Modeling Human Bodies from Video Sequences

N. D'Apuzzo[a], R. Plänkers[b], P. Fua [b], A. Gruen[a] and D. Thalmann[b]

[a]Institute of Geodesy and Photogrammetry, ETHZ, Zürich, Switzerland

[a]Computer Graphics Lab (LIG), EPFL, Lausanne, Switzerland

ABSTRACT

In this paper, we show that, given video sequences of a moving person acquired with a multi-camera system, we can track joint locations during the movement and recover shape information. We outline techniques for fitting a simplified model to the noisy 3–D data extracted from the images and a new tracking process based on least squares matching is presented. The recovered shape and motion parameters can be used to either reconstruct the original sequence or to allow other animation models to mimic the subject's actions. Our utlimate goal is to automate the process of building complete and realistic animation models of humans, given a set of video sequences.

Keywords: Human Animation, Image Sequences, 3–D Tracking, Stereo, Model Fitting

1. INTRODUCTION

Synthetic modeling of human bodies and the simulation of motion is a longstanding problem in animation and much work is involved before a near-realistic performance can be achieved. At present, it takes an experienced designer a very long time to build a complete and realistic model that closely resembles a specific person. Digital photogrammetry offers a means to obtain the necessary data faster and in a more realistic fashion. Our ultimate goal is to automate the process and to produce realistic animation models given a set of video sequences. Eventually the whole task should be performed quickly by an operator who is not necessarily an experienced graphics designer. We should be able to invite a visitor to our laboratory, make him walk in front of a set of cameras, and produce, within a single day, a realistic animation of himself.

In this paper, we show that, given video sequences of a person moving in front of the camera, we can recover shape information and joint locations, both of which are essential to instantiate the model. This is achieved with minimal human intervention: to initialize the process, the user simply has to click on the approximate location of a few key joints in one image triplet. The recovered shape and motion parameters can be used to reconstruct the original movement or to allow other animation models to mimic the subject's actions.

We concentrate on a video-based approach because of its comparatively low cost and good control of the dynamic nature of the process. While laser scanning technology provides a fairly good surface description of a static object from a given viewpoint, videogrammetry allows us in addition to measure and track particular points of interest, such as joints, and to record and track surface and point features around the object.

The problem to be solved is twofold: first, robustly extract image information from the data; second, fit the animation models to the extracted information. In this work, we use video sequences acquired with three synchronized cameras to extract tracking and stereo information.

We first introduce our approach to computing 3–D stereo information and 3–D surface trajectories. We then present the animation model we use. Finally, we introduce our fitting procedure and show how we can handle the different kinds of input information.

Authors' e-mail addresses:
[a]{nicola,agruen}@geod.ethz.ch
[b]{plaenker,fua,thalmann}@lig.di.epfl.ch
Homepage: http://www.geod.ethz.ch/p02/projects/body

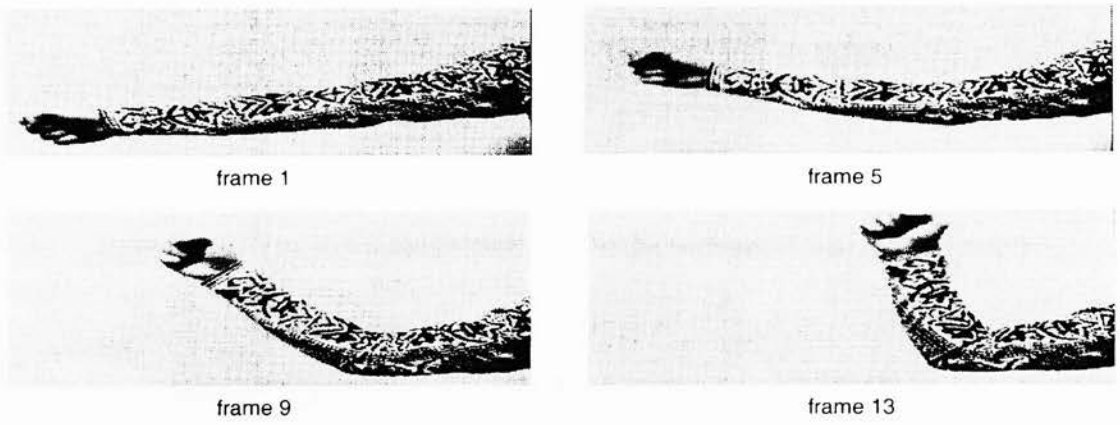


Figure 1. Four frames of a sequence (only odd lines are processed)

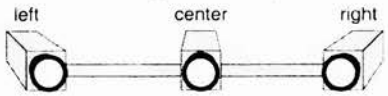


Figure 2. Arrangement of the three CCD cameras

2. 3–D LSM TRACKING ALGORITHM

Our approach to tracking is based on multi-image recording. In previous work (Fua et al.[1]), we presented a tracking of retroreflective points stuck on the skin. We have since developed an approach to tracking natural points using least squares matching techniques,[2] without using markers. To simplify the development, we have acquired sequences of the arm of a person wearing a well textured sweater (see Figure 1). The texture of the clothes facilitates the establishment of correspondences between the images of the different views and between images of subsequent frames. However, our final goal is to handle a wide range of textures, includeing bare skin.

