Articulated Soft Objects for Video-based Body Modeling

Ralf Plänkers and Pascal Fua*
Computer Graphics Lab (LIG)
Swiss Federal Institute of Technology (EPFL)
1015 Lausanne, Switzerland
Email: {Ralf.Plaenkers, Pascal.Fua}@epfl.ch

Proc. 8^{th} International Conference on Computer Vision, Vancouver, Canada, July 2001

Abstract

We develop a framework for 3–D shape and motion recovery of articulated deformable objects. We propose a formalism that incorporates the use of implicit surfaces into earlier robotics approaches that were designed to handle articulated structures. We demonstrate its effectiveness for human body modeling from video sequences. Our method is both robust and generic. It could easily be applied to other shape and motion recovery problems.

1. Introduction

Recently, many approaches to tracking and modeling articulated 3–D objects have been proposed. They have been used to capture people's motion in video sequences with potential applications to animation, surveillance, medicine, and man-machine interaction.

Such systems are promising. However, they typically use oversimplified models, such as cylinders or ellipsoids attached to articulated skeletons. Such models are too crude for precise recovery of both shape and motion. We propose a framework that retains the articulated skeleton but replaces the simple geometric primitives by soft objects. Each primitive defines a field function and the skin is taken to be a level set of the sum of these fields. This implicit surface formulation has the following advantages:

- **Effective use of stereo and silhouette data:** Defining surfaces implicitly allows us to define a distance function of data points to models that is both differentiable and computable without search.
- **Accurate shape description by a small number of parameters:** Varying a few dimensions yields models that can match different body shapes and allow both shape and motion recovery.
- **Explicit modeling of 3–D geometry:** Geometry can be taken into account to predict the expected location of image features and occluded areas, thereby making the extraction algorithm more robust.

Here, we use the model depicted by Figure 1(a-d) to demonstrate our approach in the case of human body modeling [19]. It provides *a priori* information about the shape and the allowable motions of the human body. We will refer to such a model as an *articulated soft object*. This method is equally applicable to other vertebrates, such as the horse and cow of Figure 1(e), and, more generally, to most non-polyhedral articulated objects.

Our approach, like many others, relies on optimization to deform the generic model so that it conforms to the data. This involves computing first and second derivatives of the distance function of the model to the data points. This turns out to be prohibitively complex and slow if done in a brute-force fashion. The main contribution of this paper is a mathematical formalism that greatly simplifies these computations and allows a fast and robust implementation. It extends the traditional robotics approach that were designed to handle articulated bodies [4] and allows the use of implicit surfaces.

We integrate our formalism into a complete framework for tracking and modeling and demonstrate its robustness using video sequences of complex 3–D motions. To validate it, we focus on using stereo and silhouette data because they are complementary sources of information. Stereo works well on both textured clothes and bare skin for surfaces which face the camera but fails where the angle between the view direction and the surface normal is close to orthogo-

*This work was supported in part by in part by the Swiss National Science Foundation.

nal. Silhouettes, on the other hand, provide information at the occluding contour where the surface is tangential to the view direction.

In the remainder of this paper, we first describe related approaches to articulated shape and motion recovery. We then describe our model in more detail and introduce our optimization framework. Finally, we present reconstruction results on complex human motions.

2. Related Work

Many techniques [3, 5, 6, 11, 12, 17] have recently been proposed to track human motion from video sequences. They are fairly effective but use very simplified human body models. They, usually, do not attempt to recover actual body shape and their output would not be sufficient for a truly realistic simulation. Furthermore, the approximate nature of the model is bound to impact the precision with which joint locations are recovered, thereby also limiting their applicability in a field such as medicine. For more details, we refer the interested reader to recent surveys [1, 9, 13].

Laser scanning technology provides a good, if expensive, way to recover the shape of a static person but is not well adapted for motion recovery. Furthermore, it only recovers the skin surface and, without motion data, it is difficult to accurately position the articulation structure inside the surface.

Much cheaper alternatives to shape recovery using orthogonal photographs have been implemented [10]. They can be used for applications such as populating virtual worlds. However, the geometry they recover is crude and realism is achieved by texture-mapping. Again, these systems do not capture dynamics and have only very approximate articulated structures based on anthropometric average models.

Our proposed approach addresses simultaneous recovery of shape and motion of a person from a video image sequence using a model which is sophisticated enough to recover shape and simple enough to track movement using potentially noisy image data.

3. Articulated Model and Surfaces

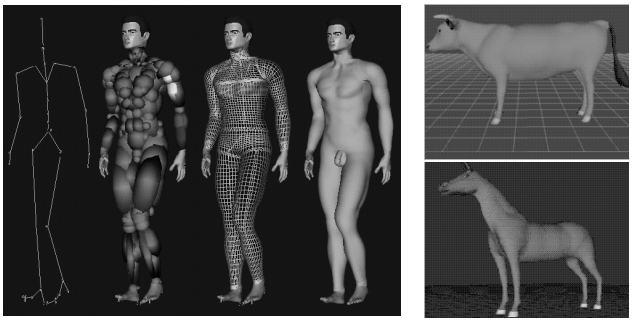
The human body model we use in this work [19] is depicted by Figure 1. It incorporates a highly effective multi-layered approach for constructing and animating realistic human bodies. The first layer is a skeleton that is a connected set of segments, corresponding to limbs and joints. A joint is the intersection of two segments, which means it is a skeleton point around which the limb linked to that point may move.

