths or scaling the template control points relative to the skeleton are straightforward to implement; examples of these operations appear in Figure 8.

## 7 Conclusion

We have developed an end-to-end system for capturing human body scans, estimating poses and kinematics, reconstructing a complete displaced subdivision surface in each pose, and combining the surfaces using $k$-nearest-neighbors scattered data interpolation in pose space. The result is an example-based, posable model that captures high definition shape changes over large ranges of motion. The interpolations are nearly interactive, with the capability of trading off speed for resolution, and the representation permits editing operations such as changing the underlying surface shape and kinematics.

Our work leaves ample room for future research. In the short term, we would like to explore more automatic techniques for pose estimation, such as fully automatic marker detection and identification or even non-rigid, markerless registration. As noted in the previous section, creases cause problems for constructing displaced subdivision surfaces. Possible solutions include finding a better template surface optimized across all poses (rather than an arbitrary "average" pose) or computing a template that itself changes from pose to pose after fitting to each one. Extending our technique to handle other degrees of freedom such as muscle load (e.g, when

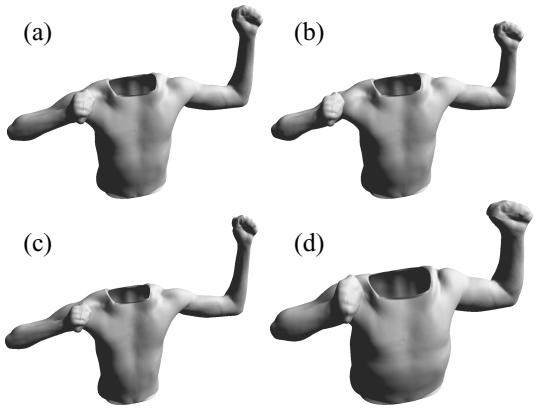


**Figure 8** By simply scaling the location of the template surface's control points relative to the bones, we can alter the appearance of the animated character. (a) Original appearance; (b) forearm 6 cm shorter; (c) 20% thinner across all body parts; (d) 44% fatter.

lifting heavy objects) would also be useful.

In the longer term, generalizing beyond a single example suggests a number of future directions. The ability to edit the surface template and the skeleton hints at the possibility of more sophisticated editing, e.g., exaggerating deformations, or even mapping deformations onto other bodies scanned in fewer poses. In addition, scanning large numbers of people would allow more degrees of freedom for modeling the human body by example, e.g., exposing controls for male-ness and female-ness [Blanz and Vetter 1999]. Finally, the posable model we have developed does not encompass dynamical behaviors or deformations due to collisions. Combining example-based techniques with anatomical and physically based modeling promises to be a fruitful area for future research.

## Acknowledgments

We would like to thank David Addleman and Christian Juhring of Cyberware for their assistance and the use of their whole-body range scanner. Thanks also go to Steve Capell for use of his subdivision code, Keith Grochow and Eugene Hsu for their motion capture work, and to Daniel Wood and others who provided assistance and feedback on this paper and the video. We also thank Michael Cohen for his helpful discussions on interpolation techniques. This work was supported by the University of Washington Animation Research Lab, industrial gifts from Intel and Microsoft Research, and NSF grant CCR-0098005.

## References

AUBEL, A., AND THALMANN, D. 2001. Interactive modeling of the human musculature. In *Proc. of Computer Animation 2001*.

BLANZ, V., AND VETTER, T. 1999. A morphable model for the synthesis of 3D faces. In *Proceedings of ACM SIGGRAPH 99*, Addison Wesley, New York, A. Rockwood, Ed., Annual Conference Series, 187–194.

BUEHLER, C., BOSSE, M., MCMILLAN, L., GORTLER, S. J., AND COHEN, M. F. 2001. Unstructured lumigraph rendering. In *Proceedings of ACM SIGGRAPH 2001*, ACM Press / ACM SIGGRAPH, E. Fiume, Ed., Annual Conference Series, ACM, 425–432.

CATMULL, E., AND CLARK, J. 1978. Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer-Aided Design 10* (Sept.), 350–355.

CHADWICK, J. E., HAUMANN, D. R., AND PARENT, R. E. 1989. Layered construction for deformable animated characters. *Computer Graphics (Proceedings of ACM SIGGRAPH 89) 23*, 3, 243–252.

CURLESS, B., AND LEVOY, M. 1995. Better optical triangulation through spacetime analysis. In *Proceedings of IEEE International Conference on Computer Vision*, 987–994.

CURLESS, B., AND LEVOY, M. 1996. A volumetric method for building complex models from range images. In *Proceedings of ACM SIGGRAPH 96*, ACM Press / ACM SIGGRAPH / Addison Wesley Longman, K. Akeley, Ed., Annual Conference Series, ACM, 303–312.

GILL, P. E., MURRAY, W., AND WRIGHT, M. H. 1989. *Practical Optimization*. Academic Press.

GOLD, S., AND RANGARAJAN, A. 1996. A graduated assignment algorithm for graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence 18*, 4, 377–388.