2.1. Image Acquisition

Three CCD cameras are arranged in a line (left, center, right) as shown in Figure 2. They acquire 768×576 8 bits synchronized image triplets.

The CCD cameras are interlaced, i.e. a full frame is split into two fields which are recorded and read out consecutively. As odd and even lines of an image are captured at different times, a saw pattern is created in the image when recording moving objects. For this reason only the odd lines of the images are processed, at the cost of reducing the spatial resolution in vertical image coordinate direction by 50 percent.

2.2. System Calibration and Orientation

For calibration and orientation, we use the reference bar method (Maas[3]). A reference bar with two retroreflective target points is moved through the object space and at each location image triplets are acquired. The image coordinates of the two target points are measured with centroid operations for each triplet. The three camera system can then be calibrated and oriented by self-calibrating bundle adjustment with the additional information of the known distance between the two points at every location.

2.3. Surface Measurement

The first task of the tracking process is the measurement of the object surface at the beginning of the sequence. The algorithm used for the measurement of the surface is analogous to the one presented by D'Apuzzo.[4] The stereo matcher is based on the adaptive least squares method.[2] It considers an image patch around a selected point. One image is used as template and the others as search images. The patch in the search image is modified by an affine transformation (translations, sheerings and scalings) and the gray levels are varied by multiplicative and additive parameters. The algorithm finds the corresponding point in the neighborhood of the initial point in the search images by minimizing the sum of the squares of the differences between the gray levels in these patches. Figure 3 shows the result of the least squares matching with an image patch of 13×13 pixels.

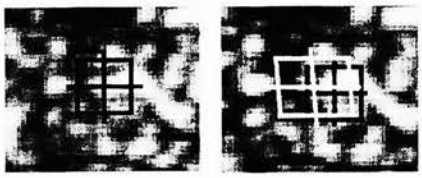


Figure 3. Least squares matching algorithm. Left: template image, right: search image

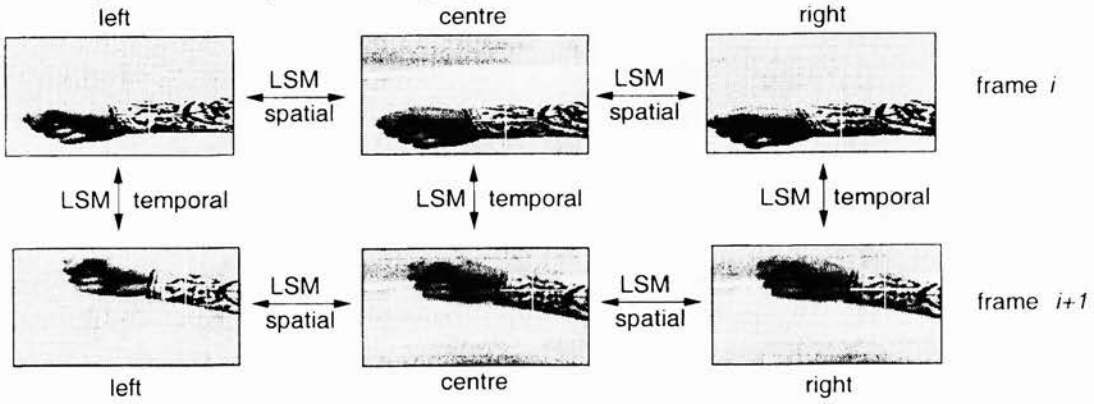


Figure 4. Tracking process

The black boxes represent the patches selected (initial location in the search image) and the white box represents the affinely transformed patch.

At the beginning of the process approximations for a few corresponding points have to be manually selected in the three images. Starting from these seed points, the stereo matcher automatically determines a dense set of corresponding points in the triplets. The 3–D coordinates of the matched points are determined by forward intersection using the calibration and orientation results.

2.4. Tracking Process

The tracking process is based on least squares matching techniques. The spatial correspondences between the three images of the different views and also the temporal correspondences between subsequent frames are determined with the same least squares matching algorithm mentioned before.

2.4.1. Tracking single points

To start the process a triplet of corresponding points in the three images is needed. This point is tracked through the sequence in the three images and therefore its 3–D trajectory can be computed. Figure 4 depicts the use of the least squares matching algorithm to track the point.

In frame i, a triplet of corresponding points in the three images is established with the least squares matching algorithm (*spatial LSM*). In each of the three images (left, center, right) a correspondent point is matched in the next frame $i + 1$ also with the same least squares matching algorithm (*temporal LSM*). Figure 5 explains how the temporal correspondences are established between subsequent frames.