Smooth implicit surfaces, also known as *metaballs*, *blobby* or *soft objects*, form the second layer [2]. They are used to simulate the gross behavior of bone, muscle, and fat tissue. The metaballs are attached to the skeleton and arranged in an anatomically-based approximation.

In order to prevent body parts from blending into each other, the body is segmented into ten distinct parts: arms, legs and torso are split into upper and lower parts. When computing the implicit surfaces, only metaballs belonging to the same segments are taken into account. We constrain the left and right limbs to be symmetric. The head, hands and feet are explicit surfaces that are attached to the body.

For display purposes a third layer, a polygonal skin surface, is constructed via B-spline patches over control points computed by ray casting [19].

The body shape and position are controlled by a *state vector* Θ, which is a set of parameters controlling joint locations and limb sizes. In this section, we first describe this state vector in more detail and, then, our implicit surface formulation.



(a) (b) (c) (d) (e)

Figure 1. Articulated soft objects: (a) Skeleton. (b) Volumetric primitives used to simulate muscles and fat tissue. (c) Polygonal surface representation of the skin. (d) Shaded rendering. (e) A cow and a horse modeled using the same technique.

3.1. State Vector

Our goal is to use video-sequences to estimate our model's shape and derive its position in each frame. Let us therefore assume that we are given N consecutive video frames and introduce position parameters for each frame.

Let B be the number of body parts in our model. We assign to each body part a variable length and width coefficient. These dimensions change from person to person but we take them to be constant within a particular sequence. This constraint could be relaxed, for example to model muscular contraction.

The model's *shape* and *position* are then described by the combined state vector

$$\Theta = \{\Theta^w, \Theta^l, \ \Theta^r, \Theta^g\} \ , \tag{1}$$

where we have broken Θ into four sub-vectors which control the following model components:

- Shape
  - $\Theta^w = \{\theta^w_b \mid b = 1..B\}$, the width of body parts.
  - $\Theta^l = \{\theta^l_b \mid b = 1..B\}$, the length of body parts.
- Motion
  - $\Theta^r = \{\theta^r_{i,f} \mid j = 1..J, f = 1..N\}$, the rotational degree of freedom of joint j of the articulated skeleton for all frames f.
  - $\Theta^g = \{\theta^g_f \mid f = 1..N\}$, the six parameters of global position and orientation of the model in the world frame for all frames f.

The size and position of the metaballs is relative to the segment they are attached to. A length parameter not only specifies the length of a skeleton segment but also the shape of the attached metaballs in the direction of the segment. Width parameters only influence the metaballs' shape in the other directions.

Motion parameters Θ^r are represented in terms of Euler angles. We can constrain joint motions to anatomically valid ranges by defining an allowable interval for each of the degrees of freedom. Other methods for describing rotations, such as quaternions or exponential maps, can be used as well.

3.2. Metaballs

First presented by Blinn [2] generalized algebraic surfaces, also called *blobby models* or *metaballs*, are defined by a summation over n 3-dimensional Gaussian density distributions, each called a *source* or *primitive*. The final surface $\mathcal{S}$ is found where the density function F equals some threshold amount:

$$\mathcal{S} = \left\{ [x,y,z]^T \in \mathbb{R}^3 \mid F(x,y,z) = T \right\}, \quad (2)$$

$$F(x,y,z) = \sum_{i=1}^{n} f_i(d_i(x,y,z)), \quad (3)$$

$$f_i(x,y,z) = b_i \, exp(-a_i \, d_i(x,y,z)). \quad (4)$$

In this work, we chose the "blobbiness" parameters to be $a_i = 2$, $b_i = 1$ and $T = 0.5$ for all sources $i = 1..n$. Field function f_i is differentiable over the whole domain and it has a long range effect because it approaches zero slowly. In the context of model fitting these two properties are very important as will be discussed in Section 4.

For simplicity's sake, in the remainder of the paper, we will omit the i index for specific metaball sources wherever the context is unambiguous.

We take the spatial distance function d to be an algebraic ellipsoidal distance, further explained in Section 3.3. We use ellipsoidal primitives because they are simple and, at the same time, allow accurate modeling of human limbs with relatively few primitives because metaballs result in a smooth surface, thus keeping the number of parameters low. Using algebraic distances for fitting purposes can result in overfitting in the high-curvature regions in some cases [18]. For our specific application, however, the ellipses only have limited degrees of freedom and are rigidly attached to a skeleton structure. Their shape is controlled by higher level width and length parameters, and, thus, such problems do not occur.

3.3. 3–D Quadratic Distance Function

To express simply the transformations of the implicit surfaces caused by their attachment to an articulated skeleton, we write the ellipsoidal distance function d of Eq. 3 in matrix notation as follows. This formulation will prove key to effectively computing the Jacobians required to implement the fitting and optimization scheme of Section 4.