GOWITZKE, B. A., AND MILNER, M. 1988. *Scientific Bases of Human Movement*, third ed. Williams & Wilkins, Baltimore, MD.

GUENTER, B., GRIMM, C., WOOD, D., MALVAR, H., AND PIGHIN, F. 1998. Making faces. In *Proceedings of ACM SIGGRAPH 98*, Addison Wesley, M. Cohen, Ed., Annual Conference Series, ACM, 55–66.

HERDA, L., FUA, P., PLÄNKERS, R., BOULIC, R., AND THALMANN, D. 2001. Using skeleton-based tracking to increase the reliability of optical motion capture. *Human Movement Science Journal 20*, 313–341.

KRISHNAMURTHY, V., AND LEVOY, M. 1996. Fitting smooth surfaces to dense polygon meshes. In *Proceedings of ACM SIGGRAPH 96*, Addison Wesley, H. Rushmeier, Ed., Annual Conference Series, ACM, 313–324.

LEE, A., MORETON, H., AND HOPPE, H. 2000. Displaced subdivision surfaces. In *Proceedings of ACM SIGGRAPH 2000*, ACM Press / ACM SIGGRAPH / Addison Wesley Longman, K. Akeley, Ed., ACM, 85–94.

LEWIS, J. P., CORDNER, M., AND FONG, N. 2000. Pose space deformations: A unified approach to shape interpolation and skeleton-driven deformation. In *Proceedings of ACM SIGGRAPH 2000*, ACM Press / ACM SIGGRAPH / Addison Wesley Longman, K. Akeley, Ed., Annual Conference Series, ACM, 165–172.

LUTTGENS, K., AND WELLS, K. F. 1982. *Kinesiology: scientific basis of human motion*, seventh ed. CBS College Publishing, New York, NY.

MAGNENAT-THALMANN, N., LAPERRIERE, R., AND THALMANN, D. 1988. Joint-dependent local deformations for hand animation and object grasping. In *Proc. Graphics Interface*, 26–33.

NG-THOW-HING, V. 1999. Physically based anatomic modeling for construction of musculoskeletal systems. In *Proceedings of the 1999 SIGGRAPH annual conference: Conference abstracts and applications*, ACM Press, New York, NY 10036, USA, Computer Graphics, ACM, 264–264.

PIGHIN, F., HECKER, J., LISCHINSKI, D., SZELISKI, R., AND SALESIN, D. H. 1998. Synthesizing realistic facial expressions from photographs. In *Proceedings of ACM SIGGRAPH 98*, Addison Wesley, M. Cohen, Ed., Annual Conference Series, ACM, 75–84.

PRAUN, E., SWELDENS, W., AND SCHRÖDER, P. 2001. Consistent mesh parameterizations. In *Proceedings of ACM SIGGRAPH 2001*, ACM Press / ACM SIGGRAPH, E. Fiume, Ed., Annual Conference Series, ACM, 179–184.

ROSE, C., COHEN, M. F., AND BODENHEIMER, B. 1998. Verbs and adverbs: Multidimensional motion interpolation. *IEEE Computer Graphics and Applications 18*, 5 (Sept./Oct.), 32–41.

SCHEEPERS, F., PARENT, R. E., CARLSON, W. E., AND MAY, S. F. 1997. Anatomy-based modeling of the human musculature. In *Proceedings of ACM SIGGRAPH 97*, T. Whitted, Ed., Annual Conference Series, ACM, 163–172.

SHEN JIANHUA, THALMANN, N. M., AND THALMANN, D. 1994. Human skin deformation from cross-sections. In *Computer Graphics Int. '94*.

SILAGHI, M.-C., PLÄNKERS, R., BOULIC, R., FUA, P., AND THALMANN, D. 1998. Local and global skeleton fitting techniques for optical motion capture. In *Proceedings of the International Workshop on Modelling and Motion Capture Techniques for Virtual Environments (CAPTECH-98)*, Springer, Berlin, N. Magnenat-Thalmann and D. Thalmann, Eds., vol. 1537 of *LNAI*, 26–40.

SLOAN, P.-P., ROSE, C., AND COHEN, M. F. 2001. Shape by example. In *Proceedings of 2001 Symposium on Interactive 3D Graphics*.

TALBOT, J. 1998. *Accurate Characterization of Skin Deformations Using Range Data*. Master's thesis, Department of Computer Science, University of Toronto.

TURK, G., AND LEVOY, M. 1994. Zippered polygon meshes from range images. In *Proceedings of ACM SIGGRAPH 94*, ACM Press, vol. 28 of *Annual Conference Series*, ACM, 311–318.

WILHELMS, J., AND GELDER, A. V. 1997. Anatomically based modeling. In *Proceedings of ACM SIGGRAPH 97*, T. Whitted, Ed., Annual Conference Series, ACM, 173–180.

ZHU, C., BYRD, R. H., LU, P., AND NOCEDAL, J. 1997. Algorithm 778. L-BFGS-B: Fortran subroutines for Large-Scale bound constrained optimization. *ACM Transactions on Mathematical Software 23*, 4 (Dec.), 550–560.