For the frame $i + 1$ a linear prediction of the position of the tracked point is made: it is assumed that the movement made from frame i to frame $i + 1$ is the same as from frame $i - 1$ to frame i. Around this predicted position in the frame $i + 1$, a search box is defined. This box is scanned for searching the position which has the best value of cross-correlation between the image of frame i and the image of frame $i + 1$. This position is considered an approximation of the exact position of the point to be tracked. The least squares matching algorithm is applied at that position and the result can be considered the exact position of the tracked point in the new frame. This process is done independently for the three images of the different views. A *spatial LSM* is executed at the positions resulting from the *temporal LSMs* and if no significant differences occurs between the two matches, the point can be considered exactly tracked. Figures 6 and 7 show the results of tracking six single points through the sequence.

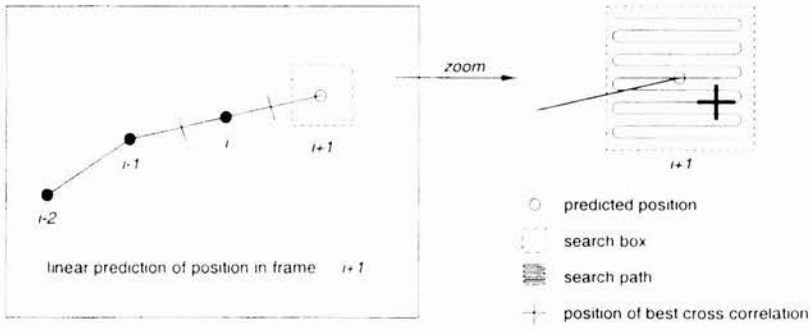


Figure 5. Tracking in image space: at the position of best cross correlation is applied *LSM temporal*

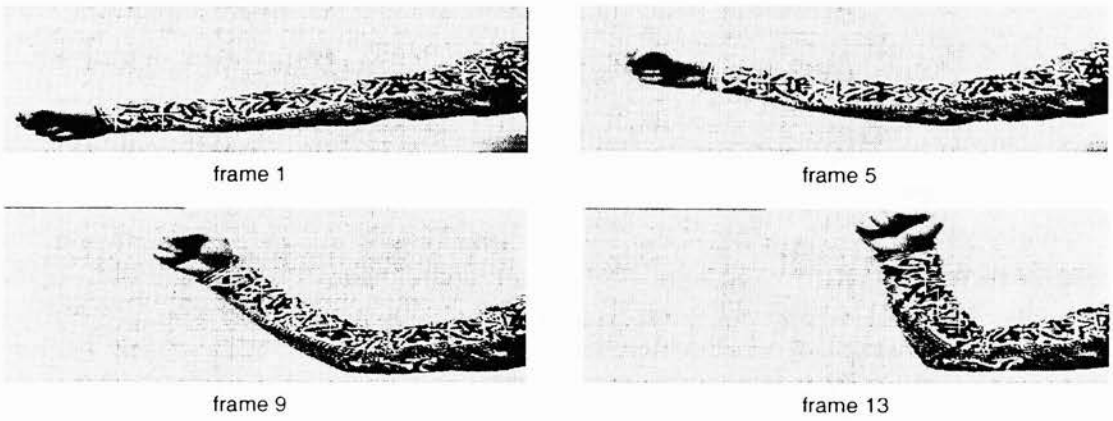


Figure 6. Tracking six single points

Figure 7. 2–D trajectories of six single points

Figure 8. 3–D trajectories of six tracked points. Left: front view, right: side view

Figure 9. Vector field of trajectories of surface tracking

The 3–D trajectory of the tracked point is determined by computing the 3–D coordinates of the point through the sequence by forward intersection. Besides, velocities and accelerations are also computed. Figure 8 shows the trajectories of the six single points.