For a specific metaball and a state vector Θ, we define the 4×4 matrix

$$\mathbf{Q}_\Theta = \mathbf{L}_{\Theta^{w,l}} \cdot \mathbf{C}_{\Theta^{w,l}} \cdot \mathbf{S}_{\Theta^{l,r}}, \quad (5)$$

where $\mathbf{L}$,$\mathbf{C}$ and $\mathbf{S}$ are defined below.

- $\mathbf{LC}_{\Theta^{w,l}} = \mathbf{L}_{\Theta^{w,l}} \cdot \mathbf{C}_{\Theta^{w,l}}$ is the scaling and translation along the principal axes:

$$\mathbf{LC}_{\Theta^{w,l}} = \begin{bmatrix} \frac{1}{\theta^w l_x} & 0 & 0 & -\theta^w c_x \\ 0 & \frac{1}{\theta^w l_y} & 0 & -\theta^w c_y \\ 0 & 0 & \frac{1}{\theta^l l_z} & -\theta^l c_z \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

  where $L = (l_x, l_y, l_z)$ are the radii of an ellipsoid, i.e. half the axis length along the principal directions and $C = (c_x, c_y, c_z)$ is the primitive's center. Coefficients θ^l and θ^w from the state vector Θ control relative *length* and *width* of a metaball. They are shared among groups of metaballs according to segment assignment.

- $\mathbf{S}_{\Theta^{l,r}}$ is the skeleton induced transformation, a 4×4 rotation-translation matrix from the world frame to the frame to which the metaball is attached. Given rotation $\theta \in \Theta^r$ of a joint J, we write:

$$\mathbf{S}_{\Theta^{l,r}} = \mathbf{E} \cdot \mathbf{J}_\theta = \mathbf{E} \cdot \mathbf{R}_\theta \cdot \mathbf{J}_0, \quad (6)$$

  where $\mathbf{E}$ is the homogeneous 4×4 transformation from the joint frame to the quadric frame. $\mathbf{J}_\theta$ is the transform from world frame to joint frame, including the rotation parameterized by θ and $\mathbf{R}_\theta$ is the homogeneous rotation matrix of θ around some axis $\mathbf{a}$ with $\mathbf{J}_0 = \mathbf{R}_\theta^{-1} \cdot \mathbf{J}_\theta$.

Given the $\mathbf{Q}_\Theta$ matrix, we combine the quadric and the articulated skeleton transformations by writing the distance function of Eq. 3 as:

$$d(\mathbf{x}, \Theta) = \mathbf{x}^T \cdot \mathbf{Q}_\Theta^T \cdot \mathbf{Q}_\Theta \cdot \mathbf{x} \ . \quad (7)$$

$d(\mathbf{x}, \Theta)$ defines an ellipsoidal quadratic distance field.

We can now compute the global field function F of Eq. 3 by plugging Eq. 7 into the individual field functions of Eq. 4 and adding up these fields for all primitives. In other words, the field function from which the model surface is derived can be expressed in terms of the $\mathbf{Q}_\Theta$ matrices, and so can its derivatives as will be seen later. These matrices will therefore constitute the basic building blocks of our optimization scheme's implementation.

Note that we employ a generic body model consisting of a fixed number of properly placed and sized metaballs. The parameters to optimize for are high level limb parameters $\Theta^{l,w}$ which implicitly adjust the lower level metaball parameters L, C. This constrains the parameter space considerably during optimization for proper body corpulence.

4. Least Squares Framework

From a fitting point of view, the body model of Section 3 embodies a rough knowledge about the shape of the body and can be used to constrain the search space. Our goal is to derive its degrees of freedom so that it conforms as faithfully as possible to the image data.

Here we use motion sequences such as the ones shown in Figure 5. The expected output of our system is the instantiated state vector Θ of Eq. 1 that describes the model's shape and motion.

In standard least-squares fashion, we use the image data to write $nobs$ observation equations y of the form

$$y_i(\Theta) = obs_i - \epsilon_i \ , \ 1 \leq i \leq nobs \ , \quad (8)$$

where ϵ_i is the deviation from the model. We will then minimize $v^T P v$ where $v = [\epsilon_1, \ldots, \epsilon_{nobs}]$ is the vector of residuals and P is a usually diagonal weight matrix associated to the observations.

Our system must be able to deal with observations coming from different sources that may not be commensurate with each other. Formally we can rewrite the observation equations of Equation 8 as

$$y_i^{type}(\Theta) = obs_i^{type} - \epsilon_i \ \ , 1 \leq i \leq nobs \ \ , \quad (9)$$

with weight p_i^{type}, where $type$ is one of the possible types of observations we use. In our work, $type$ can be object space coordinates, silhouette rays or temporal constraints. However, other information cues can easily be integrated. See [14] for further details about weighing the different observations.

To solve our minimization problem, we use an implementation of the Levenberg-Marquardt algorithm [15] that can handle the large number of parameters and observations we must deal with as well as hard constraints, such as joint limits.

5. Data Constraints

In our approach, we use a well defined and differentiable error and distance measure. As shown in Section 3.3, to compute the algebraic distance between a data observation and the model in its current state, no search is needed. We only need to evaluate the $\mathbf{Q}_\Theta$ matrices of Eq. 5 and the set of quadratic terms described by Equation 7.

In this section, we will show that the same holds true for the derivatives of the field function with respect to the state variables. As a result, the derivatives of the residuals of Section 4 are also easily computable using our matrix formalism. This is important because the least-squares solver we use takes advantage of differential information for faster and more robust optimization, as do most powerful optimizers.

We now turn to the detailed implementation of the 3–D point and silhouette observations.

5.1. 3–D Point Observations

A 3–D point $\mathbf{x}$, e.g. from stereo, is integrated into the framework by adding one observation of the form

$$obs = F(\mathbf{x}, \Theta) - T \ . \quad (10)$$

This constrains the point to lie on the surface parameterized by the state vector Θ and the threshold T. To compute the Jacobian of the error function of Eq. 3, we estimate the differential with respect to parameter $\theta \in \Theta$ as follows:

$$\frac{\partial}{\partial \theta} F(\mathbf{x}, \Theta) = \frac{\partial}{\partial \theta} \sum_{i=1}^{n} f_i(d_i(\mathbf{x}, \Theta)) \quad (11)$$

and the respective differentials of the field and the distance function can be shown to be

$$\frac{\partial}{\partial \theta} f(d(\mathbf{x}, \Theta)) = \frac{\partial f}{\partial d} \frac{\partial d}{\partial \theta} = -2 \frac{\frac{\partial}{\partial \theta} d(\mathbf{x}, \Theta)}{e^{2d(\mathbf{x}, \Theta)}} \quad (12)$$

$$\frac{\partial}{\partial \theta} d(\mathbf{x}, \Theta) = 2 * \mathbf{x}^T \cdot \mathbf{Q}_\Theta^T \cdot \ \frac{\partial}{\partial \theta} \mathbf{Q}_\Theta \ \cdot \mathbf{x} \quad (13)$$

From this set of equations, we see that the derivatives consist of modules most of which are constant for all observations and constraints and need only be computed once. In the appendix 8.1 we will show how to derive the differential of Q which can also be modularized.

Because of the long range effect of the exponential field function the fitting succeeds even when the model is not very close to the data. Also, in least-squares sense an error measure that approaches zero instead of becoming ever greater with growing distance has the effect of filtering outliers. Other methods need great care and effort to avoid misfits due to erroneous or noisy data.

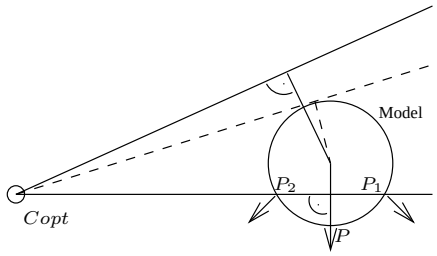


Figure 2. A silhouette ray for a circular object is represented by a dashed line. The camera is depicted by $Copt$. The ray touches the object at exactly one point and it's slope is perpendicular to the surface normal. The two other rays don't satisfy both of the silhouette criteria of Section 5.2.

5.2. Silhouettes Observations

A silhouette point in the image defines a line of sight to which the surface must be tangential as depicted by Figure 2. Let $\theta \in \Theta$ be an element of the state vector. For each value θ, we define the implicit surface

$$\mathcal{S}(\theta) = \{[x, y, z]^T \in R^3, F(x, y, z, \theta) = T\} \ . \tag{14}$$

Let $[x(\theta), y(\theta), z(\theta)]$ be the point on the line of sight where it is tangential to $\mathcal{S}(\theta)$. By definition $[x(\theta), y(\theta), z(\theta)]$ satisfies two constraints:

1. The point has to be on the surface, therefore $F(x(\theta), y(\theta), z(\theta), \theta) = T$.
2. The normal to $S(\theta)$ is perpendicular to the line of sight at $[x(\theta), y(\theta), z(\theta)]$.

We integrate silhouette observations into our framework by performing an initial search (using Brent's line minimization [15]) along the line of sight to find the point that is closest to the model at its current configuration. This point is added as an observation having the same error function as 3–D point observations in Eq. 10. Its position is updated after each iteration. This enforces the first constraint. The second constraint is taken into account during the computation of the Jacobian for silhouette observations [18] and we provide a full derivation in appendix 8.2. Its complexity stems from the fact that the point moves along the line of sight during optimization. The Jacobian has to take this into account. This involves computing second order derivatives:

$$\begin{aligned}
\frac{\partial^2 d}{\partial x_i \partial x_j} &= 2 * \frac{\partial \mathbf{x}}{\partial x_i}^T \cdot \mathbf{Q}_\Theta^T \cdot \mathbf{Q}_\Theta \cdot \frac{\partial \mathbf{x}}{\partial x_j} \ , \\
\frac{\partial^2 d}{\partial x_i \partial \theta} &= 2 * \mathbf{x}^T \cdot \quad \frac{\partial}{\partial \theta}\mathbf{Q}_\Theta^T \ \cdot \mathbf{Q}_\Theta + \mathbf{Q}_\Theta^T \cdot \ \frac{\partial}{\partial \theta}\mathbf{Q}_\Theta \quad \cdot \frac{\partial \mathbf{x}}{\partial x_i} \cdot
\end{aligned}$$

As was the case for 3–D point observations, the silhouette observation derivatives can easily be computed thanks to the modular matrix notation. They can also be implemented efficiently because most parts do not change between observations and need to be computed only once.