2.4.2. Tracking the surface

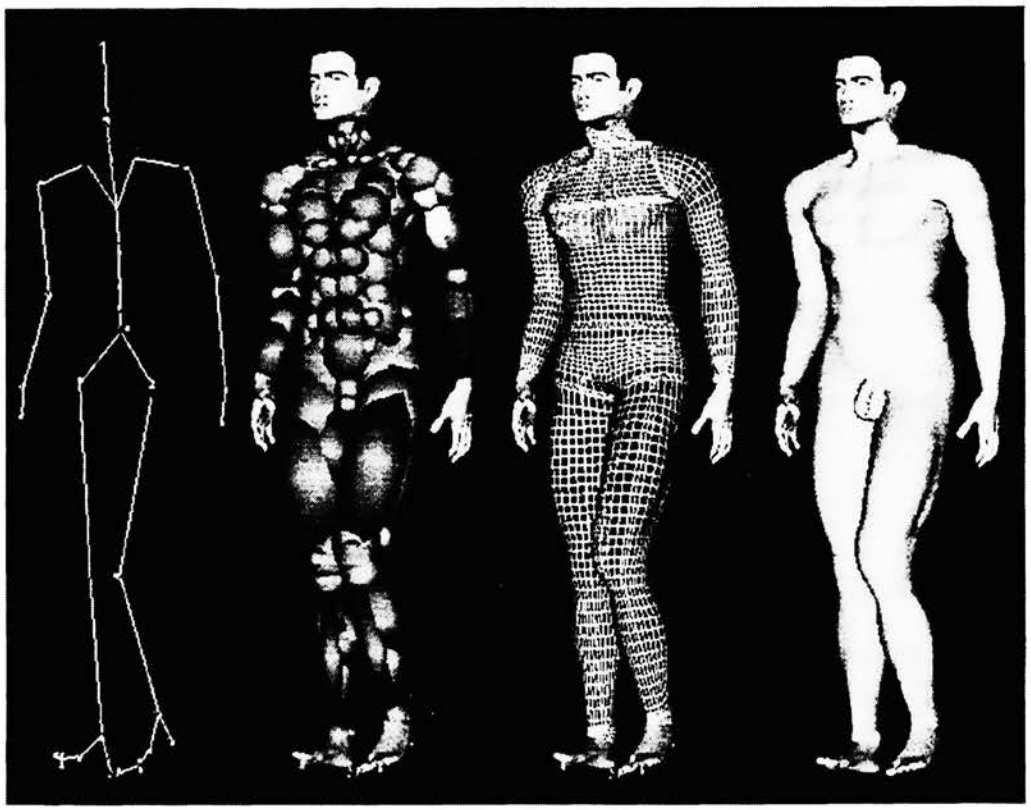
The tracking algorithm is applied to all the points measured on the surface at the beginning of the sequence. The result can be seen as a vector field of trajectories (position, velocity and acceleration). An important advantage of this general tracking scheme of all the points is that at the end the results can be checked for consistency and local uniformity of the movement. Figure 9 shows the result after passing through two filters.

The first filter consists of thresholds for the velocity and the acceleration of the movement. The second filter checks for the local (in space and time) uniformity of the movement. Since the human body can be considered an articulated moving object, the resulting vector field of trajectories must be locally uniform, i.e. the velocity vector must be nearly constant in sufficiently small regions at a particular time. To check this property, the single trajectories are compared to local (in space and time) mean values of the velocity vector. If the differences are too large, the trajectory is considered false and it is truncated or removed. The results of this filtering are not only trajectories without errors, but also surface measurement (in form of a 3–D points cloud) at each time instance without errors. The possible errors in the surface measurement done at the beginning of the sequence are removed during the tracking process, since they probably generate false trajectories.

The tracking system starts with a set of points that constantly becomes smaller during the process, because erroneous trajectories are removed or truncated by the filters and because points disappear from the scene during the sequence. To improve the system, it is therefore essential to regenerate lost points and to create new ones as they become visible. New neighborhood heuristics will also be added to the filters.

3. MODELS

In this section, we first describe the complete model that we use for animation purposes. This model has too many degrees of freedom to be effectively fit to noisy data without a-priori knowledge. We therefore introduce a simplified model that we have used to derive an initial shape and position. In this work, we will use this knowledge to initialize the complete one before refining it.



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

Figure 10. The layered human body model: (a) Skeleton. (b) Ellipsoidal metaballs used to simulate muscles and fat tissue. (c) Polygonal surface representation of the skin. (d) Shaded rendering.

3.1. Complete Animation Model

Generally, virtual humans bodies are structured as articulated bodies defined by a skeleton. When an animator specifies an animation sequence, he defines the motion using this skeleton.

A skeleton 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 where the limb linked to that point may move.

Our model[5] is depicted by Figure 10. It incorporates a highly effective multi-layered approach for constructing and animating realistic human bodies. Ellipsoidal metaballs are used to simulate the gross behavior of bone, muscle, and fat tissue; they are attached to the skeleton and arranged in an anatomically-based approximation. The skin construction is made in a three step process. First, the implicit surface resulting from the combination of the metaballs influence is automatically sampled along cross-sections with a ray casting method.[6,5] Second, the sampled points constitute control points of a B-spline patch for each body part (limbs, trunk, pelvis, neck). Third, a polygonal surface representation is constructed by tessellating those B-spline patches for seamless joining different skin pieces together and final rendering. The method, simple and intuitive, combines the advantages of implicit, parametric and polygonal surface representation, producing very realistic and robust body deformations. By applying smooth blending twice (metaball potential field blending and B-spline basis blending), the model's data size is significantly reduced.

Since the overall appearance of a human body is very much influenced by its internal muscle structures, the layered model is the most promising for realistic human animation. The key advantage of the layered methodology is that once the layered character is constructed, only the underlying skeleton need be scripted for animation; consistent yet expressive shape deformations are generated automatically.