Figure 4 illustrates the importance of silhouette information. In this example, we used a single stereo pair and allowed for changes in the model's posture and the shape parameters of the arms. In Figure 4(c) only the 3–D information is used. The fitting tends to incorrectly move the model further away from the cloud and to compensate by inflating the arms to keep contact with the point cloud. The noisy stereo data is too ambiguous to sufficiently constrain the model. The silhouettes are needed to constrain it, as shown in Figure 4(d) where we fitted to both stereo and silhouette information.

The behavior shown in Figure 4 is very similar to that observed when fitting ellipses to noisy 2–D points, especially in the case of nonuniform distribution like points obtained from stereo reconstruction. To illustrate this point, in Figure 3, we have implemented a 2–D version of the constraints of Section 5.2. In this example, we use 9 points that lie on an ellipse and recompute the ellipse parameters by least-squares minimization. In the absence of noise, the recovered ellipse is of course the right one. But, as shown in the figure's top row, the recovered ellipse becomes very different from the real one even with only a small amount of added noise. The 2–D equivalent of silhouette constraints are tangent constraints such as the ones depicted by lines in the upper left image of Figure 3(a). On the figure's bottom row, we show the result of performing again the least-square minimization using the same noisy points as before but with two added silhouette constraints. On average, using the tangent information yields results that are much closer to the true solution. The more silhouette constraints we add the better the solution.

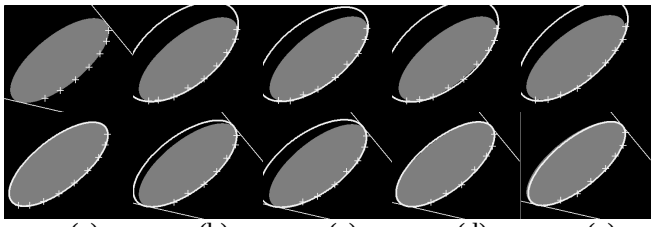
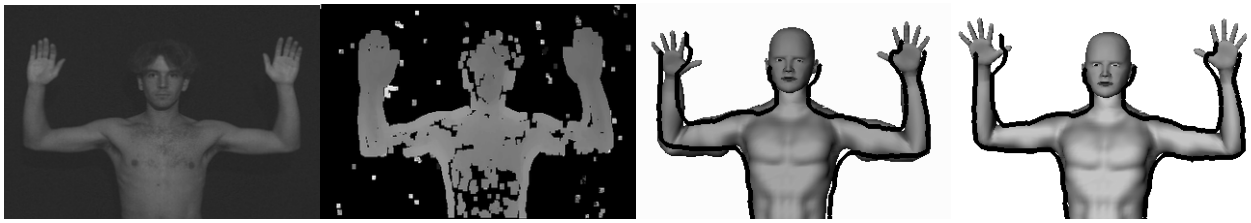


Figure 3. Fitting an ellipse to data points. (a) Top: 9 data points lying on an ellipse and two tangents. Bottom: In the absence of noise, the fitted ellipse is perfectly superposed with the original one. (b,c,d,e) Four independent fitting results using randomized data points. Top: Without using the tangents. Bottom: Using the tangents.

5.3. Segmentation

Thanks to the metaball framework segmentation is not necessary. In theory, all metaballs form a single smooth surface, i.e. the whole body model generates a single force field being the sum of each primitive's fields and the distance of an observation to the model has been defined as



(a) (b) (c) (d)

Figure 4. The importance of silhouette information for shape modeling. (a) Original image. (b) Disparity map from a stereo pair. (c) No silhouette constraints were used and the fitting puts the model too far away from the cloud. The system compensates by incorrectly enlarging the primitives. The silhouettes provide stricter constraints for the model. (d) Result of the fitting with silhouettes.

the value of the field at that point (Eq. 3 and 10). Thus, we can compute the error function and the model's Jacobian without search as a simple evaluation of the metaball matrix formulation of Section 3 and its derivatives.

In practice we separated the body into 10 distinct parts and we perform an initial search to assign an observation to the closest body part.

6. Results

The image from Figure 4 is one of a trinocular sequence in which the subject performed a complex upper body motion. It was acquired by three cameras in an L configuration taking non-interlaced images at 30 frames/sec, with an effective resolution of 640×400. Our stereo algorithm [7] produced very dense point clouds which are then filtered yielding about 4000 evenly distributed 3–D points on the surface of the subject, even without textured clothes. The body outline consists of about 1000 pixels and may be obtained in many ways. We are currently experimenting with robust contour tracking techniques as well as re-projecting the model to guide the process.

We use the framework presented in this paper for tracking the human figure as well as recovering its shape parameters. This is basically a two-step process. After manual initialization in the first frame, the system optimizes posture on a frame-by-frame basis (tracking). Then, the system optimizes posture as well as shape for all frames simultaneously (fitting). We refer the interested reader to [14] for further details.

In Figure 5(a), we show four frames of that sequence and the result of the tracking and fitting process. The skeleton and the metaballs have been resized to conform to the subject's corpulence. The model's head was generated from a single video sequence [8].

We applied the algorithm with the same generic model to a different subject having different corpulence as shown in Figure 5(b). The system resized the model correctly and the shape and motion of this subject was captured as well.

7. Conclusion

We have presented an extensible framework for video-based modeling using articulated 3–D soft objects. The volumetric models we use are sophisticated enough to recover shape and simple enough to track motion using potentially noisy image data. This has allowed us to validate our approach using complex video-sequences without engineering the environment or adding markers.

The implicit surface approach to modeling we advocate extends earlier robotics approaches designed to handle articulated bodies. It has a number of advantages for our purposes. First, it allows us to define a distance function from data points to models that is both differentiable and computable without search. Second, it lets us describe accurately both shape and motion using a fairly small number of parameters. Last, the explicit modeling of 3–D geometry lets us predict the expected location of image features and occluded areas, thereby making the extraction algorithm more robust.

Our approach relies on optimization to deform the generic model so that it conforms to the image data. This involves computing first and second derivatives of the distance function from model to data points. The main contribution of this paper is a mathematical formalism that greatly simplifies these computations and allows a fast and robust implementation. This is in many ways orthogonal to recent approaches to human body tracking as we address the question of how to best represent the human body for tracking and fitting purposes. The actual optimization algorithm could easily be replaced by others, e.g. probability or multi-modal based techniques.

In our current work, we rely on cheap and easily installed video cameras to provide data. This, we hope, will lead to

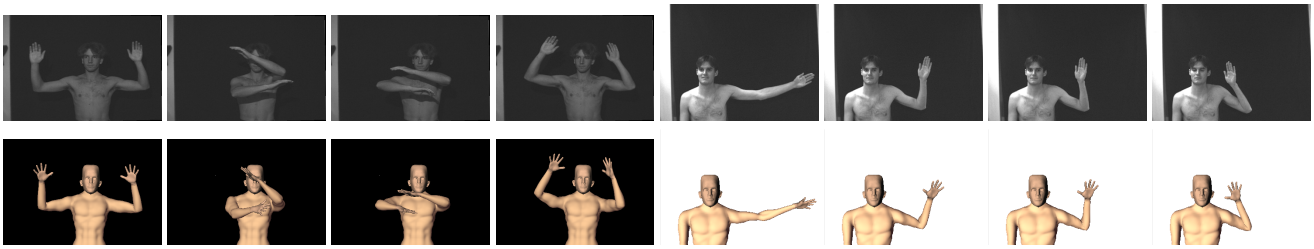


Figure 5. In the top row are the original sequences of upper body motions of different persons. Results of the tracking and fitting are shown in the bottom row. Although the two persons have very different body sizes the system adjusts the generic model accordingly.

practical applications in the field of medicine or entertainment. It would also be interesting to test our approach using high quality data coming from a new breed of dynamic 3–D scanners [16]. Our technique will provide the relative position of the skeleton inside the data and a standard joint angle based description of the subject's motion. Having high-resolution front and back data coverage of the subject should allow us to recover very high-quality and animatable body models.

In future work, we will apply this method to a larger number of sequences of subjects with different corpulences. If the body shapes change drastically, this will involve devising automated algorithms for adding or removing metaballs to the generic model. To further increase robustness, we also plan to take dynamics into account. The current model constrains the shape and imposes joint angle limits. It will be augmented by bio-medical constraints that bound the acceleration and speed or more complex interactions.

8. Appendix

8.1. Jacobian of a Quadric

Matrix notation allows us to easily compute the differentials element-wise. Shape parameters θ_w control the size and position of the metaballs but they are independent of the skeleton structure. Thus, we can differentiate $\mathbf{Q}$ as follows:

$$\mathbf{Q}_{\Theta^w} = \mathbf{L}_{\Theta^w} \cdot \mathbf{C}_{\Theta^w} \cdot \mathbf{S} \tag{15}$$

$$\frac{\partial}{\partial \theta^w}\mathbf{L} = \begin{bmatrix} -\frac{1}{\theta_w^2 l_x} & 0 & 0 & 0 \\ 0 & -\frac{1}{\theta_w^2 l_y} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{16}$$

$$\frac{\partial}{\partial \theta^w}\mathbf{C} = \begin{bmatrix} 0 & 0 & 0 & -c_x \\ 0 & 0 & 0 & -c_y \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{17}$$

$$\frac{\partial}{\partial \theta^w}\mathbf{Q} = \frac{\partial}{\partial \theta^w}\mathbf{L} \cdot \mathbf{C} + \mathbf{L} \cdot \frac{\partial}{\partial \theta^w}\mathbf{C} \cdot \mathbf{S} \tag{18}$$

For shape parameters θ_l the differentiation is more complex as they control the metaballs as well as the lengths of skeleton segments. However, element-wise differentiation is done in a similar way.

For rotational parameters θ_r the differentiation can be shown to be:

$$\frac{\partial}{\partial \theta^r}\mathbf{Q} = \mathbf{L} \cdot \mathbf{C} \cdot \mathbf{E} \cdot [\mathbf{a}\times] \cdot \mathbf{J}_\theta \tag{19}$$

$$\frac{\partial}{\partial \theta^r}\mathbf{Q} \cdot \mathbf{x} = \mathbf{L} \cdot \mathbf{C} \cdot \mathbf{E} \cdot \mathbf{a} \times \mathbf{jx} \ , \tag{20}$$

with $[\mathbf{a}\times]$ being the cross-product operator of axis-vector $\mathbf{a}$. The vector $\mathbf{jx} = \mathbf{J}_\theta \cdot \mathbf{x}$ is the vector from joint center to observation.