3.2. Skeleton and State Vector

The state of the skeleton is described by the combined state vector

$$S_{body} = [S_{motion}, S_{skel}] \ . \tag{1}$$

The *static* or *init* state of the skeleton S_{skel} consists of the rotations and translations from each joint to the preceding one. It is fixed for a given instance of the body model. The variable or ***motion*** state vector S_{motion} contains the actual values for each degree of freedom (DOF), i.e. the angle around the z-axis towards the next DOF. They reflect the position and posture of the body with respect to its rest position. All DOFs have only a single-angle freedom, since more complicated articulations are split into several, single DOF joints.

For any given limb or body part a partial motion state vector for its parent joint can be written as $S_{part} = [S_{pre}, \Theta_\kappa]$, where S_{pre} is the state vector of the preceding joints, and Θ_κ is a rotation angle around the z-axis of that joint.

The position of joints in a global world referential is obtained by multiplying the local coordinates by a transformation matrix. This matrix is computed recursively by multiplying all the transformation matrices that correspond to the preceding joints in the body hierarchy:

$$X_j = \prod_i D^i(S) * X_w \ , \tag{2}$$

with $X_{j,w} = [x, y, z]^T$ being joint local, resp. world global, coordinates and the homogeneous transformation matrices D^i, which depend on the state vector S, ranging from the root articulation's first to the reference articulations's last DOF. These matrices are split into static and motion matrices, according to the state vector. They are of the form

$$D = D_{rot_z} * D_{ini} \ . \tag{3}$$

The rotation matrix D_{rot_z} is defined by the motion state vector. It is a sparse matrix allowing only a rotation around the local z-axis (Θ_κ). The static transformation $D_{ini} = (RX + sT)$ is a matrix directly taken from the standard skeleton. These matrices translate by the bone length and rotate the local coordinate system from the joint to its parent. The matrix entries are calculated using values s from the state vector S_{skel}. The variable coefficient s is necessary because the exact size of the limbs may vary from person to person. For the first DOF of an articulation, D_{ini} is usually dense but the other DOFs have no translation and the rotational part consists only of a permutation of the axes to ensure that the DOF rotates around the z-axis.

The articulations may consist of several DOFs, each having its own transformation matrix D. For example, the elbow joint has two DOFs flexion and twisting: $D^{elbow} = D_{rot_{twist}} * D_{ini_{twist}} * D_{rot_{flex}} * D_{ini_{flex}}$.

3.3. Simplified Model of a Limb

To be able to robustly estimate the skeleton's position and to reduce the number of DOFs, we replace the multiple metaballs of Section 3.1 by only three metaballs attached to each limb. In an earlier approach,[7,1] we used only one ellipsoid per limb. This had the advantage of fast calculations but the errors introduced by the imperfection of the model, were big enough to lead to unsatisfactory fittings. We therefore decided to use a slightly more complicated model which is capable of better modeling the shape of human limbs. Figure 11 shows the model we have used to recover the shape and motion from the arm sequences of Figure 1.

The metaballs are rigidly attached to the skeleton. They have a fixed orientation and a fixed position relative to the length of the limb. Only their size, i.e. their radii, are subject to modification by the fitting process.

The different body parts are segmented before the fitting starts. This is simply done during the initialization phase where the model takes an approximate posture which is good enough to assign a 3–D observation to the closest limb. Thus, we do not have to wait for a motion of the person to split a limb such as the arm into two parts, upper arm and forearm, as is the case in the work of Kakadiaris and Metaxas.[8]

More sophisticated models that include both global and local deformations, such as tapered superquadrics[8] or evolving surfaces,[9] may be able to approximate more closely the exact shape of the limb. However, they require the setting of more parameters and are thus harder to fit.

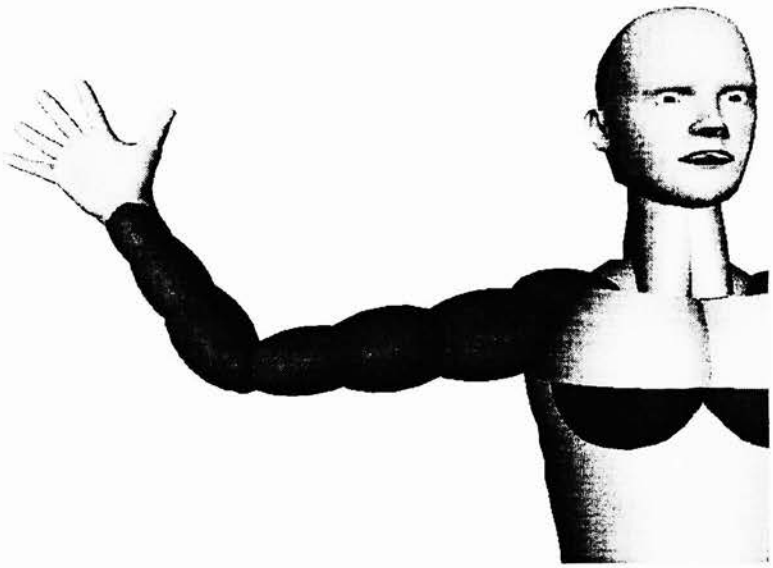


Figure 11. Simplified model for fitting. Although the metaballs are displayed as distinct ellipsoids, they blend into each other to form a single smooth surface.