Eq. 20 can be efficiently implemented because it only consists of a simple cross-product transformed into the quadric's frame.

8.2. Silhouettes and Implicit Surfaces

A silhouette point in the image defines a line of sight to which the surface must be tangential. Let

$$Sl = [Sl_x, Sl_y, Sl_z] \tag{21}$$

be a unit vector that defines the direction of the line of sight. Let $N1$ and $N2$ be two additional vectors such that $N1, N2, Sl$ form an orthonormal referential.

Let θ be a parameter of the implicit surface $\mathcal{S}(\theta)$ defined as

$$\mathcal{S}(\theta) = \{[x, y, z]^T \in R^3, f(x, y, z, \theta) = T\} \ , \tag{22}$$

and let $x(\theta), y(\theta), z(\theta)$ be the point on the line of sight where it is tangential to $\mathcal{S}(\theta)$.
By definition $x(\theta), y(\theta), z(\theta)$ satisfies

$$f(x(\theta), y(\theta), z(\theta), \theta) = T \ . \tag{23}$$

To exploit this constraint in the least-squares context, one must be able to compute

$$\frac{df(x(\theta),y(\theta),z(\theta),\theta)}{d\theta} = \frac{\partial f}{\partial x}\frac{dx}{d\theta} + \frac{\partial f}{\partial y}\frac{dy}{d\theta} + \frac{\partial f}{\partial z}\frac{dz}{d\theta} + \frac{\partial f}{\partial \theta} \quad (24)$$

which requires the computation of $\frac{dx}{d\theta}$,$\frac{dy}{d\theta}$ and $\frac{dz}{d\theta}$. These can be derived as follows.
The line of sight is tangential to the surface at $(x(\theta), y(\theta), z(\theta))$, therefore

$$\begin{aligned} \frac{\partial f(x(\theta),y(\theta),z(\theta),\theta)}{\partial x} Sl_x &+ \\ \frac{\partial f(x(\theta),y(\theta),z(\theta),\theta)}{\partial y} Sl_y &+ \\ \frac{\partial f(x(\theta),y(\theta),z(\theta),\theta)}{\partial z} Sl_z &= 0\ . \end{aligned} \quad (25)$$

Differentiating this with respect to θ yields:

$$\begin{aligned} 0 &= (Sl_x\frac{\partial^2 f}{\partial x\partial x} + Sl_y\frac{\partial^2 f}{\partial x\partial y} + Sl_z\frac{\partial^2 f}{\partial x\partial z})\frac{dx}{d\theta} \\ &+ (Sl_x\frac{\partial^2 f}{\partial x\partial y} + Sl_y\frac{\partial^2 f}{\partial y\partial y} + Sl_z\frac{\partial^2 f}{\partial y\partial z})\frac{dy}{d\theta} \\ &+ (Sl_x\frac{\partial^2 f}{\partial x\partial z} + Sl_y\frac{\partial^2 f}{\partial y\partial z} + Sl_z\frac{\partial^2 f}{\partial z\partial z})\frac{dz}{d\theta} \\ &+ (Sl_x\frac{\partial^2 f}{\partial x\partial \theta} + Sl_y\frac{\partial^2 f}{\partial y\partial \theta} + Sl_z\frac{\partial^2 f}{\partial z\partial \theta}) \end{aligned} \quad (26)$$

Furthermore, $(x(\theta), y(\theta), z(\theta))$ is constrained to move along the line of sight, therefore

$$N1_x\frac{dx}{d\theta} + N1_y\frac{dy}{d\theta} + N1_z\frac{dz}{d\theta} = 0 \quad (27)$$

$$N2_x\frac{dx}{d\theta} + N2_y\frac{dy}{d\theta} + N2_z\frac{dz}{d\theta} = 0 \quad (28)$$

Equations 26, 27 and 28 are three linear equations in the three unknowns $\frac{dx}{d\theta}$, $\frac{dy}{d\theta}$ and $\frac{dz}{d\theta}$ that can thus be computed. $\frac{df(x(\theta),y(\theta),z(\theta),\theta)}{d\theta}$ can then be derived using the chain rule of Equation 24.

Apart from Equations 12 and 13 one also needs to implement the following differentials:

$$\begin{aligned} \frac{\partial \mathbf{x}}{\partial x_i} &= [x_1, x_2, x_3, 0]^T,\ x_i = 1,\ x_{j\neq i} = 0 \\ \frac{\partial d}{\partial x_i} &= 2 * \mathbf{x}^T \cdot \mathbf{Q}_\Theta^T \cdot \mathbf{Q}_\Theta \cdot \frac{\partial \mathbf{x}}{\partial x_i} \\ \frac{\partial^2 d}{\partial x_i \partial x_j} &= 2 * \frac{\partial \mathbf{x}}{\partial x_i}^T \cdot \mathbf{Q}_\Theta^T \cdot \mathbf{Q}_\Theta \cdot \frac{\partial \mathbf{x}}{\partial x_j} \\ \frac{\partial^2 d}{\partial x_i \partial \theta} &= 2 * \mathbf{x}^T \cdot \left(\frac{\partial}{\partial \theta}\mathbf{Q}_\Theta^T\right) \cdot \mathbf{Q}_\Theta + \mathbf{Q}_\Theta^T \cdot \left(\frac{\partial}{\partial \theta}\mathbf{Q}_\Theta\right) \cdot \frac{\partial \mathbf{x}}{\partial x_i} \\ \frac{\partial^2 f}{\partial x_i \partial x_j} &= \frac{\partial f}{\partial d}\frac{\partial^2 d}{\partial x_i \partial x_j} + \frac{\partial^2 f}{\partial x_i \partial x_j}\frac{\partial d}{\partial x_i}\frac{\partial d}{\partial x_j} \\ &= -2 * exp(-2 * d(\mathbf{x},\theta)) * \frac{\partial^2 d}{\partial x_i \partial x_j} - 2\frac{\partial d}{\partial x_i}\frac{\partial d}{\partial x_j} \\ \frac{\partial^2 f}{\partial x_i \partial \theta} &= \frac{\partial f}{\partial d}\frac{\partial^2 d}{\partial x_i \partial \theta} + \frac{\partial^2 f}{\partial d \partial d}\frac{\partial d}{\partial x_i}\frac{\partial d}{\partial \theta} \\ &= -2 * exp(-2 * d(\mathbf{x},\theta)) * \frac{\partial^2 d}{\partial x_i \partial \theta} - 2\frac{\partial d}{\partial x_i}\frac{\partial d}{\partial \theta} \end{aligned}$$

References

[1] J.K. Aggarwal and Q. Cai. Human motion analysis: a review. *Computer Vision and Image Understanding*, 73(3):428–440, 1999.

[2] J. F. Blinn. A generalization of algebraic surface drawing. *ACM Transactions on Graphics*, 1(3):235–256, 1982.

[3] Ch. Bregler and J. Malik. Tracking People with Twists and Exponential Maps. In *Conference on Computer Vision and Pattern Recognition*, Santa Barbara, CA, June 1998.

[4] J.J. Craig. *Introduction to robotics: mechanics and control*, chapter 5. Electrical and Computer Engineering. Addison-Wesley, 2nd edition, 1989.

[5] Q. Delamarre and O. Faugeras. 3D Articulated Models and Multi-View Tracking with Silhouettes. In *International Conference on Computer Vision*, Corfu, Greece, September 1999.

[6] J. Deutscher, A. Blake, and I. Reid. Articulated Body Motion Capture by Annealed Particle Filtering. In *CVPR*, Hilton Head Island, SC, 2000.

[7] P. Fua. Reconstructing Complex Surfaces from Multiple Stereo Views. In *International Conference on Computer Vision*, pages 1078–1085, Cambridge, MA, June 1995. Also available as Tech Note 550, Artificial Intelligence Center, SRI International.

[8] P. Fua. Regularized Bundle-Adjustment to Model Heads from Image Sequences without Calibration Data. *International Journal of Computer Vision*, 38(2):153–171, July 2000.

[9] D.M. Gavrila. The Visual Analysis of Human Movement: A Survey. *Computer Vision and Image Understanding*, 73(1), January 1999.

[10] A. Hilton, D. Beresford, T. Gentils, R. Smith, and W. Sun. Virtual People: Capturing Human Models to Populate Virtual Worlds. In *Computer Animation*, Geneva, Switzerland, May 1999.

[11] I.A. Kakadiaris and D. Metaxas. 3D Human body model acquisition from multiple views. *International Journal of Computer Vision*, 30(3):191–218, December 1998.

[12] F. Lerasle, G. Rives, M. Dhome, and A. Yassine. Human Body Tracking by Monocular Vision. In *European Conference on Computer Vision*, pages 518–527, Cambridge, England, April 1996.

[13] T.B. Moeslund and E. Granum. A Survey of Computer Vision-Based Human Motion Capture. *Computer Vision and Image Understanding*, 81(3), March 2001.

[14] R. Plänkers and P. Fua. Tracking and Modeling People in Video Sequences. *Computer Vision and Image Understanding*, 81:285–302, March 2001.

[15] W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling. *Numerical Recipes, the Art of Scientific Computing*. Cambridge U. Press, Cambridge, MA, 1986.

[16] H. Saito and T. Kanade. Shape Reconstruction in Projective Grid Space from Large Number of Images. In *Conference on Computer Vision and Pattern Recognition*, Ft. Collins, CO, June 1999.

[17] H. Sidenbladh, M. J. Black, and D.J. Fleet. Stochastic tracking of 3D human figures using 2D image motion. In *European Conference on Computer Vision*, June 2000.

[18] S. Sullivan, L. Sandford, and J. Ponce. Using geometric distance fits for 3–d. object modeling and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(12):1183–1196, December 1994.

[19] D. Thalmann, J. Shen, and E. Chauvineau. Fast Realistic Human Body Deformations for Animation and VR Applications. In *Computer Graphics International*, Pohang, Korea, June 1996.