3.4. Metaballs and their Mathematical Description

3.4.1. Definition[10]

In the basic formulation proposed by Blinn,[11] metaballs or *blobs* are defined by a set of points $P_i(x_i, y_i, z_i)$ where each point is the source of a potential field. Each source is defined by a *field function* $F_i(x, y, z)$ that maps $\mathbb{R}^3$ to $\mathbb{R}$ (or a subset of $\mathbb{R}$). At a given point $P(x, y, z)$ of the Euclidean space, the fields of all sources are computed and added together, leading to the global field function

$$F(x, y, z) = \sum_{i=1}^{n} F_i(x, y, z) \quad . \tag{4}$$

A curved surface can then be defined from the global field function F by giving a threshold value T and rendering the following equipotential surface $\mathbf{S}$ for this threshold:

$$\mathbf{S} = \left\{ (x, y, z) \in \mathbb{R}^3 \mid F(x, y, z) = T \right\} \quad . \tag{5}$$

Conceptually it is usually simpler to consider field function F_i as the composition of two functions[12]: the *distance function* d_i which maps $\mathbb{R}^3$ to $\mathbb{R}^+$, and the *potential function* f_i which maps $\mathbb{R}^+$ to $\mathbb{R}^3$:

$$F(x, y, z) = \sum_{i=1}^{n} f_i(d_i(x, y, z)) \quad . \tag{6}$$

The function $f_i(d)$ characterizes the distance between a given point $P(x, y, z)$ and the source point $P_i(x_i, y_i, z_i)$. Typically d_i is defined as a function of a user-provided parameter $r_a \in \mathbb{R}^+$ (called *effective radius*) which expresses the growing speed of the distance function. The most obvious solution for $d_i(x, y, z)$ is the Euclidean distance, but several other functions have been proposed in the literature, especially when the potential source is not reduced to a single point or its field is not equally distributed in space.

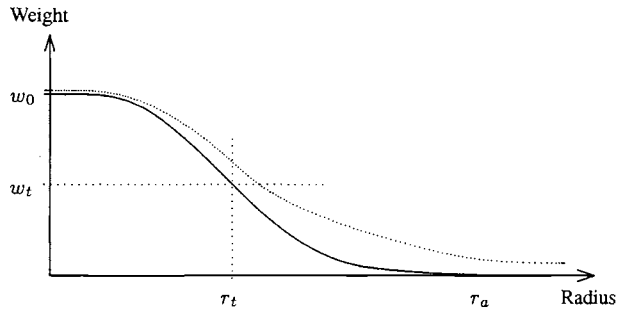


Figure 12. Two metaball field functions. The solid curve is a typical piecewise function and the dotted curve is our exponential function. r_t is the visible size of the primitive which depends on the threshold w_t and the effective radius r_a.

3.4.2. Distance function

In this work, we only consider ellipsoids as primitives because they are relatively simple but, nevertheless, allow modeling of human limbs with a reasonable low number of primitives and thus number of parameters. We represent the distance function d_i by the I/O distance of the ellipsoid that is

$$d_i(x,y,z) = \left(\frac{x}{l_x}\right)^2 + \left(\frac{y}{l_y}\right)^2 + \left(\frac{z}{l_z}\right)^2 \quad , \tag{7}$$

where $L_i = (l_x, l_y, l_z)$ are the radii of the ellipsoid, i.e. half the axis length along the principal directions.

3.4.3. Potential function

The field value at any point P in space is defined by the distances between P and the source points P_i. Figure 12 shows a typical curve of a density distribution. The center of the primitive, its source, has the greatest density. The value of the primitive's density, or *weight*, decreases toward the element's outer edge, or effective radius. The visible size of a primitive, called the *threshold radius*, is determined by the effective radius and weight.

Field functions should satisfy two criteria:

1. Extrema
   The contribution at the source itself will be some maximum value w_0, and the field will drop smoothly to zero at a distance r_a, the effective radius.

2. Smoothness
   In order to blend multiple metaballs smoothly and gradually, $f'(0) = f'(r_a) = 0$.

A single, lower degree polynomial cannot meet both criteria, hence either piecewise quadric or high order polynomials have been proposed. Their disadvantage are a high complexity and thus high computational cost.

Shen[10] used a much simpler approach in his work on human body modeling: $f_i = w_i(1-d)$, where d is defined as in Equation 7 and w_i defines the maximum weight w_0 for primitive i. This simplified field function satisfies only the first criterion. The second criterion is satisfied by using cubic B-spline blending. The simplification used by Shen yields excellent results for modeling purposes, as can be seen in Figure 10.

3.4.4. Usage in this work

However, here we are attempting to fit the model to 3–D data by minimizing an objective function. In order to do so, we need to work on a well-defined mathematical basis and the smoothness criterion is essential when fitting a shape with multiple metaballs. We therefore use an exponential field function:

$$f_i = w_i \left(\frac{1}{e^d}\right)^2 = w_i * exp(-2d) \quad , \tag{8}$$

with d also being defined as in Equation 7 and the weight being fixed for the moment ($w_0 = 1, w_t = 0.5$). We might leave the weight as a parameter for the fitting in the future since it allows to easily reproduce sharper edges.

An exponential field function is also more effective in the least squares fitting framework because its derivatives are very easy to compute. As shown in Figure 12, its equipotential surface **S** (Eq. 5) is only slightly different from the standard representation and, more importantly, it never falls to zero.

The last property has two consequences:

1. Each blob has an influence on all other blobs of the same limb, although, it will become very small for distant blobs. This is obviously undesired for modeling since the designer looses local control.
2. At the same time as each blob influences all other blobs, each blob is influenced by all observations in our fitting framework. This allows us to work with only a rough initialization of the model's posture. Since the observations are already segemented and associated to body parts, the unlimited influence does not pose any problems.

4. FITTING THE MODELS TO IMAGE DATA

From a fitting point of view, the body model of Section 3.3 embodies a rough knowledge about the shape of the body and can be used to constrain the search space. Our goal is to fix its degrees of freedom so that is conforms as faithfully as possible to the image data.

Here we use motion sequences such as the one shown in Figure 1 and corresponding stereo data computed using the method of Section 2. Thus, the expected output of our system is a state vector that describes the shape of the metaballs and a set of joint angles corresponding to their positions in each frame.

In this section, we introduce the least squares framework we use and show how we can exploit the tracking and stereo data that we derive from the images.

4.1. Least Squares Framework

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

$$f_i(S) = obs_i - \epsilon_i \ \ , 1 \leq i \leq nobs \ \ , \tag{9}$$

where S is the state vector of Equation 1 that defines the shape and position of the limb and ϵ_i is the deviation from the model. We will then minimize

$$v^T P v \Rightarrow Min \ , \tag{10}$$

where v is the vector of residuals and P is a weight matrix associated with the observations (P is usually introduced as diagonal).

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 observations equations of Equation 9 as

$$f_i^{type}(S) = obs_i^{type} - \epsilon_i \ \ , 1 \leq i \leq nobs \ \ , \tag{11}$$

with weight p_i^{type}, where *type* is one of the possible types of observations we use. In this paper, *type* is restricted to object space coordinates, although other information cues can easily be integrated.

The individual weights of the different types of observations have to be homogenized prior to estimation according to:

$$\frac{p_i^k}{p_j^l} = \frac{\left(\sigma_j^l\right)^2}{\left(\sigma_i^k\right)^2} \ , \tag{12}$$

where σ_j^l, σ_i^k are the a priori standard deviations of the observations obs_i, obs_j of type k, l.

Applying least-squares estimation implies the joint minimum

$$\sum_{type=1}^{nt} v^{type} P_{type} v^{type} \Rightarrow Min \ , \tag{13}$$

with nt the number of observation types, which then leads to the well-known normal equations which need to be solved using standard techniques.

Since our overall problem is non-linear, the results are obtained through an iteration process.

Figure 13. Arm model after being fitted to frames 1, 9 and 13 from the images of Figure 1

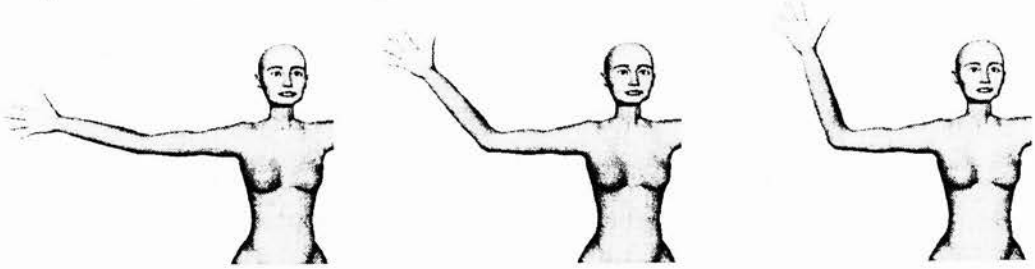


Figure 14. The recovered animation parameters applied to a completely different animation model

4.2. Using Stereo Data

Let us assume that we are given a 3–D point that has been computed using stereo data. We want to minimize the distance of the reconstructed limb to all such "attractor" points. Given the implicit description of our metaballs, the simplest way to achieve this result is to write an pseudo-observation equation of the form:

$$\sum_{i=1}^{np} w_i \cdot exp(-2d_i) = w_t - \epsilon \tag{14}$$

$$\sum_{i=1}^{np} exp\left(-2\left(\left(\frac{x_i}{lx_i}\right)^2 + \left(\frac{y_i}{ly_i}\right)^2 + \left(\frac{z_i}{lz_i}\right)^2\right)\right) = \frac{1}{2} - \epsilon \ . \tag{15}$$

where np is the number of primitives for this body part, $P_i\,(x, y, z)$ is the 3–D observation transformed into the local coordinates of primitive i with radii $L_i(lx, ly, lz)$. We use Equation 15 which is the same than Equation 14 except for the fixed weights $w_t = \frac{1}{2}, w_i = 1, i \in [1, np]$.

The optimization is effected wrt. the primitives' radii L_i and the DOFs which reside in the transformation of each observation from world global to primitive local coordinates. According to Equation 2, each P_i can be written as a function of its world coodinates and the elements of state Vector S. In practice, we experienced better convergence by iteratively alternating between primitive parameters and skeleton parameters instead of optimizing them simultaneously. For more detail we refer the interested reader to a previous publication.[1]

4.3. Preliminary Results

By fitting our model of the right arm to the stereo information obtained from the images of Figure 1, we can reconstruct the positions and shapes depicted by Figure 13. The joint angles stored in the state vector can then be used to animate the very different virtual human of Figure 14.

5. CONCLUSION AND FUTURE WORK

In this paper, we have shown that given video sequences of a moving person acquired with a multi-camera system, we can recover shape information and track joint locations during the motion. We have outlined techniques for fitting a simplified model to noisy stereo data and we have presented a new tracking process based on least squares matching. The recovered shape and motion parameters can be used to create a realistic animation. Our ultimate goal is to produce automatically, with minimal

human intervention, realistic animation models given a set of video sequences. The capability we intend to develop will be of great applicability in animation areas, since the techniques used nowadays require a very long time of manual work to generate and animate sophisticated models of humans. Automating the process will allow an increase of realism with simultaneous decrease of costs.

In future work, will next produce some synthetic data in order to test the accuracy of the system. At the moment we are applying the algorithms on sequences of full bodies in motion.

We will also investigate the possibilities of having the model guide the tracking process. If a point vanishes due to occlusion we can employ the model to predict where and when it will appear again.

The next step for the fitting process consists of refining the model from only three primitives to the full set of the animation model. This would allow us to model the filmed person very realistically. The simplified model only allows to get a good approximation of the skeleton but a rather rough approximation of the shape.

REFERENCES

1. P. Fua, A. Gruen, R. Plänkers, N. D'Apuzzo, and D. Thalmann, "Human body modeling and motion analysis from video sequences," in *International Archives of Photogrammetry and Remote Sensing*, vol. 32(B5), pp. 866–873, (Hakodate, Japan), 1998.
2. A. Gruen, "Adaptive least squares correlation: a powerful image matching technique," *South African Journal of Photogrammetry, Remote Sensing and Cartography* **14(3)**, pp. 175–187, 1985.
3. H.-G. Maas, "Image sequence based automatic multi-camera system calibration techniques," in *International Archives of Photogrammetry and Remote Sensing*, vol. 32(B5), pp. 763–768, (Hakodate, Japan), 1998.
4. N. D'Apuzzo, "Automated photogrammetric measurement of human faces," in *International Archives of Photogrammetry and Remote Sensing*, vol. 32(B5), pp. 402–407, (Hakodate, Japan), 1998.
5. 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.
6. J. Shen and D. Thalmann, "Interactive shape design using metaballs and splines," in *Implicit Surfaces*, April 1995.
7. P. Fua, R. Plänkers, and D. Thalmann, "From image synthesis to image analysis: Using human animation models to guide feature extraction," in *Fifth International Symposium on the 3-D Analysis of Human Movement*, (Chattanooga, TN), July 1998.
8. I. Kakadiaris, D. Metaxas, and R. Bajcsy, "Active part-decomposition, shape and motion estimation of articulated objects: A physics-based approach," in *Conference on Computer Vision and Pattern Recognition*, pp. 980–984, 1994.
9. R. Malladi, J. Sethian, and B. Vemuri, "Shape modeling with front propagation: A level set approach," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **17**(2), pp. 158–175, 1995.
10. J. Shen, *Human Body Modeling and Deformations*. PhD thesis, EPFL, Lausanne, Switzerland, 1996.
11. J. F. Blinn, "A generalization of algebraic surface drawing," *ACM Transactions on Graphics* **1**(3), pp. 235–256, 1982.
12. C. Blanc and C. Schlick, "Extended field functions for soft objects," in *Eurographics Workshop on Implicit Surface 95*, pp. 21–32, (Grenoble, France), 1995.